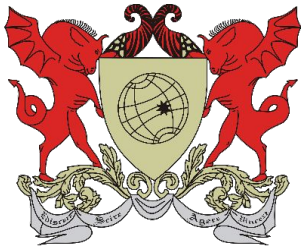


Redes Neurais Recorrentes

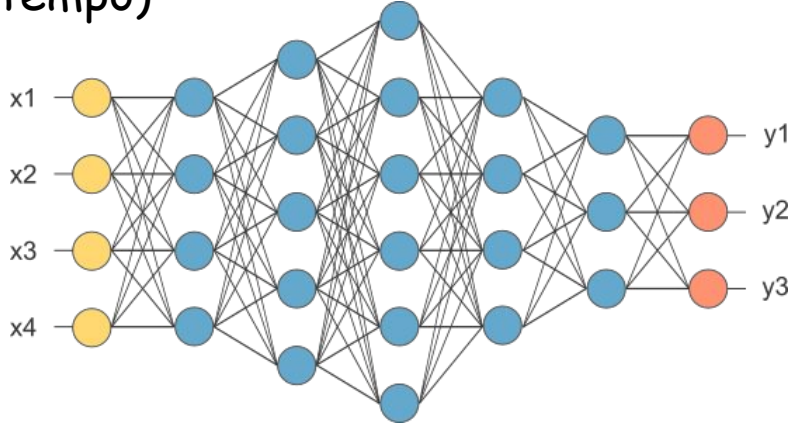
Alcione de Paiva Oliveira - DPI/UFV

Baseado no tutorial http://colah.github.io/posts/2015-08-Understanding-LSTMs/ e vários outros

Redes Neurais Recorrentes

Introdução

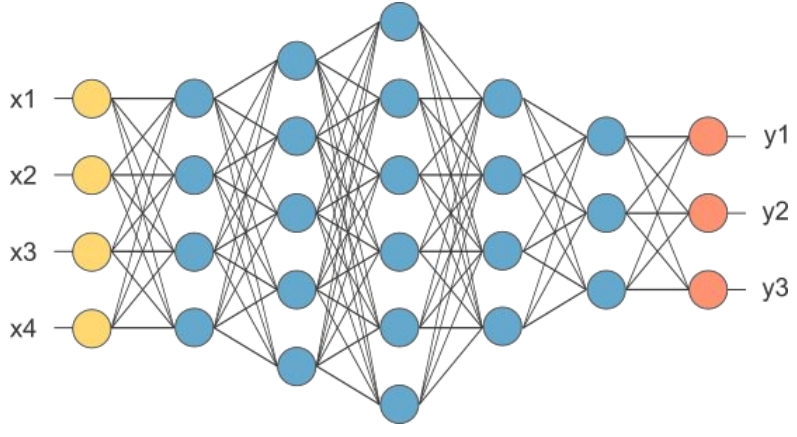
- As redes neurais tradicionais são ótimas para classificar.
- Elas recebem uma entrada de tamanho fixo emitem uma saída de tamanho fixo.
- Cada entrada é analisada separadamente, não levando em consideração as entradas anteriores (contexto e tempo)



Redes Neurais Recorrentes

Introdução

- Uma característica importante de todas as redes neurais até agora é que elas não têm memória. Cada entrada mostrada para eles é processada independentemente, sem estado mantido entre entradas.



Redes Neurais Recorrentes

Introdução

- Embora as representações derivadas de redes convolucionais sejam uma melhoria pois oferecem alguma sensibilidade à ordem das palavras, sua sensibilidade à ordem é restrita a padrões locais, e desconsidera a ordem dos padrões que estão distantes na sequência.

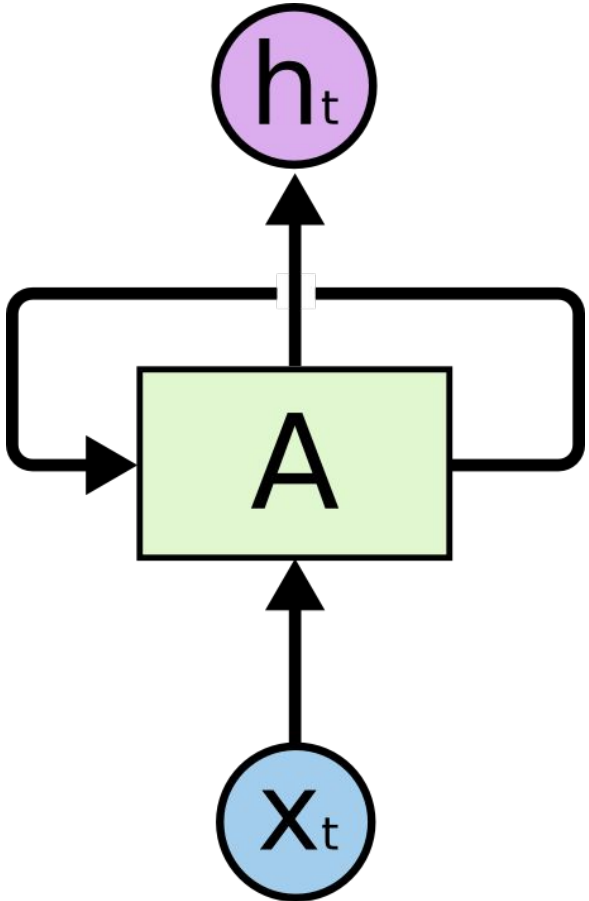
Redes Neurais Recorrentes

Introdução

- Elas não são adequadas para analisar sequências longas, tais como sentenças em linguagem natural.
- Para esses casos deve-se usar as Redes neurais recorrentes (RNN)

Redes Neurais Recorrentes

RNN



Redes Neurais Recorrentes

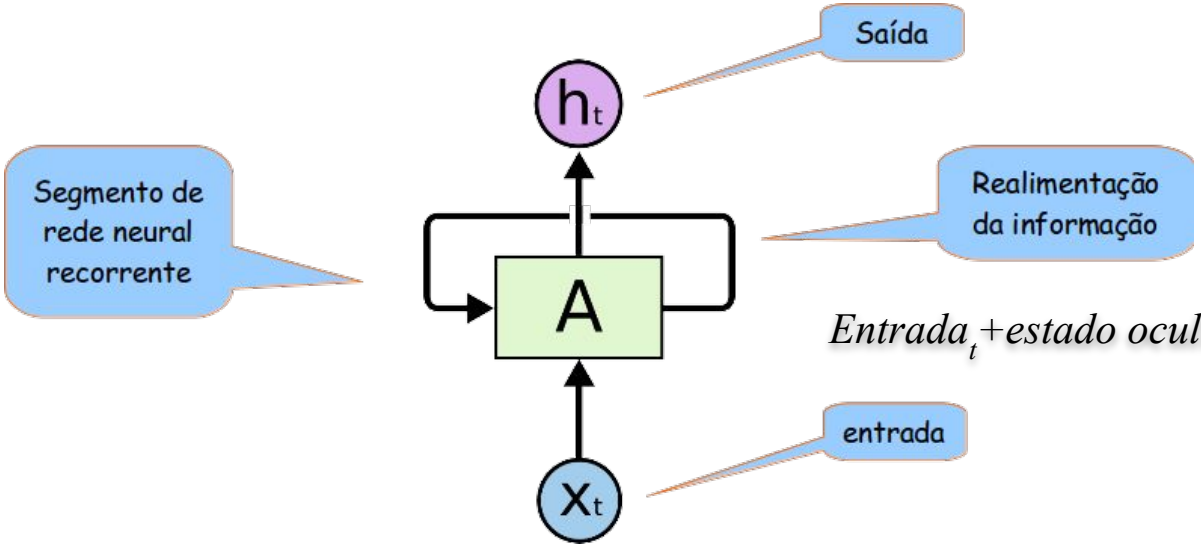
RNN

- Redes Neurais Recorrentes (RNNs) [Elman, 1990] permitem representar entradas sequenciais arbitrariamente dimensionadas em vetores de tamanho fixo, enquanto retém as propriedades estruturais das entradas.

Redes Neurais Recorrentes

RNN

- É necessário o uso de redes que possam "lembrar" o que aconteceu anteriormente: contexto.
- As redes neurais recorrentes (RNN) procuram resolver esse problema.

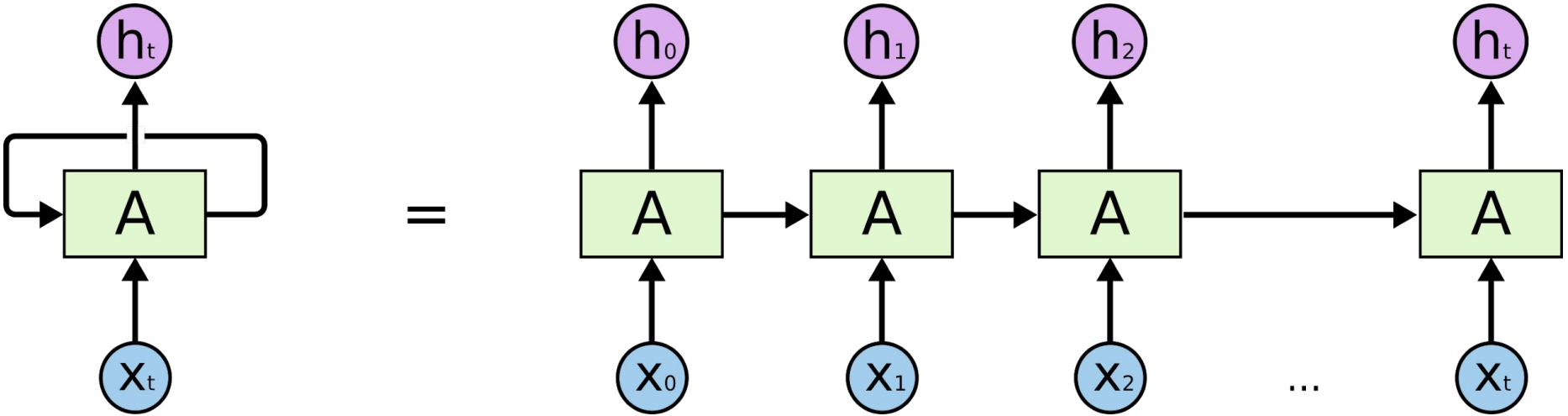


$$Entrada_t + estado\ oculto_{t-1} \rightarrow estado\ oculto_t \rightarrow saída$$

Redes Neurais Recorrentes

RNN

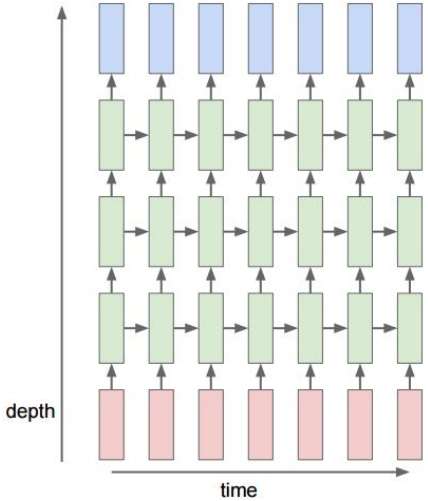
- Se considerarmos a realimentação como uma entrada em uma nova rede no tempo t+1, fica mais fácil visualizar o funcionamento da rede.
- Dessa forma fica clara a adequação da rede ao tratamento de listas e sequências.



Redes Neurais Recorrentes

RNN

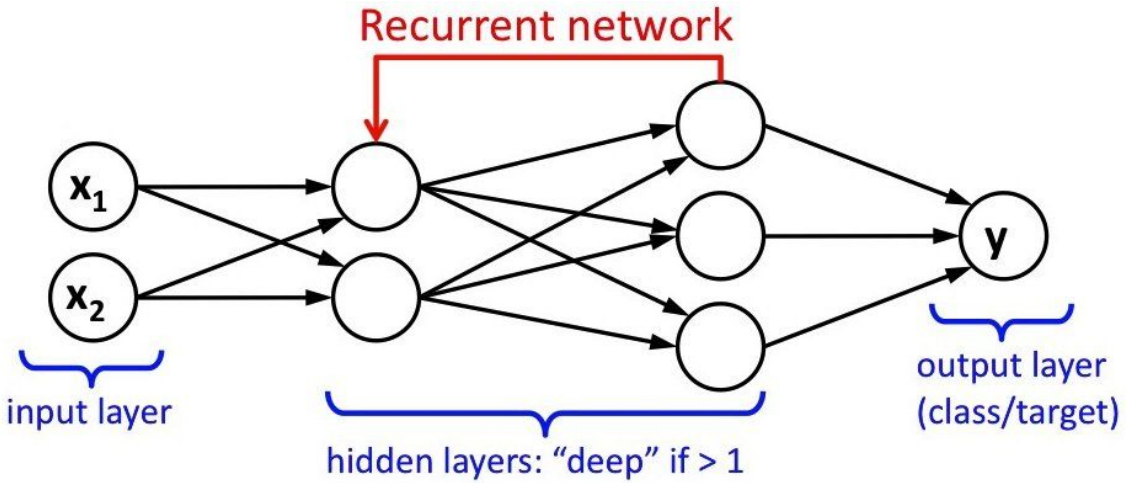
- Se considerarmos a realimentação como uma entrada em uma nova rede no tempo t+1, fica mais fácil visualizar o funcionamento da rede.
- Dessa forma fica clara a adequação da rede ao tratamento de listas e sequências.



Redes Neurais Recorrentes

RNN

- Redes Neurais Recorrentes é o melhor modelo para regressão, porque levam em consideração os valores passados.



Redes Neurais Recorrentes

RNN

- Fórmula geral

$$\boldsymbol{h}^{(t)} = f(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)}; \boldsymbol{\theta}),$$

Redes Neurais Recorrentes

RNN

Aplicações:

- Tradução automática (inglês -> francês)
- Fala para texto
- Previsão de mercado
- Rotulagem de cena (combinada com CNN)
- Direção da roda de carro. (Combinado com a CNN)

Redes Neurais Recorrentes

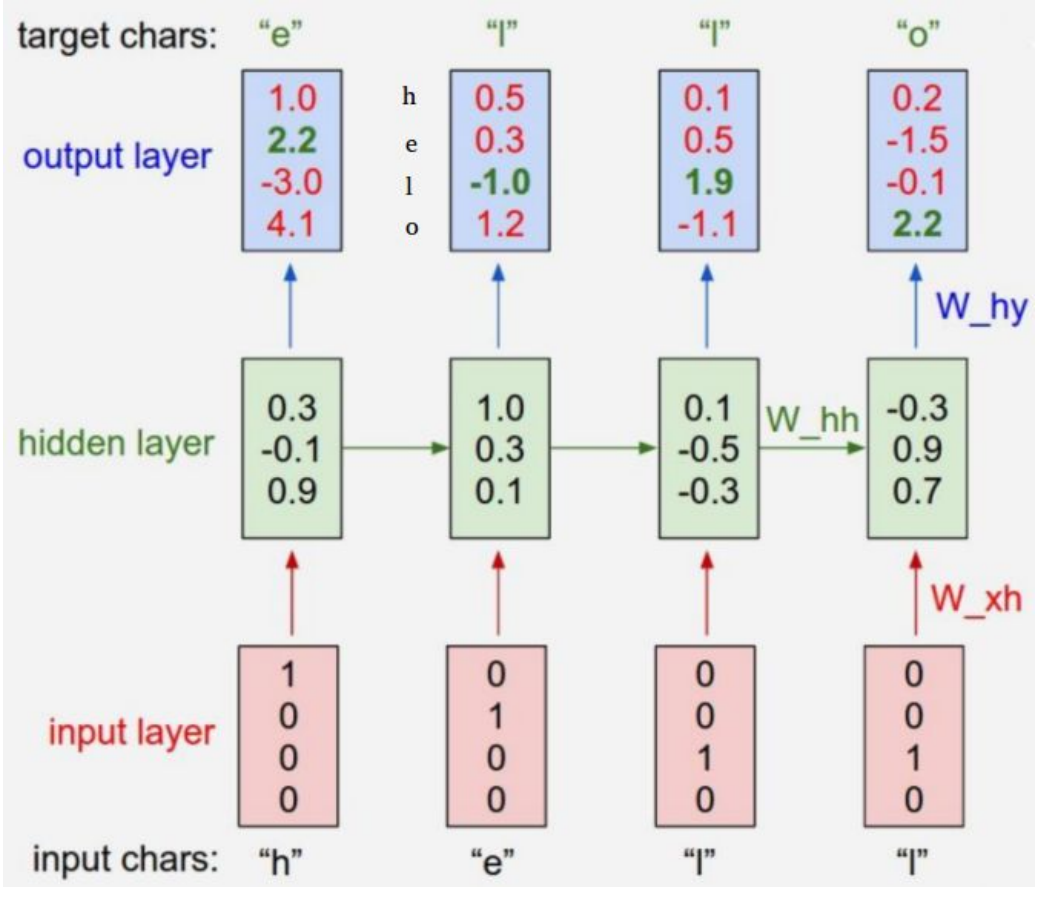
RNN

Exemplo

Deseja-se que a RNN complete a palavra. Dado o vocabulário [h, e, l, o] e a sequência de entrada h, e, l, l.

Observe que após o primeiro 'h', a rede quer dar a resposta errada (a correta está em verde).

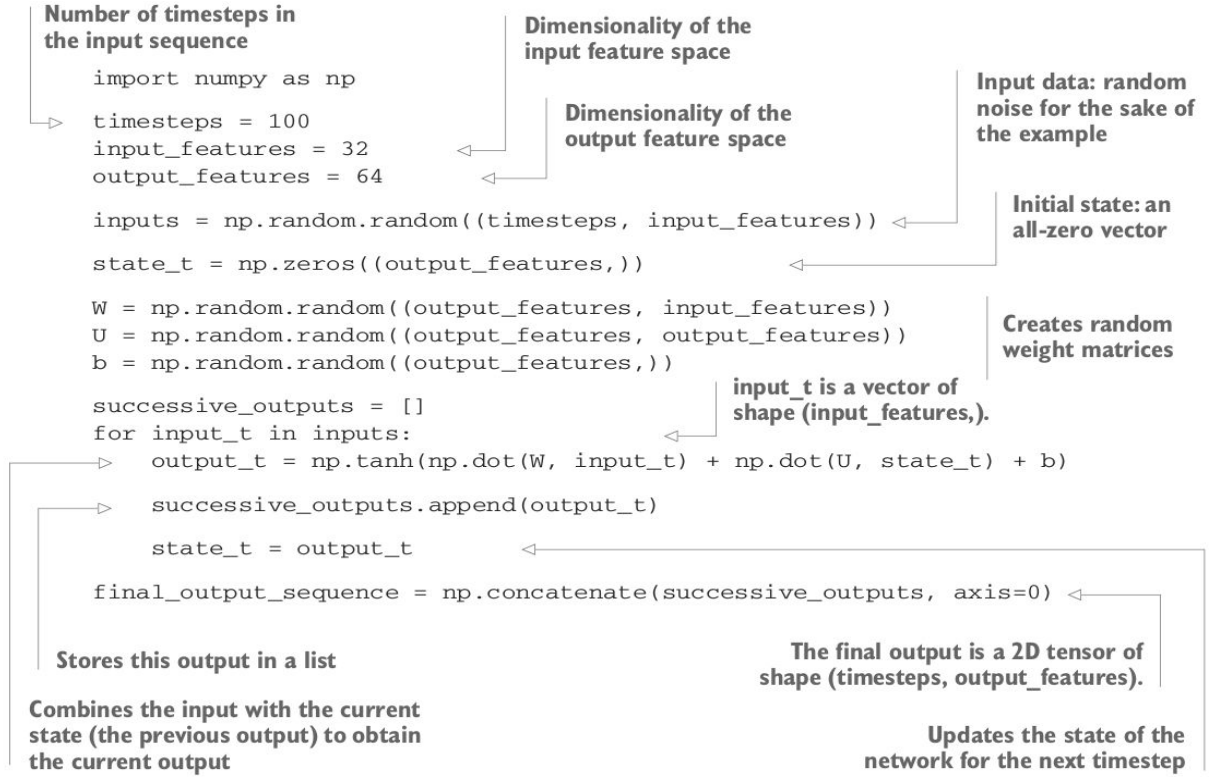
Porém, mas perto do fim, depois do segundo 'l' que a RNN quer dar a resposta correta 'o'. Aqui a ordem de entrada importa.



Redes Neurais Recorrentes

RNN

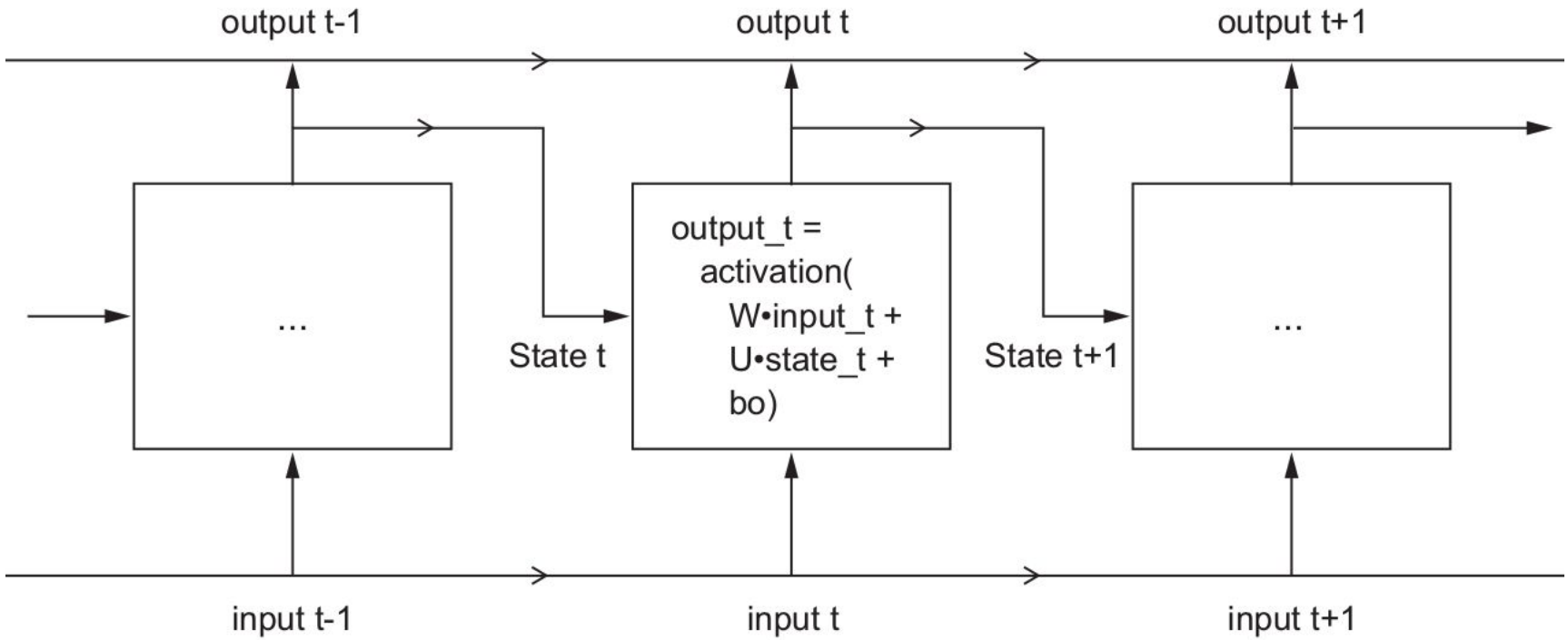
Exemplo simples de código em Numpy



Redes Neurais Recorrentes

RNN

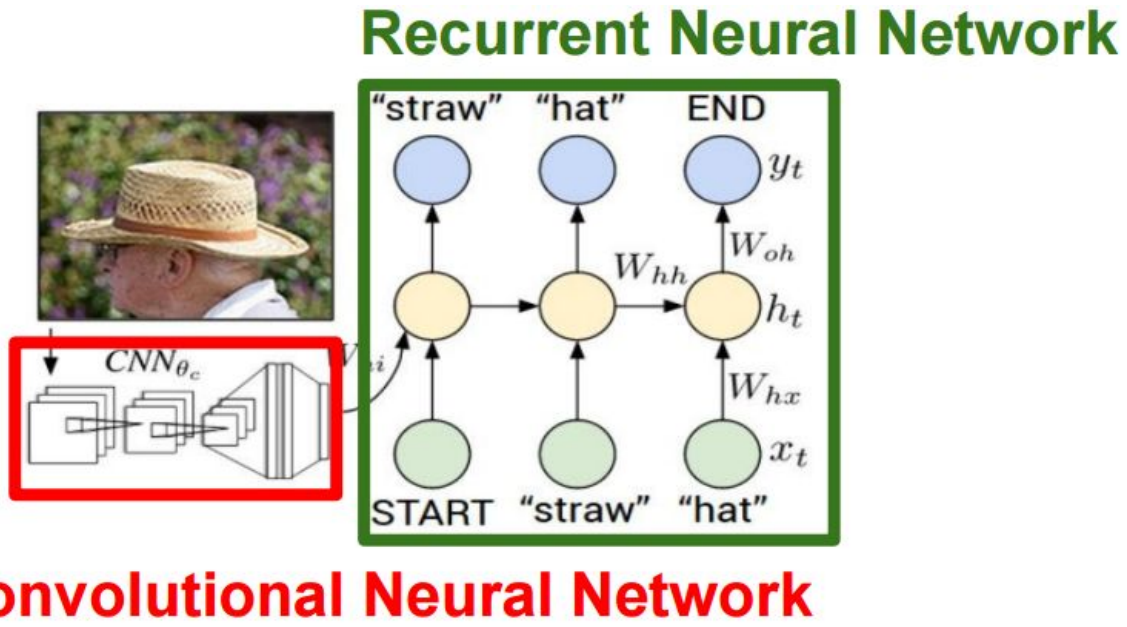
Exemplo simples de código em Numpy



Redes Neurais Recorrentes

RNN

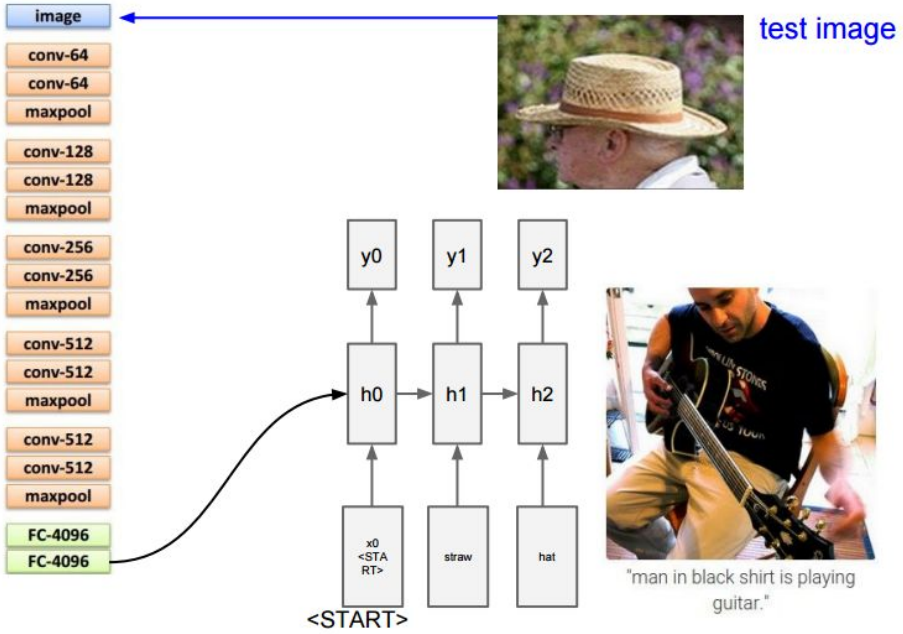
Se você conectar uma rede neural de convolução, com uma RNN pré-treinada, a RNN será capaz de descrever o que "vê" na imagem.



Redes Neurais Recorrentes

RNN

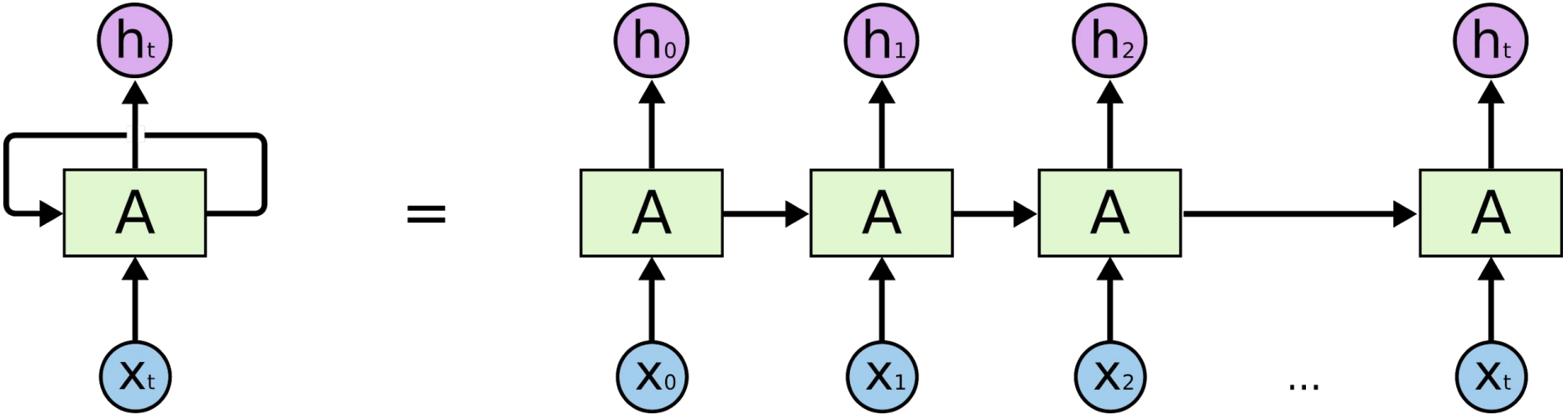
Basicamente, obtemos uma CNN pré-treinada (ou seja, VGG) e conectamos a última camada FC a uma RNN. Depois disso, tudo é treinado novamente.



Redes Neurais Recorrentes

RNN

- RNNs são ótimas para lidar com dependências de curta e média distância. No entanto, quando é preciso lidar com dependências distantes, as RNNs tradicionais não conseguem memorizar.
- Ela também sofre do problema de dissipação de gradiente (vanishing gradient)



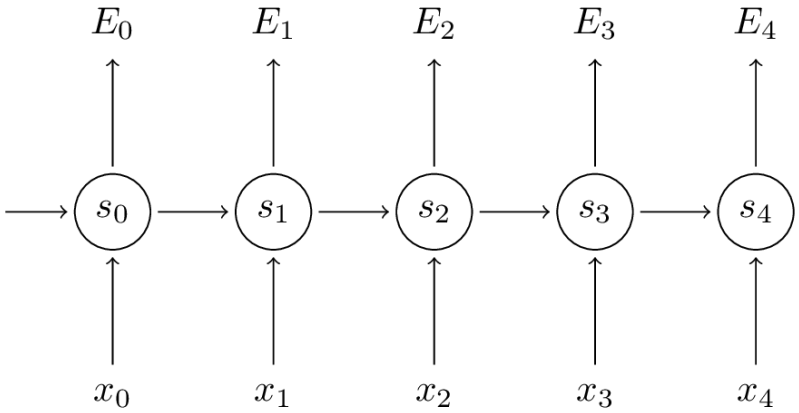
Redes Neurais Recorrentes

RNN

Problema de dissipação de gradiente (vanishing gradient)

- O erro total é a soma de cada erro no passo de tempo t

$$\frac{\partial E}{\partial W} = \sum_{t=1}^{T} \frac{\partial E_t}{\partial W}$$



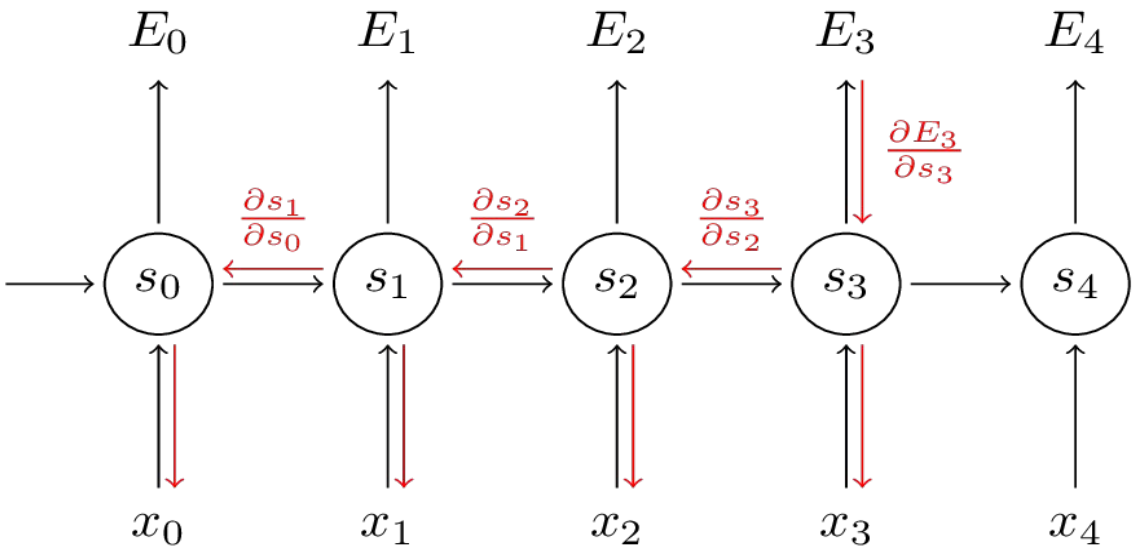
Redes Neurais Recorrentes

RNN

Problema de dissipação de gradiente (vanishing gradient)

- Aplicação da regra da cadeia

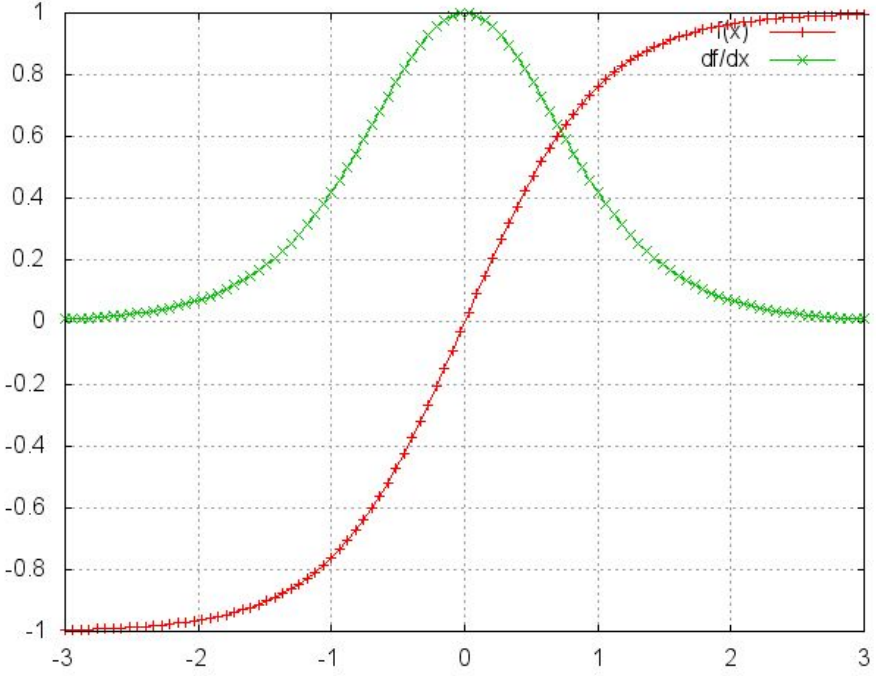
$$\frac{\partial E_3}{\partial V} = \sum_{k=0}^{3} \frac{\partial E_3}{\partial \hat{y}_3} \frac{\partial \hat{y}_3}{\partial s_3} \frac{\partial s_3}{\partial s_k} \frac{\partial s_k}{\partial s_W}$$



Redes Neurais Recorrentes

RNN

Problema de dissipação de gradiente (vanishing gradient)



Redes Neurais Recorrentes

RNN

Problema de dissipação de gradiente (vanishing gradient)

- No caso de modelagem de linguagem ou sistema de pergunta e resposta, as etapas de tempo longe não são levados em consideração quando for prever a próxima palavra

Exemplo:

Jane walked into the room. John walked in too. It was late in the day. Jane said hi to _____

Redes Neurais Recorrentes

RNN

Problema de dissipação de gradiente (vanishing gradient)

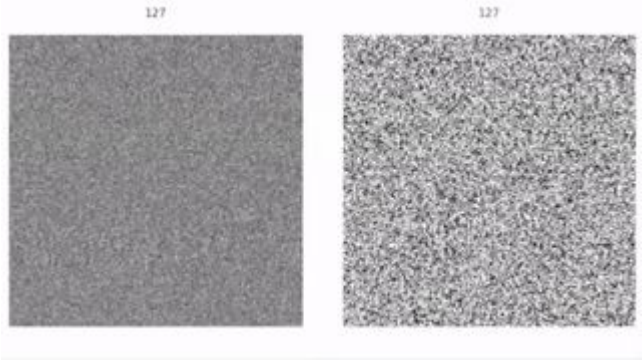
- Felizmente, existem algumas maneiras de combater o problema. A inicialização adequada da matriz W pode reduzir a dissipação de gradientes, ou funções de ativação ReLU em vez de tanh ou sigmóide.

- Uma solução ainda mais popular é usar as arquiteturas Long Short-Term Memory (LSTM) ou Gated Recurrent Unit (GRU).

Redes Neurais Recorrentes

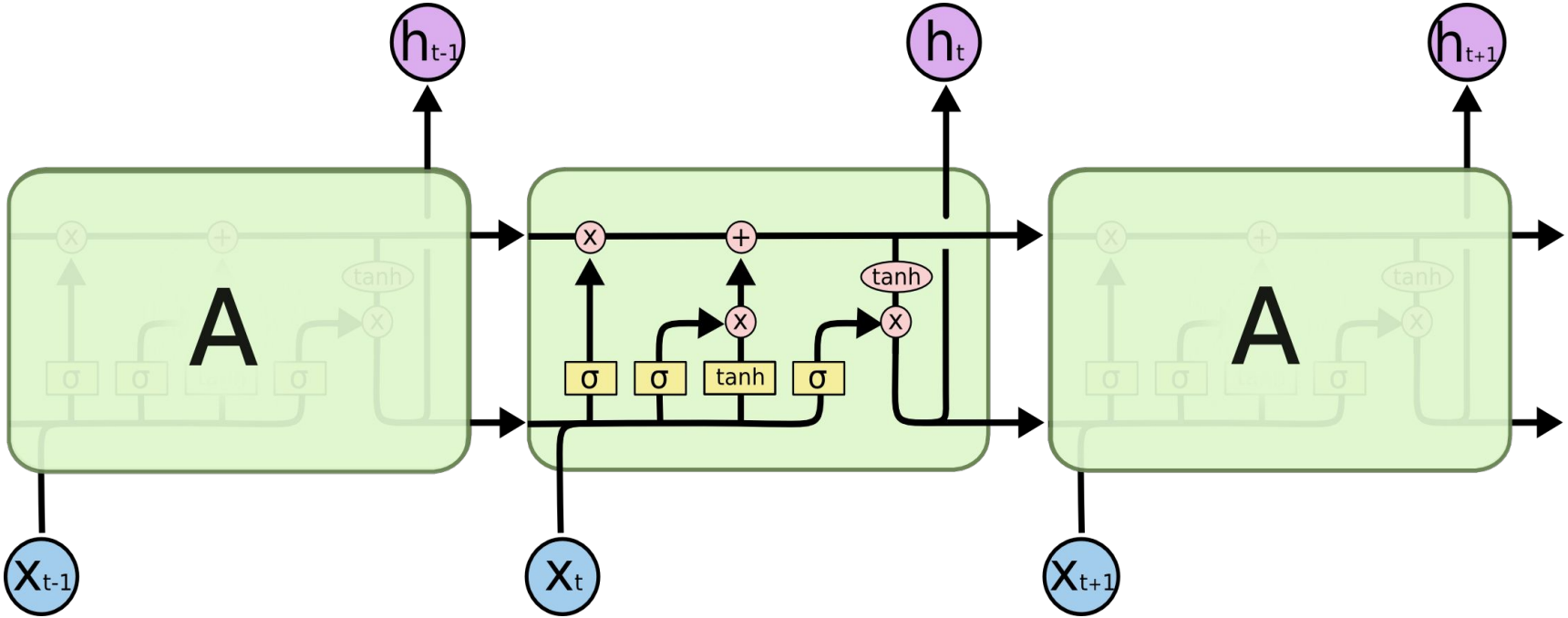
LSTM

Em meados dos anos 90, os pesquisadores alemães Sepp Hochreiter e Juergen Schmidhuber apresentaram uma variação da rede recorrente denominada de Long Short Term Term Memory (LSTM), como uma solução para o problema da dissipação de gradiente.



Redes Neurais Recorrentes

LSTM



Redes Neurais Recorrentes

LSTM

- LSTM é uma arquitetura de rede neural recorrente (RNN) que lembra valores em intervalos arbitrários. Os valores armazenados não são modificados à medida que o aprendizado avança.

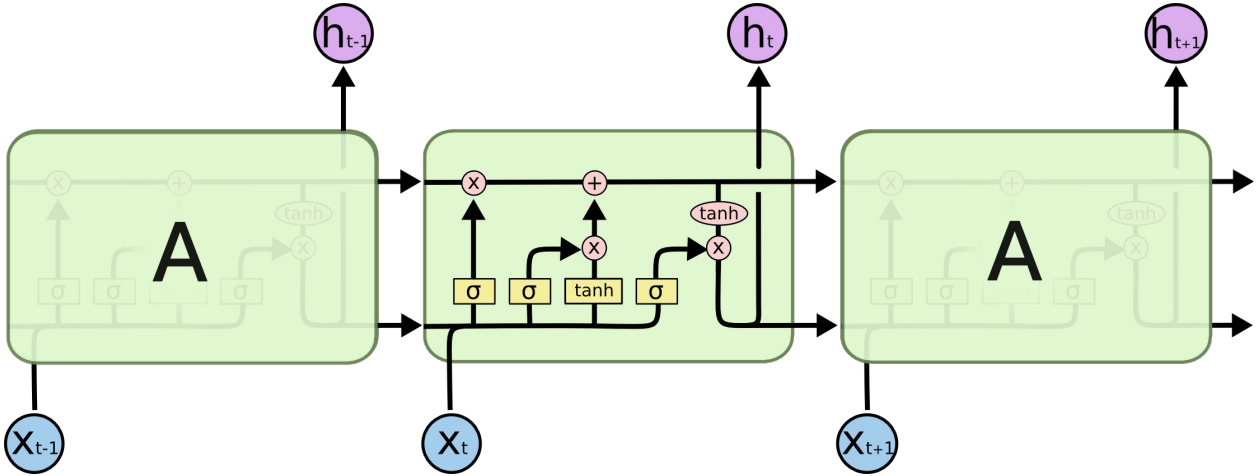
- LSTM é adequada para classificar, processar e prever séries temporais com atrasos temporários de tamanho e duração desconhecidos entre eventos importantes.

- A insensibilidade relativa ao comprimento do intervalo proporciona uma vantagem ao LSTM em relação a RNNs alternativas, modelos de Markov escondidos e outros métodos de aprendizagem de sequências.

Redes Neurais Recorrentes

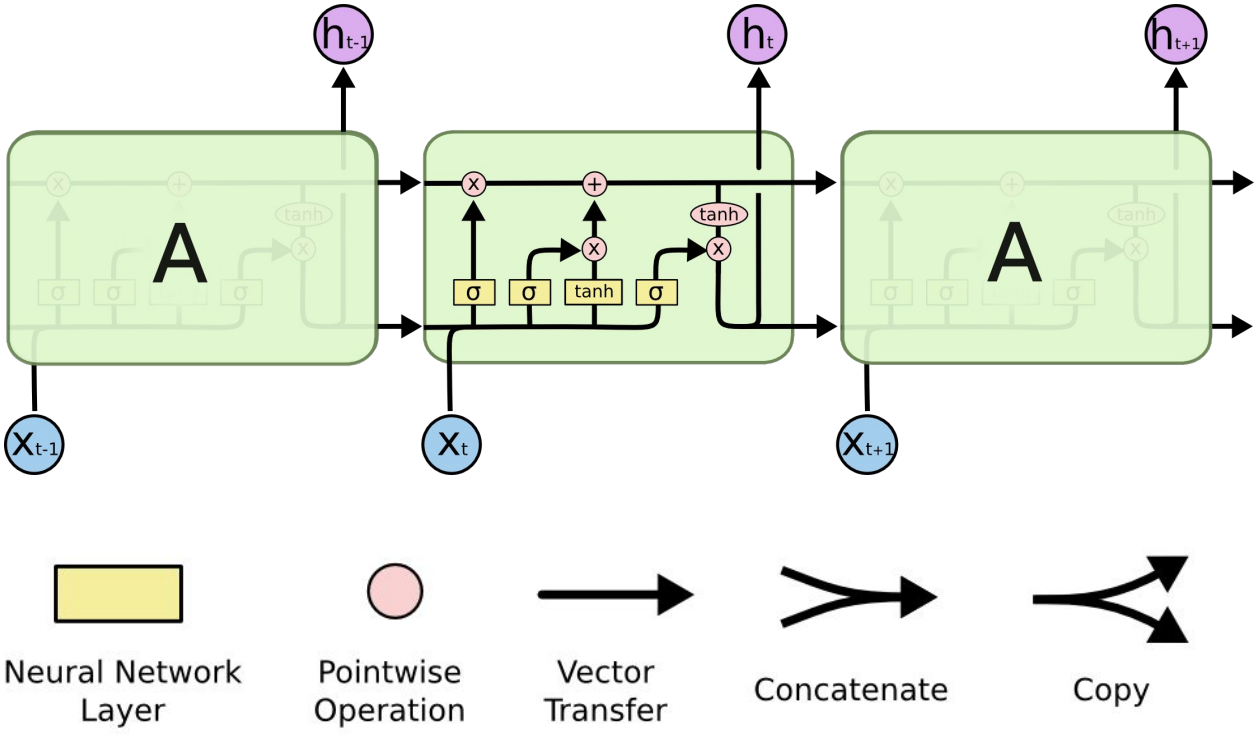
LSTM

Os LSTMs também possuem a estrutura de cadeia, mas o módulo de repetição tem uma estrutura diferente. Em vez de ter uma única camada de rede neural, existem quatro, interagindo de forma específica.



Redes Neurais Recorrentes

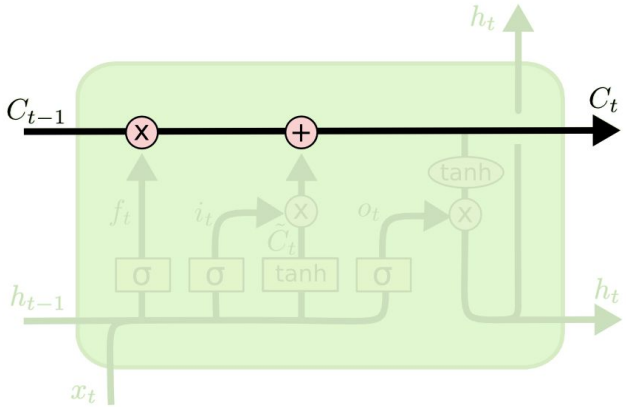
LSTM



Redes Neurais Recorrentes

LSTM

- A chave para LSTMs é o estado da célula C_t, a linha horizontal que passa pela parte superior do diagrama.
- O estado da célula é como uma correia transportadora. Ele corre diretamente pela cadeia inteira, com apenas algumas pequenas interações lineares. É possível que a informação apenas flua ao longo dela inalterada.

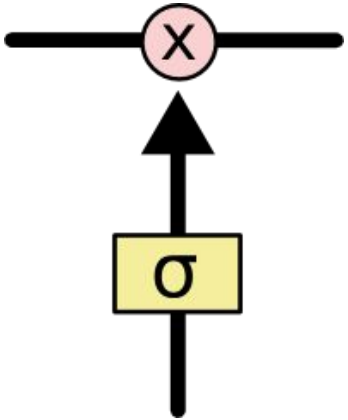


Redes Neurais Recorrentes

LSTM

- LSTM tem a capacidade de remover ou adicionar informações ao estado da célula, cuidadosamente reguladas por estruturas chamadas portas (gates).

- Gates é uma forma de inserir opcionalmente informações. São compostas de uma camada de rede neural sigmóide e uma operação de composição de funções.

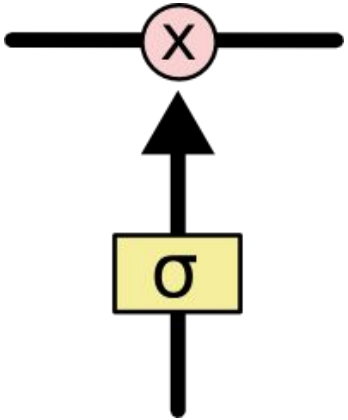


Redes Neurais Recorrentes

LSTM

- A camada sigmóide produz números entre zero e um, descrevendo quanto de cada componente deve passar. Um valor de zero significa "deixar nada", enquanto um valor de um significa "passar tudo".

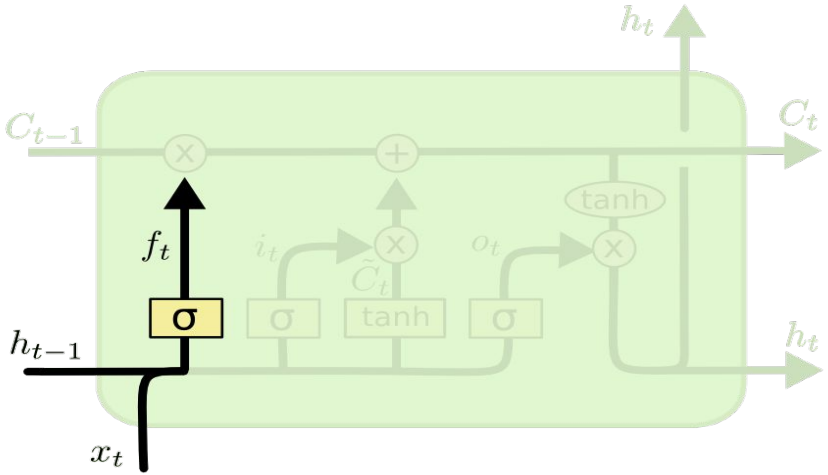
- LSTM tem três desses gates, para proteger e controlar o estado da célula.



Redes Neurais Recorrentes

LSTM

- O primeiro passo é decidir quais as informações eliminar do estado celular.
- Esta decisão é feita por uma camada sigmóide chamada "camada de porta de esquecimento".
- Ela examina h_{t-1} e x_t e emite um número entre 0 e 1 para cada número no estado celular C_{t-1}. 1 representando "manter completamente" enquanto um 0 representa "elimine completamente".



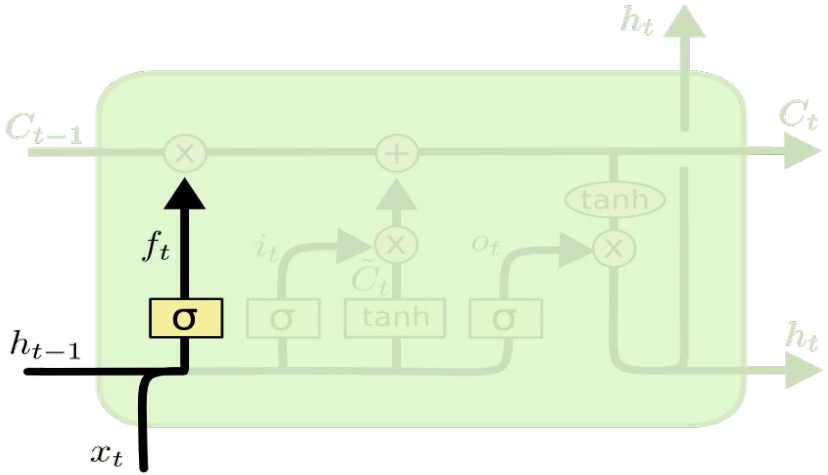
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Redes Neurais Recorrentes

LSTM

- Suponha o exemplo de um modelo de linguagem tentando prever a próxima palavra com base em todas as anteriores.

- Em tal problema, o estado celular pode incluir o gênero do assunto atual, para que os pronomes corretos possam ser usados. Quando vemos um novo assunto, queremos esquecer o gênero do assunto antigo.



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

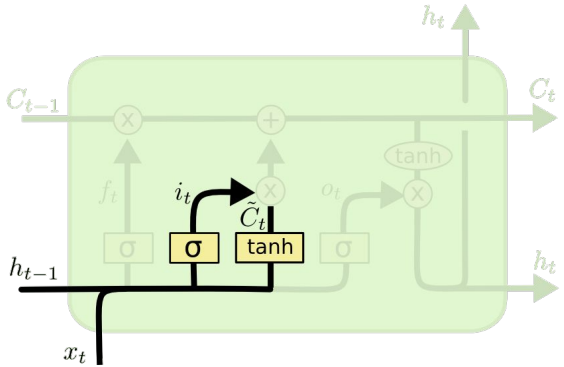
Redes Neurais Recorrentes

LSTM

- O próximo passo é decidir quais novas informações armazenar no estado celular.

- Isso tem duas partes. Primeiro, uma camada sigmóide chamada "camada de porta de entrada" decide quais valores vamos atualizar. Em seguida, uma camada *tanh* cria um vetor de novos valores candidatos, C_t, que podem ser adicionados ao estado.

- No próximo passo, estes são combinados para criar uma atualização para o estado.



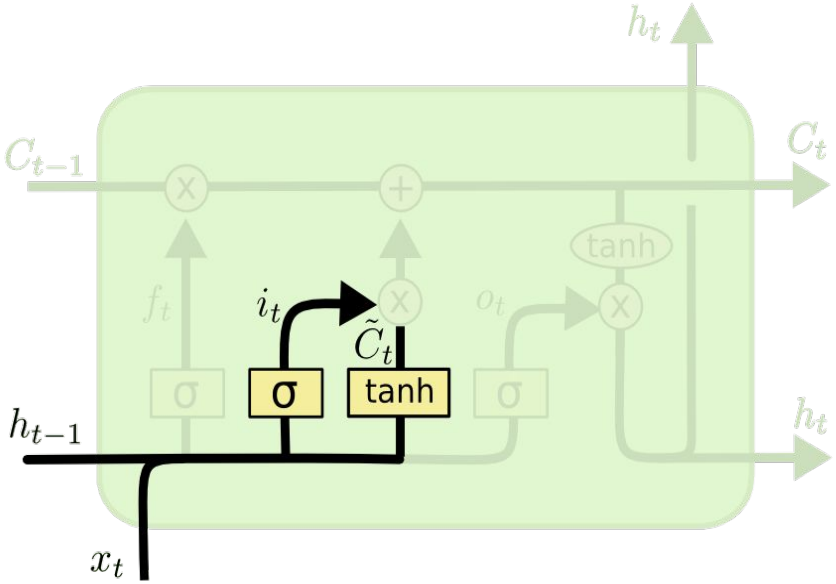
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

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

Redes Neurais Recorrentes

LSTM

- No exemplo do modelo de linguagem, pode ser necessário adicionar o gênero do novo assunto ao estado da célula, para substituir o antigo.



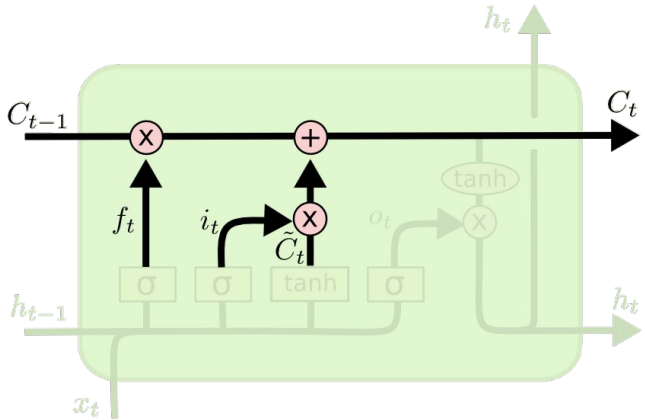
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

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

Redes Neurais Recorrentes

LSTM

- O próximo passo é atualizar o estado da célula antiga, C_{t-1}, para o novo estado C_t, conforme definido nas etapas anteriores.
- É multiplicado o estado anterior por f_t, apagando o que foi decidido anteriormente. Então, é adicionado $\tilde{C}_t$. Estes são os novos valores candidatos, na proporção que foi decidida para a atualização de cada valor do estado.

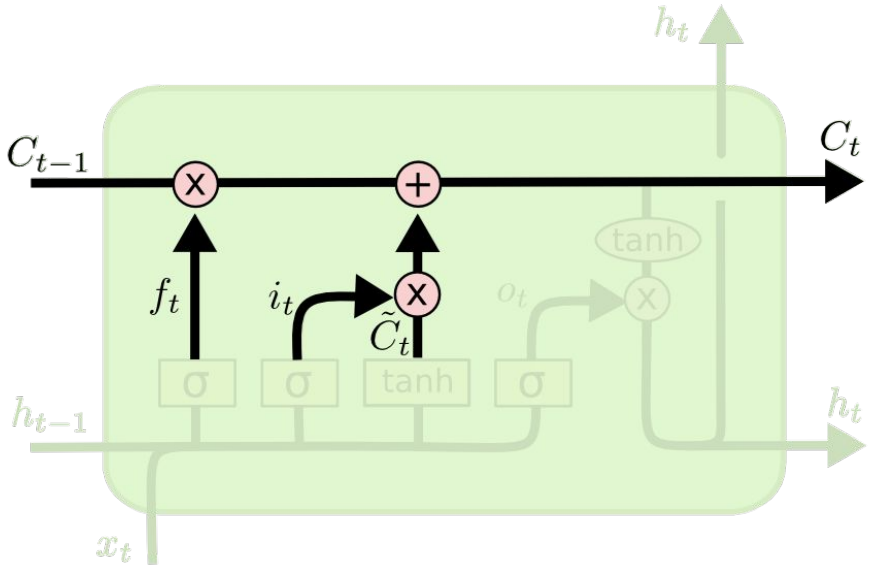


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Redes Neurais Recorrentes

LSTM

- No exemplo do modelo de linguagem, é aqui que é, realmente, eliminada a informação sobre o gênero do sujeito e adicionada as novas informações, como foi decidido nas etapas anteriores.

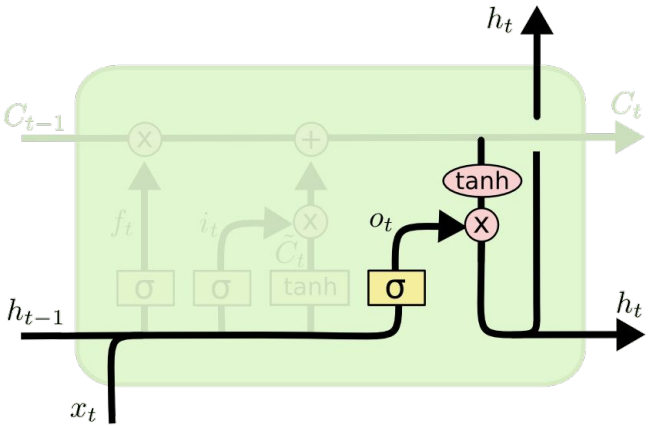


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Redes Neurais Recorrentes

LSTM

- Finalmente, é decidido o que será emitido. Esta saída será baseada no estado da célula, mas será uma versão filtrada.
- É executada uma camada sigmóide que decide quais partes do estado é emitida. Em seguida, é passado o estado através do tanh (para forçar os valores para ficar entre -1 e 1) e multiplicado pela saída da porta sigmóide, de modo a emitir as partes decididas.



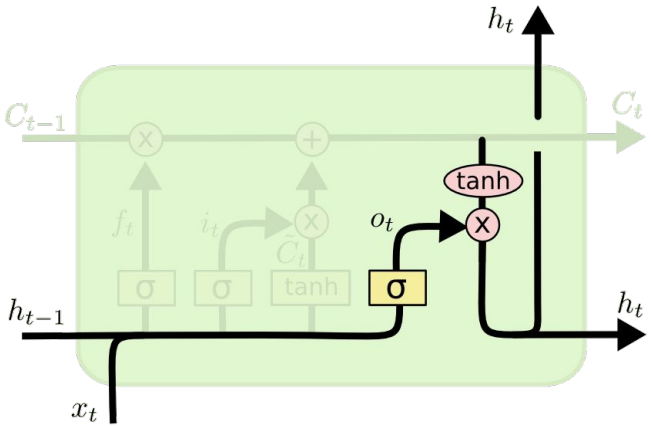
$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

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

Redes Neurais Recorrentes

LSTM

- No exemplo do modelo de linguagem, uma vez que acabou de ver um sujeito, pode querer exibir informações relevantes para um verbo, caso seja o que vem depois.
- Por exemplo, pode produzir se o sujeito é singular ou plural, para que possamos saber em que forma um verbo deve ser conjugado se for o que se segue.

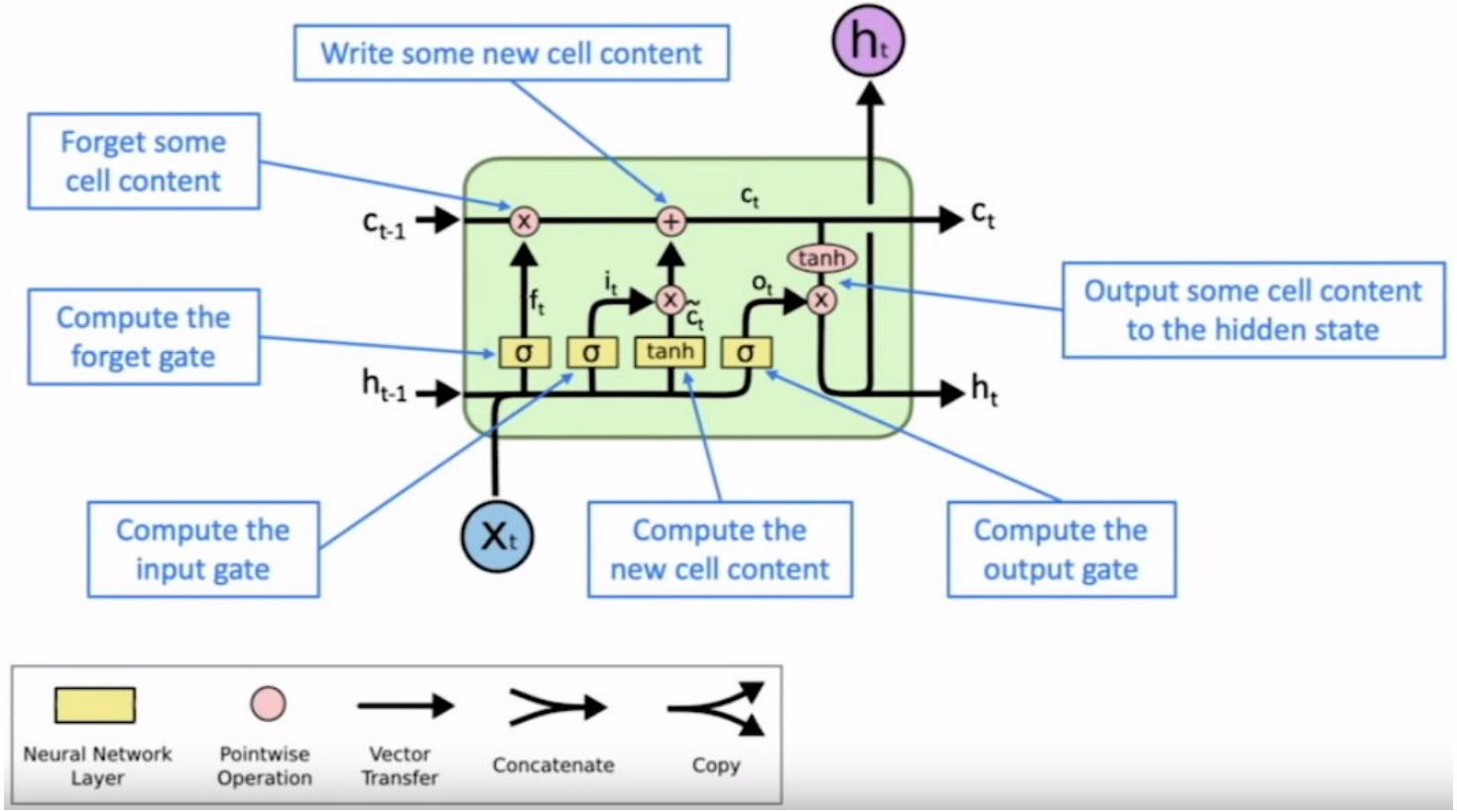


$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

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

Redes Neurais Recorrentes

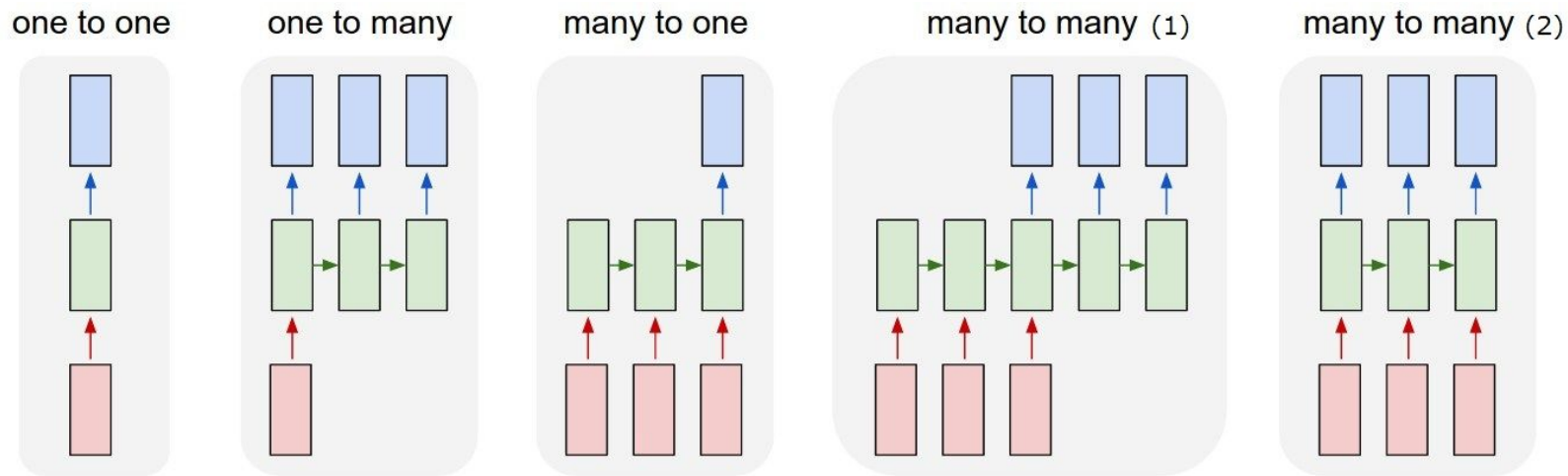
LSTM



Redes Neurais Recorrentes

LSTM

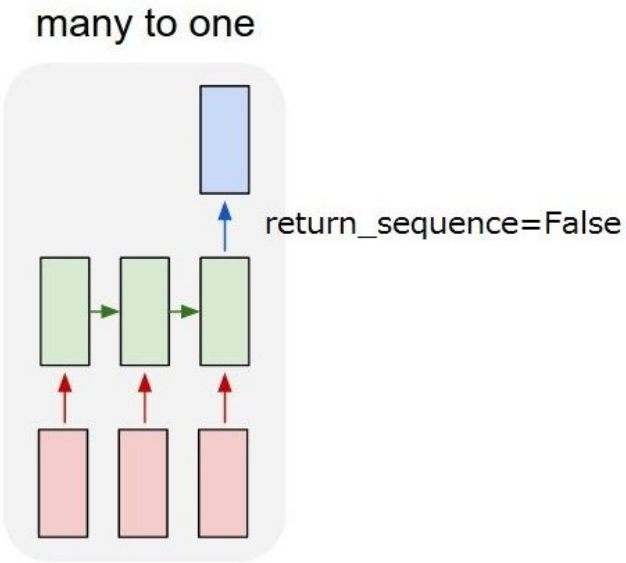
- Arquiteturas

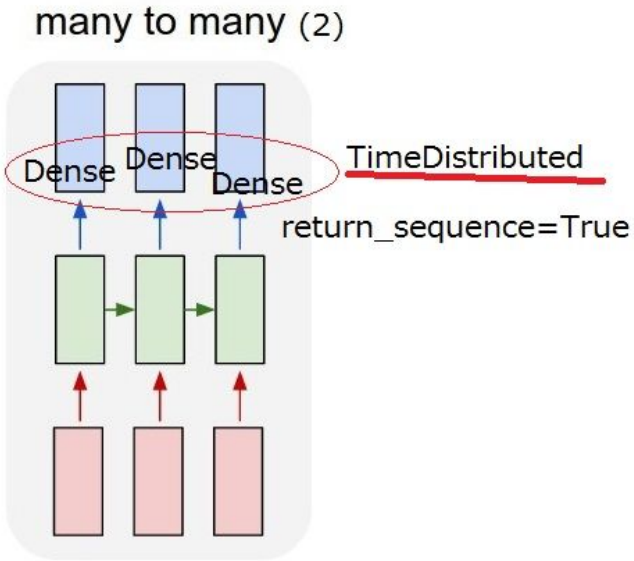


Redes Neurais Recorrentes

LSTM

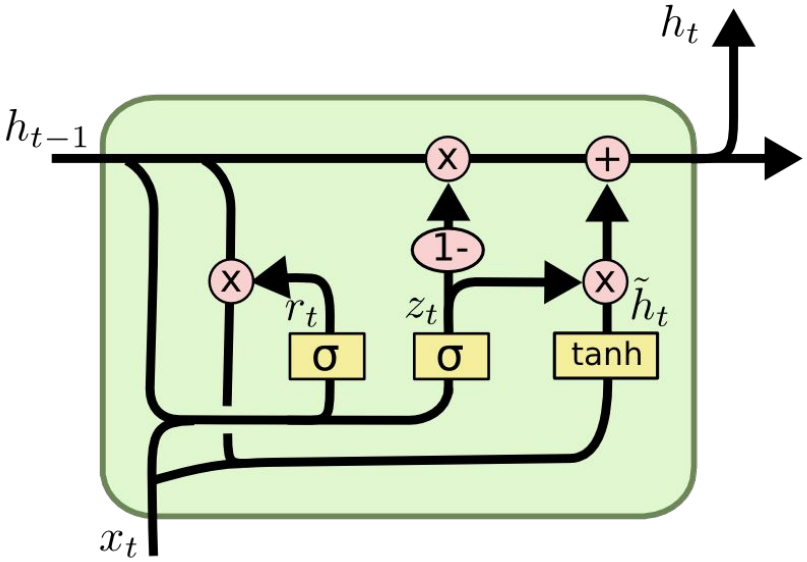
- Arquiteturas





Redes Neurais Recorrentes

GRU



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

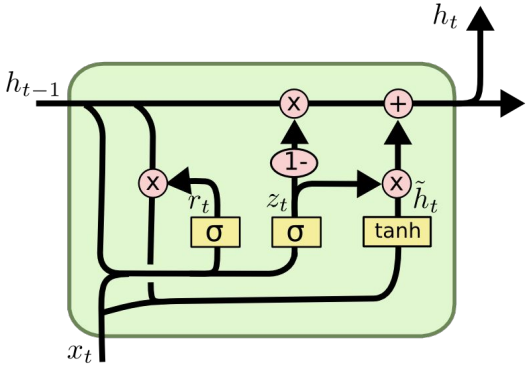
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Redes Neurais Recorrentes

GRU

- Uma variação da LSTM é a Gated Recurrent Unit, ou GRU, introduzida por Cho et al. (2014). Ele combina as portas de esquecimento e de entrada em uma única "porta de atualização". Ele também mescla o estado da célula e o estado oculto, e faz algumas outras mudanças.

- O modelo resultante é mais simples do que os modelos LSTM padrão e tem se tornado cada vez mais popular.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Redes Neurais Recorrentes

Links interessantes

https://deeplearning4j.org/lstm.html

https://www.depends-on-the-definition.com/guide-sequence-tagging-neural-networks-python/

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

https://medium.com/towards-data-science/lstm-by-example-using-tensorflow-feb0c1968537

http://web.stanford.edu/class/cs224n/

http://www.renom.jp/notebooks/tutorial/basic_algorithm/LSTM/notebook.html

http://ufldl.stanford.edu/tutorial/

http://www.wildml.com/2015/10/recurrent-neural-networks-tutorial-part-3-backpropagation-through-time-and-vanishing-gradients/

https://codeburst.io/recurrent-neural-network-4ca9fd4f242

Redes Neurais Recorrentes

Livros

GOLDBERG Y. *Neural Network Methods for Natural Language Processing*. Morgan & Claypool ed. 2017.

GULLI A., PAL S. *Deep Learning with Keras*. Packt Publishing. 2017.

AURÉLIEN G. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc. 2017.