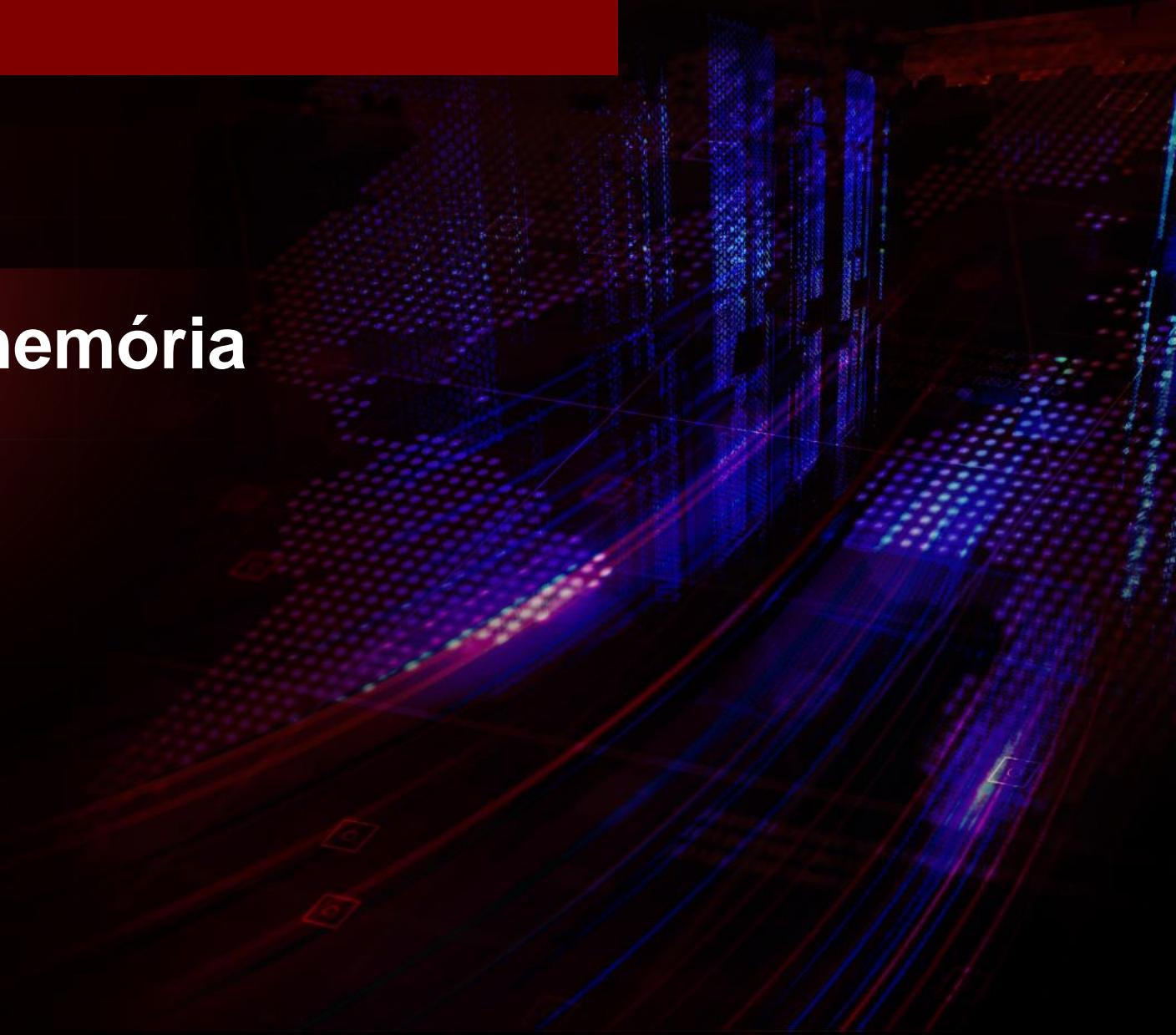


REDES NEURAIS

Redes recorrentes com memória
de longo prazo

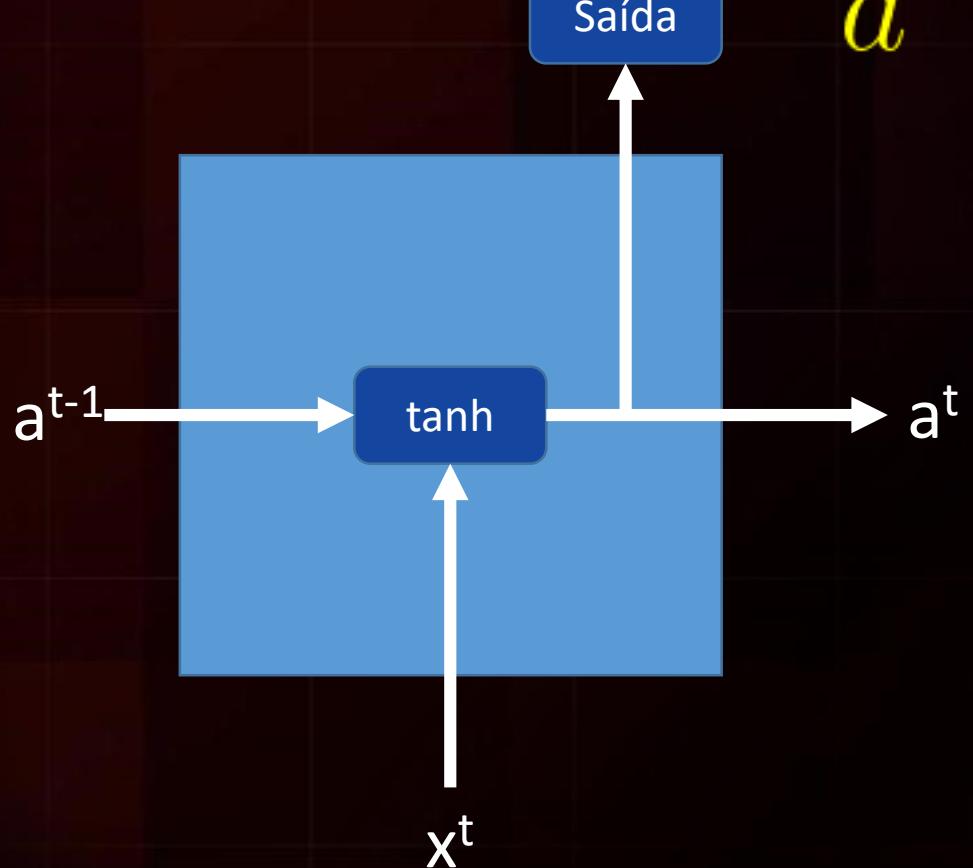


TÓPICOS

1. Desaparecimento/explosão do gradiente
2. Unidades com memória de Longo Prazo:
 - GRU e LSTM
3. Possíveis extensões



UNIDADE TRADICIONAL



$$a^t = \varphi(\mathbf{w}_a[\mathbf{a}^{t-1}, \mathbf{x}^t] + \mathbf{b}_a)$$

PROBLEMAS COM O GRADIENTE

PROBLEMA COMUM AO UTILIZAR O MÉTODO DO GRADIENTE DESCENDENTE E O ALGORITMO DE RETROPROPAGAÇÃO: DESAPARECIMENTO OU EXPLOSÃO

- **Desaparecimento do gradiente (vanishing):** o gradiente se torna ínfimo, impossibilitando o ajuste dos pesos
- **Explosão do gradiente:** dependendo da função de ativação, o valor do gradiente pode se tornar muito elevado saturando (estourando) as sinapses

PROBLEMAS COM O GRADIENTE

- Recordando o cálculo do gradiente nos neurônios ocultos:

$$\delta_j = f'(v_j) \sum_k \delta_k w_{kj}$$

- O gradiente do neurônio j consiste na multiplicação da derivada da função de ativação de j pelo somatório das derivadas dos neurônios posteriores k, camada a camada

PROBLEMAS COM O GRADIENTE

- Embora o problema não seja tão aparente nas redes alimentadas adiante, pode complicar o treinamento das redes recorrentes

$$\delta_j^1 \rightarrow f'(v_j^1) f'(v_j^2) \dots f'(v_j^t)$$

$$\varphi^t \rightarrow 0 \text{ se } \varphi < 1$$

$$\varphi^t \rightarrow \infty \text{ caso contrário}$$

$$\delta_j^{t-1} = f'(v_j^{t-1}) \sum \delta_j^t w_{jj}$$

CONSIDERAÇÕES

- Visando resolver esse problema, o modelo LSTM foi proposto em 1997
- A rede LSTM permite que tanto o sinal na fase de propagação quanto o sinal do gradiente na retropropagação sejam preservados
- Resolvendo o problema do desaparecimento/explosão do gradiente

LONG-SHORT TERM MEMORY

- Proposta em 1997 pelos professores Sepp Hochreiter e Jürgen Schmidhuber
- Ampliada por Felix Gers em 2000
 - Inclusão da porta de esquecimento



Fontes: https://de.wikipedia.org/wiki/Sepp_Hochreiter
<https://www.brainpreservation.org/team/juergen-schmidhuber/>

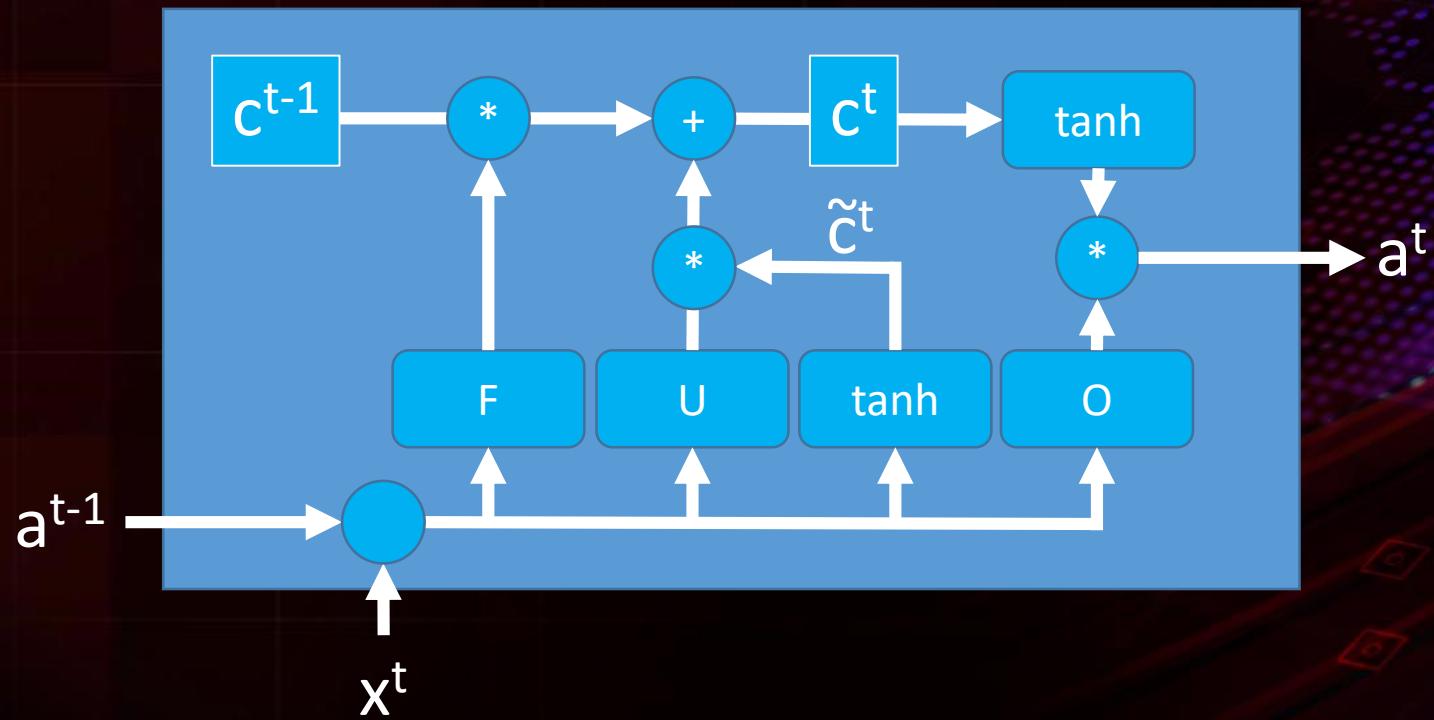
LONG-SHORT TERM MEMORY

- Uma célula LSTM é composta por três gates:
entrada, saída e esquecimento (forget)
 - Gate de entrada (ou de atualização): Gate U
 - Gate de saída: Gate O
 - Gate de esquecimento: Gate F
- A célula é capaz de recordar sinais arbitrários do passado a partir da configuração dos gates – controle do fluxo de informação
- Teoricamente, sinais podem ser mantidos por longos períodos

LONG-SHORT TERM MEMORY

A célula LSTM possui dois sinais de memória

- c^t – Sinal interno (estado da célula)
- a^t – Sinal externo (sinal de saída)



LONG-SHORT TERM MEMORY-LSTM

$$\tilde{\mathbf{c}}^t = \tanh(\mathbf{w}_a[\mathbf{a}^{t-1}, \mathbf{x}^t] + \mathbf{b}_c)$$

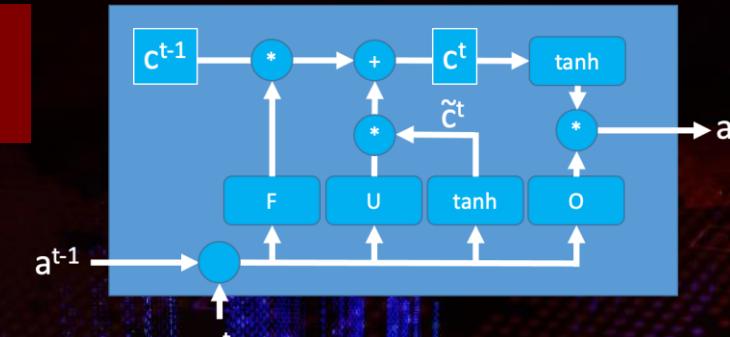
$$\Gamma_u = \varphi(\mathbf{w}_u[\mathbf{a}^{t-1}, \mathbf{x}^t] + \mathbf{b}_u)$$

$$\Gamma_f = \varphi(\mathbf{w}_f[\mathbf{a}^{t-1}, \mathbf{x}^t] + \mathbf{b}_f)$$

$$\Gamma_o = \varphi(\mathbf{w}_o[\mathbf{a}^{t-1}, \mathbf{x}^t] + \mathbf{b}_o)$$

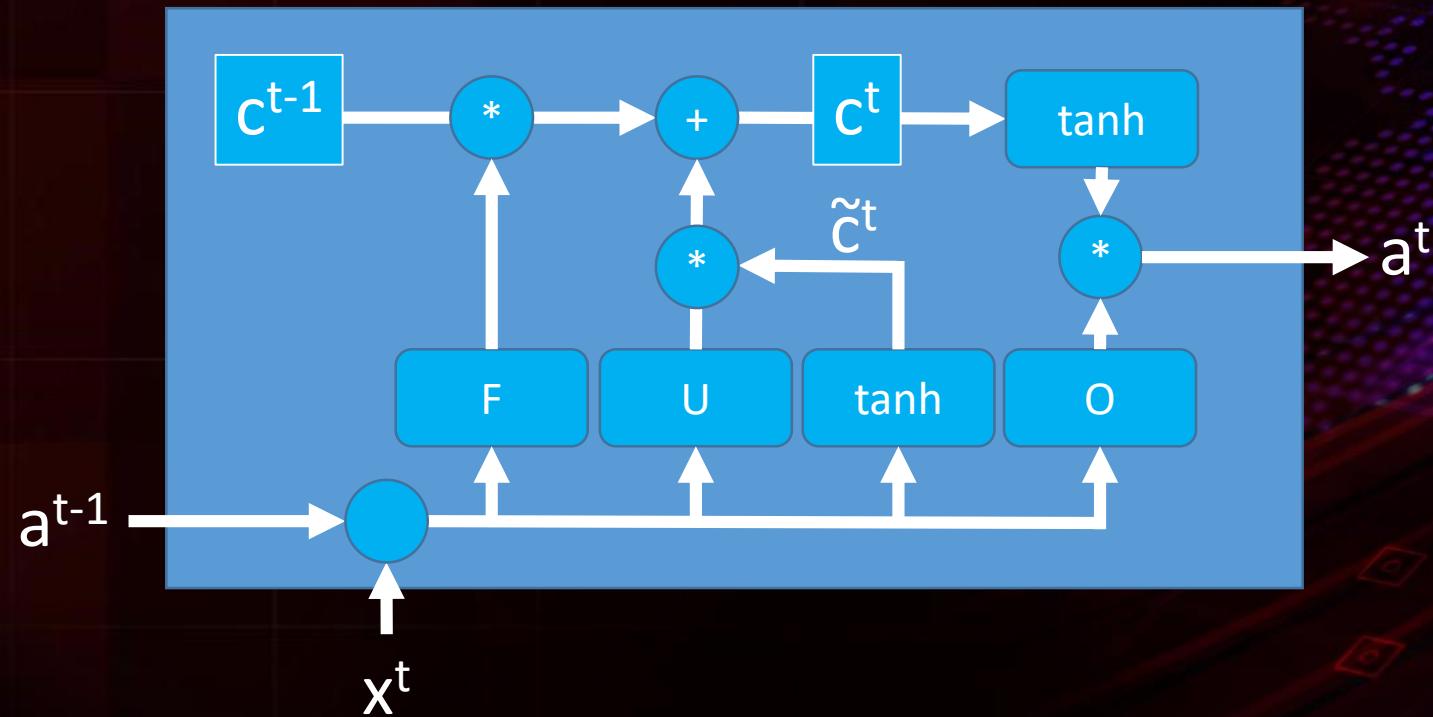
$$\mathbf{c}^t = \Gamma_u * \tilde{\mathbf{c}}^t + \Gamma_f \mathbf{c}^{t-1}$$

$$a^t = \Gamma_o * \mathbf{c}^t$$



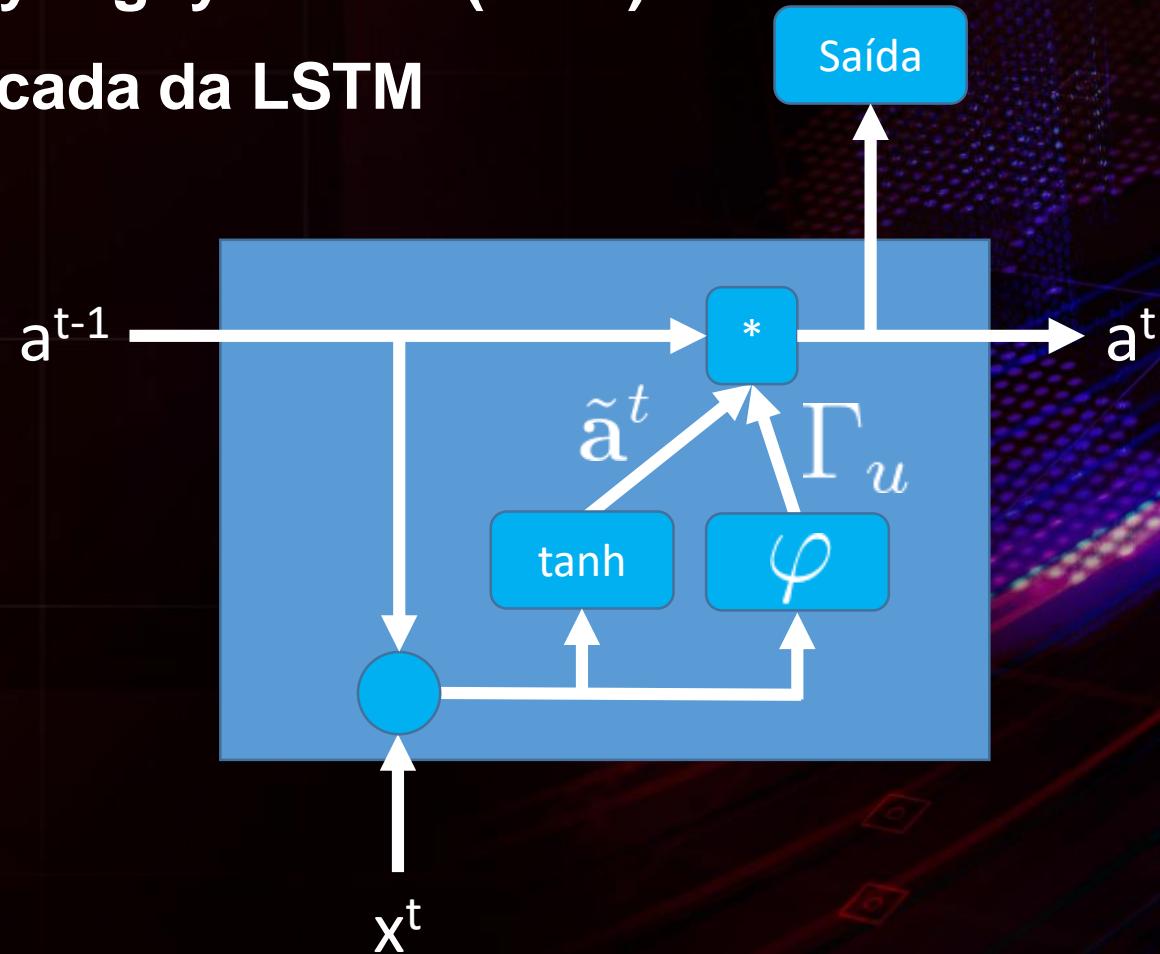
LONG-SHORT TERM MEMORY-LSTM

$$\mathbf{E}_f^t = \varphi(\mathbf{w}_g(\mathbf{a}_a^t[\mathbf{d}_f^t \mathbf{x}^{t-1}] * \mathbf{b}_g)) \mathbf{b}_c)$$



GATED RECURRENT UNIT-GRU

- Proposta por Kyunghyun Cho (2014)
- Versão Simplificada da LSTM
- Duas versões



Fonte:
https://cims.nyu.edu/people/profiles/CHO_Kyunghyun.html

GRU (SIMPLIFICADA)

NEURÔNIO CONTÉM UM GATE DE ATUALIZAÇÃO Γ_u

- Responsável por indicar quais sinais são preservados ao longo do tempo

$$\tilde{\mathbf{a}}^t = \tanh(\mathbf{w}_a[\mathbf{a}^{t-1}, \mathbf{x}^t] + \mathbf{b}_a)$$

$$\Gamma_u = \varphi(\mathbf{w}_u[\mathbf{a}^{t-1}, \mathbf{x}^t] + \mathbf{b}_u)$$

$$\mathbf{a}^t = \Gamma_u * \tilde{\mathbf{a}}^t + (1 - \Gamma_u) * \mathbf{a}^{t-1}$$

GRU (COMPLETA)

NEURÔNIO GRU (FULL): GATE DE RELEVÂNCIA

$$\tilde{\mathbf{a}}^t = \tanh(\mathbf{w}_a[\Gamma_r * \mathbf{a}^{t-1}, \mathbf{x}^t] + \mathbf{b}_a)$$

$$\Gamma_r = \varphi(\mathbf{w}_r[\mathbf{a}^{t-1}, \mathbf{x}^t] + \mathbf{b}_r)$$

$$\Gamma_u = \varphi(\mathbf{w}_u[\mathbf{a}^{t-1}, \mathbf{x}^t] + \mathbf{b}_u)$$

$$\mathbf{a}^t = \Gamma_u * \tilde{\mathbf{a}}^t + (1 - \Gamma_u) * \mathbf{a}^{t-1}$$

GRU x LSTM

GRU

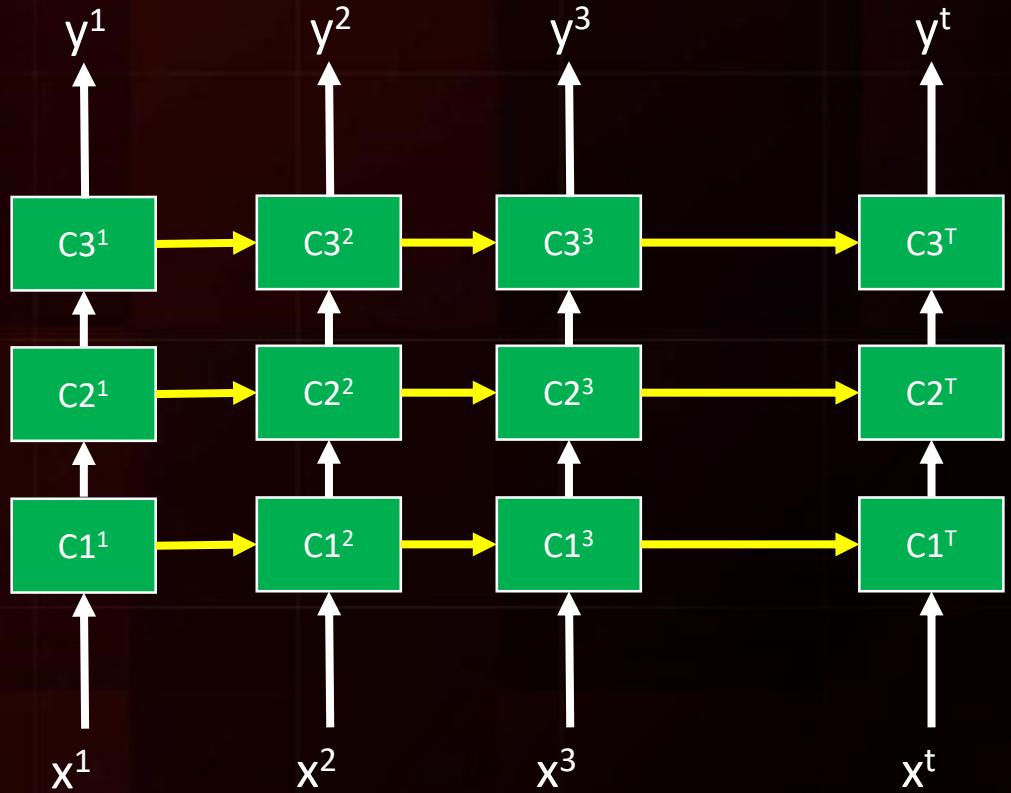
- Um (ou dois) gates
- Dois (ou três) matrizes de pesos
- Apenas um sinal interno

LSTM

- Três gates
- Quatro matrizes de pesos
- Dois sinais internos (a e c)

Por possuir mais parâmetros, a LSTM tem mais flexibilidade. Na prática, os resultados são similares

REDES PROFUNDAS

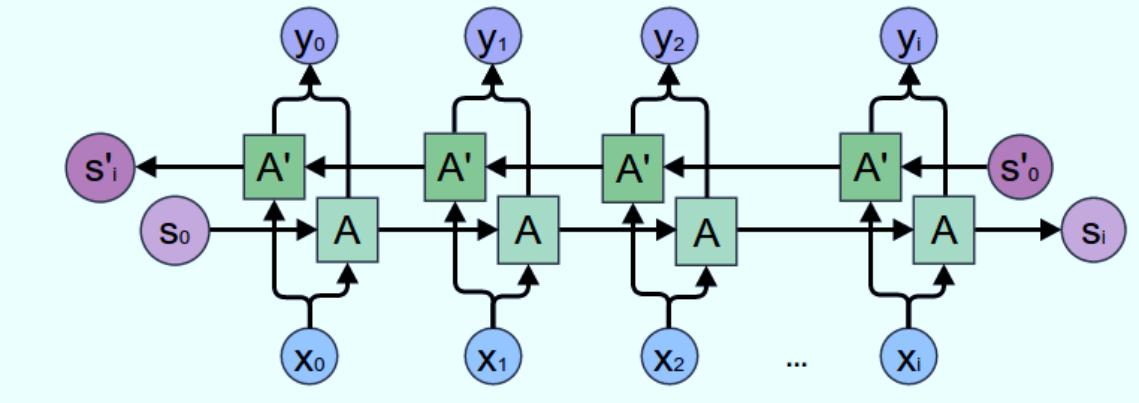


- Geralmente não se usa muitas camadas
- Excesso de parâmetros dificulta o treinamento

VARIACÕES

Redes LSTM (ou GRU) Bidirecionais

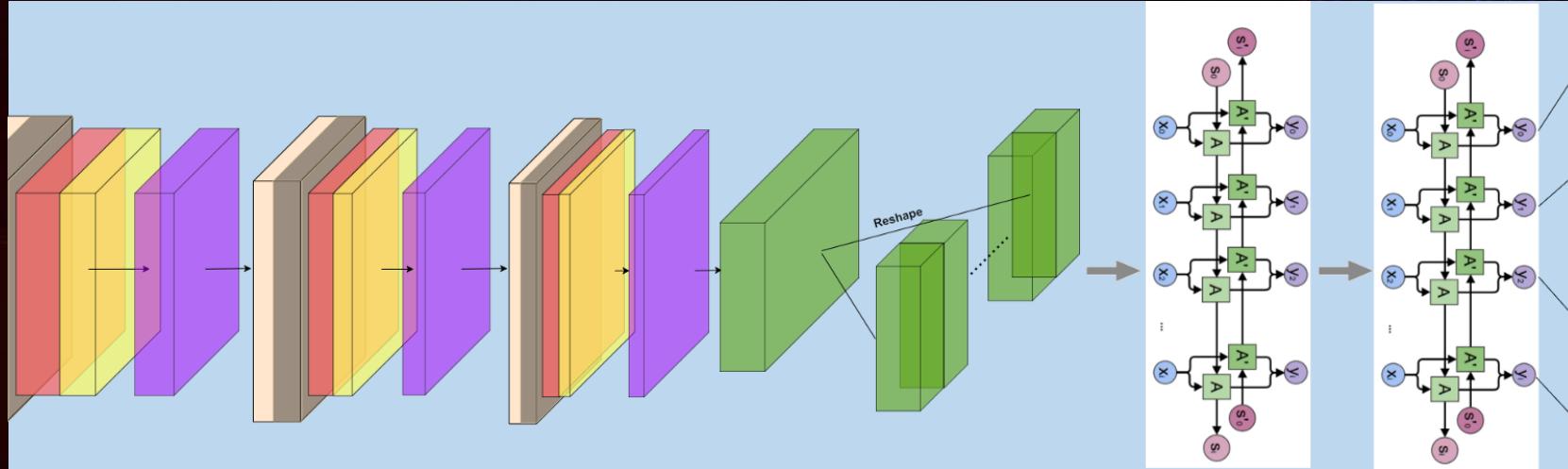
- Precisamos avaliar a ativação dos neurônios nos dois sentidos $[\vec{a}, \vec{a}]$
- A saída depende das ativações calculadas nos dois sentidos:
$$\hat{y} = g(W_y[\vec{a}, \vec{a}] + b)$$
- Problema: precisamos avaliar a sequência inteira antes de gerar uma saída



Fonte:
<http://colah.github.io/posts/2015-09-NN-Types-FP/>

VARIACÕES

- RNNs Convolucionais



- Redes Transformers etc.

Fonte: <https://towardsdatascience.com/an-approach-towards-convolutional-recurrent-neural-networks-a2e6ce722b19>

O QUE VIMOS?

- Conhecemos o problema relacionado ao desaparecimento ou explosão do gradiente
- Aprendemos sobre as redes com memória de Longo Prazo: GRU e LSTM

PRÓXIMA AULA

- Na nossa próxima aula, última videoaula, iremos realizar alguns experimentos práticos com a rede GRU em Pytorch

ATÉ A PRÓXIMA VIDEOAULA