

# Aprendizado de Máquina em Linguagem Natural

---

Beatriz Albiero

<https://github.com/beatrizalbiero>

Felipe Salvatore

<https://felipessalvatore.github.io/>

November 21, 2017

**USP:**University of São Paulo

# Conceitos básicos

---

Um algoritmo de **aprendizado de máquina**  
é um algoritmo que é capaz  
de usar dados para  
realizar uma tarefa

**Regressão**

$\mathbf{x}_1, \dots, \mathbf{x}_N$



$y_1, \dots, y_N$

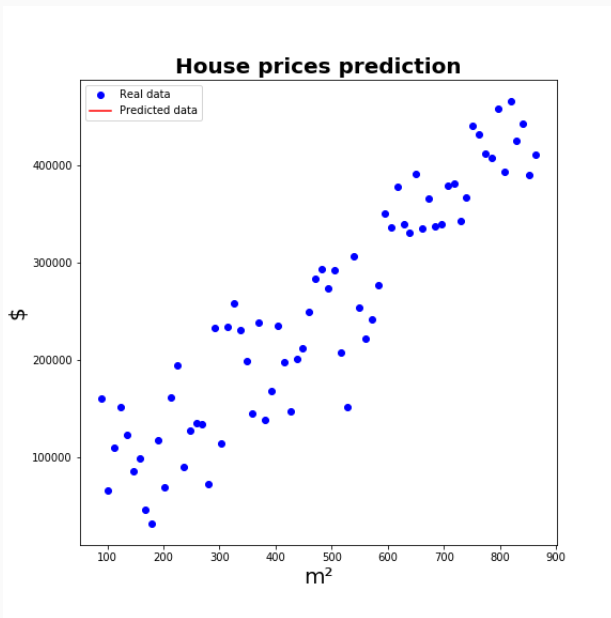
**Classificação**

$\mathbf{x}_1, \dots, \mathbf{x}_N$

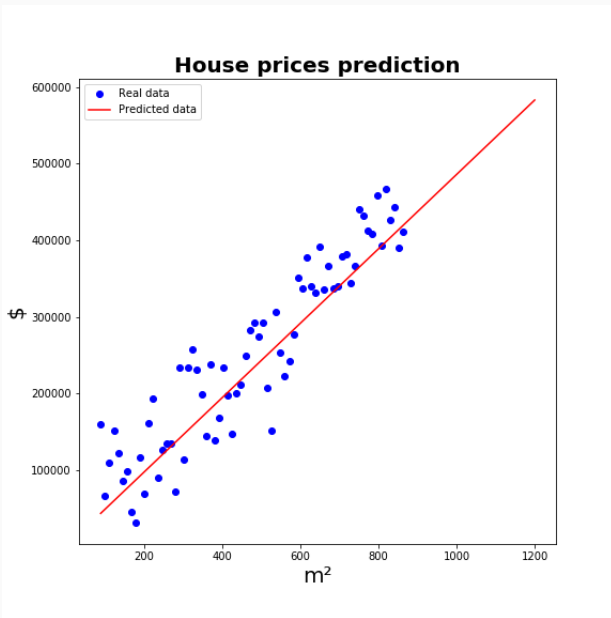


$L_1, \dots, L_N$

# Regressão



# Regressão



## Classificação: Fashion MNIST [6]



→  $y = \text{camisa}$



→  $y = \text{vestido}$



→  $y = \text{bolsa}$



→  $y = \text{tênis}$

# Aprendizado por redes neurais

O que sabemos sobre o cérebro:

- neurônios em rede
- neurônios emitem sinais elétricos (disparam)
- dendritos e axônios

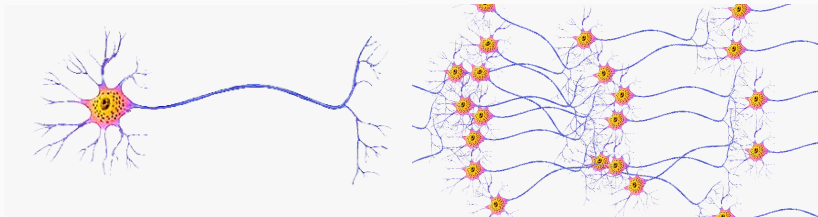


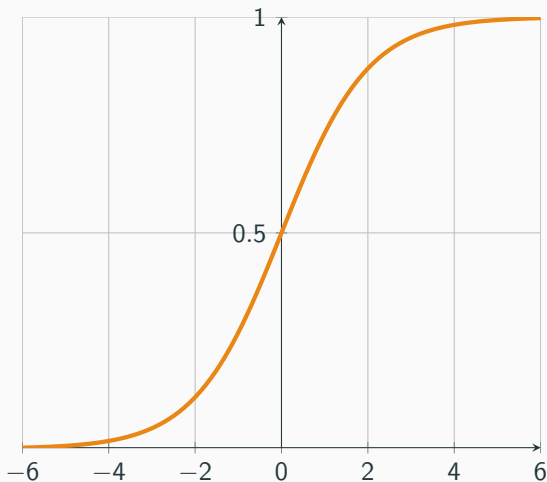
imagem retirada de [1]

Traduzindo para o modelo de redes neurais artificiais:

- Energia recebida: **Inputs**
- Energia enviada: **Output**
- Força entre uma conexão e outra: **Peso ( $\theta$ )**
- Carga mínima: **Threshold**
- Uma função que recebe um input e emite um output mas leva em consideração um threshold mínimo: **Função de ativação**

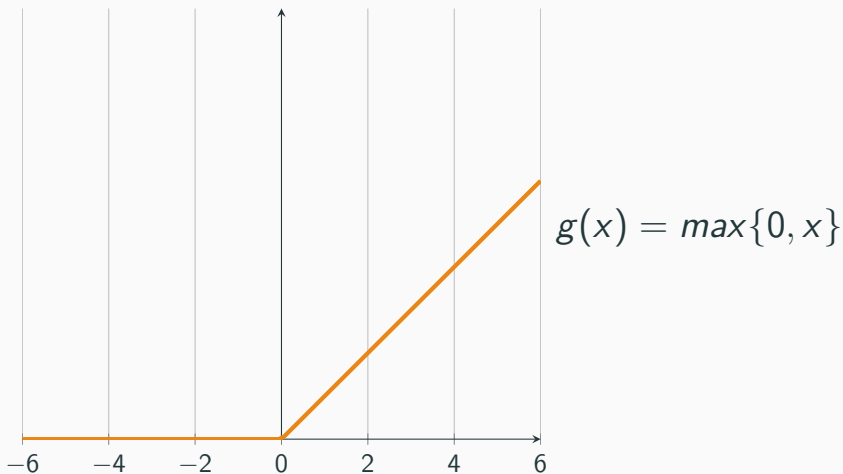


## Função de ativação: função sigmoide

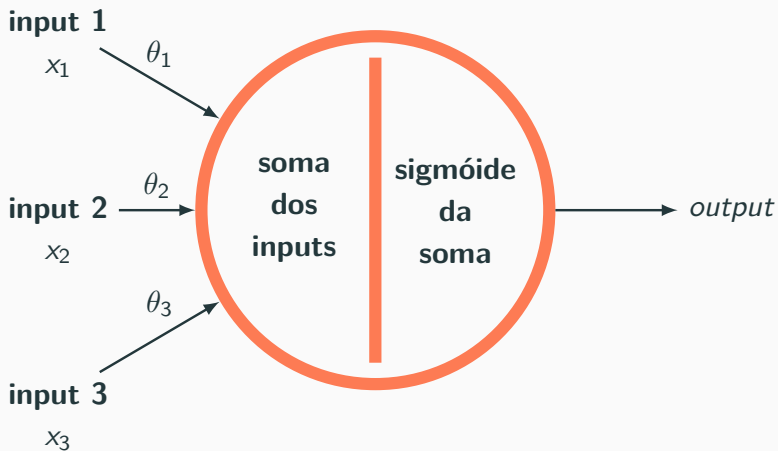


$$\sigma(x) = \frac{1}{1+e^{-x}}$$

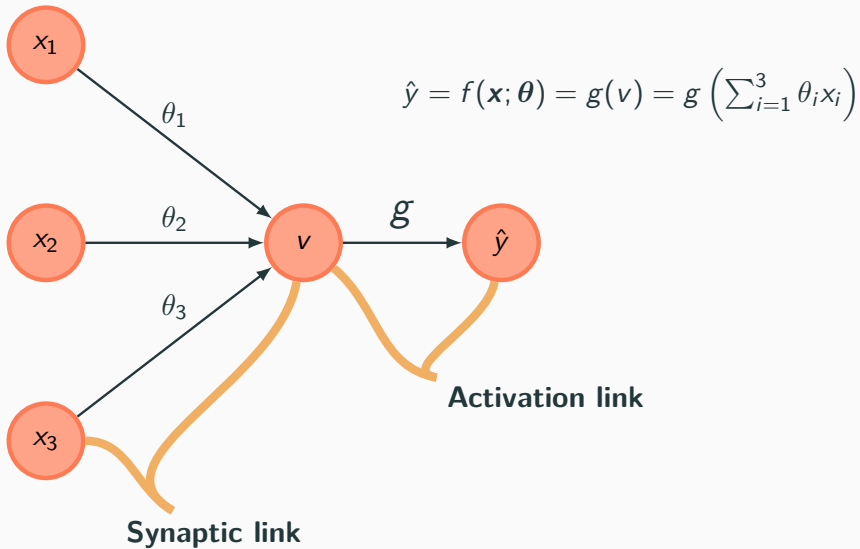
## função de ativação: ReLU (Rectified Linear Unit)



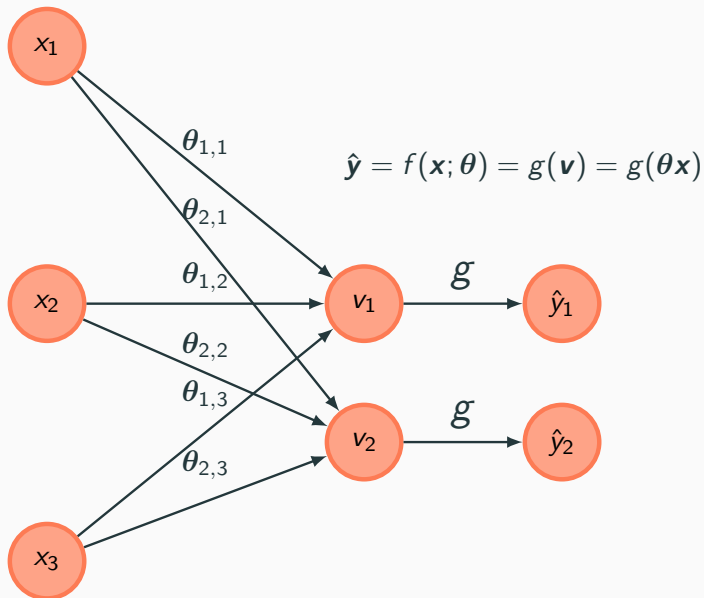
# Perceptron



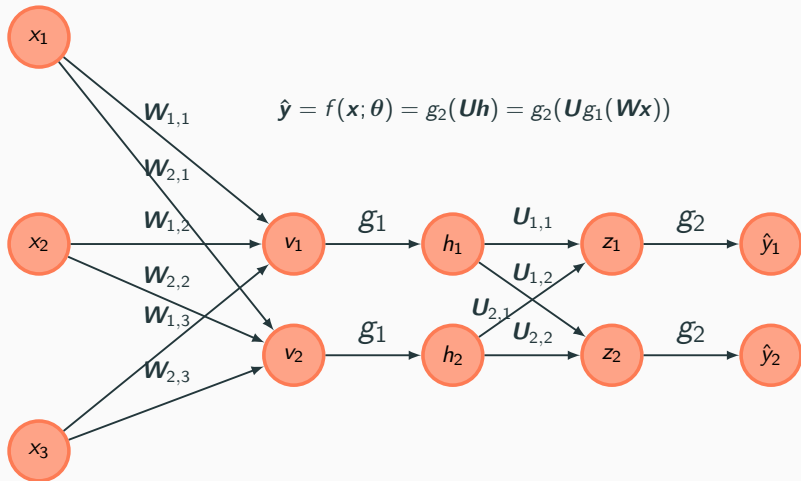
# Perceptron



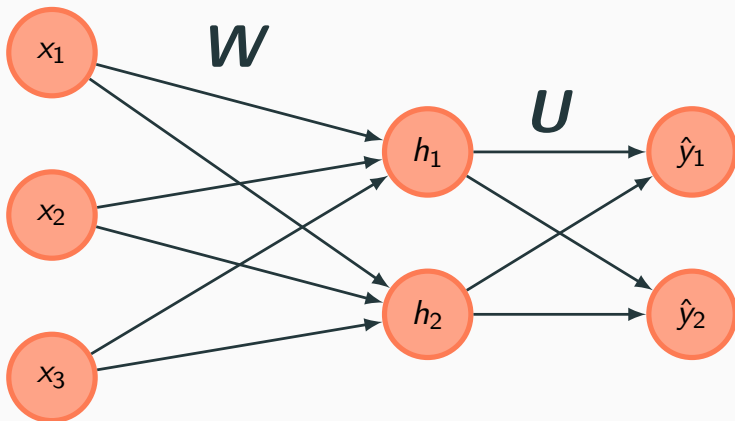
# Multi layer perceptron – Feedforward neural network



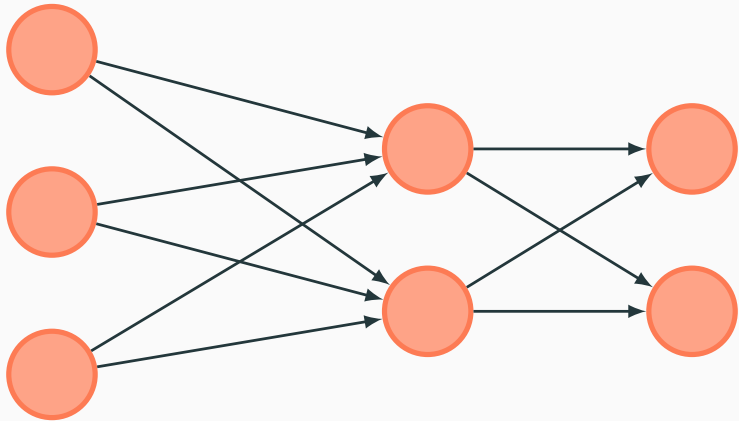
# Multi layer perceptron – Feedforward neural network



## Multi layer perceptron – Feedforward neural network

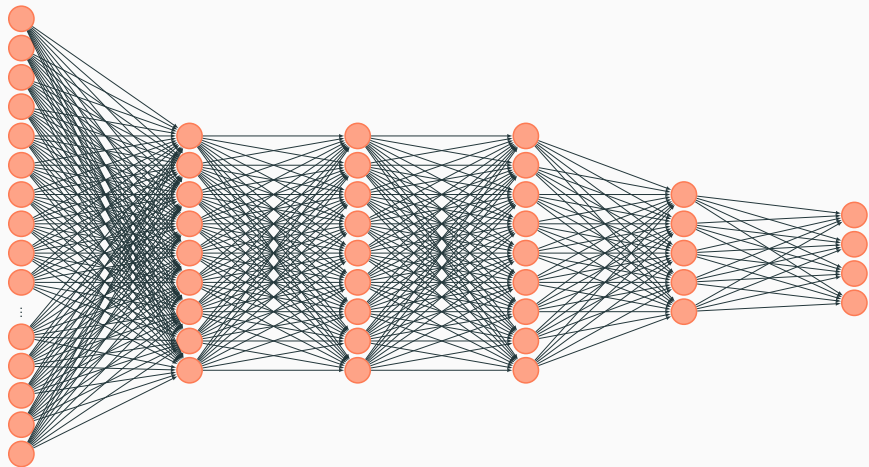


## Multi layer perceptron – Feedforward neural network





# Deep feedforward network



**Features** são características ou traços do objeto do aprendizado.

**Features** são características ou traços do objeto do aprendizado.

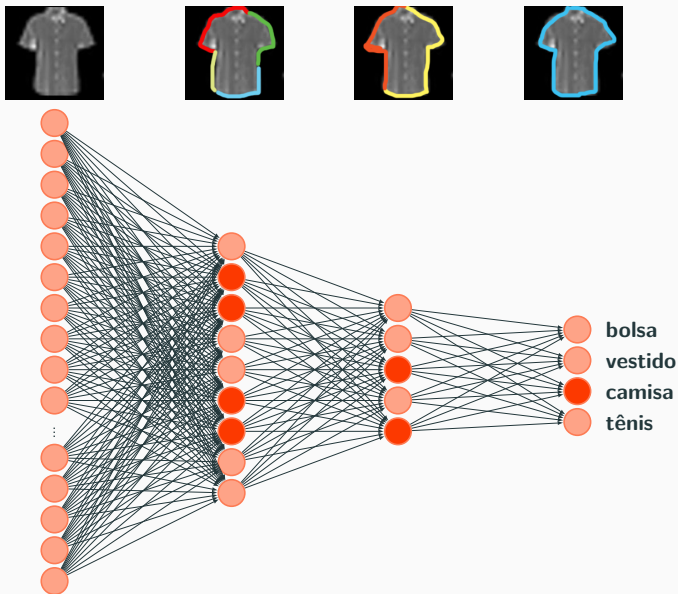
O aprendizado irá acontecer a partir do **reconhecimento de padrões** entre **features**.

**Features** são características ou traços do objeto do aprendizado.

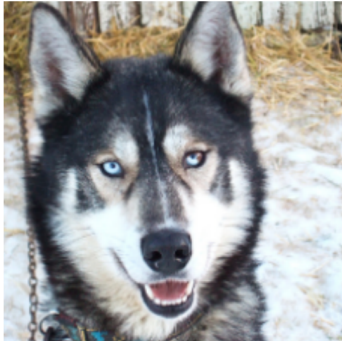
O aprendizado irá acontecer a partir do **reconhecimento de padrões** entre **features**.

Treinaremos o modelo a **ativar as mesmas unidades simultaneamente** quando diante de um **determinado padrão**.

# Classificação de imagens



# Classificação de imagens



(a) Husky classified as wolf

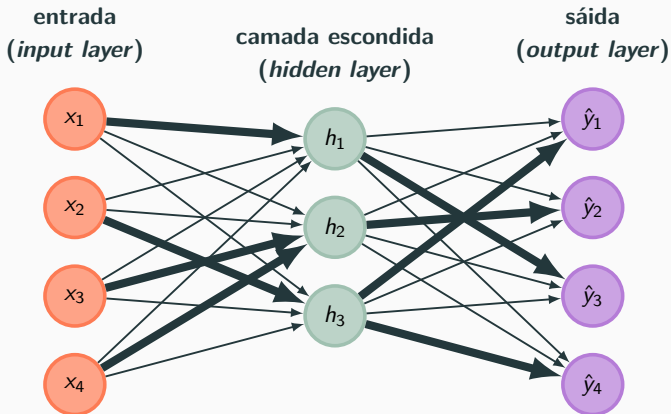


(b) Explanation

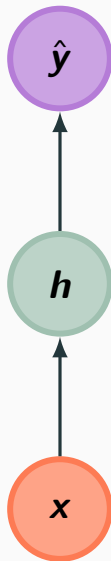
**Figure 11: Raw data and explanation of a bad model's prediction in the "Husky vs Wolf" task.**

imagem retirada de [5]

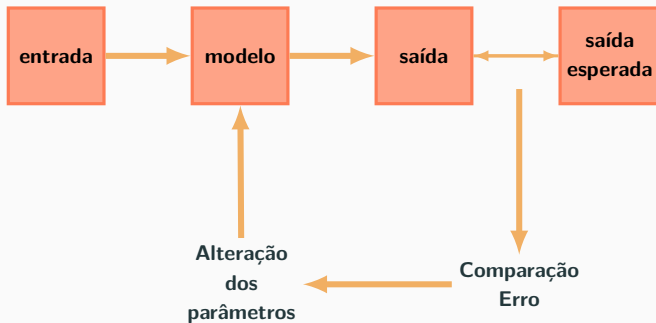
# Feedforward neural network (rede neural)



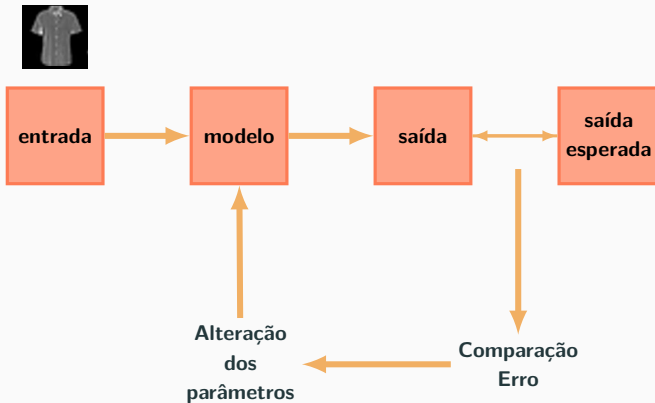
## Versão resumida de uma rede neural



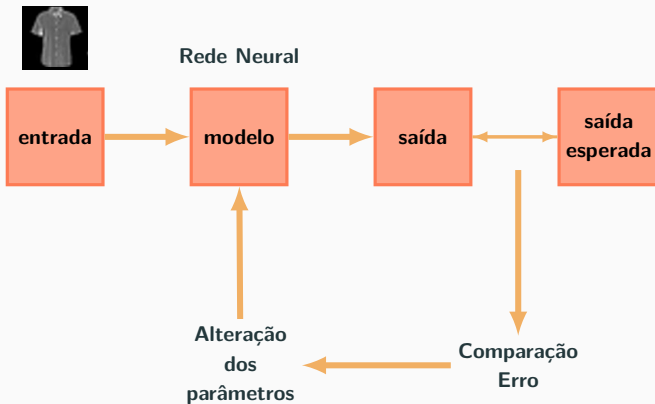




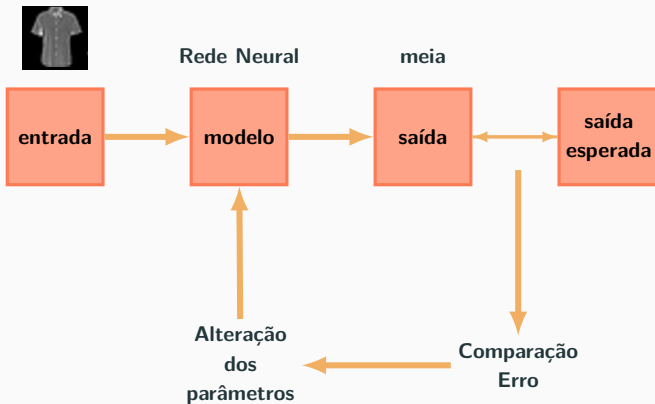
# Treinamento



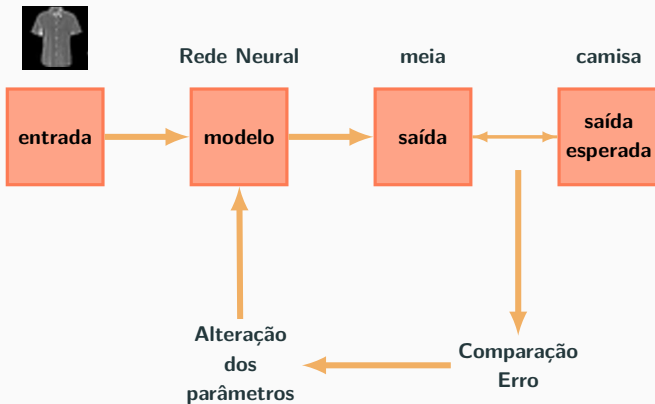
# Treinamento



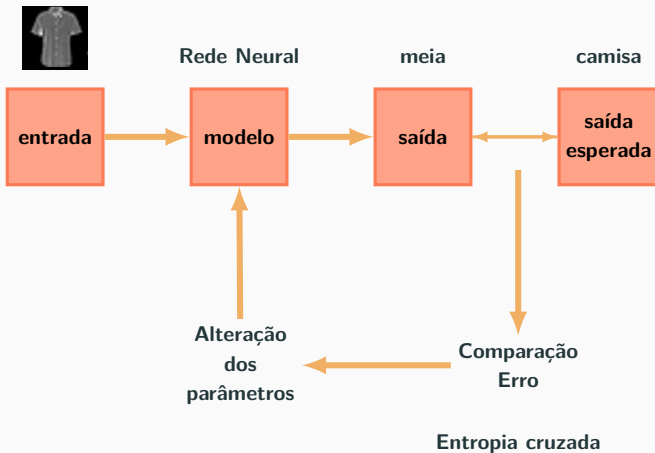
# Treinamento



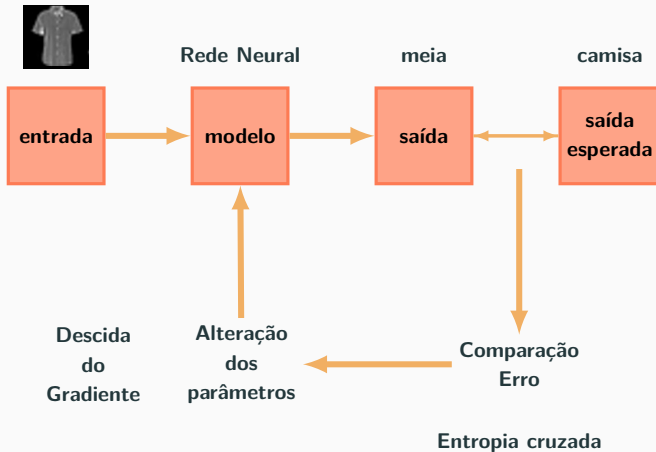
# Treinamento



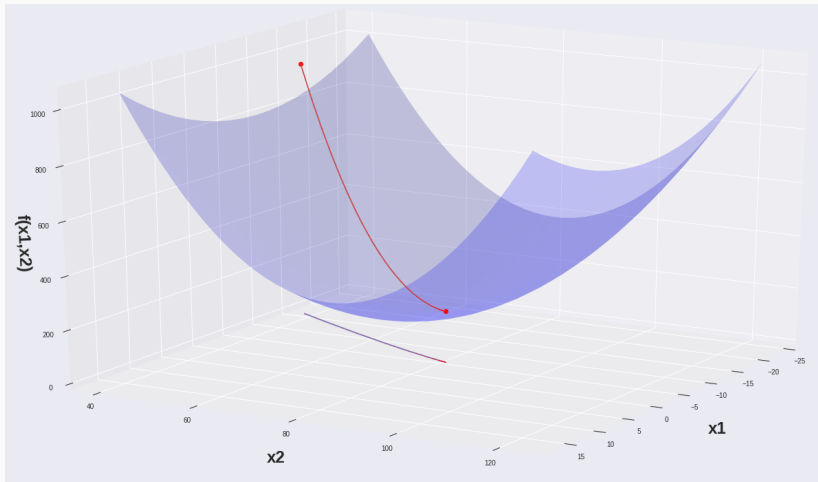
# Treinamento



# Treinamento



# Descida do gradiente





# Aprendizado de máquina em linguística

---

# On Learning the Past Tenses of English Verbs

Problema:

- Aprendizado dos verbos "irregulares" no passado do inglês

# On Learning the Past Tenses of English Verbs

Problema:

- Aprendizado dos verbos "irregulares" no passado do inglês
- Famílias de verbos irregulares:

# On Learning the Past Tenses of English Verbs

Problema:

- Aprendizado dos verbos "irregulares" no passado do inglês
- Famílias de verbos irregulares:
  - “blow–blew, grow–grew, know–knew, throw–threw”
  - “bind–bound, find–found, grind–ground, wind–wound”
  - “drink–drank, shrink–shrank, sink–sank, stink–stank”

# On Learning the Past Tenses of English Verbs

Problema:

- Aprendizado dos verbos "irregulares" no passado do inglês
- Famílias de verbos irregulares:
  - “blow–blew, grow–grew, know–knew, throw–threw”
  - “bind–bound, find–found, grind–ground, wind–wound”
  - “drink–drank, shrink–shrank, sink–sank, stink–stank”
- Chomsky vs Rumelhart e McClelland

# On Learning the Past Tenses of English Verbs

Problema:

- Aprendizado dos verbos "irregulares" no passado do inglês
- Famílias de verbos irregulares:
  - “blow–blew, grow–grew, know–knew, throw–threw”
  - “bind–bound, find–found, grind–ground, wind–wound”
  - “drink–drank, shrink–shrank, sink–sank, stink–stank”
- Chomsky vs Rumelhart e McClelland
- Regras (Racionalismo) vs Analogias (Conexionismo)

# On Learning the Past Tenses of English Verbs

Exemplo de entrada  $\mathbf{x}$  e saída  $\mathbf{y}$ :

$(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}) = (\text{begin}, \text{began}).$

$(\mathbf{x}^{(2)}, \mathbf{y}^{(2)}) = (\text{love}, \text{loved})$

$(\mathbf{x}^{(3)}, \mathbf{y}^{(3)}) = (\text{drink}, \text{drank})$

$(\mathbf{x}^{(4)}, \mathbf{y}^{(4)}) = (\text{hate}, \text{hated})$

$(\mathbf{x}^{(5)}, \mathbf{y}^{(5)}) = (\text{grow}, \text{grew})$

$(\mathbf{x}^{(6)}, \mathbf{y}^{(6)}) = (\text{bind}, \text{bound})$

$(\mathbf{x}^{(7)}, \mathbf{y}^{(7)}) = (\text{hit}, \text{hit})$

...

# On Learning the Past Tenses of English Verbs

$x$  será uma combinação de traços (features) fonológicos.

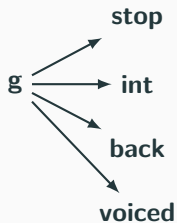
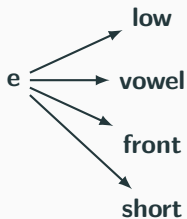
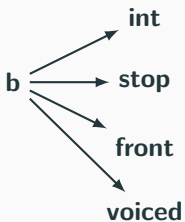
**Table 1:** Categorização de Fonemas em 4 dimensões simples

		Place					
		Front		Middle		Back	
		V/L	U/S	V/L	U/S	V/L	U/S
Int.	Stop	b	p	d	t	g	k
	Nasal	m	-	n	-	ŋ	-
Cont	Fric	v/D	f/T	z	s	ʒ/j	ʃ/C
	Liq/SV	w/l	-	r	-	y	h
Vowel	High	E	i	ʊ	^	U	u
	Low	A	e	ɪ	a/α	W	o



# Explicando as features

Exemplo: begin (bEgiN)



# Wickelfeatures

Wickelfeatures: Trigramas de features

Exemplo: begin

beg

egi

gin

# Wickelfeatures

Wickelfeatures: Trigramas de features

Exemplo: begin

beg

egi

gin

[	<i>int, low, int</i>	]
[	<i>int, vowel, int</i>	]
[	<i>int, front, int</i>	]
[	<i>int, short, int</i>	]
[	<i>stop, low, stop</i>	]
[	<i>stop, vowel, stop</i>	]
[	<i>stop, front, stop</i>	]
[	<i>stop, short, stop</i>	]
[	$\vdots$	]
[	<i>voiced, front, voiced</i>	]
[	<i>voiced, short, voiced</i>	]

# Wickelfeatures

Wickelfeatures: Trigramas de features

Exemplo: begin

beg

$$\begin{bmatrix} \textit{int, low, int} \\ \textit{int, vowel, int} \\ \textit{int, front, int} \\ \textit{int, short, int} \\ \textit{stop, low, stop} \\ \textit{stop, vowel, stop} \\ \textit{stop, front, stop} \\ \textit{stop, short, stop} \\ \vdots \\ \textit{voiced, front, voiced} \\ \textit{voiced, short, voiced} \end{bmatrix}$$

egi

$$\begin{bmatrix} \textit{vowel, int, vowel} \\ \textit{vowel, stop, vowel} \\ \textit{vowel, back, vowel} \\ \textit{vowel, voiced, vowel} \\ \textit{low, int, high} \\ \textit{low, stop, high} \\ \textit{low, back, high} \\ \textit{low, voiced, high} \\ \vdots \\ \textit{short, int, short} \\ \textit{short, voiced, short} \end{bmatrix}$$

gin

$$\begin{bmatrix} \textit{int, \dots} \\ \vdots \end{bmatrix}$$


# On Learning the Past Tenses of English Verbs

$$\mathbf{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \begin{array}{l} \text{int, vowel, int} \\ \text{int, vowel, cont} \\ \text{int, vowel, vowel} \\ \\ \text{low, int, high} \\ \text{low, int, low} \\ \\ \text{short, voiced, short} \end{array}$$

# On Learning the Past Tenses of English Verbs

$$\hat{\mathbf{y}} = \begin{bmatrix} 0.234 \\ 0.967 \\ 0.123 \\ \vdots \\ 0.152 \\ 0.876 \\ \vdots \\ 0.765 \end{bmatrix} \quad \begin{array}{l} \text{int, vowel, int} \\ \text{int, vowel, cont} \\ \text{int, vowel, vowel} \\ \\ \text{low, int, high} \\ \text{low, int, low} \\ \\ \text{short, voiced, short} \end{array}$$

# On Learning the Past Tenses of English Verbs

$$\hat{\mathbf{y}} = \begin{bmatrix} 0.234 \\ 0.967 \\ 0.123 \\ \vdots \\ 0.152 \\ 0.876 \\ \vdots \\ 0.765 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 1 \\ 0 \\ \vdots \\ 0 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = \mathbf{y}$$


$(\hat{\mathbf{y}} - \mathbf{y})^2$   
erro quadrado médio

# On Learning the Past Tenses of English Verbs

Resultados (PROS):



# On Learning the Past Tenses of English Verbs

Resultados (PROS):

- Identificou padrões corretamente entre todos os 420 verbos do treinamento;

# On Learning the Past Tenses of English Verbs

Resultados (PROS):

- Identificou padrões corretamente entre todos os 420 verbos do treinamento;
- Taxa de acerto de **92%** para verbos regulares ausentes no treinamento;

# On Learning the Past Tenses of English Verbs

Resultados (PROS):

- Identificou padrões corretamente entre todos os 420 verbos do treinamento;
- Taxa de acerto de **92%** para verbos regulares ausentes no treinamento;
- Taxa de acerto de **84%** para verbos irregulares ausentes no treinamento;

# On Learning the Past Tenses of English Verbs

## Resultados (PROS):

- Identificou padrões corretamente entre todos os 420 verbos do treinamento;
- Taxa de acerto de **92%** para verbos regulares ausentes no treinamento;
- Taxa de acerto de **84%** para verbos irregulares ausentes no treinamento;
- **U-shaped Development**

# On Learning the Past Tenses of English Verbs

Resultados (CONS):

Resultados (CONS):

- Falha ao tentar fazer predições com palavras que compartilham muitas features em comum;  
Exemplo: "algalgal" - Oygangand

## Resultados (CONS):

- Falha ao tentar fazer predições com palavras que compartilham muitas features em comum;  
Exemplo: "algalgal" - Oygangand
- Problemas como uma teoria da mente

# As Irregularidades no Português Brasileiro

Desafios:



# As Irregularidades no Português Brasileiro

Desafios:

- "Wug Test";

Exemplos: "poguir", "redir", "atover"

# As Irregularidades no Português Brasileiro

Desafios:

- "Wug Test";  
Exemplos: "poguir", "redir", "atover"
- Adaptar a rede para a língua portuguesa

# As Irregularidades no Português Brasileiro

Desafios:

- "Wug Test";  
Exemplos: "poguir", "redir", "atover"
- Adaptar a rede para a língua portuguesa
- Melhorar o desempenho da rede

# Modelos de linguagem e redes recorrentes

---

# Definition

Nos chamamos de **modelo de linguagem** uma distribuição de probabilidade sobre uma sequência de tokens em uma língua natural.

$$P(x_1, x_2, x_3, x_4) = p$$

**Usamos esse modelo em:**

- reconhecimento de fala
- tradução automática
- text auto-completion
- correção de texto
- resposta automatizada
- sumarização

- Probabilidade condicional

$$P(A|B) = \frac{P(A, B)}{P(B)}$$

- Independência

$$P(A|B) = P(A)$$

- Regra da cadeia

$$P(A, B, C) = P(A)P(B|A)P(C|A, B)$$

## Como calculamos essas probabilidades?

Regra da cadeia:

$$P(x_1, x_2, x_3, x_4) = P(x_1)P(x_2|x_1)P(x_3|x_1x_2)P(x_4|x_1x_2x_3)$$

Para simplificação fazemos uma **suposição de Markov**, i.e., para um  $n$  específico assumimos certas independências, assim cada palavra depende apenas das últimas  $n - 1$  palavras:

$$P(x_1, \dots, x_T) = \prod_{t=1}^T P(x_t|x_1, \dots, x_{t-1}) = \prod_{t=1}^T P(x_t|x_{t-(n+1)}, \dots, x_{t-1})$$

# Modelos baseados em estatísticas de $n$ -gramas

A escolha de  $n$  leva a modelos diferentes.

**Modelo de unigrama** ( $n = 1$ ):

$$P_{uni}(x_1, x_2, x_3, x_4) = P(x_1)P(x_2)P(x_3)P(x_4)$$

em que  $P(x_i) = \text{count}(x_i)$ .

**Modelo de bigrama** ( $n = 2$ ):

$$P_{bi}(x_1, x_2, x_3, x_4) = P(x_1)P(x_2|x_1)P(x_3|x_2)P(x_4|x_3)$$

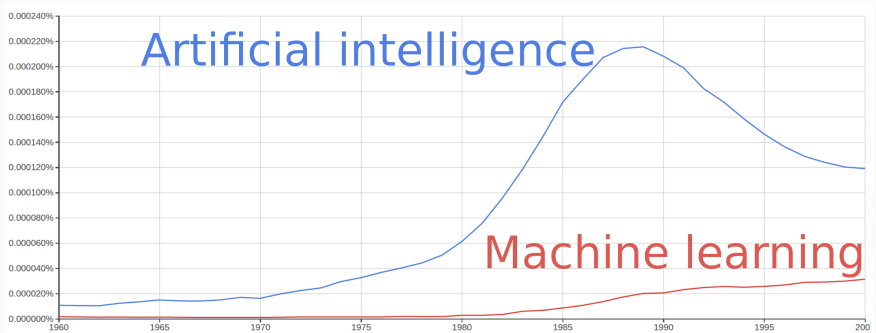
em que

$$P(x_i|x_j) = \frac{\text{count}(x_i, x_j)}{\text{count}(x_j)}$$



# Estatísticas de $n$ -gramas

<https://books.google.com/ngrams>



# Modelos baseados em estatísticas de $n$ -gramas

- Quanto maior o  $n$  melhor a performance do modelo.
- Quanto maior o  $n$  maior o uso de memória!

*"Using one machine **with 140 GB RAM for 2.8 days**, we built an unpruned model on 126 billion tokens."*

*Scalable Modified Kneser-Ney Language Model Estimation by Heafield et al.*

# Modelos de linguagem como predição de data sequencial

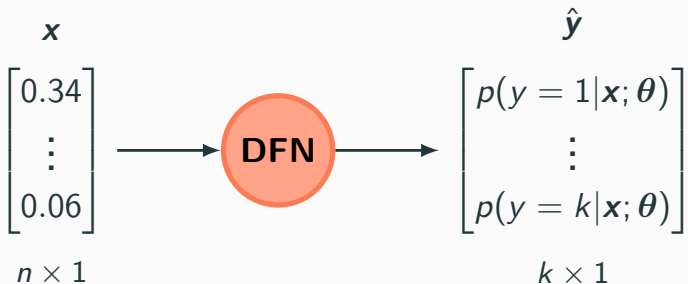
Em vez de usar uma abordagem que seja específica para o domínio da linguagem natural, podemos usar um modelo para predição de dados sequencias: **uma rede recorrente (RNN)**.

Nossa tarefa de aprendizado é estimar a distribuição de probabilidade

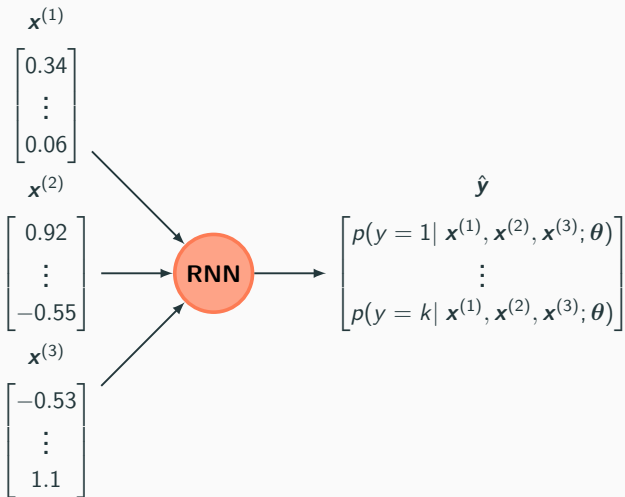
$$P(x_n = \text{palavra}_{j^*} | x_1, \dots, x_{n-1})$$

para qualquer  $(n - 1)$  sequencia de palavras  $x_1, \dots, x_{n-1}$ .

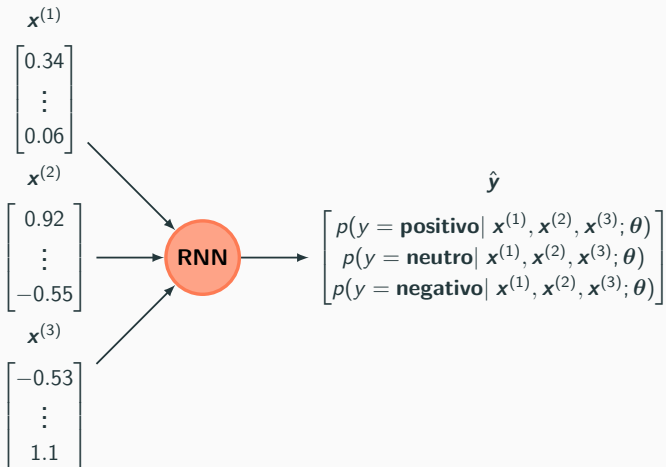
## Classificação com uma deep feedforward network



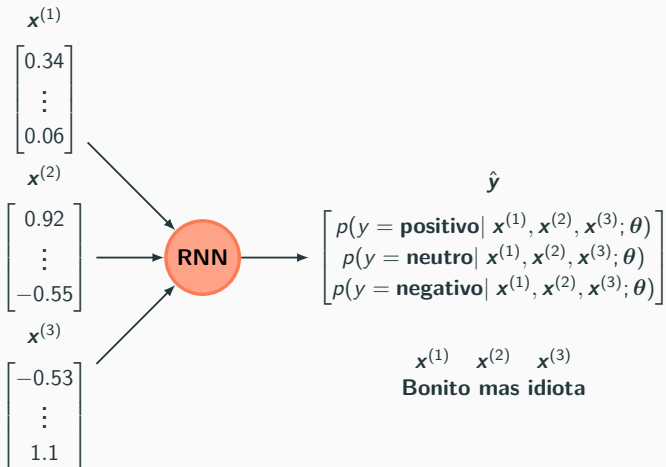
# Classificação com uma RNN



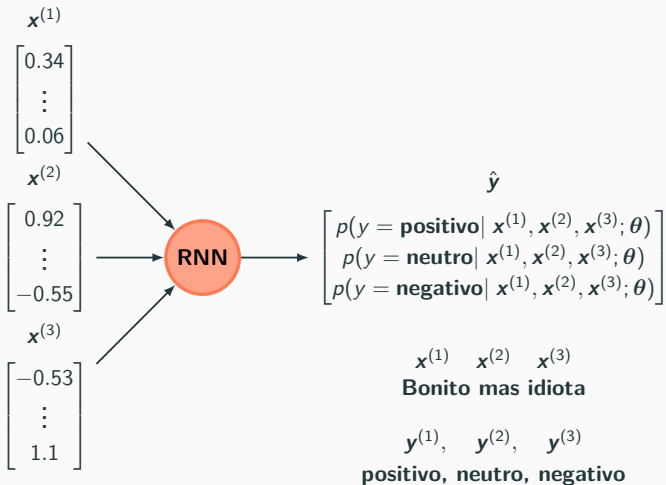
# Classificação com uma RNN



# Classificação com uma RNN

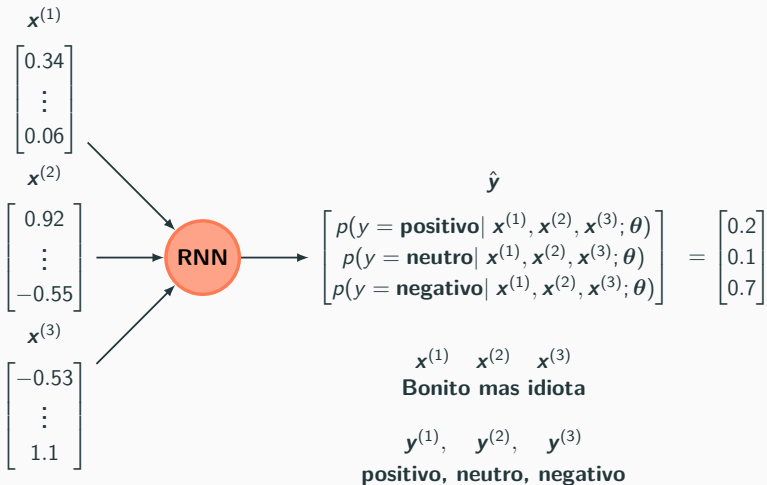


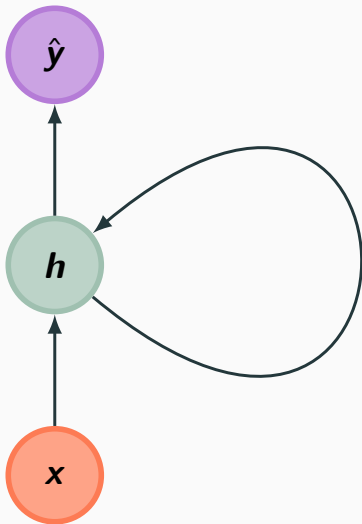
# Classificação com uma RNN

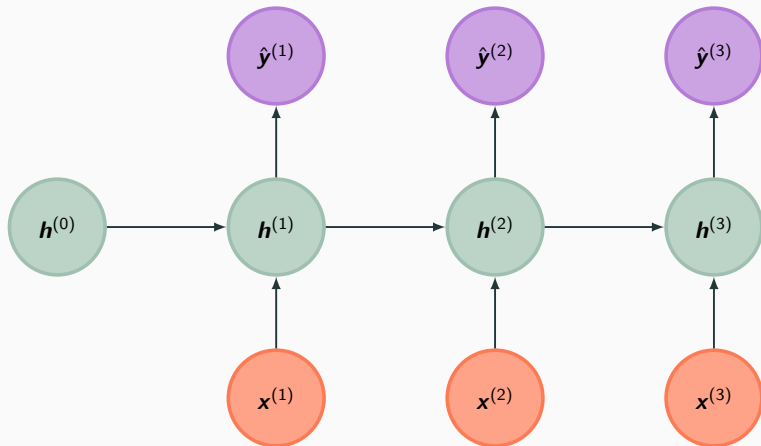




# Classificação com uma RNN







## Exemplo de um dataset

Nos separamos um corpus  $C$  com  $T$  tokens e vocabulário  $\mathbb{V}$ .

Exemplo: **Make Some Noise**, the Beastie Boys.

*Yes, here we go again, give you more, nothing lesser  
Back on the mic is the anti-depressor  
Ad-Rock, the pressure, yes, we need this  
The best is yet to come, and yes, believe this  
...*

- $T = 378$
- $|\mathbb{V}| = 186$

## Exemplo de um dataset

O dataset é uma coleção de pares  $(\mathbf{x}, \mathbf{y})$  em que  $\mathbf{x}$  é uma palavra e  $\mathbf{y}$  é a palavra imediatamente a direita. Por exemplo:

$$(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}) = (\text{Yes, here}).$$

$$(\mathbf{x}^{(2)}, \mathbf{y}^{(2)}) = (\text{here, we})$$

$$(\mathbf{x}^{(3)}, \mathbf{y}^{(3)}) = (\text{we, go})$$

$$(\mathbf{x}^{(4)}, \mathbf{y}^{(4)}) = (\text{go, again})$$

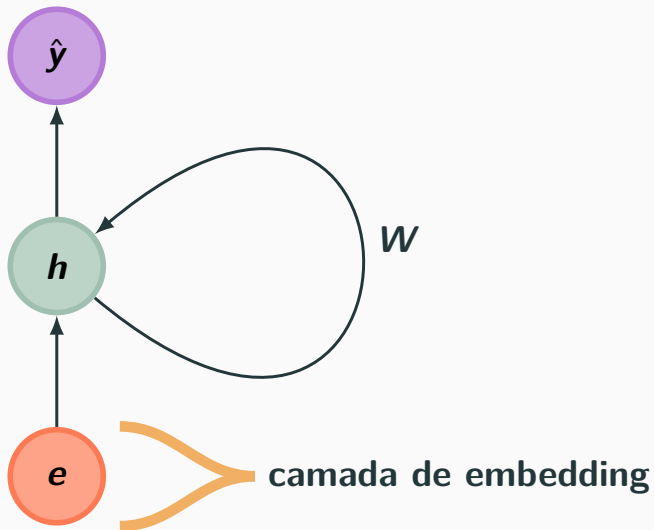
$$(\mathbf{x}^{(5)}, \mathbf{y}^{(5)}) = (\text{again, give})$$

$$(\mathbf{x}^{(6)}, \mathbf{y}^{(6)}) = (\text{give, you})$$

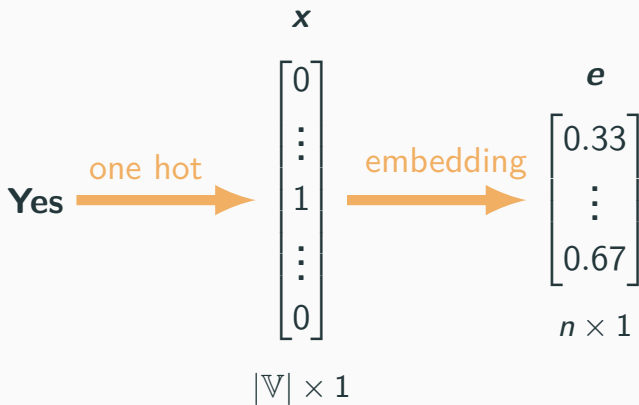
$$(\mathbf{x}^{(7)}, \mathbf{y}^{(7)}) = (\text{you, more})$$

...

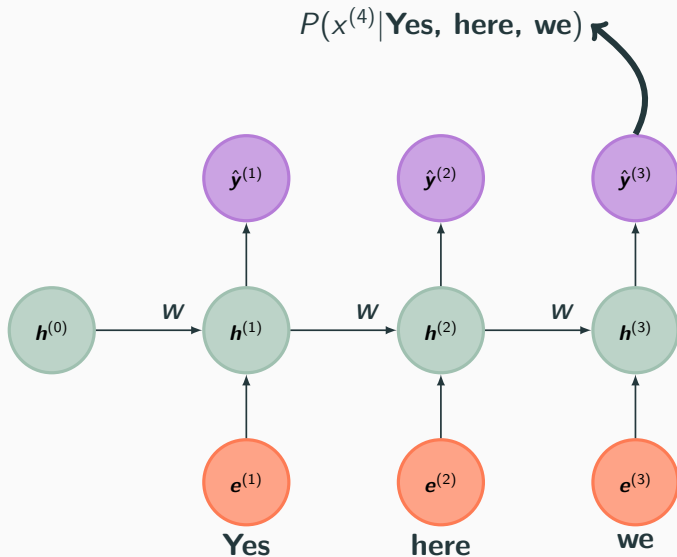
# O modelo de linguagem com RNN



# Word Embeddings



# O modelo de linguagem com RNN





# Dificuldades ao se treinar uma RNN

- Quando nos inicializamos  $\mathbf{W}$  de modo que  $\|\mathbf{W}\| < 1$ , os gradientes dos passos mais antigos vão sumir (**vanishing problem**).
- E quando  $\|\mathbf{W}\| > 1$ , os gradientes dos passos mais antigos vão explodir (**exploding problem**).
- Como resultado os gradientes dos passos mais próximos ao passo final vão ter mais influência do que os passos mais distantes. Isso é ruim para capturar **longas dependências**.

# Dificuldades ao se treinar uma RNN

- Quando nos inicializamos  $\mathbf{W}$  de modo que  $\|\mathbf{W}\| < 1$ , os gradientes dos passos mais antigos vão sumir (**vanishing problem**).
- E quando  $\|\mathbf{W}\| > 1$ , os gradientes dos passos mais antigos vão explodir (**exploding problem**).
- Como resultado os gradientes dos passos mais próximos ao passo final vão ter mais influência do que os passos mais distantes. Isso é ruim para capturar **longas dependências**.

Eu fui morar na **frança**, com uma \_\_\_\_\_.

# Dificuldades ao se treinar uma RNN

- Quando nos inicializamos  $\mathbf{W}$  de modo que  $\|\mathbf{W}\| < 1$ , os gradientes dos passos mais antigos vão sumir (**vanishing problem**).
- E quando  $\|\mathbf{W}\| > 1$ , os gradientes dos passos mais antigos vão explodir (**exploding problem**).
- Como resultado os gradientes dos passos mais próximos ao passo final vão ter mais influência do que os passos mais distantes. Isso é ruim para capturar **longas dependências**.

Eu fui morar na **frança**, com uma \_\_\_\_\_. (**francesa**)

# Dificuldades ao se treinar uma RNN

- Quando nos inicializamos  $\mathbf{W}$  de modo que  $\|\mathbf{W}\| < 1$ , os gradientes dos passos mais antigos vão sumir (**vanishing problem**).
- E quando  $\|\mathbf{W}\| > 1$ , os gradientes dos passos mais antigos vão explodir (**exploding problem**).
- Como resultado os gradientes dos passos mais próximos ao passo final vão ter mais influência do que os passos mais distantes. Isso é ruim para capturar **longas dependências**.

Eu fui morar na **frança**, com uma \_\_\_\_\_. (**francesa**)

Eu fui morar na **frança**, nesse tempo fiquei estudando a língua \_\_\_\_\_.

# Dificuldades ao se treinar uma RNN

- Quando nos inicializamos  $\mathbf{W}$  de modo que  $\|\mathbf{W}\| < 1$ , os gradientes dos passos mais antigos vão sumir (**vanishing problem**).
- E quando  $\|\mathbf{W}\| > 1$ , os gradientes dos passos mais antigos vão explodir (**exploding problem**).
- Como resultado os gradientes dos passos mais próximos ao passo final vão ter mais influência do que os passos mais distantes. Isso é ruim para capturar **longas dependências**.

Eu fui morar na **frança**, com uma \_\_\_\_\_. (**francesa**)

Eu fui morar na **frança**, nesse tempo fiquei estudando a língua \_\_\_\_\_. (**francesa**)

# Dificuldades ao se treinar uma RNN

- Quando nos inicializamos  $\mathbf{W}$  de modo que  $\|\mathbf{W}\| < 1$ , os gradientes dos passos mais antigos vão sumir (**vanishing problem**).
- E quando  $\|\mathbf{W}\| > 1$ , os gradientes dos passos mais antigos vão explodir (**exploding problem**).
- Como resultado os gradientes dos passos mais próximos ao passo final vão ter mais influência do que os passos mais distantes. Isso é ruim para capturar **longas dependências**.

Eu fui morar na **frança**, com uma \_\_\_\_\_. (**francesa**)

Eu fui morar na **frança**, nesse tempo fiquei estudando a língua \_\_\_\_\_. (**francesa**)

Eu fui morar na **frança** durante três anos e cinco meses com dois amigos, o Carlos e o Lucas. Foi bem legal, nesse tempo fiquei estudando a língua \_\_\_\_\_.

# Dificuldades ao se treinar uma RNN

- Quando nos inicializamos  $\mathbf{W}$  de modo que  $\|\mathbf{W}\| < 1$ , os gradientes dos passos mais antigos vão sumir (**vanishing problem**).
- E quando  $\|\mathbf{W}\| > 1$ , os gradientes dos passos mais antigos vão explodir (**exploding problem**).
- Como resultado os gradientes dos passos mais próximos ao passo final vão ter mais influência do que os passos mais distantes. Isso é ruim para capturar **longas dependências**.

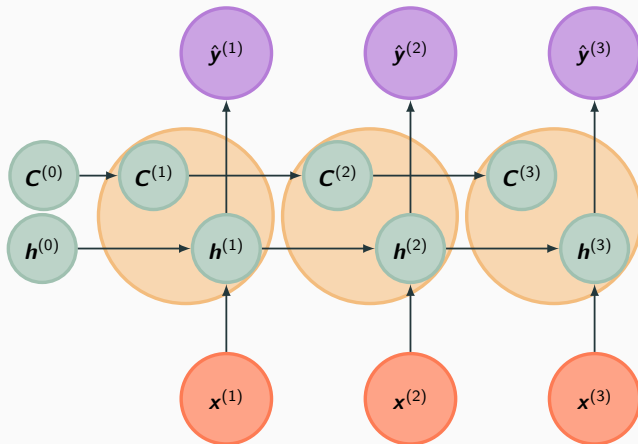
Eu fui morar na **frança**, com uma \_\_\_\_\_. (**francesa**)

Eu fui morar na **frança**, nesse tempo fiquei estudando a língua \_\_\_\_\_. (**francesa**)

Eu fui morar na **frança** durante três anos e cinco meses com dois amigos, o Carlos e o Lucas. Foi bem legal, nesse tempo fiquei estudando a língua \_\_\_\_\_. (**francesa**)

# Algumas soluções

- GRU (Gated Recurrent Unit)
- LSTM (Long Short-Term Memory).





# Exemplo de aplicação: TrumpBot

<https://github.com/felipessalvatore/MyTwitterBot>



**Felipe Salvatore**

@Felipessalvador

Hillary can make america great again.

[@greta](#) [@MarkBurnettTV](#)

[#DinheiroNãoCompra](#) [#SecretBallot](#)

[#خسوف\\_القمر](#)

Traduzir do inglês

15:10 - 7 de ago de 2017



**Felipe Salvatore**

@Felipessalvador

Obama is all beautiful. I agree with people attacking me. Amazing. [@CLewandowski\\_](#)

[#SecretBallot](#) [@garyplayer](#) [@greta](#)

Traduzir do inglês

14:40 - 7 de ago de 2017

## Exemplo de aplicação: SakaBot

<https://github.com/felipessalvatore/MyTwitterBot>



**Felipe Salvatore**

@Felipessalvador



Eduardo Cunha deve ser denunciado pelos frigoríficos ainda. Podem apostar no máximo  
[#AGoodDayIncludes](#) [#لعبه\\_مریم](#)

10:19 - 7 de ago de 2017



**Felipe Salvatore**

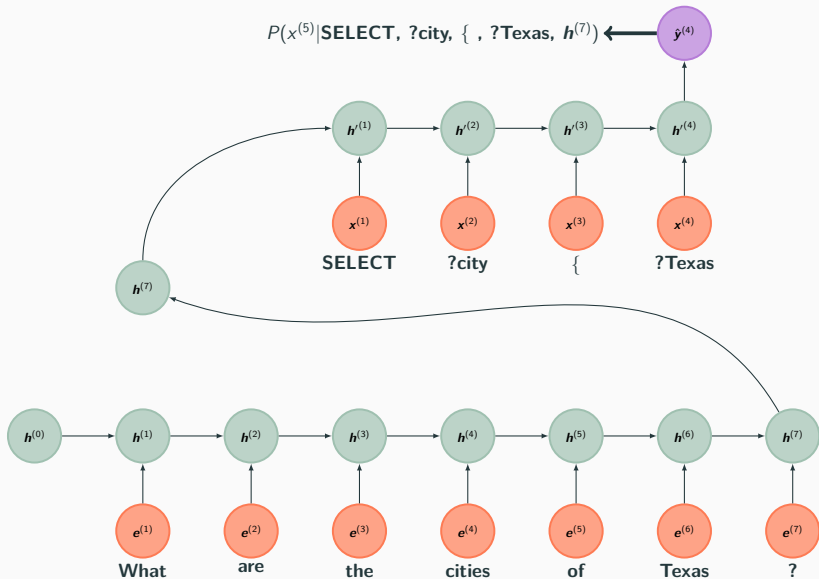
@Felipessalvador



Neymar é na verdade algo que o cara vomitou na rua. Lá ele se torna mais rico  
[#WannaOneDebut](#)  
[#العيسي\\_للطلاب\\_اشتكوني\\_للمظالم](#)

09:19 - 7 de ago de 2017

# Tradução automática: Inglês $\rightarrow$ SPARQL[3]



# LSTM e GRU são apenas o começo

**Perplexidade** é uma medida de quantas palavras diferentes igualmente prováveis podem seguir uma sequência de palavras (pior caso  $|\mathbb{V}|$ , melhor caso 1).

Olhando o corpus **Penn Treebank (PTB)** ( $|\mathbb{V}| = 10000$ ):

Model	Val	Test
Mikolov et al (2011)[4]	163.2	149.9
Zaremba et al (2014)[7]	82.62	78.29



Richard

@RichardSocher

Seguindo



When Zoph & Le at Google got 62 perplexity on PTB, I thought it'd be impossible to beat. Amazing progress in AI atm.

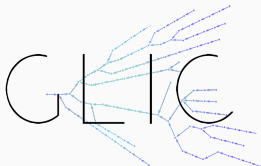
[arxiv.org/abs/1708.02182](https://arxiv.org/abs/1708.02182)

Traduzir do inglês

Model results over Penn Treebank (PTB)	Params	Val	Test
Grave et al. (2016) - LSTM	—	—	82.3
Grave et al. (2016) - LSTM + continuous cache pointer	—	—	72.1
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	24M	75.7	73.2
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	51M	71.1	68.5
Zilly et al. (2016) - Variational RHN (tied)	23M	67.9	65.4
Zoph & Le (2016) - NAS Cell (tied)	25M	—	64.0
Zoph & Le (2016) - NAS Cell (tied)	54M	—	62.4
Melis et al. (2017) - 4-layer skip connection LSTM (tied)	24M	60.9	58.3
AWD-LSTM - 3-layer LSTM (tied)	24M	<b>60.0</b>	<b>57.3</b>
AWD-LSTM - 3-layer LSTM (tied) + continuous cache pointer	24M	<b>53.9</b>	<b>52.8</b>

01:47 - 8 de ago de 2017

# Obrigado!



<https://glicusp.wordpress.com/>



<https://www.ime.usp.br/~liamf/>

# References I



metodosupera neuronios glossario do cerebro.

<http://metodosupera.com.br/neuronios-glossario-do-cerebro/>.



I. Goodfellow, Y. Bengio, and A. Courville.

***Deep Learning.***

MIT Press, 2017.



F. F. Luz and M. Finger.

**Semantic parsing natural language into sparql: an lstm encoder- decoder neural net approach.**

2017.



T. Mikolov, S. Kombrink, L. Burget, J. Cernocký, and S. Khudanpur.

**Extensions of recurrent neural network language.**

*IEEE*, pages 5528–5531, 2011.

## References II



M. T. Ribeiro, S. Singh, and C. Guestrin.

**"why should i trust you?": Explaining the predictions of any classifier.**

In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144, New York, NY, USA, 2016. ACM.



H. Xiao, K. Rasul, and R. Vollgraf.

**Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.**



W. Zaremba, I. Sutskever, and O. Vinyals.

**Recurrent neural network regularization.**

*CoRR*, abs/1409.2329, 2014.