



Universidade de Brasília

Departamento de Ciência da Computação



Redes Neurais Recorrentes

Curso ENAP - Processamento de Linguagem Natural

Prof. Dr. Vinícius Ruela Pereira Borges

`viniciusrpb@unb.br`

Brasília-DF, 2024

- Esses slides foram redigidos e produzidos pelo Prof. Dr. Vinícius R. P. Borges;
- Material didático de referência:
 - Capítulo 9 do livro “Speech and Language Processing. Daniel Jurafsky & James H. Martin, 2021.”
 - Slides do curso “CS224n: Natural Language Processing with Deep Learning” - Stanford University

- Motivação

- Motivação
- Definição

- Motivação
- Definição
- Modelos de Linguagem

- Motivação
- Definição
- Modelos de Linguagem
- Redes Neurais Recorrentes de Elman

- Motivação
- Definição
- Modelos de Linguagem
- Redes Neurais Recorrentes de Elman
- Implementação

- Motivação
- Definição
- Redes Neurais Recorrentes de Elman
- Modelos de Linguagem
- Implementação
- Considerações Finais

Motivação



- Processamento de dados sequenciais;



¹Fonte: <https://www.bussoladoinvestidor.com.br/tudo-sobre-graficos-de-analise-tecnica/>

● Processamento de dados sequenciais;

Data de Referência: 05/04/2024 - 05/04/2024
Estação: BRASILIA A001
*Dados disponíveis em tempo real (sem controle de qualidade).

Banco CSV

Data	Hora	Temperatura (°C)			Umidade (%)			Pto. Orvalho (°C)			Pressão (hPa)			Vento			Radiação KJ/m²	Chuva mm
		UTC	Inst.	Máx.	Mín.	Inst.	Máx.	Mín.	Inst.	Máx.	Mín.	Inst.	Máx.	Mín.	Vel. (m/s)	Dir. (°)		
05/04/2024	0000		21.8	22.1	21.2	91.0	92.0	89.0	20.2	20.3	19.8	888.5	888.5	887.7	1.2	90.0	3.3	0.0
05/04/2024	0100		21.6	21.9	21.2	93.0	94.0	90.0	20.4	20.4	19.9	888.7	888.7	888.5	1.4	91.0	3.1	0.0
05/04/2024	0200		21.2	21.7	21.0	96.0	96.0	93.0	20.4	20.5	20.1	888.6	888.7	888.4	0.5	138.0	2.8	0.0
05/04/2024	0300		20.6	21.3	20.6	96.0	97.0	95.0	19.9	20.6	19.8	888.4	888.6	888.4	0.4	269.0	1.9	0.0
05/04/2024	0400		20.4	20.6	20.0	97.0	97.0	96.0	20.0	20.0	19.5	888.2	888.4	888.1	1.0	151.0	1.8	0.0
05/04/2024	0500		20.4	20.5	20.0	98.0	98.0	97.0	20.0	20.1	19.5	887.7	888.2	887.7	0.7	65.0	1.8	0.0
05/04/2024	0600		20.7	20.9	20.4	97.0	98.0	97.0	20.2	20.5	20.0	887.0	887.7	886.9	1.0	129.0	2.7	0.0
05/04/2024	0700		19.5	20.7	19.5	97.0	97.0	97.0	19.1	20.2	19.1	886.7	887.1	886.7	0.4	136.0	2.1	0.0
05/04/2024	0800		20.3	20.3	19.4	98.0	98.0	97.0	20.0	20.0	19.0	886.7	886.8	886.7	1.2	71.0	2.8	0.0
05/04/2024	0900		20.0	20.5	20.0	97.0	98.0	97.0	19.6	20.1	19.6	887.1	887.1	886.7	1.5	68.0	3.7	0.0
05/04/2024	1000		20.5	20.5	19.9	97.0	97.0	97.0	19.9	19.9	19.4	887.6	887.6	887.1	0.6	190.0	3.1	78.00
05/04/2024	1100		21.2	22.0	20.5	86.0	87.0	85.0	18.9	20.2	18.9	888.6	888.6	887.6	2.2	98.0	3.9	529.00
05/04/2024	1200		21.9	22.0	21.2	82.0	88.0	82.0	18.8	19.3	18.6	889.2	889.2	888.6	2.4	84.0	4.0	746.40

2

²Fonte: <https://portal.inmet.gov.br>

- As redes neurais estudadas até o momento não capturam relações de dependência em dados temporais;
 - em uma MLP, as entradas são independentes entre si.

- As redes neurais estudadas até o momento não capturam relações de dependência em dados temporais;
 - em uma MLP, as entradas são independentes entre si.
- Ademais, necessitam que todas as entradas possuam a mesma dimensão;
 - vide o vetor TF-IDF;

- As redes neurais estudadas até o momento não capturam relações de dependência em dados temporais
 - em uma MLP, as entradas são independentes entre si.
- Ademais, necessitam que todas as entradas possuam a mesma dimensão
 - vide o vetor TF-IDF;
- Como resultado, a capacidade de capturar contexto em textos é limitada
 - entendimento do contexto requer a análise de várias palavras ou sentenças.

Definição



- Redes neurais recorrentes (recurrent neural network - RNN) são uma família de redes neurais para processar dados sequenciais;

- Redes neurais recorrentes (recurrent neural network - RNN) são uma família de redes neurais para processar dados sequenciais;
- Dentre os tipos dados sequenciais mais comuns, citam-se:
 - Séries temporais;

- Redes neurais recorrentes (recurrent neural network - RNN) são uma família de redes neurais para processar dados sequenciais;
- Dentre os tipos dados sequenciais mais comuns, citam-se:
 - Séries temporais;
 - Textos;

- Redes neurais recorrentes (recurrent neural network - RNN) são uma família de redes neurais para processar dados sequenciais;
- Dentre os tipos dados sequenciais mais comuns, citam-se:
 - Séries temporais;
 - Textos;
 - Áudio;

- Redes neurais recorrentes (recurrent neural network - RNN) são uma família de redes neurais para processar dados sequenciais;
- Dentre os tipos dados sequenciais mais comuns, citam-se:
 - Séries temporais;
 - Textos;
 - Áudio;
 - Sinais.

- Exemplo:

*A partida estava truncada, mas um gol heróico no final
livrou o Brasil da derrota.*

- Exemplo:

*A partida estava truncada, mas um gol heróico no final
livrou o Brasil da derrota.*

- A construção correta de uma frase depende da ordem lógica das palavras;

- Exemplo:

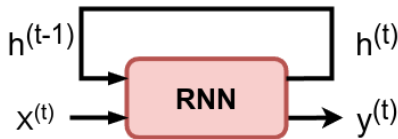
*A partida estava truncada, mas um gol heróico no final
livrou o Brasil da derrota.*

- A construção correta de uma frase depende da ordem lógica das palavras;
- Cada palavra é considerada como uma parte da entrada em um instante de tempo.

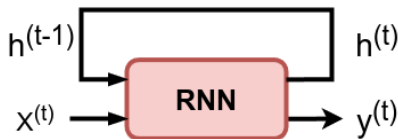
- Uma rede neural recorrente (recurrent neural network - RNN) é uma família de redes neurais para processar dados sequenciais;
- RNNs contemplam qualquer rede que contém um ciclo dentro das conexões de rede;

Definição

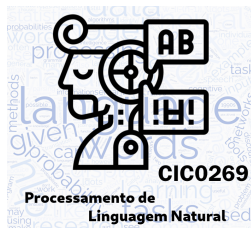
- Uma rede neural recorrente (recurrent neural network - RNN) é uma família de redes neurais para processar dados sequenciais;
- RNNs contemplam qualquer rede que contém um ciclo dentro das conexões de rede;



- Os valores dos pesos de um neurônio dependem diretamente (ou indiretamente) dos seus próprios valores de saída obtidos de um estado anterior.



RNN de Elman



- Matematicamente, as equações que definem uma RNN são descritas abaixo:

$$h^{(t)} = \sigma(W_h h^{(t-1)} + W_x x^{(t)}) \quad (1)$$

$$\hat{y}^{(t)} = \sigma(W_s h^{(t)}) \quad (2)$$

$$h^{(t)} = \sigma(W_h h^{(t-1)} + W_x x^{(t)}) \quad (3)$$

- $x^{(t)} \in \mathbb{R}^d$ é um vetor da palavra de entrada associada ao tempo t ;

$$h^{(t)} = \sigma(W_h h^{(t-1)} + W_x x^{(t)}) \quad (4)$$

- $x^{(t)} \in \mathbb{R}^d$ é um vetor da palavra de entrada associada ao tempo t ;
- $h^{(t-1)} \in \mathbb{R}^{D_h}$ é o vetor de saída da função não-linear no instante tempo $t - 1$;

$$h^{(t)} = \sigma(W_h h^{(t-1)} + W_x x^{(t)}) \quad (5)$$

- $W_x \in \mathbb{R}^{D_h \times d}$ é a matriz de pesos que balanceia o vetor $x^{(t)}$;

$$h^{(t)} = \sigma(W_h h^{(t-1)} + W_x x^{(t)}) \quad (6)$$

- $W_x \in \mathbb{R}^{D_h \times d}$ é a matriz de pesos que balanceia o vetor $x^{(t)}$;
- $W_h \in \mathbb{R}^{D_h \times D_h}$ é a matriz de pesos que balanceia a saída $h^{(t-1)}$ do tempo anterior;

$$h^{(t)} = \sigma(W_h h^{(t-1)} + W_x x^{(t)}) \quad (7)$$

- $W_x \in \mathbb{R}^{D_h \times d}$ é a matriz de pesos que balanceia o vetor $x^{(t)}$;
- $W_h \in \mathbb{R}^{D_h \times D_h}$ é a matriz de pesos que balanceia a saída $h^{(t-1)}$ do tempo anterior;
- $\sigma()$ é a função de ativação não-linear, no caso a sigmóide.

$$\hat{y}^{(t)} = \sigma(W_s h^{(t)}) \quad (8)$$

- $\hat{y}^{(t)}$ é um tipo de saída da rede, no caso acima, modulada pela função sigmóide.

$$\hat{y}^{(t)} = \sigma(W_s h^{(t)}) \quad (9)$$

- $\hat{y}^{(t)}$ é um tipo de saída da rede, no caso acima, modulada pela função sigmóide.
- poderia ser outro tipo de função de ativação: tangente hiperbólica (tanh), relu, softmax... depende da aplicação!

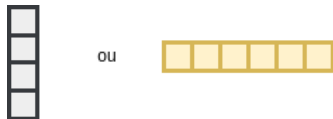
- Vetores com valores numéricos



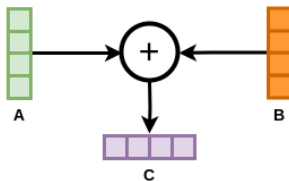
ou



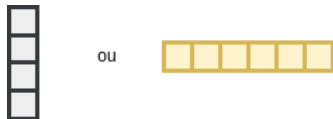
- Vetores com valores numéricos



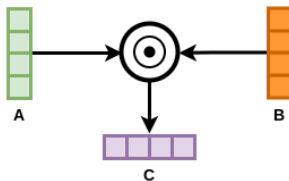
- Soma ponto-a-ponto: $c = a + b$



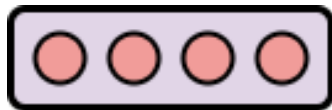
- Vetores com valores numéricos



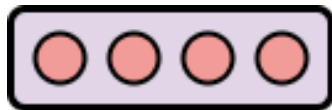
- Multiplicação ponto-a-ponto: $c = a \odot b$



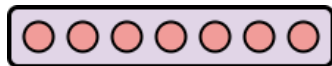
- Matriz de pesos contendo 4 neurônios (unidades internas):



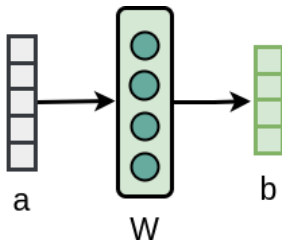
- Matriz de pesos contendo 4 neurônios (unidades internas):



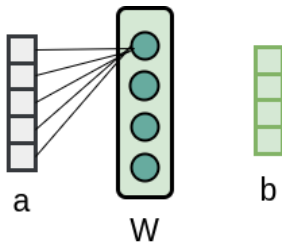
- Matriz de pesos contendo 7 neurônios (unidades internas):



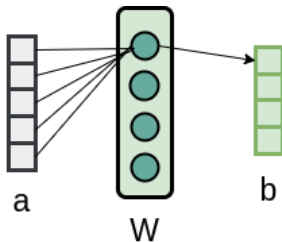
- Produto cruzado entre um vetor e a matriz de pesos



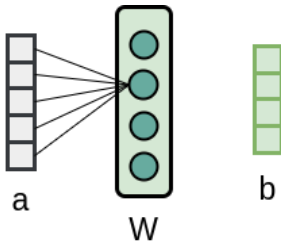
- Produto cruzado entre um vetor e a matriz de pesos
($a = Wb$):
- Exemplo:



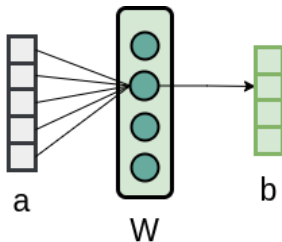
- Produto cruzado entre um vetor e a matriz de pesos
- Exemplo:



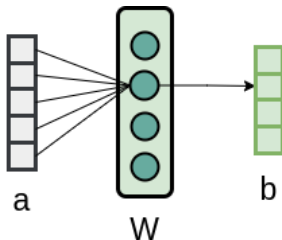
- Produto cruzado entre um vetor e a matriz de pesos
- Exemplo:



- Produto cruzado entre um vetor e a matriz de pesos
- Exemplo:



- Produto cruzado entre um vetor e a matriz de pesos



- e assim por diante...

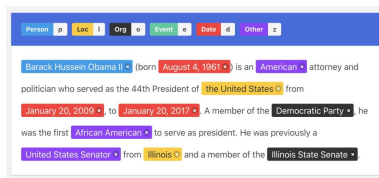
- Seria possível uma rede neural analisar uma sentença de textos de modo que...

- Seria possível uma rede neural analisar uma sentença de textos de modo que...
- o processamento de um texto ocorra palavra por palavra, respeitando sua ordem e

- Seria possível uma rede neural analisar uma sentença de textos de modo que...
- o processamento de um texto ocorra palavra por palavra, respeitando sua ordem e
- a rede consiga reter relações de dependência a curto e longo prazos?

- A principal tarefa de uma RNN em PLN é servir como um modelo de linguagem;

- A principal tarefa de uma RNN em PLN é servir como um modelo de linguagem;
- A partir de um modelo de linguagem, pode-se estender sua aplicação para outras tarefas:
 - Reconhecimento de entidades nomeadas;



- A principal tarefa de uma RNN em PLN é servir como um modelo de linguagem;
- A partir de um modelo de linguagem, pode-se estender sua aplicação para outras tarefas:
 - Reconhecimento de entidades nomeadas;
 - Classificação de textos;

- A principal tarefa de uma RNN em PLN é servir como um modelo de linguagem;
- A partir de um modelo de linguagem, pode-se estender sua aplicação para outras tarefas:
 - Reconhecimento de entidades nomeadas;
 - Classificação de textos;
 - Geração automática de textos.

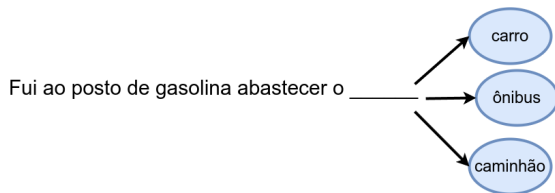
- A principal tarefa de uma RNN em PLN é servir como um modelo de linguagem;
- A partir de um modelo de linguagem, pode-se estender sua aplicação para outras tarefas:
 - Reconhecimento de entidades nomeadas;
 - Classificação de textos;
 - Geração automática de textos.
- Além disso, as RNNs servem como base para abordagens mais complexas como a Long Short-Term Memory (LSTM) e a Gated Recurrent Unit (GRU).

Modelos de Linguagem

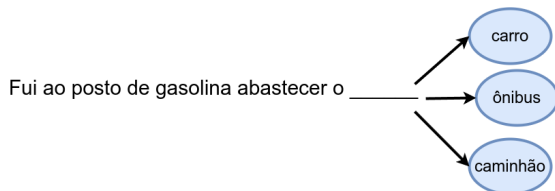


- Um modelo de linguagem possui a tarefa de prever a próxima palavra de um texto.

- Um modelo de linguagem possui a tarefa de prever a próxima palavra de um texto.



- Um modelo de linguagem possui a tarefa de prever a próxima palavra de um texto.



- Essa próxima palavra pode ser qualquer uma no vocabulário do *corpus*.

- Dada uma sequência de palavras, calcule a distribuição de probabilidade P da próxima palavra;

- Dada uma sequência de palavras, calcule a distribuição de probabilidade P da próxima palavra;

$$P(x^{(t+1)} | x^{(t)}, x^{(t-1)}, \dots, x^{(1)})$$

- em que $x^{(t+1)}$ é qualquer palavra no vocabulário $\mathbb{V} = \{w_1, w_2, \dots, w_N\}$;
- t é a variável associada ao instante de tempo.

- Se temos um texto contendo M palavras, o modelo de linguagem associado ao texto é definido como:

$$\begin{aligned} P(x^{(1)}, x^{(2)}, \dots, x^{(M)}) &= P(x^{(1)}) \times P(x^{(2)}|x^{(1)}) \times \dots \\ &\quad \dots \times P(x^{(t)}|x^{(t-1)}, x^{(t-2)}, \dots, x^{(1)}) \end{aligned} \tag{10}$$

- Se temos um texto contendo M palavras, o modelo de linguagem associado ao texto é definido como:

$$\begin{aligned} P(x^{(1)}, x^{(2)}, \dots, x^{(M)}) &= P(x^{(1)}) \times P(x^{(2)}|x^{(1)}) \times \dots \\ &\quad \dots \times P(x^{(t)}|x^{(t-1)}, x^{(t-2)}, \dots, x^{(1)}) \\ P(x^{(1)}, x^{(2)}, \dots, x^{(M)}) &= \prod_{t=1}^M P(x^t|x^{(t-1)}, \dots, x^{(1)}) \quad (11) \end{aligned}$$

$$P(x^{(1)}, x^{(2)}, \dots, x^{(M)}) = \prod_{t=1}^M P(x^t | x^{(t-1)}, \dots, x^{(1)}) \quad (12)$$

- Um modelo n -gram pode ser empregado para implementar o modelo de linguagem acima;


Fui ao posto de gasolina abastecer o _____

Pega a distribuição de
probabilidade

Fui ao posto de gasolina abastecer o _____

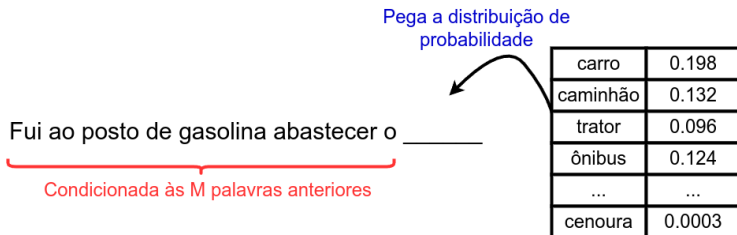
Fui ao posto de gasolina abastecer o _____

Pega a distribuição de
probabilidade

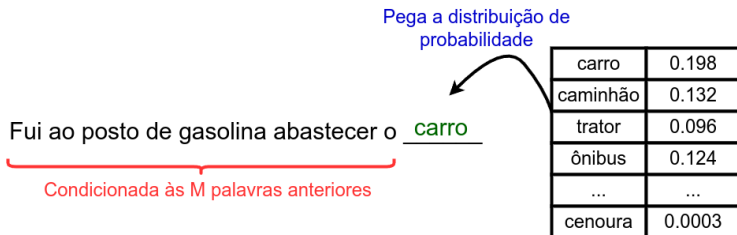


carro	0.198
caminhão	0.132
trator	0.096
ônibus	0.124
...	...
cenoura	0.0003

Modelos de Linguagem



Modelos de Linguagem



Modelo de Linguagem utilizando RNNs

- Em PLN, podemos empregar uma RNN para funcionar como um modelo de linguagem;
- O objetivo é permitir a propagação de informação relacionada ao contexto através da rede durante os instantes de tempo.

Modelo de Linguagem Baseado em RNN



- Duas alterações são essenciais para se utilizar uma RNN como um modelo de linguagem:

- Duas alterações são essenciais para se utilizar uma RNN como um modelo de linguagem:
- **1.** Representação de uma palavra em linguagem natural em uma estrutura vetorial;

Adaptações na RNN de Elman

- Duas alterações são essenciais para se utilizar uma RNN como um modelo de linguagem:
- **1.** Representação de uma palavra em linguagem natural em uma estrutura vetorial;
- **2.** Ajuste da saída para refletir uma distribuição de probabilidade em relação às palavras do vocabulário.

$$x^{(t)} = W_e l^{(t)} \quad (13)$$

- $l^{(t)}$ é a representação inteira da palavra no instante de tempo t ;

$$x^{(t)} = W_e l^{(t)} \quad (14)$$

- $l^{(t)}$ é a representação inteira da palavra no instante de tempo t ;
 - construa um vocabulário, em que cada palavra está associada a um índice inteiro;
 - utilize esse índice como a representação inteira.

$$x^{(t)} = W_e l^{(t)} \quad (15)$$

- $l^{(t)}$ é a representação inteira da palavra no instante de tempo t ;
- $W_e \in \mathbb{R}^{N \times D_e}$ é a matriz de pesos da camada *Embedding*, que gera um espaço denso;

$$x^{(t)} = W_e l^{(t)} \quad (16)$$

- $l^{(t)}$ é a representação inteira da palavra no instante de tempo t ;
- $W_e \in \mathbb{R}^{N \times D_e}$ é a matriz de pesos da camada *Embedding*, que gera um espaço denso;
 - $N = |\mathbb{V}|$ é o tamanho do vocabulário \mathbb{V} do *corpus*;

$$x^{(t)} = W_e l^{(t)} \quad (17)$$

- $l^{(t)}$ é a representação inteira da palavra no instante de tempo t ;
- $W_e \in \mathbb{R}^{N \times D_e}$ é a matriz de pesos da camada *Embedding*, que gera um espaço denso;
 - $N = |\mathbb{V}|$ é o tamanho do vocabulário \mathbb{V} do *corpus*;
 - D_e é a dimensão do vetor de Embeddings (um hiperparâmetro!).

$$\hat{y}^{(t)} = \textit{softmax}(W_s h^{(t)}) \quad (18)$$

- $\hat{y}^{(t)}$ é a distribuição de probabilidade sobre o vocabulário do *corpus* em um instante de tempo t .

$$\hat{y}^{(t)} = \textit{softmax}(W_s h^{(t)}) \quad (19)$$

- $\hat{y}^{(t)}$ é a distribuição de probabilidade sobre o vocabulário do *corpus* em um instante de tempo t .
- $\hat{y}^{(t)}$ é a próxima palavra a ser predita com base:

$$\hat{y}^{(t)} = \text{softmax}(W_s h^{(t)}) \quad (20)$$

- $\hat{y}^{(t)}$ é a distribuição de probabilidade sobre o vocabulário do *corpus* em um instante de tempo t .
- $\hat{y}^{(t)}$ é a próxima palavra a ser predita com base:
 - no estado anterior (codificação do contexto) $h^{(t-1)}$;

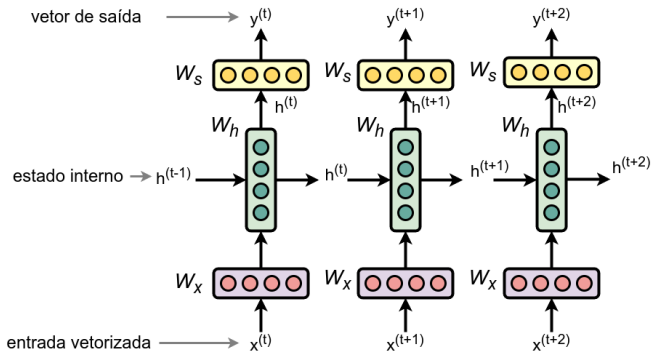
$$\hat{y}^{(t)} = \text{softmax}(W_s h^{(t)}) \quad (21)$$

- $\hat{y}^{(t)}$ é a distribuição de probabilidade sobre o vocabulário do *corpus* em um instante de tempo t .
- $\hat{y}^{(t)}$ é a próxima palavra a ser predita com base:
 - no estado anterior (codificação do contexto) $h^{(t-1)}$;
 - e a última palavra analisada a partir de seu vetor $x^{(t)}$.

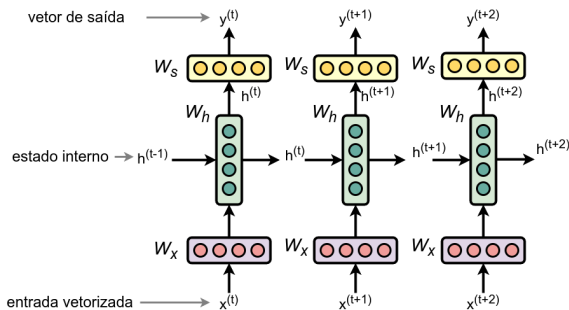
$$\hat{y}^{(t)} = \text{softmax}(W_s h^{(t)}) \quad (22)$$

- $\hat{y}^{(t)}$ é a distribuição de probabilidade sobre o vocabulário do *corpus* em um instante de tempo t .
- $\hat{y}^{(t)}$ é a próxima palavra a ser predita com base:
- $W_s \in \mathbb{R}^{N \times D_h}$ e $\hat{y} \in \mathbb{R}^N$, em que $N = |\mathbb{V}|$ é o tamanho do vocabulário \mathbb{V} do *corpus*.

Representação Gráfica

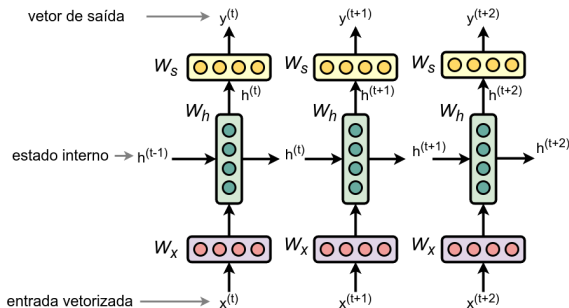


Representação Gráfica



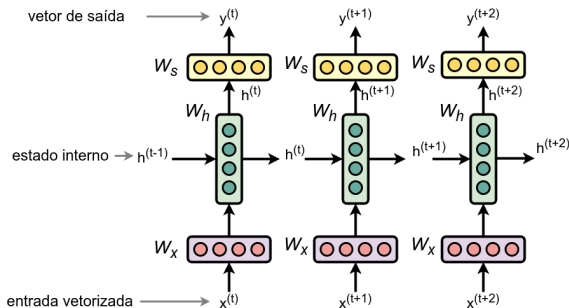
- Camada oculta contém neurônios;
 - podemos ter várias neurônios - também chamadas de unidades internas!

Representação Gráfica



- Camada oculta contém neurônios;
 - podemos ter várias neurônios - também chamadas de unidades internas!
- Um neurônio realiza uma operação matricial linear em suas entradas;

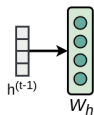
Representação Gráfica



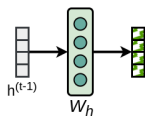
- Camada oculta contém neurônios;
 - podemos ter várias neurônios - também chamadas de unidades internas!
- Em seguida, uma função de ativação ($\tanh()$ ou $\text{ReLU}()$) é aplicada.


 $h^{(t-1)}$

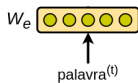
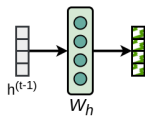
palavra^(t)

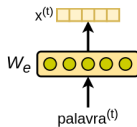
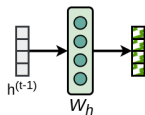


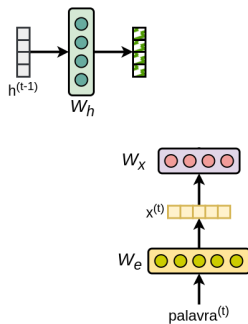
palavra^(t)

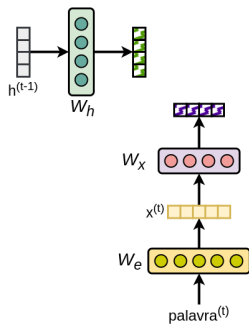


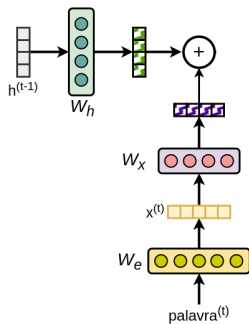
palavra^(t)

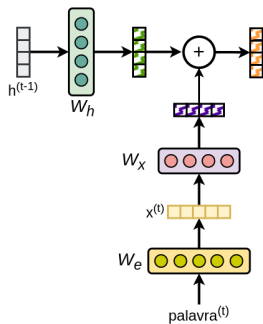


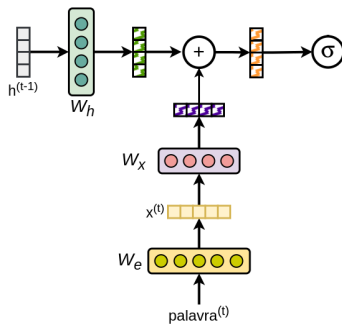


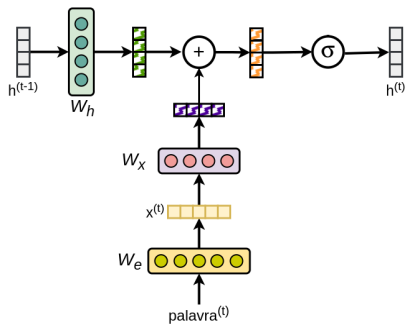


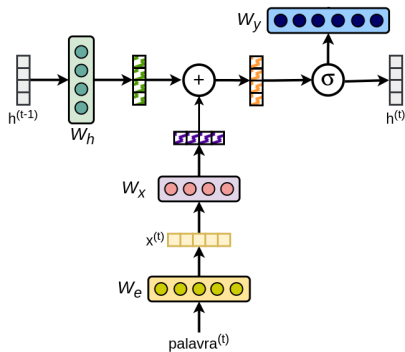


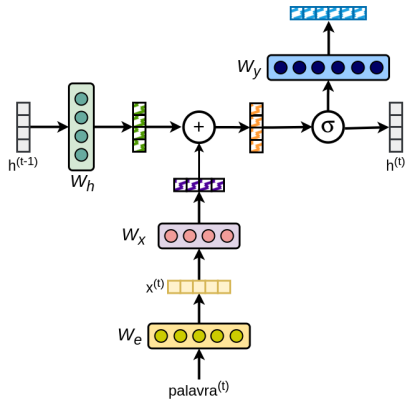


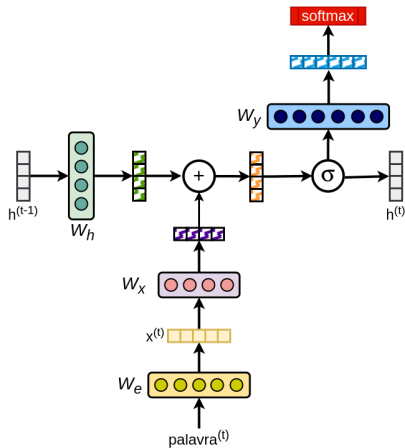


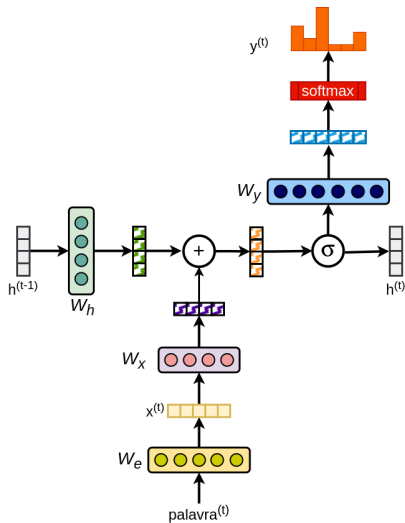




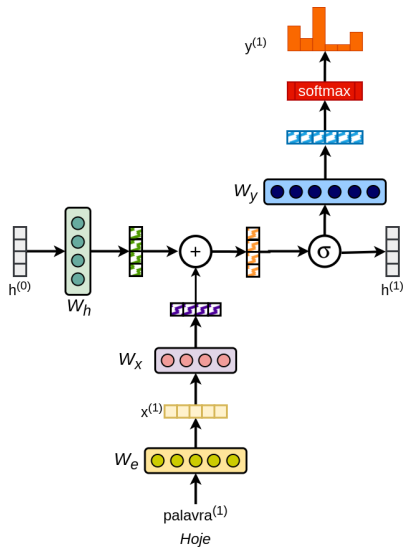




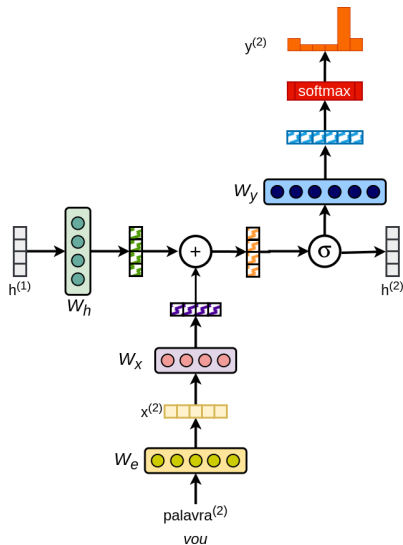




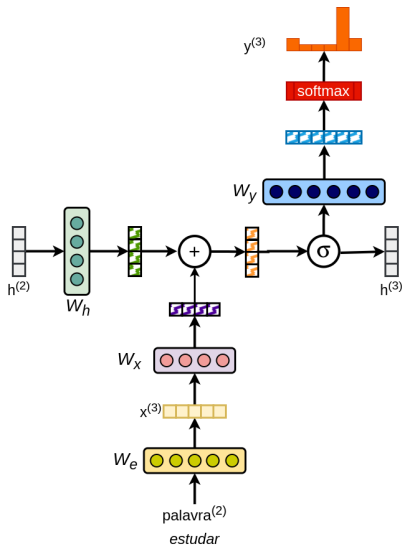
Processando uma Sentença



Processando uma Sentença



Processando uma Sentença



- Pegue o maior *corpus* possível para treinar sua RNN;

Treinamento de um modelo de linguagem

- Pegue o maior *corpus* possível para treinar sua RNN;
- Processe sentença por sentença.

- Pegue o maior *corpus* possível para treinar sua RNN;
- Processe sentença por sentença. Suponha a sentença:
O ponto forte da equipe é o jogo aéreo.

Treinamento de um modelo de linguagem

O ponto forte da equipe é o jogo aéreo.

- Para cada palavra dessa sentença, reestrute o *corpus* como se segue maneira:

palavra	próxima palavra
O	ponto
ponto	forte
forte	da
da	equipe
equipe	é
...	...
aéreo	"<END>"

Treinamento de um modelo de linguagem

O ponto forte da equipe é o jogo aéreo.

- Para cada palavra dessa sentença, reestrute o *corpus* como se segue maneira:

notação (vetorial)	palavra	próxima palavra	notação
$x^{(1)}$	O	ponto	$y^{(1)}$
$x^{(2)}$	ponto	forte	$y^{(2)}$
$x^{(3)}$	forte	da	$y^{(3)}$
$x^{(4)}$	da	equipe	$y^{(4)}$
$x^{(5)}$	equipe	é	$y^{(5)}$
...
$x^{(M)}$	aéreo	"<END>"	$y^{(M)}$

Treinamento de um modelo de linguagem

- Em seguida, pegue cada par $(x^{(t)}, y^{(t)})$ (palavra, próxima palavra) e coloque como entrada da RNN;

Treinamento de um modelo de linguagem

- Em seguida, pegue cada par $(x^{(t)}, y^{(t)})$ (palavra, próxima palavra) e coloque como entrada da RNN;
- Obtenha a predição $\hat{y}^{(t)}$ para cada instante de tempo t ;

Treinamento de um modelo de linguagem

- Em seguida, pegue cada par $(x^{(t)}, y^{(t)})$ (palavra, próxima palavra) e coloque como entrada da RNN;
- Obtenha a predição $\hat{y}^{(t)}$ para cada instante de tempo t ;
 - em outras palavras, determine a distribuição de probabilidade de cada palavra condicionada às palavras analisadas até o momento.

- Calcule a função *loss* utilizando
 - $\hat{y}^{(t)}$: a próxima palavra de acordo com a probabilidade predita pela RNN;
 - $y^{(t)}$: a verdadeira próxima palavra (gabarito).
- A função *loss* empregada em uma RNN é comumente a entropia cruzada:

$$J^{(t)}(\theta) = CE(y^{(t)}, \hat{y}^{(t)}) = - \sum_{w \in \mathbb{V}} y_w^{(t)} \log \hat{y}_w^{(t)} = - \log \hat{y}_{x^{(t+1)}}^{(t)} \quad (23)$$

- A função *loss* empregada em uma RNN é comumente a entropia cruzada:

$$J^{(t)}(\theta) = CE(y^{(t)}, \hat{y}^{(t)}) = - \sum_{w \in \mathbb{V}} y_w^{(t)} \log \hat{y}_w^{(t)} = - \log \hat{y}_{x^{(t+1)}}^{(t)} \quad (24)$$

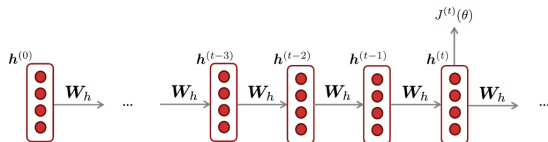
- A função *loss* geral para todo os dados de treinamento:

$$J(\theta) = \frac{1}{T} \sum_{t=1}^T J^{(t)}(\theta) = \frac{1}{T} \sum_{t=1}^T - \log \hat{y}_{x^{(t+1)}}^{(t)} \quad (25)$$

- O cálculo da função *loss* é proibitivo para ser feito de uma única vez após processar todo o *corpus*;
- Para amortizar esses cálculos, deve-se calcular a *loss* após processar uma parte do *corpus*
 - uma sentença ou um documento, por exemplo.
- Sabemos que o Gradiente Descendente nos permite calcular a função *loss* e os gradientes para pequenas quantidades de dados e atualizá-los.

- Processo feito para uma sentença (ou um batch de sentenças):
 - 1 Calcular a *loss* $J(\theta)$;
 - 2 Calcular os gradientes;
 - 3 Atualizar os pesos;
- Repita o processo acima para um novo batch...

Otimização dos parâmetros



- A derivada de $J(\theta)$ em relação a uma matriz de pesos repetida é:

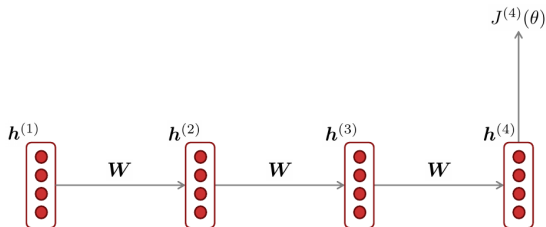
$$\frac{\partial J^{(t)}}{\partial W_h} = \sum_{i=1}^t \frac{\partial J^{(t)}}{\partial W_h} \Big|_{(i)} \quad (26)$$

- 1 Calcular a *loss* $J(\theta)$;
 - 2 Calcular os gradientes;
 - 3 Atualizar os pesos;
- Repita o processo acima para um novo batch...

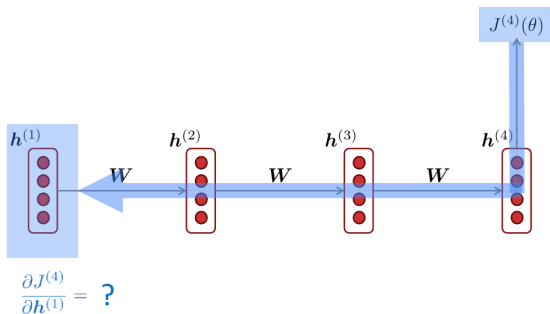
Implementação



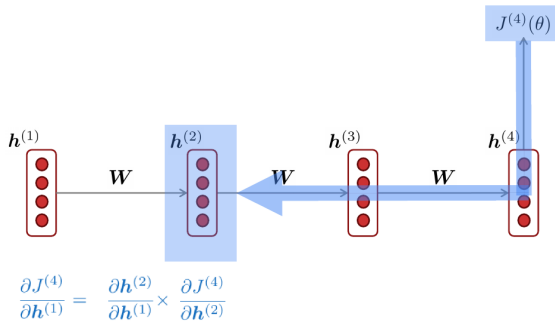
O Gradiente que Encolhe!



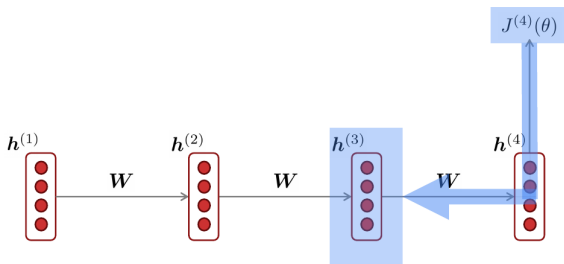
O Gradiente que Encolhe!



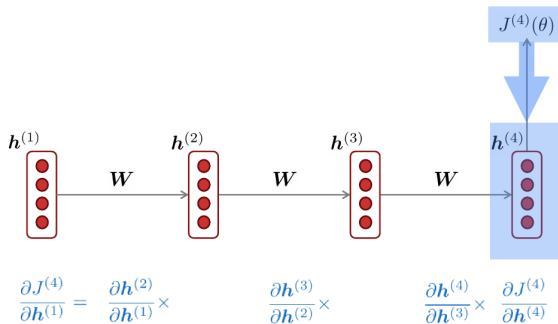
O Gradiente que Encolhe!



O Gradiente que Encolhe!

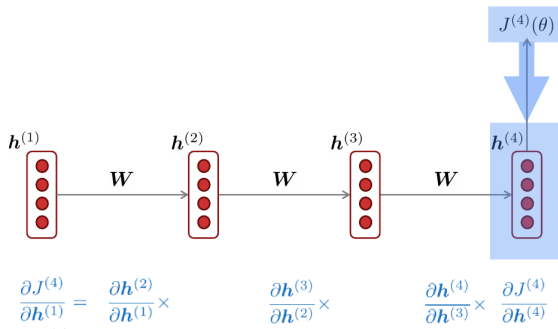


O Gradiente que Encolhe!



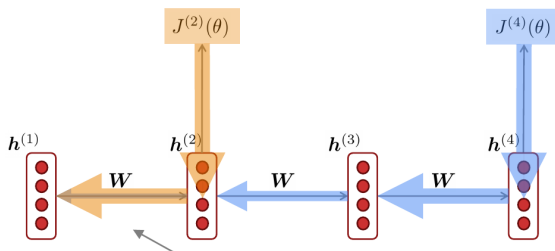
- O que acontece se as derivadas resultarem em valores pequenos?

O Gradiente que Encolhe!



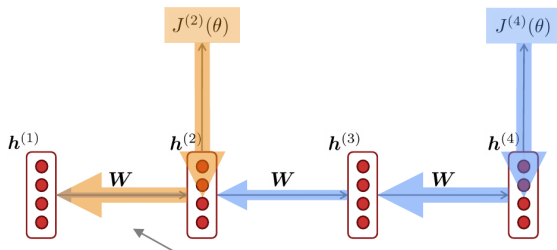
- O que acontece se as derivadas resultarem em valores pequenos?
- A magnitude do gradiente diminui com o passar do *backpropagation* durante o tempo.

O Gradiente que Encolhe!



- O gradiente mais distante (em azul) é perdido com o passar do tempo, pois sua magnitude é bem menor do que era quando foi calculada (mais forte em amarelo).

O Gradiente que Encolhe!



- Os pesos da RNN tem efeito a curto prazo, e não a longo prazo!
- A preservação das relações de longa dependência são mínimas em uma RNN.

O Gradiente que Explode!

- Por outro lado, a magnitude gradiente pode aumentar caso as atualizações dos pesos sejam significativas;

$$\theta^{new} = \theta^{old} - \alpha \nabla_{\theta} J(\theta) \quad (27)$$

- Isso gera atualizações discrepantes nos parâmetros do modelo;
 - valores de *loss* altos e com bastante variações entre as épocas.

Considerações Finais



- Capaz de processar textos de entrada de qualquer tamanho;
- O tamanho do modelo não está relacionado ao tamanho do texto de entrada;
- Treinamento e geração das saídas levam em conta informações no tempo;
- Os pesos são compartilhados durante o tempo.

- Treinamento e cálculo da saída são lentos e computacionalmente caros;
- Difficulty of accessing information from a long time ago
- Cannot consider any future input for the current state

- As RNNs propagam Recurrent neural networks propagate weight matrices from one timestep to the next. Recall the goal of a RNN implementation is to enable propagating context information through faraway time-steps. For example, consider the following two sentences:

Implementação





Universidade de Brasília

Departamento de Ciência da Computação



Redes Neurais Recorrentes

Curso ENAP - Processamento de Linguagem Natural

Prof. Dr. Vinícius Ruela Pereira Borges

`viniciusrpb@unb.br`

Brasília-DF, 2024