

Aprendizado de Máquina sobre textos

1001513 – Aprendizado de Máquina 2
Turma A – 2023/2
Prof. Murilo Naldi



naldi@ufscar.br



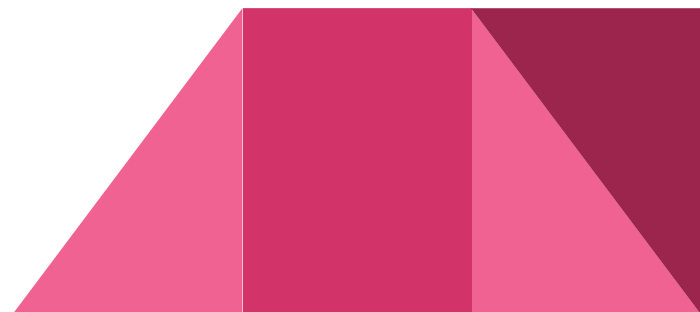
Agradecimentos

- Forte agradecimento as pessoas que colaboraram com a produção deste material: Prof. Helena Caseli e Diego Silva

Disclaimer

A aula de hoje é focada em processamento de texto,

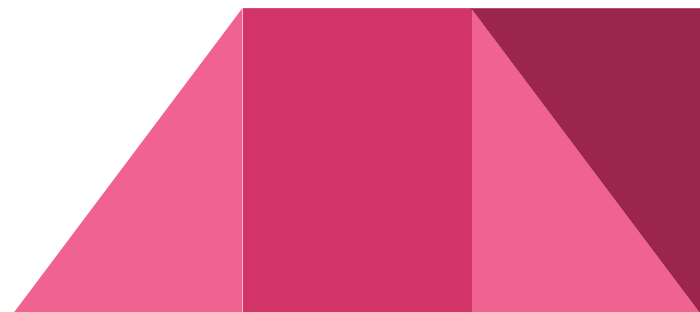
- Mas não caracteriza PLN
 - PLN foca em técnicas que permitam ao computador processar linguagem natural de forma semelhante aos humanos
 - Envolve linguística (estuda os fatos da linguagem)
 - Processar => entender, extrair conhecimento, comunicar-se, etc
- Nós iremos focar em texto



Dica

Uma dica importante para aula de hoje é o Natural Language Toolkit

- www.nltk.org/
- Fornece uma introdução prática à programação para processamento de linguagem.
- Possui livro escrito pelos criadores do NLTK
 - www.nltk.org/book/



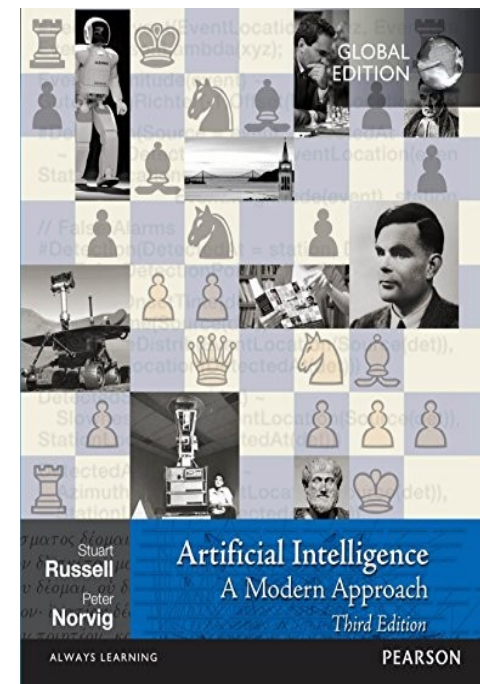
Modelo de linguagem

Vamos usar **Russell e Norvig** (capítulo 22.1)

“As linguagens formais, tais como as linguagens de programação Java ou Python, têm **modelos de linguagem** precisamente definidos.”

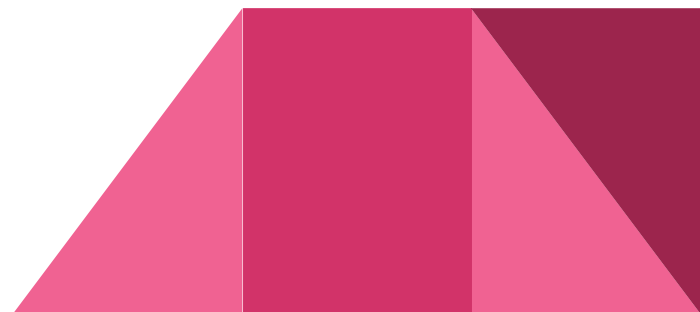
- Definimos regras gramaticais e semânticas

“o ‘significado’ de ‘ $2 + 2$ ’ é 4, e o significado de ‘1/0’ é que será sinalizado erro”



Modelo de linguagem

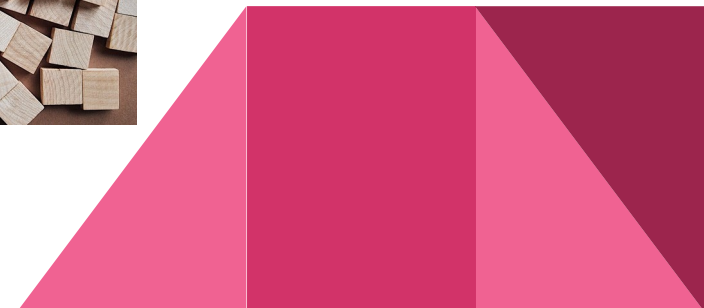
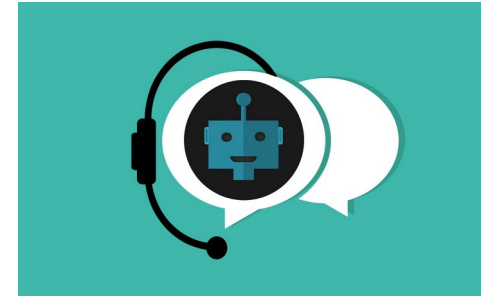
“Finalmente, as linguagens naturais são difíceis de lidar porque são muito grandes e em constante mutação. Assim, os **modelos de nossa língua são**, na melhor das hipóteses, **uma aproximação.**”



Aplicações

Várias como:

- Análise de sentimentos
- Chatbots
- Assistente de voz
- Tradução automática
- Detecção de *fake news*
- Outras...



Exemplo Clássico - Bag of Words (BoW)

Ideia geral é contabilizar as palavras em uma **matriz de frequência**

- Assim, cada **palavra** é um **atributo**
- Cada **sentença ou documento** é um **exemplo**

Documento	Abacate	Avião	Beterraba	Casa	Dados
1	213	0	35	0	0
2	18	0	123	0	0
3	0	0	0	0	0
4	0	7	0	3	7
5	0	2	0	5	15
6	14	0	0	17	0
7	0	0	12	0	0

Representação - Bag of Words (BoW)

Os valores $w(t_i, \mathbf{x})$ representam os termos de um documento baseado em sua ocorrência ou frequência

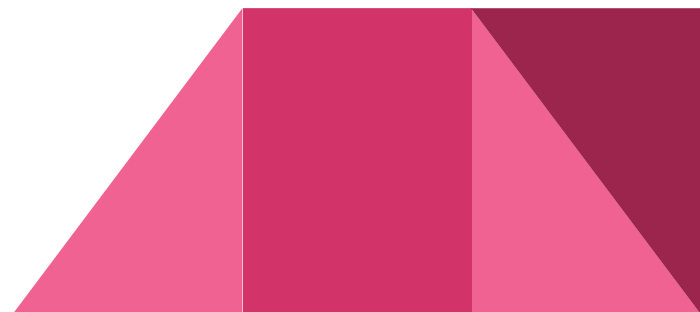
Tabela 23.2 Representação dos documentos usando o modelo espaço-vetorial

Documentos	t_1	t_2	t_3	...	t_d	Categoria
\mathbf{x}_1	$w(t_1, \mathbf{x}_1)$	$w(t_2, \mathbf{x}_1)$	$w(t_3, \mathbf{x}_1)$...	$w(t_d, \mathbf{x}_1)$	y_1
\mathbf{x}_2	$w(t_1, \mathbf{x}_2)$	$w(t_2, \mathbf{x}_2)$	$w(t_3, \mathbf{x}_2)$...	$w(t_d, \mathbf{x}_2)$	y_2
\mathbf{x}_3	$w(t_1, \mathbf{x}_3)$	$w(t_2, \mathbf{x}_3)$	$w(t_3, \mathbf{x}_3)$...	$w(t_d, \mathbf{x}_3)$	y_3
\vdots	\vdots	\vdots	\vdots	\ddots	\vdots	\vdots
\mathbf{x}_n	$w(t_1, \mathbf{x}_n)$	$w(t_2, \mathbf{x}_n)$	$w(t_3, \mathbf{x}_n)$...	$w(t_d, \mathbf{x}_n)$	y_n

Representação - Bag of Words (BoW)

O termo $w(t_i, \mathbf{x})$ pode ser calculado de várias formas:

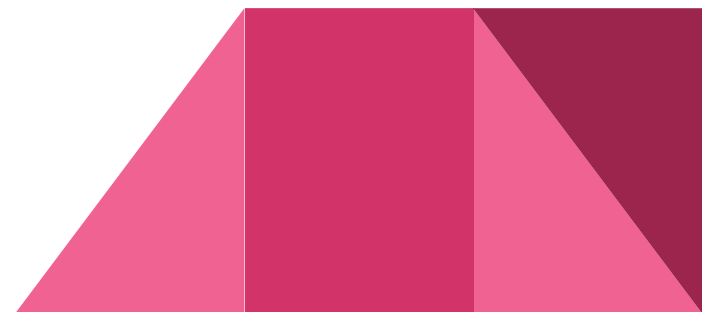
- Binária: neste tipo de representação, o termo recebe o valor um, caso apareça no documento, e zero, caso não apareça.



Representação - Bag of Words (BoW)

O termo $w(t_i, \mathbf{x})$ pode ser calculado de várias formas:

- Binária: neste tipo de representação, o termo recebe o valor um, caso apareça no documento, e zero, caso não apareça.
- Frequência do termo (TF – term frequency): os pesos dos termos correspondem à quantidade de vezes que o termo apareceu no documento.



Exemplo - Bag of Words (BoW)

Raw Text

**Bag-of-words
vector**

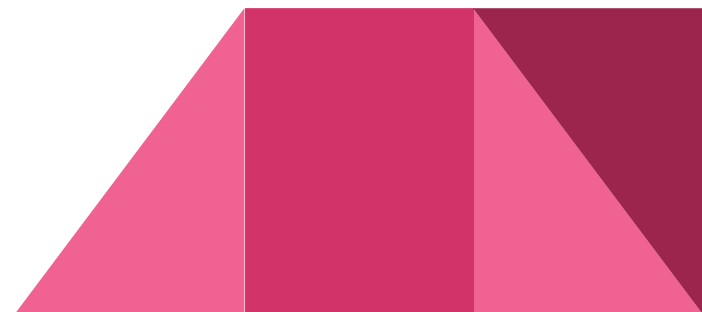
it is a puppy and it
is extremely cute

it	2
they	0
puppy	1
and	1
cat	0
aardvark	0
cute	1
extremely	1
...	...

Representação - Bag of Words (BoW)

Problemas:

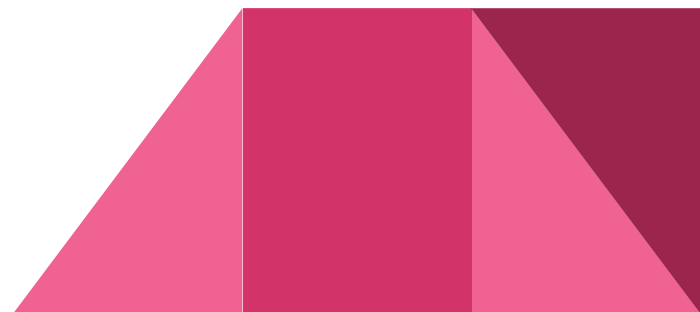
- Só constar que a palavra existe em um texto é pouco informativo
 - Perde características do texto
- Utilizar a frequência absoluta “privilegia” algumas palavras
 - Aparecem muito no texto



Exemplo - Bag of Words (BoW)

O termo $w(t_i, \mathbf{x})$ pode ser calculado de várias formas:

- Binária: neste tipo de representação, o termo recebe o valor um, caso apareça no documento, e zero, caso não apareça.
- Frequência do termo (TF – term frequency): os pesos dos termos correspondem à quantidade de vezes que o termo apareceu no documento.
- Frequência do termo-frequência inversa dos documentos (TF-IDF – term frequency-inverse document frequency): produto da frequência do termo pela frequência inversa dos documentos

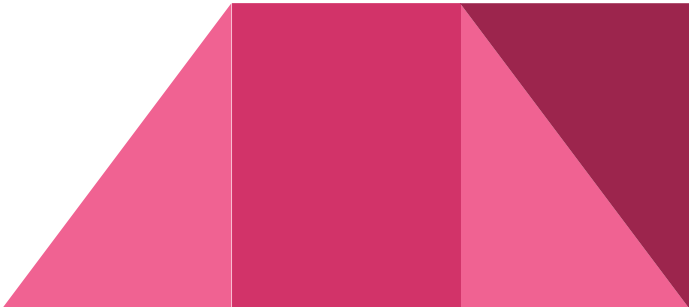


Representação - Bag of Words (BoW)

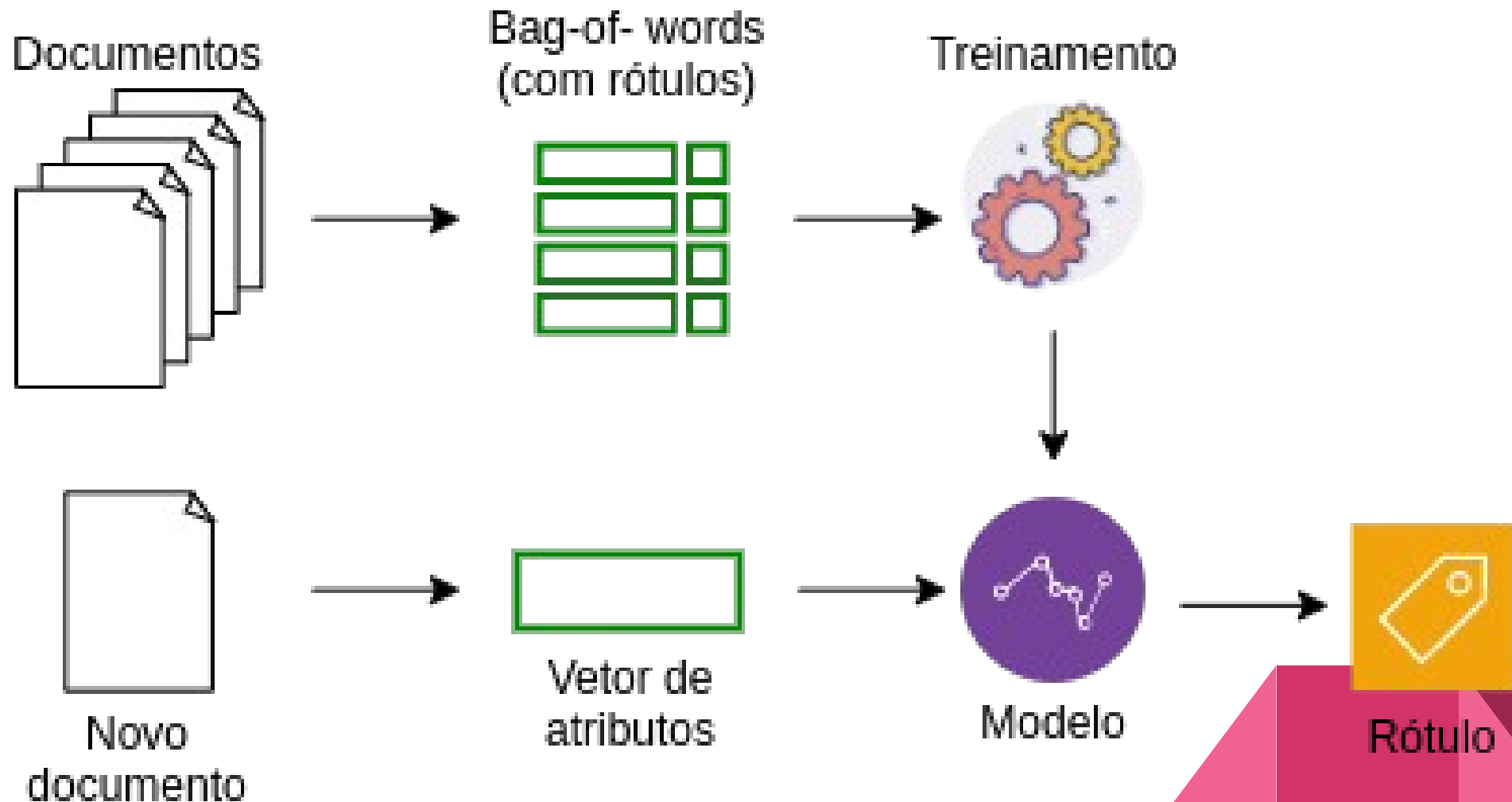
TF-IDF: term frequency-inverse document frequency

- tf é a frequência relativa do termo no documento
- idf é constante para o **corpus**
 - Inversamente proporcional ao número de documentos em que o termo aparece

$$tf(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \quad idf(t, D) = \log \frac{|D|}{|d \in D: t \in d|}$$

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D)$$


Bag of Words na classificação





problem?



Redução - Bag of Words (BoW)

Tratamento dos termos

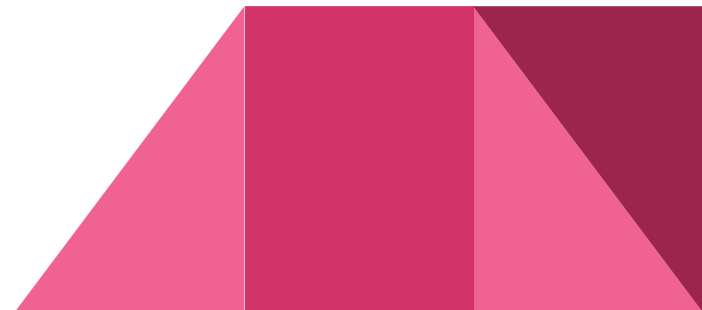
- Conversão dos termos para letras minúsculas



Redução - Bag of Words (BoW)


Tratamento dos termos

- Conversão dos termos para letras minúsculas
- Remoção de *stopwords*
 - Palavras que não agregam informação como “A”, “o”, “mas”, “então”, “se”, “um”, “uma”, ...
 - Podem ser específicas de domínio ou escolhidas em dicionário



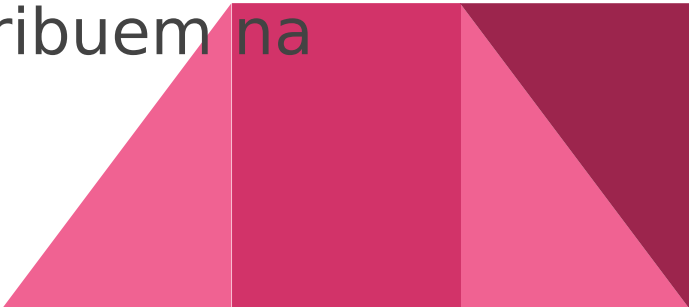
Redução - Bag of Words (BoW)

Tratamento dos termos

- Conversão dos termos para letras minúsculas
 - Remoção de *stopwords*
 - Palavras que não agregam informação como “A”, “o”, “mas”, “então”, “se”, “um”, “uma”, ...
 - Podem ser específicas de domínio ou escolhidas em dicionário
 - Remoção de palavras raras, com baixa frequência no conjunto de documentos que não contribuem na identificação da categoria
- 

Redução - Bag of Words (BoW)

Tratamento dos termos

- Conversão dos termos para letras minúsculas
 - Remoção de *stopwords*
 - Palavras que não agregam informação como “A”, “o”, “mas”, “então”, “se”, “um”, “uma”, ...
 - Podem ser específicas de domínio ou escolhidas em dicionário
 - Remoção de palavras raras, com baixa frequência no conjunto de documentos que não contribuem na identificação da categoria
 - Estemização e Lematização
- 

Estemização

Reduz o termo ao seu radical e varia com o idioma

- Para português, um dos algoritmos mais utilizados na é o RSLP
- Os algoritmos de estemização geralmente removem os afixos e o final das palavras

Palavra original	Estemização
Palavras da língua inglesa	
<i>studies</i>	<i>studi</i>
<i>university</i>	<i>univers</i>
<i>walking</i>	<i>walk</i>
Palavras da língua portuguesa	
empobrecendo	empobrec
felicíssimo	felic
segurados	segur

Lematização

Reduz o termo à sua forma canônica (lema)

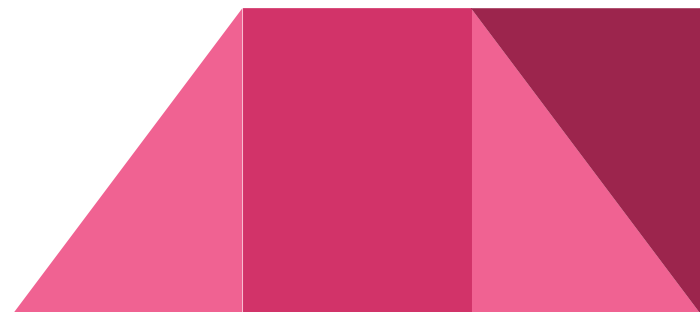
- Usa análise morfológica
- Mais complexo porque leva em consideração a semântica

Palavra original	Lematização
<i>studies</i>	<i>study</i>
<i>university</i>	<i>university</i>
<i>walking</i>	<i>walk</i>
empobrecendo	empobrecer
felicíssimo	feliz
segurados	segurar

Termos compostos - Bag of Words (BoW)

Termos compostos

- “Machine learning” vs “machine” & “learning”
- **n-gramas**
 - Alguns conceitos necessitam de mais de uma palavra para fazer sentido
 - Algumas palavras precisam de outras palavras para definir significado (semântica)
 - Exemplo: sinônimos, contexto

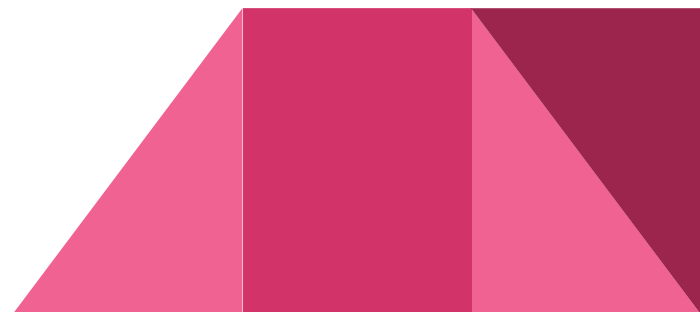


Variedade - Bag of Words (BoW)

Palavras diferentes sempre viram atributos separados

- “O banqueiro perdeu dinheiro”
- “A banqueira perdeu dinheiro”
- “O dono do banco perdeu dinheiro”

Banco = banqueiro = banqueira = “Banco”
(apesar da ambiguidade, que é outro problema)

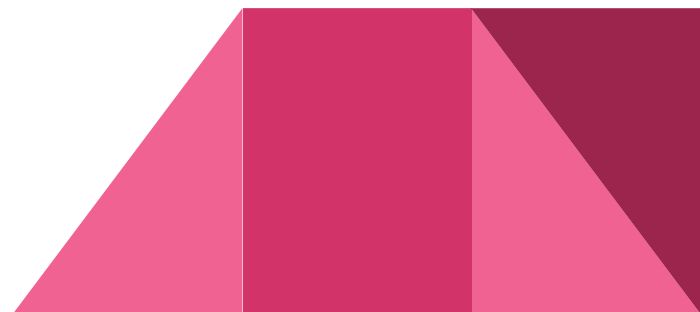


Significado - Bag of Words (BoW)

Mas e o caso “Paris” vs “Berlim”?

Elas são tão diferentes quanto “abobrinha” e “bicicleta”?

E se eu estou desprezando a ordem das palavras, onde fica a sintática? E a semântica?



Ordem das palavras

O que define a ordem das palavras?

- Cada macaco no seu ____? ____
- Macaco cada no galho seu X Cada macaco no seu galho

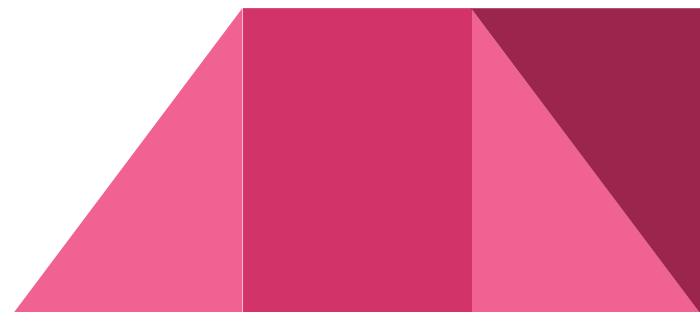
Como ensinar o computador a definir qual é a próxima palavra ou a ordem correta delas?



Ordem das palavras

Geralmente definida com base na probabilidade de ocorrência

- Cada macaco no seu galho
 - $P(\text{galho} \mid \text{cada macaco no seu})$
- Macaco cada no galho seu X Cada macaco no seu galho
 - $P(\text{macaco cada no galho seu}) < P(\text{cada macaco no seu galho})$

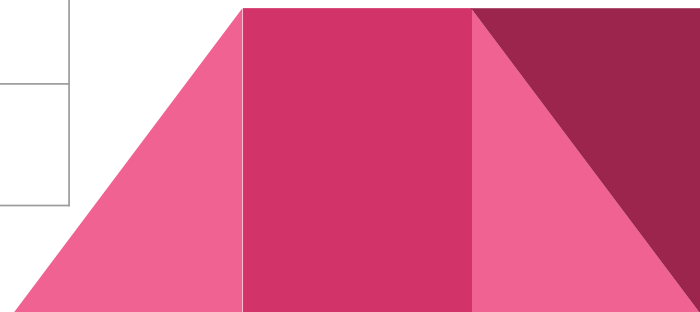


Ordem das palavras

Aplicações de geração de texto

- Sequência já gerada: Cada macaco no seu _____

Próxima palavra	Probabilidade
galinheiro	0,0003
anzol	0,000002
galho	0,004
toca	0,00005



Ordem das palavras

Aplicações de tradução

- Probabilidade de uma sentença

Possíveis sentenças	Probabilidade
Macaco cada no galho seu	0,00003
Cada macaco no seu galho	0,0005
Galho no seu cada macaco	0,000004

Ordem das palavras

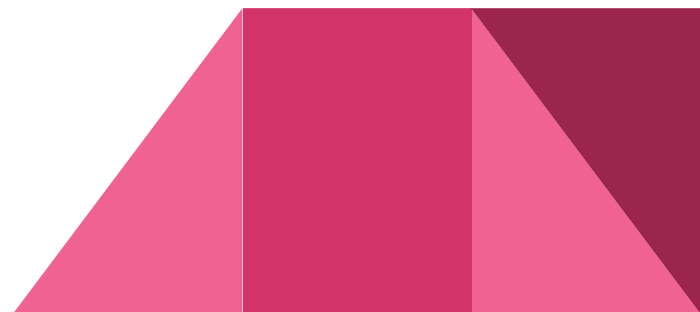
Aplicações de correção

- Probabilidade de uma sentença corrigida
 - Sentença original: saudade corta como aço de navaia
 - Opções de correção: navalha ou navais

Possíveis sentenças	Probabilidade
saudade corta como aço de navalha	0,005
saudade corta como aço de navais	0,00002

Modelo de linguagem

- Modelo computacional que infere a probabilidade de uma sequência de palavras
- Usado para prever
 - A próxima palavra dada uma sequência
 $P(w_5 \mid w_1, w_2, w_3, w_4)$
 - A ocorrência de uma sequência de palavras
 $P(w_1, w_2, w_3, w_4, w_5)$



Modelo de linguagem

Regra da cadeia

- A probabilidade de uma sentença pode ser estimada pela multiplicação das probabilidades de cada palavra, estimada com base nas palavras anteriores
 - $P(w_1, w_2, w_3, w_4, w_5) = P(w_1) \times P(w_2|w_1) \times P(w_3|w_1, w_2) \times P(w_4|w_1, w_2, w_3) \times P(w_5|w_1, w_2, w_3, w_4)$
 - $P(\text{Cada macaco no seu galho}) = P(\text{Cada}) \times P(\text{macaco}|\text{Cada}) \times P(\text{no}|\text{Cada macaco}) \times P(\text{seu}|\text{Cada macaco no}) \times P(\text{galho}|\text{Cada macaco no seu})$

Modelo de linguagem

- **Estimando probabilidades**

- A partir de um grande corpus
- Com base nas contagens das sequências

$$P(\text{galho}|\text{Cada macaco no seu}) = \frac{\text{freq}(\text{Cada macaco no seu galho})}{\text{freq}(\text{Cada macaco no seu})}$$

- Mas não é necessário “olhar” para toda a sequência

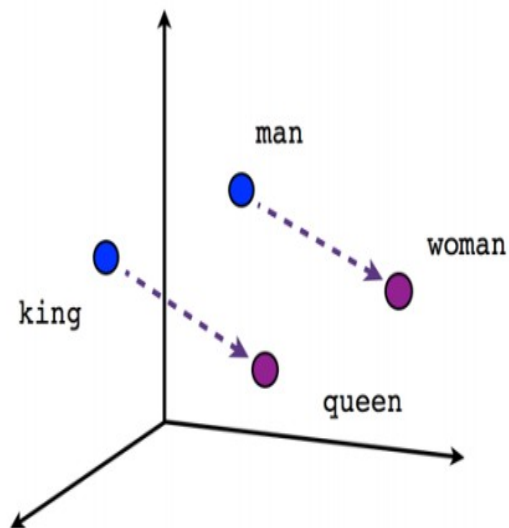
$$P(<s> \text{Cada macaco no seu galho} </s>) = P(\text{Cada} | <s>) \times P(\text{macaco} | \text{Cada}) \times P(\text{no} | \text{Cada macaco}) \times P(\text{seu} | \text{Cada macaco no}) \times P(\text{galho} | \text{Cada macaco no seu}) \times P(</s> | \text{Cada macaco no seu galho})$$

Word Embeddings

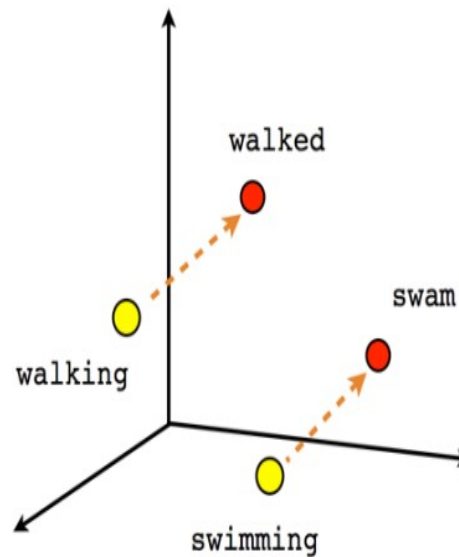
A ideia é **transformar uma palavra em um ponto em um novo espaço**, de tal forma que palavras relacionadas se encontram próximas uma da outra

- Usa **Seq2Vec** para estimar o *embedding*
 - $\mathbb{R}^W \rightarrow \mathbb{R}^E$
- Palavras **semanticamente similares** devem estar próximas no novo espaço
- No espaço transformado é desejável conseguir **fazer operações** do tipo (“França” - “Paris”) + “Berlim” \approx “Alemanha”

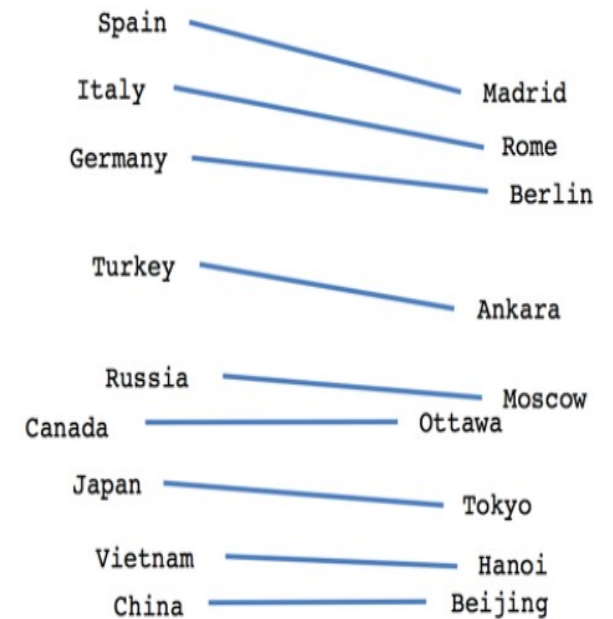
Word Embeddings



Male-Female



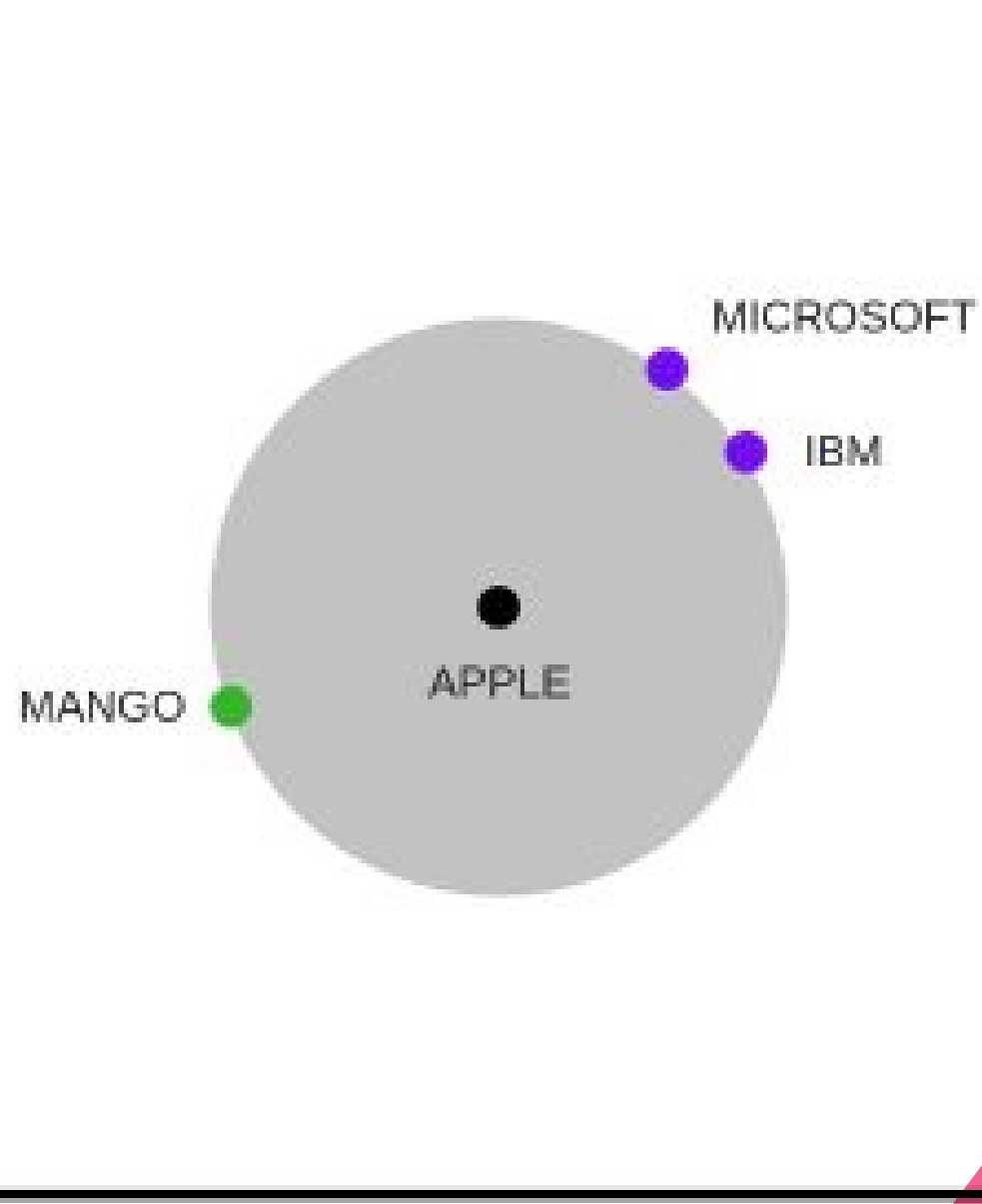
Verb tense



Country-Capital

$v(\text{king}) + v(\text{woman}) - v(\text{man}) \sim v(\text{queen})$
 $v(\text{king}) - v(\text{royal}) \sim v(\text{man})$
 $v(\text{queen}) - v(\text{royal}) \sim v(\text{woman})$

Word Embeddings



Word Embeddings

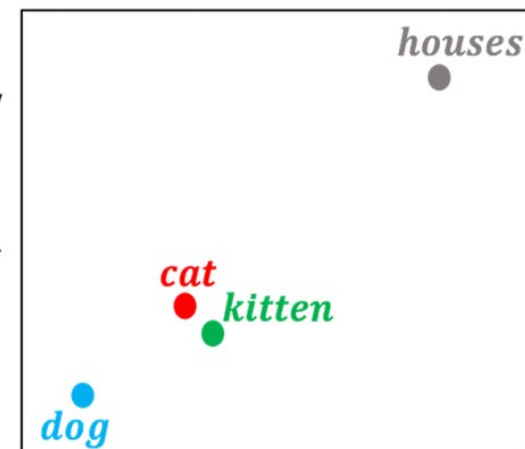
Representações
vetoriais densas
(valores
diferentes de
zero)

A dimensão é
fixa (p. ex. 300)

Fonte:
<https://medium.com/mlearning-ai/word-embeddings-how-do-organizations-use-them-for-building-recommendation-systems-e0341cf5e638>

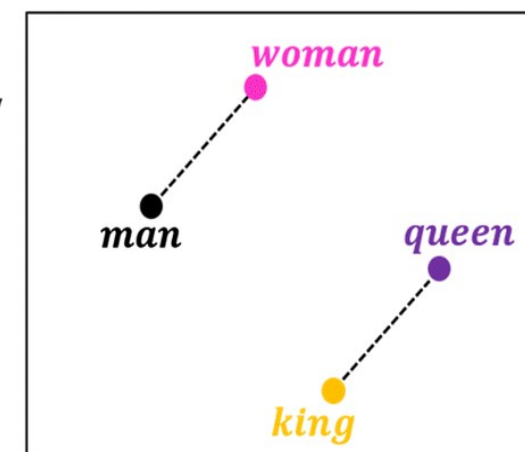
	living being	feline	human	gender	royalty	verb	plural
<i>cat</i> →	0.6	0.9	0.1	0.4	-0.7	-0.3	-0.2
<i>kitten</i> →	0.5	0.8	-0.1	0.2	-0.6	-0.5	-0.1
<i>dog</i> →	0.7	-0.1	0.4	0.3	-0.4	-0.1	-0.3
<i>houses</i> →	-0.8	-0.4	-0.5	0.1	-0.9	0.3	0.8

Dimensionality
reduction of
word
embeddings
from 7D to 2D
→



<i>man</i> →	0.6	-0.2	0.8	0.9	-0.1	-0.9	-0.7
<i>woman</i> →	0.7	0.3	0.9	-0.7	0.1	-0.5	-0.4
<i>king</i> →	0.5	-0.4	0.7	0.8	0.9	-0.7	-0.6
<i>queen</i> →	0.8	-0.1	0.8	-0.9	0.8	-0.5	-0.9

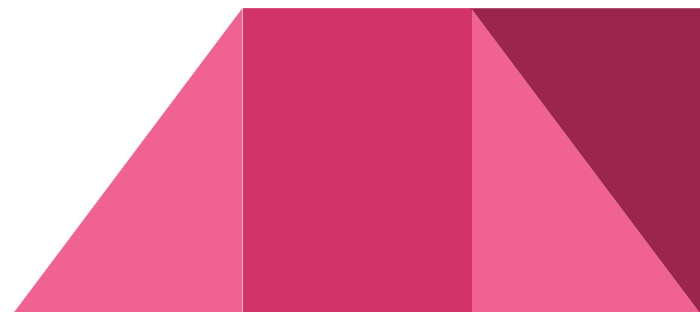
Dimensionality
reduction of
word
embeddings
from 7D to 2D
→



Embedding Models

Existem diferentes modelos comumente utilizados para gera *embeddings*, por exemplo:

- Word2Vec
- FastText
- Wang2Vec
- Glove



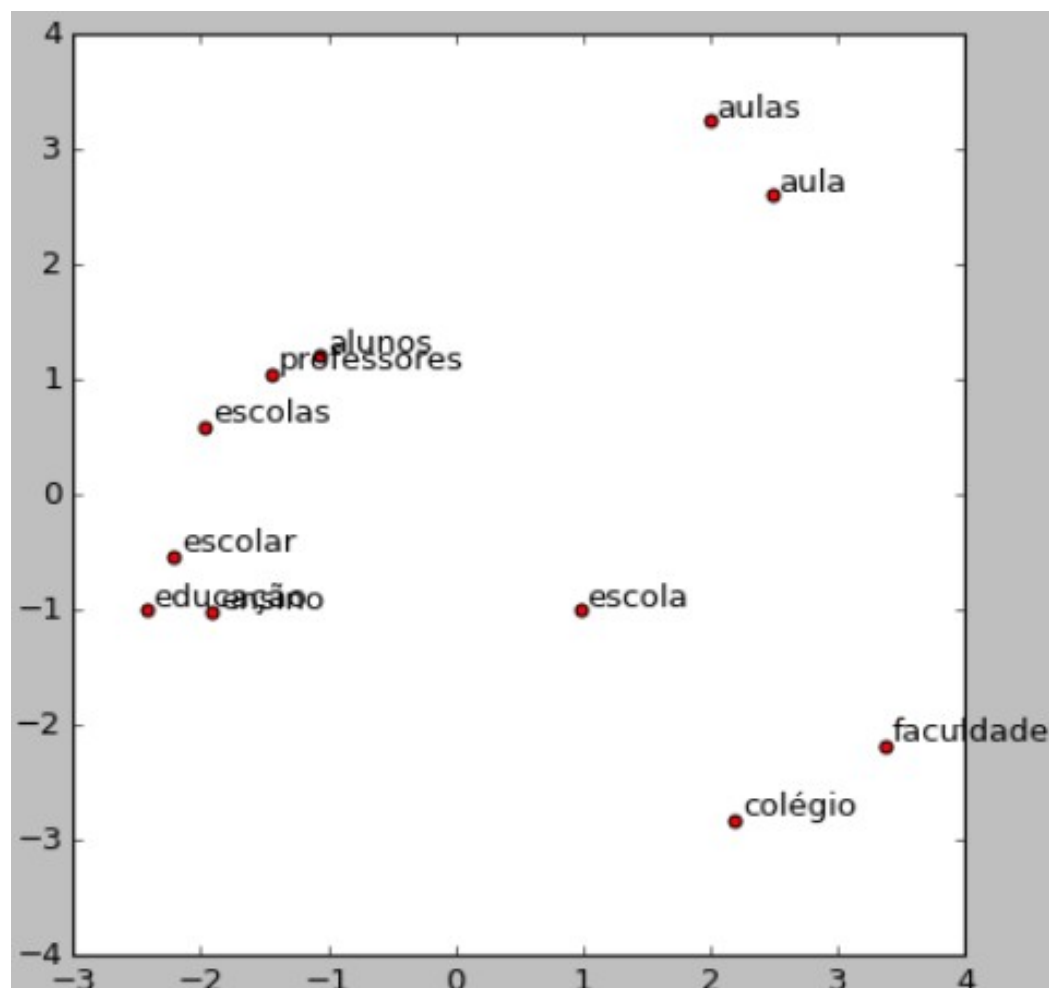
Word Embeddings

Para o português:

*NILC word
embeddings:*

[nilc.icmc.usp.br/
embeddings](http://nilc.icmc.usp.br/embeddings)

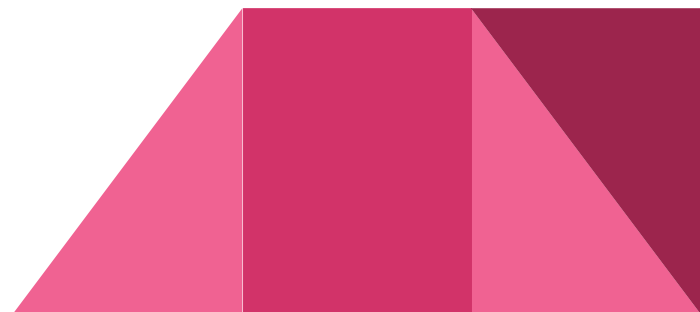
```
[('alunos', 0.6777069568634033),  
 ('escolas', 0.6750764846801758),  
 ('ensino', 0.6746147871017456),  
 ('colégio', 0.673301100730896),  
 ('faculdade', 0.6269355416297913),  
 ('aula', 0.6062946319580078),  
 ('aulas', 0.6045212745666504),  
 ('educação', 0.6030406355857849),  
 ('professores', 0.5954739451408386),  
 ('escolar', 0.5880724191665649)]
```



Word2Vec - Arquiteturas

Veremos dois tipos de arquiteturas que utilizam *Word2Vec* para aprender um vetor por palavra do *corpus*

- Continuous Bag of Words (CBOW)
- Skip-gram



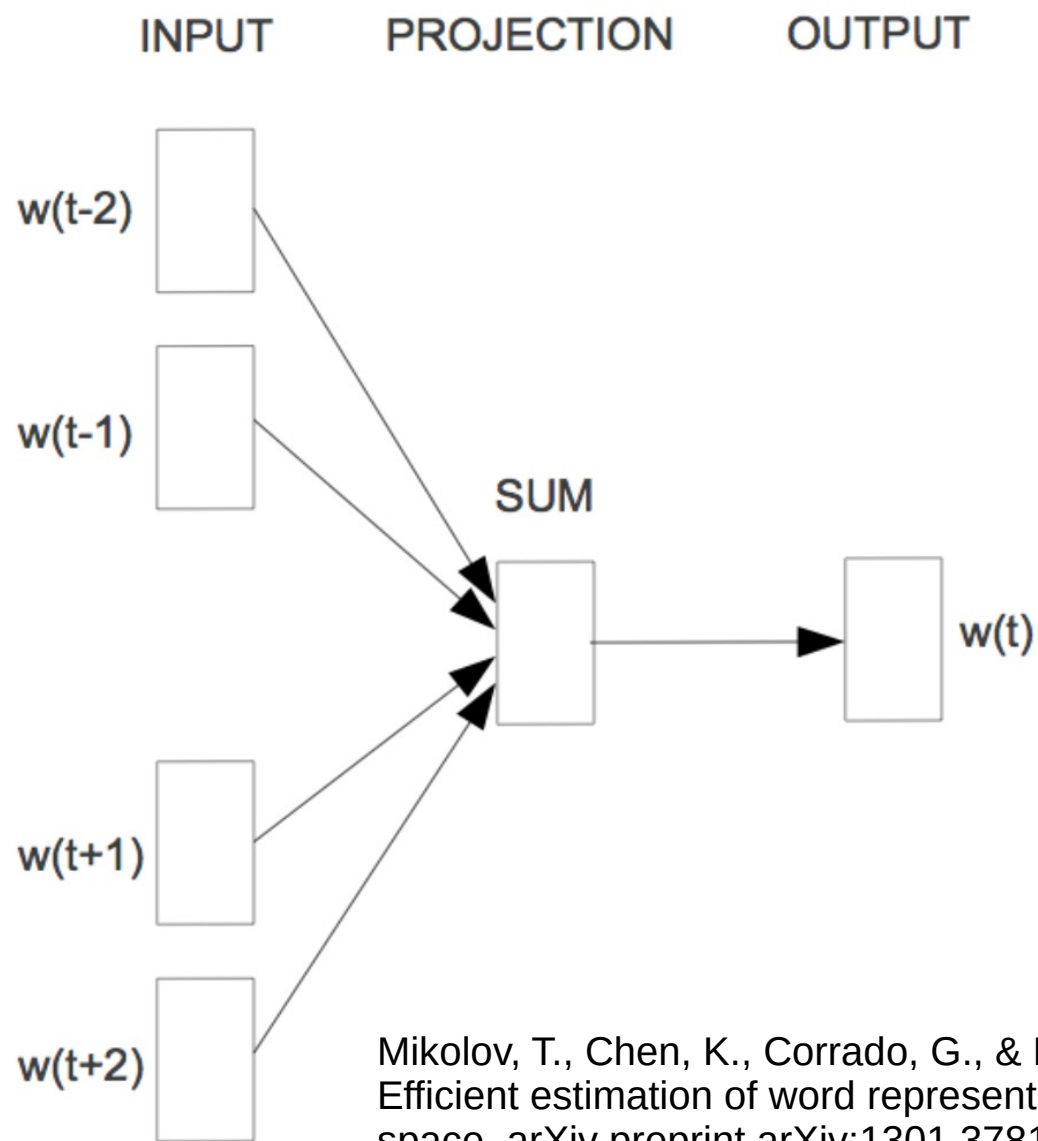
Continuous Bag of Words (CBOW)

Muito similar a uma rede neural feed-forward, que tenta prever uma palavra alvo de uma lista de palavras que consigam expressar contexto

- A intuição por trás desse modelo é simples:
 - Considerando a frase “cada macaco no seu galho”, escolhemos a palavra “macaco” como atributo alvo e o contexto será [“cada”, “no”, “seu”, “galho”]



Word Embeddings

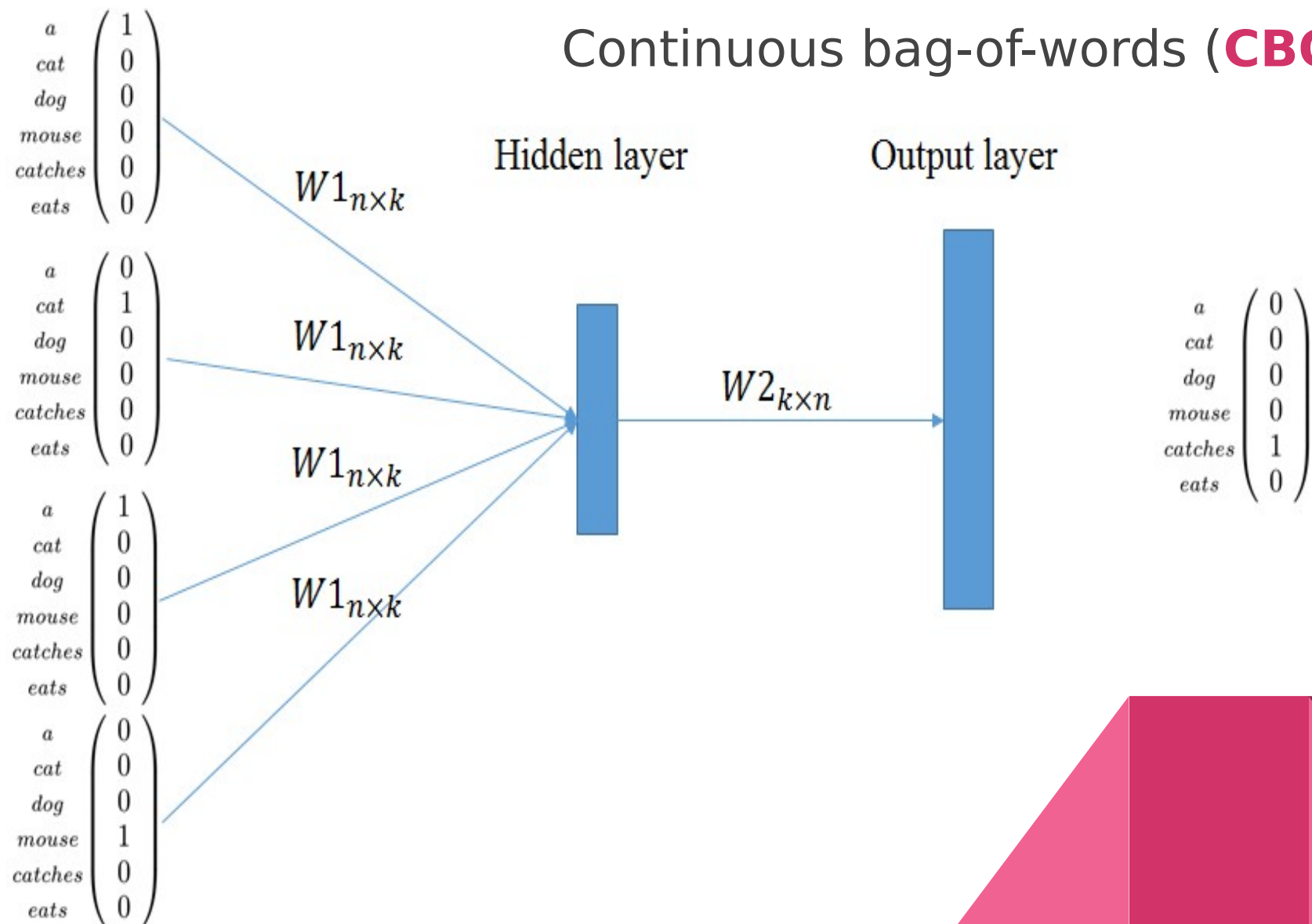


Continuous
bag-of-words (**CBOW**)

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

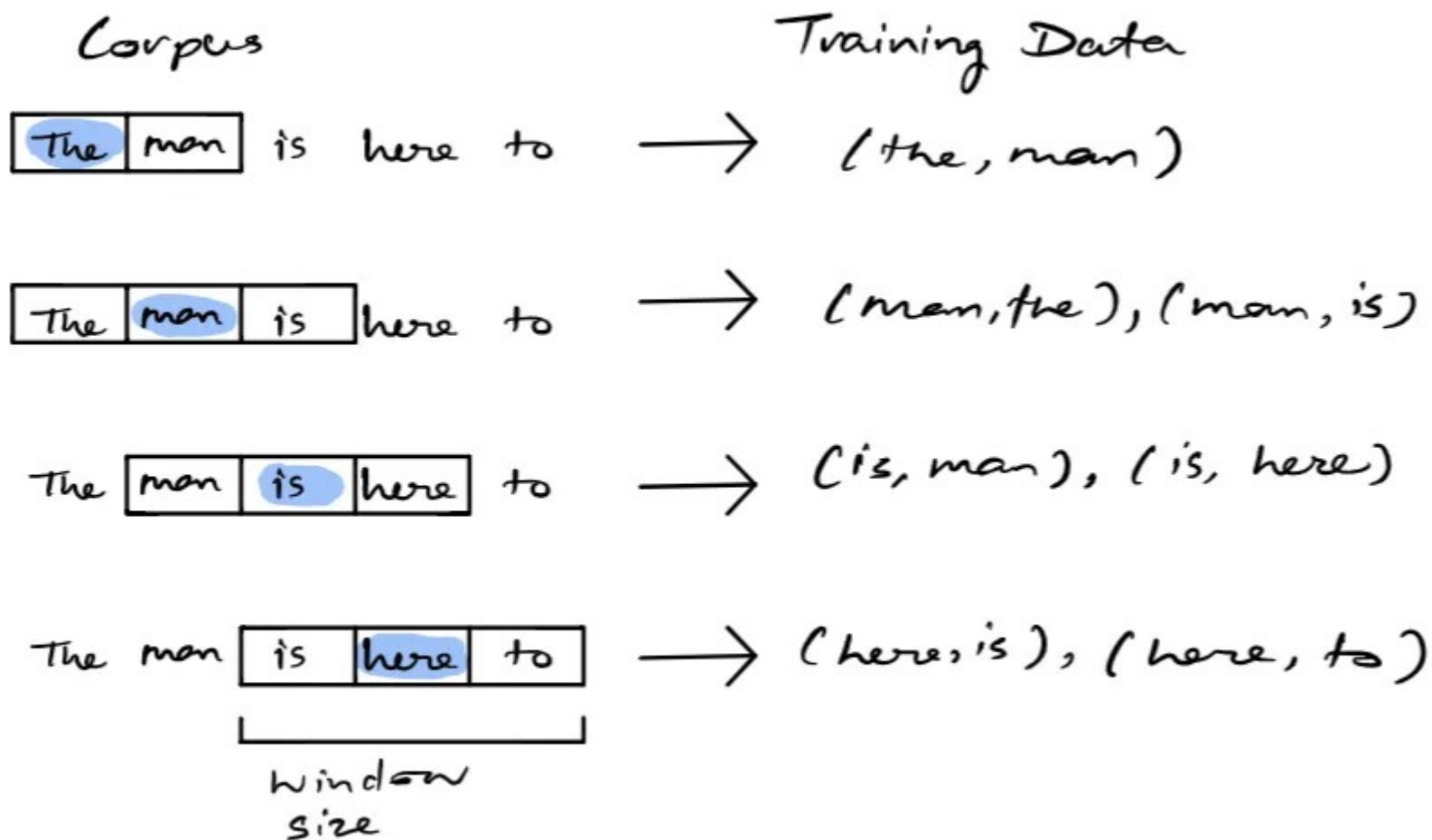
Word Embeddings

Continuous bag-of-words (**CBOW**)



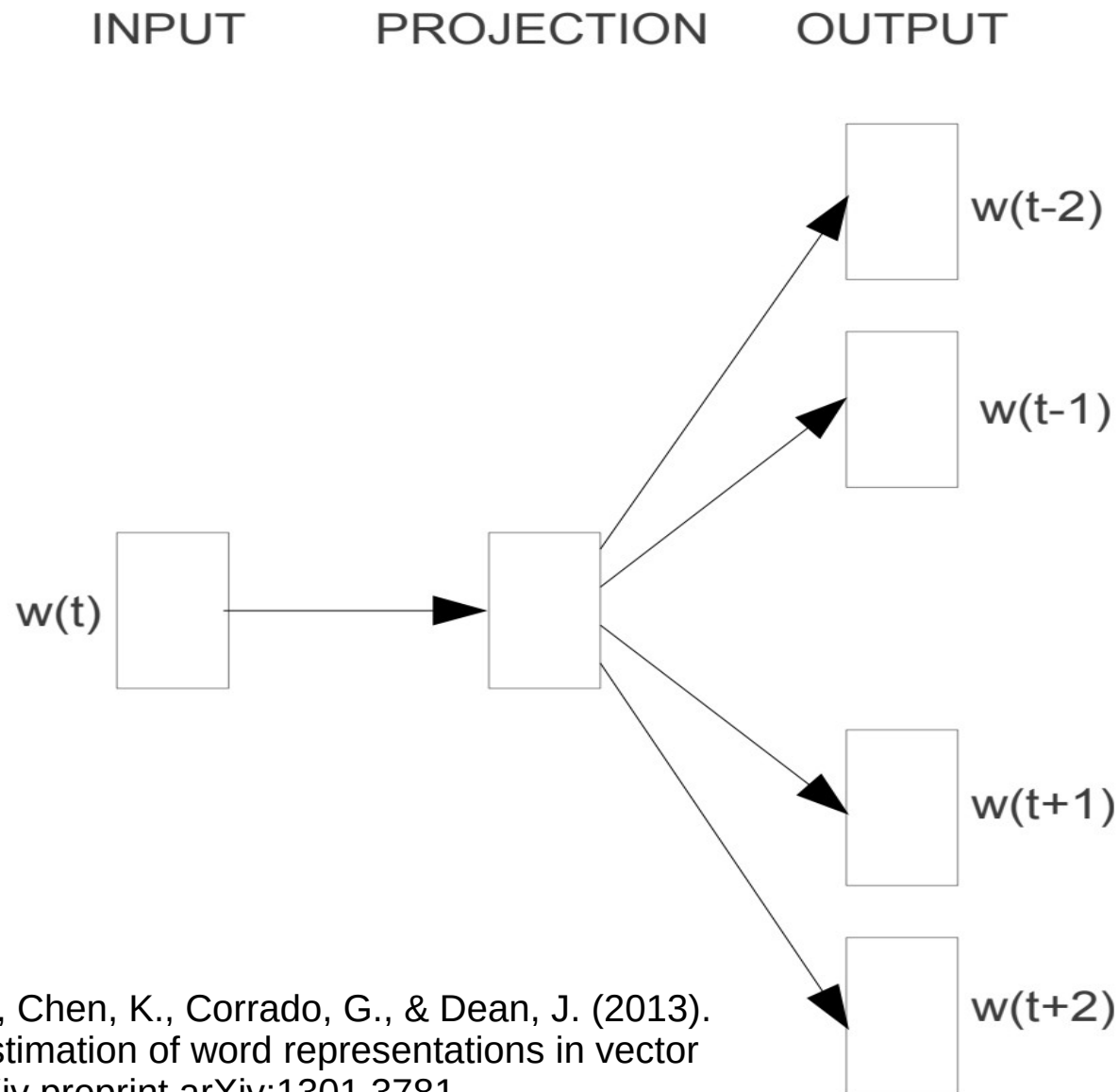
Continuous Bag of Words (CBOW)

Rede neural simples com uma camada intermediária treinada para prever a probabilidade de palavras estarem no contexto de uma dada palavra



Vatsal 2021.
Word2Vec
Explained.
towardsdatascience.com/word2vec-explained-49c52b4ccb71

Word Embeddings

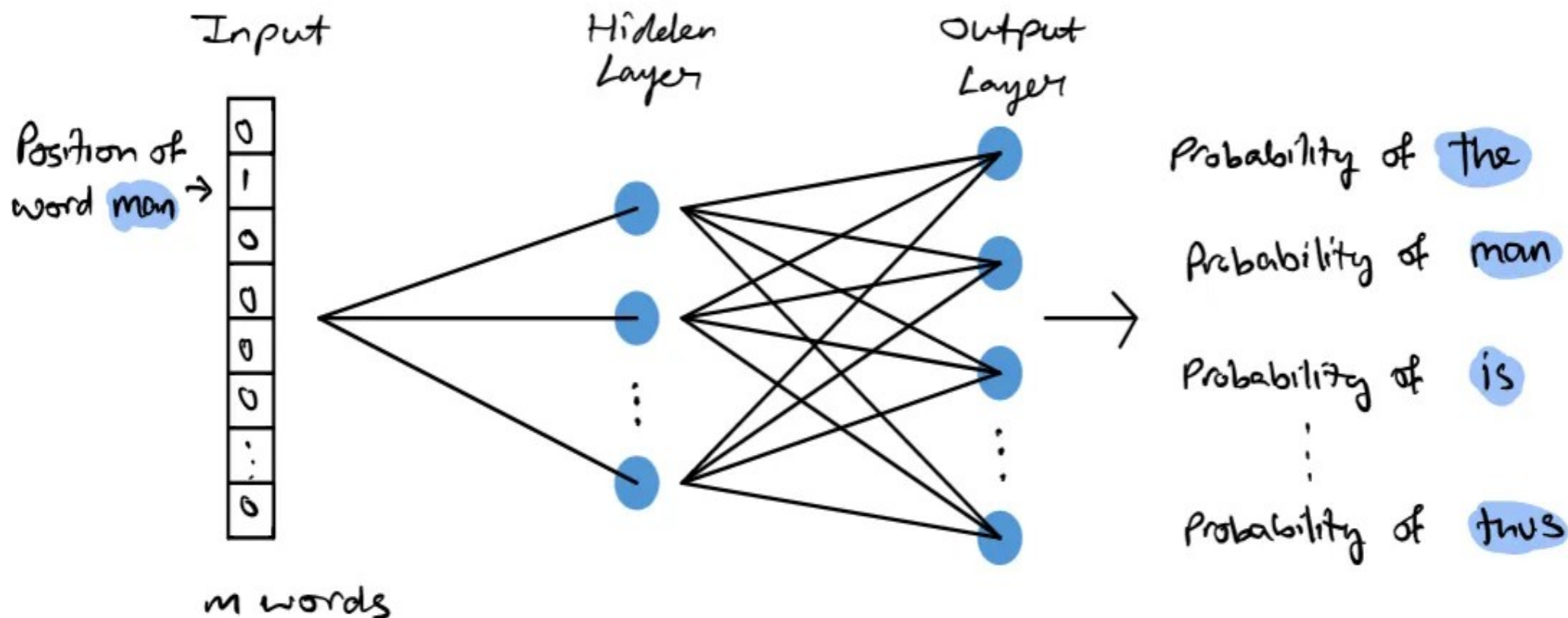


Skip-gram

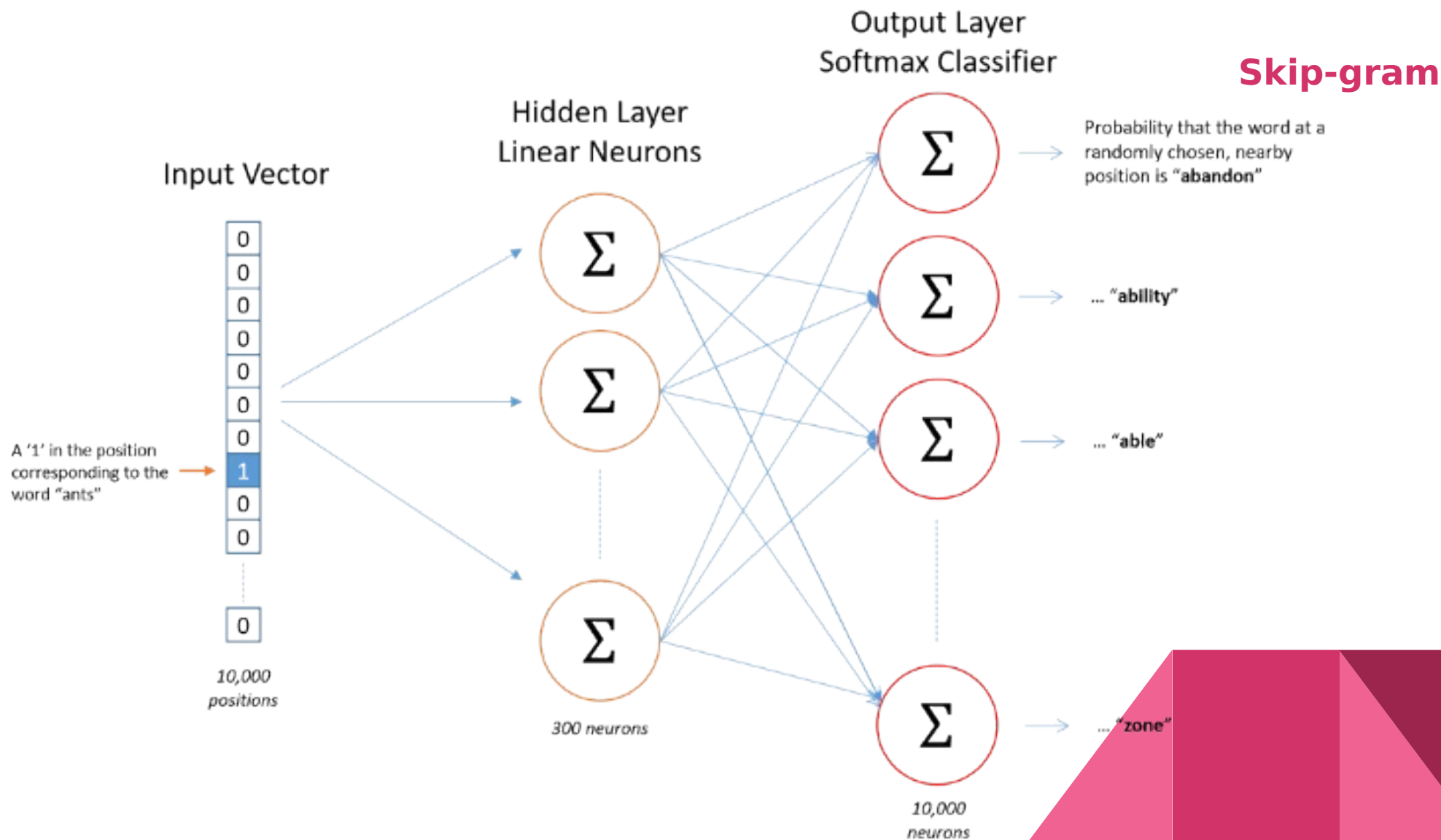
Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Word Embeddings

Skip-gram



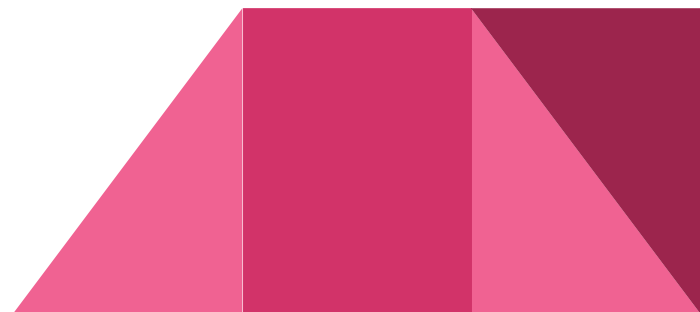
Word Embeddings



Word Embeddings

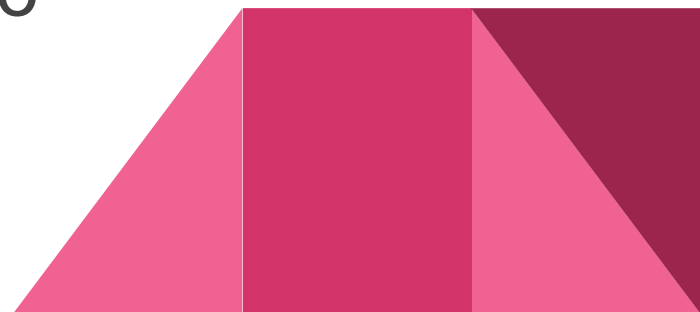
Observações

- Como usar para classificação?
- Modelos pré-treinados
 - <http://nilc.icmc.usp.br/nilc/index.php/repositorio-de-word-embeddings-do-nilc>
- Outras línguas e modelos multilíngues
- Especificidades de domínio



Literatura é extensa...

- Como dissemos, existem outros modelos como FastText, Wang2Vec, Glove, etc...
 - No geral, são extensões do modelos word2vec
 - Por exemplo, Glove incorpora estatísticas globais de co-ocorrência dos termos para obter os vetores
 - Estatística globais consideram a co-ocorrência de termos dado um *corpus*

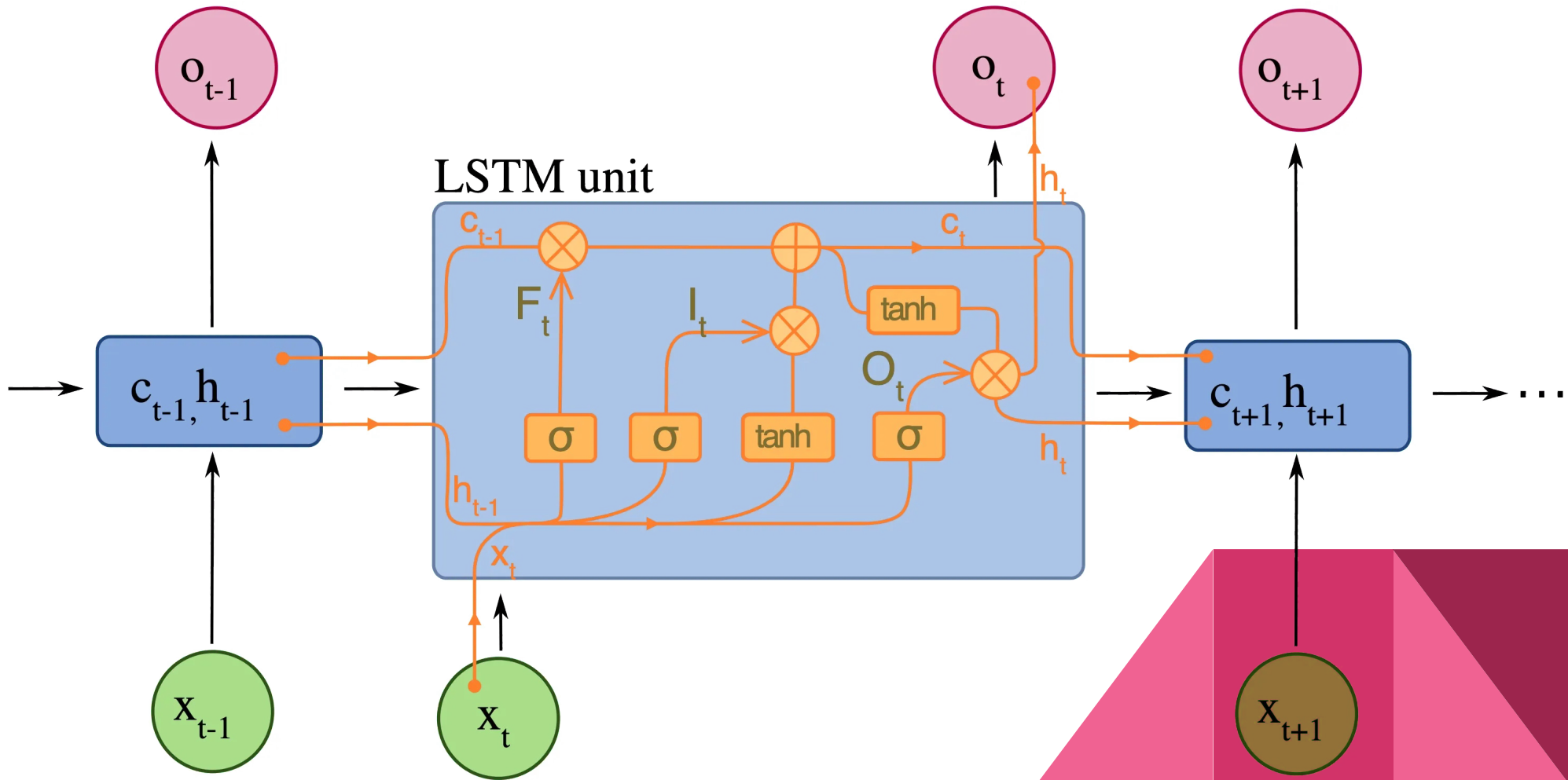


ELMo - *Embeddings from Language Models*

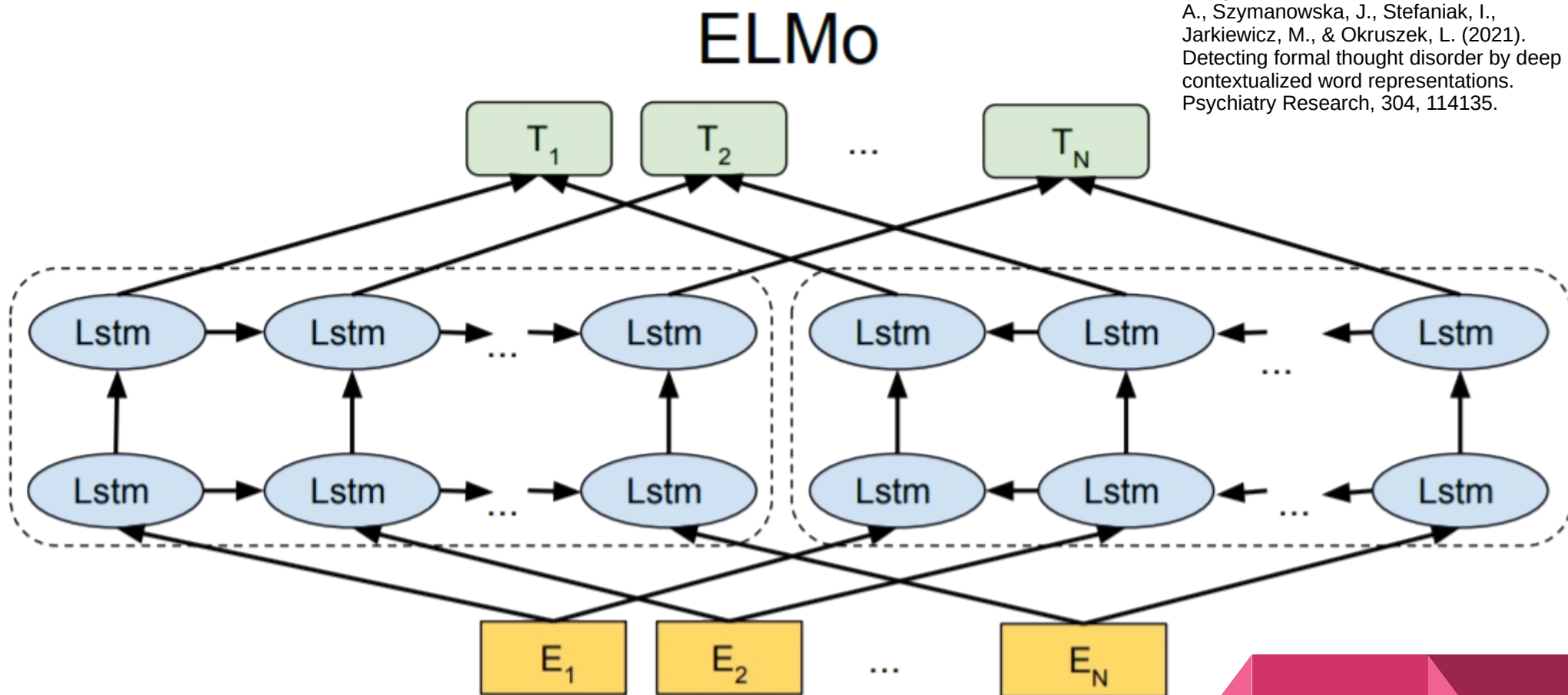
- Modelo que usa redes neurais profundas para gera representações vetoriais contextualizadas
 - Modela características complexas do uso da palavra alvo (e.g. sintaxe e semântica)
 - Modela também como esse uso varia entre múltiplos contextos linguísticos (i.e., para modelar polissemia)
 - Exemplo: prato: 'vasilha', 'comida', 'iguaria', 'receptáculo de balança', 'instrumento musical' etc

Long Short Term Memory (LSTM) networks

- Redes feitas para evitar dependência a longo prazo, capazes de passar informações passadas



Outros Modelos - ELMo



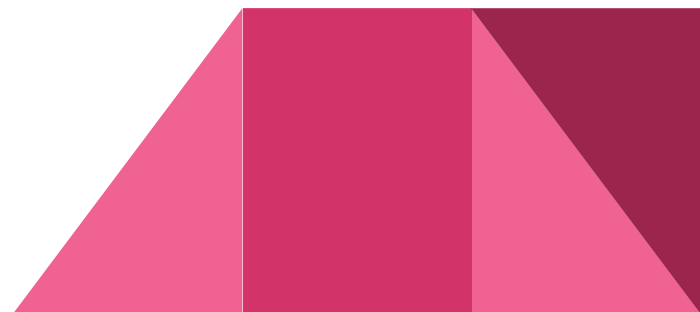
Sarzynska-Wawer, J., Wawer, A., Pawlak, A., Szymanowska, J., Stefaniak, I., Jarkiewicz, M., & Okruszek, L. (2021). Detecting formal thought disorder by deep contextualized word representations. *Psychiatry Research*, 304, 114135.

Long short-term memory (LSTM) são redes recorrentes feitas especialmente para reconhecer sequências de entradas

Exemplo ELMo

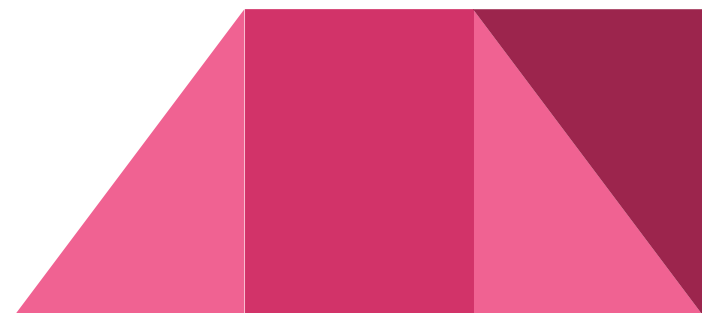
“Jack while talking over the **cell** phone entered the prison **cell** to extract blood **cell** samples of Jill and made an entry in the excel **cell** about the blood sample collection.”

- Como saber o significado de “**cell**”?
 - LSTM pode modelar contexto



Exemplo ELMo

- Primeiras duas camadas são biLSTMs com convoluções de letras
 - Calcula probabilidades de *tokens* nos dois sentidos
 - Com uma camada de representação de *token* e uma *softmax* de saída



Arquiteturas *encoder-decoder*

Auto-encoder

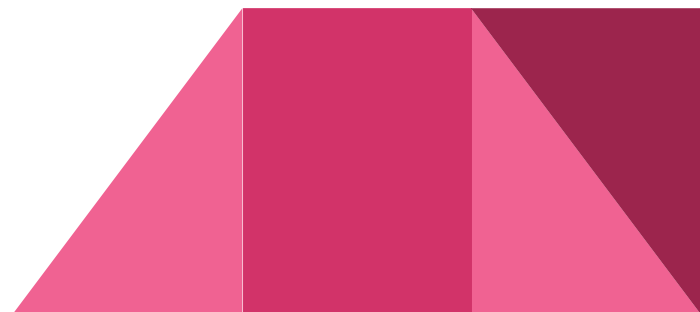
- Usado para aprender códigos de dados não rotulados (não supervisionado)
- Mapear sequência para um código
 - Decodificar o código em sequência



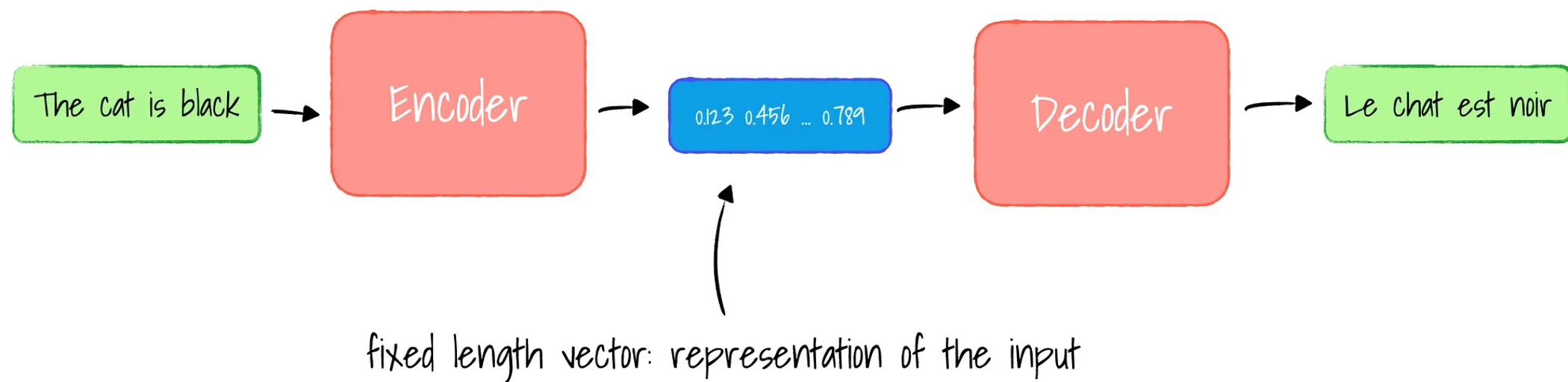
Arquiteturas encoder-decoder

Aplicações de Seq2Seq de tamanhos diferentes

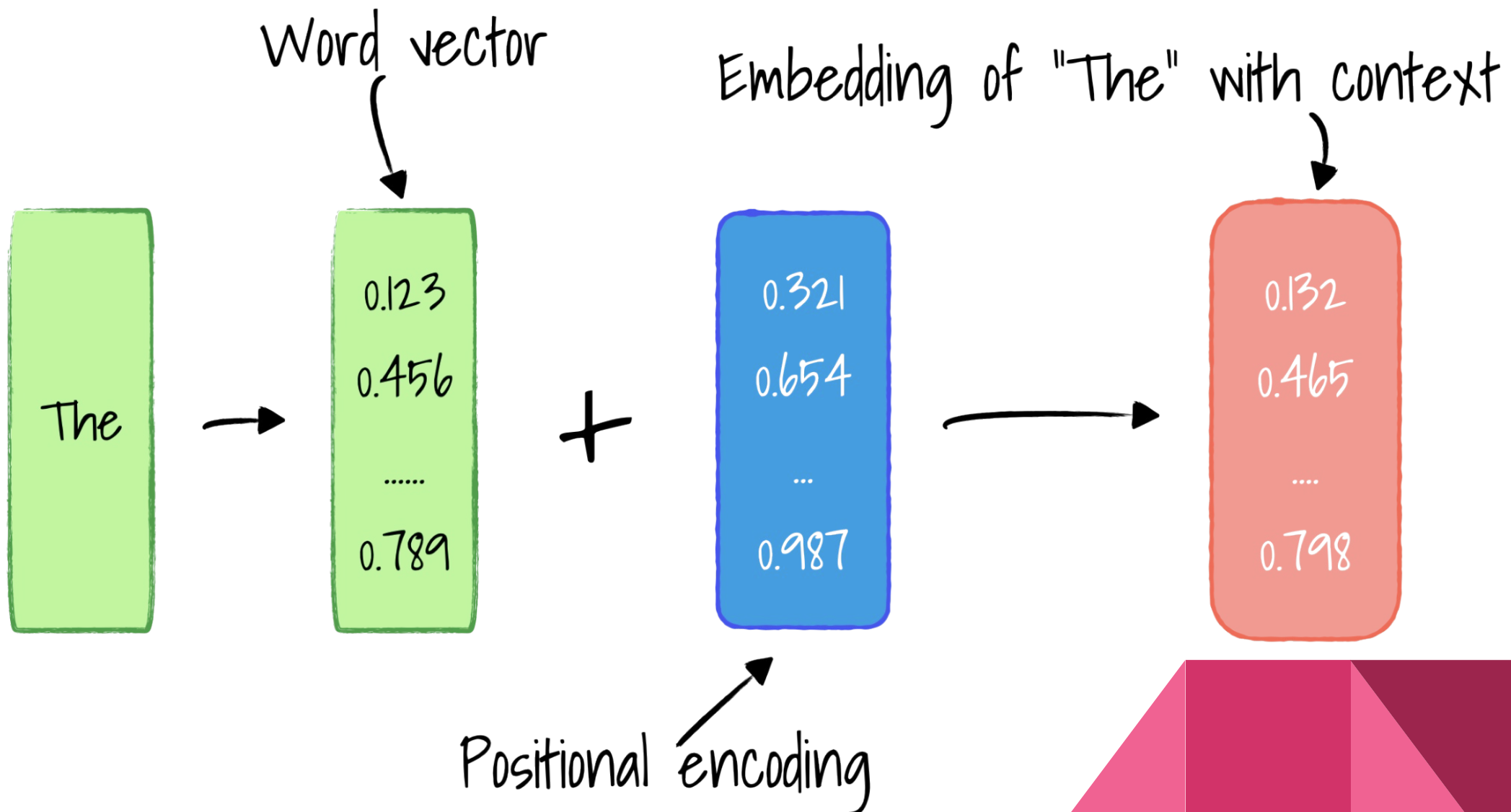
- Sumarização
- Tradução



Arquiteturas encoder-decoder



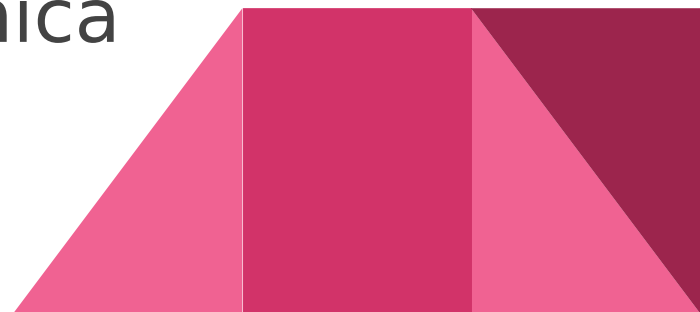
Positional encoders



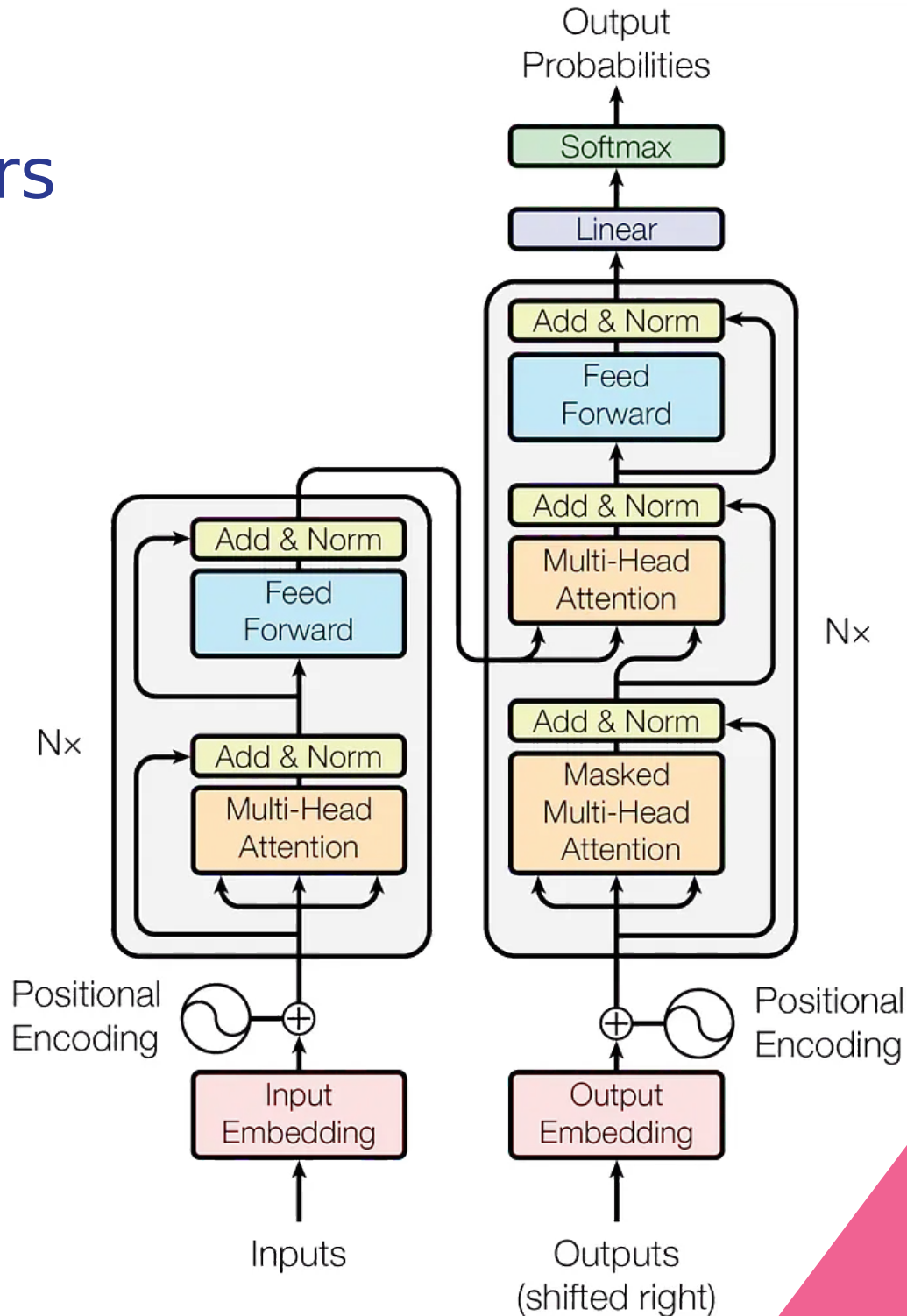
Arquiteturas encoder-decoder

Contudo, o problema de usar uma RNN com arquitetura *encoder-decoder* é o tamanho fixo dos vetores

- Entrada e saída
- Desafiador para entradas grandes, pois a sentença é processada de uma única vez
 - Que transforma em saída única
 - Traduzida ao mesmo tempo

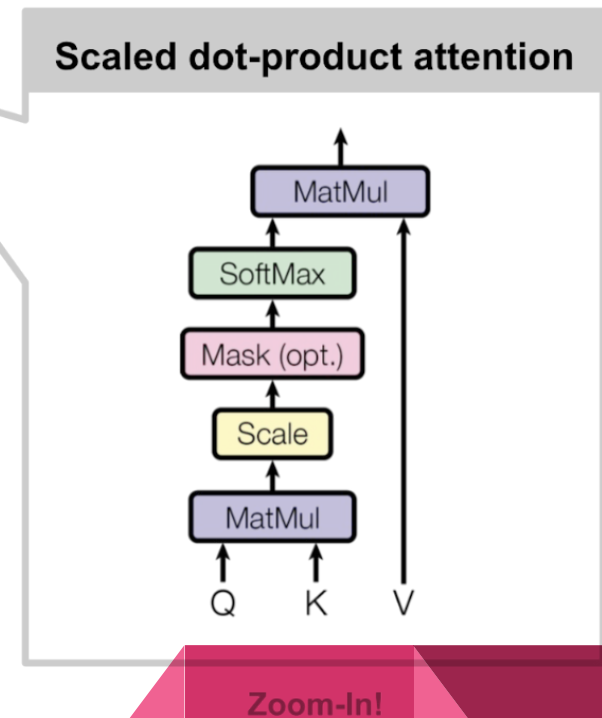
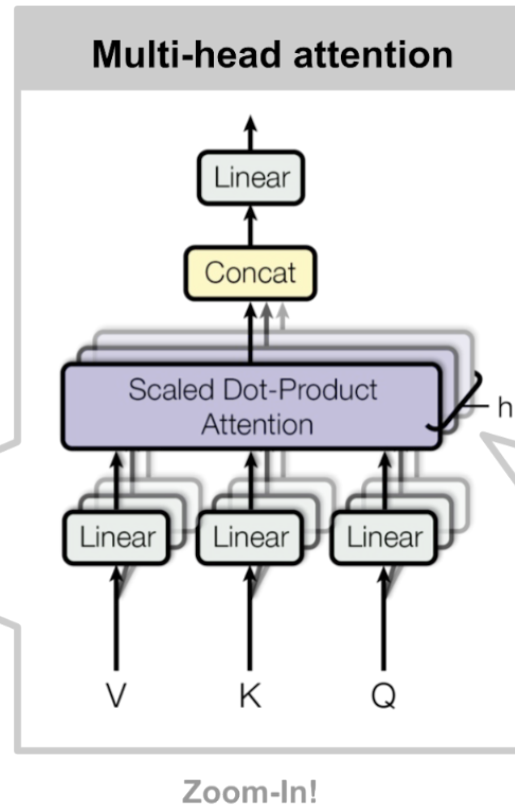
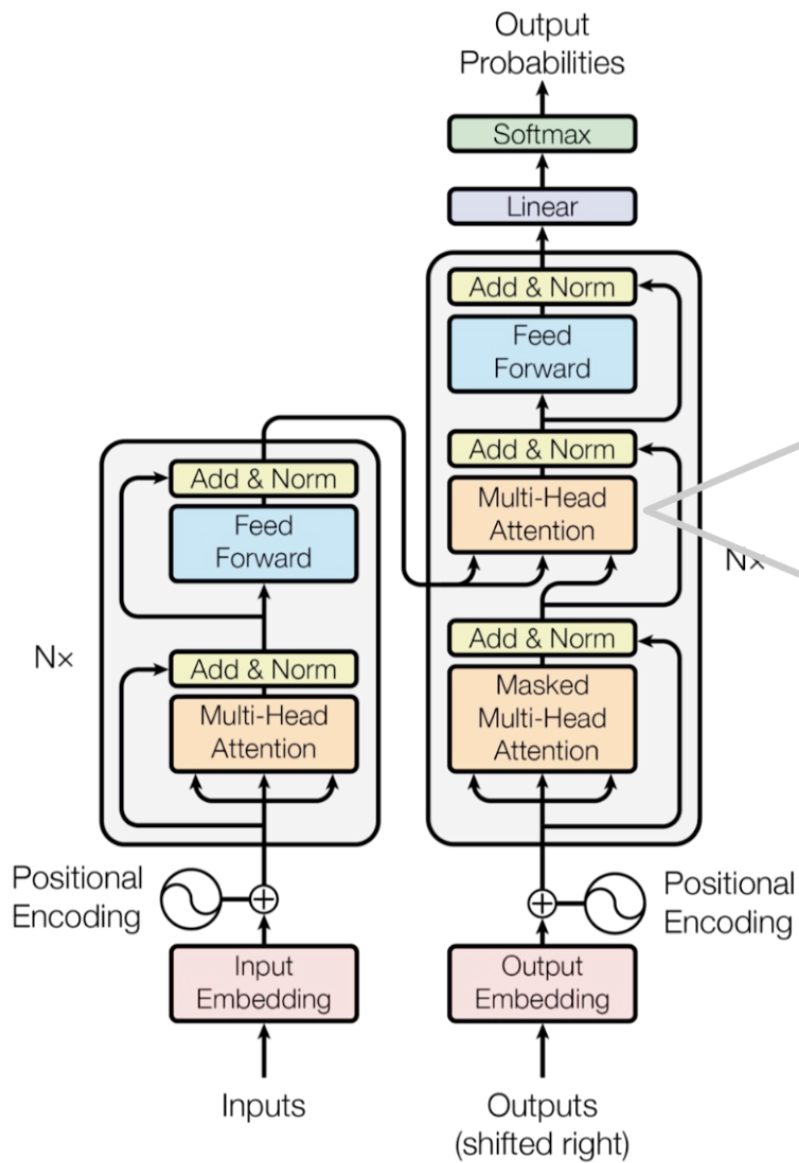


Transformers



VASWANI, Ashish et al. Attention is all you need. In: **Proceedings of the 31st International Conference on Neural Information Processing Systems**. 2017. p. 6000-6010.

Transformers

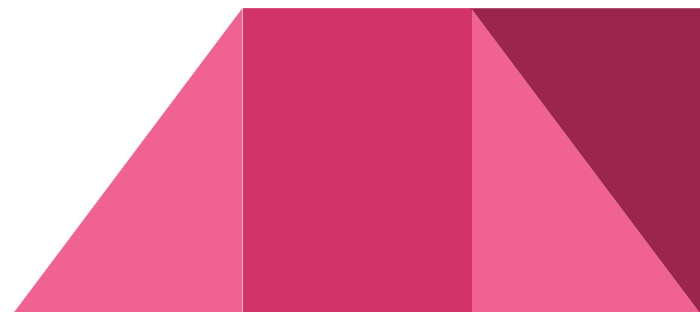


VASWANI, Ashish et al. Attention is all you need. In: **Proceedings of the 31st International Conference on Neural Information Processing Systems**. 2017. p. 6000-6010.

Multi-head attention

Multi-head attention define qual parte da sentença o encoder deve dar prioridade e qual não

- Calcula quão relevante uma palavra é em relação as outras na setença
 - Cria um vetor de atenção, que reflete as relações contextuais das palavras na sentença
- Múltiplos vetores → *Multi-head*
 - Normalizados em um



Multi-head attention

Multiple Attention

(how relevant is a word in the sentence relevant to other words)

The	The	Cat	Is	Black
Cat	The	Cat	Is	Black
Is	The	Cat	Is	Black
Black	The	Cat	Is	Black

Averaged Attention Vectors

[0.80 0.13 0.04 0.03]

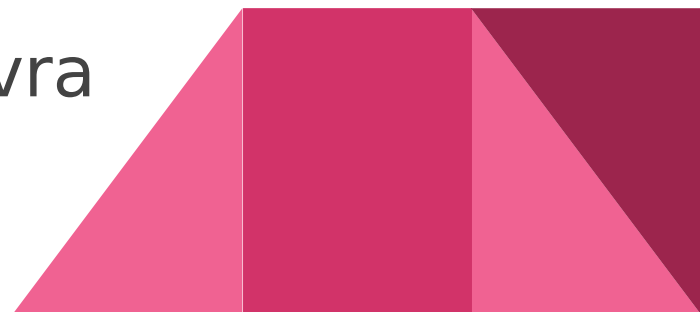
[0.20 0.62 0.08 0.10]

[0.05 0.25 0.42 0.28]

[0.04 0.33 0.12 0.51]

Decoder embedding

- Decoder é treinado com embeddings
- Os vetores posicionais da entrada são usados para gerar contexto
 - E alimentam o *decoder*
- No decoder, os vetores de atenção são mascarados para que as palavras na tradução considerem apenas as anteriores
 - Simulando uma tradução palavra a palavra na ordem da frase



Masked input

Multiple Attention

(how relevant is a word in the sentence relevant to other words)

1st pass	Le	Chat	Est	Noir
2nd pass	Le	Chat	Est	Noir
3rd pass	Le	Chat	Est	Noir
4th pass	Le	Chat	Est	Noir

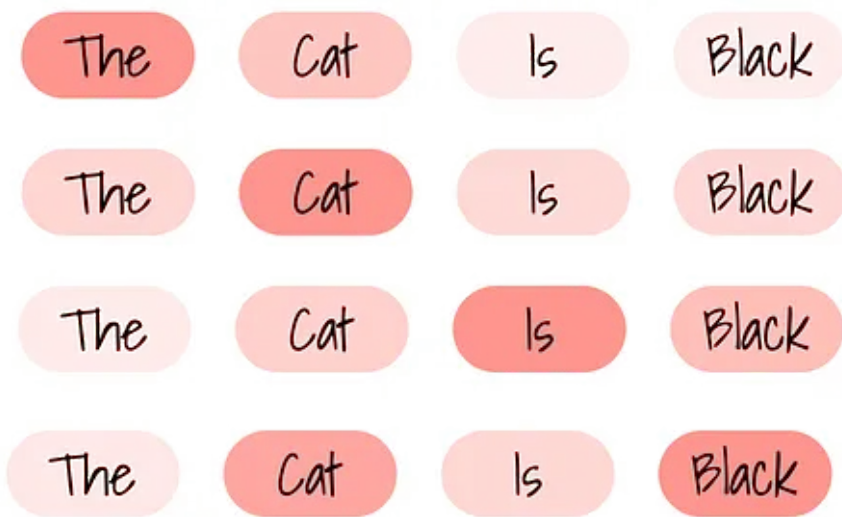
Masked Input

(mask the words appearing later so the attention network can't use them)

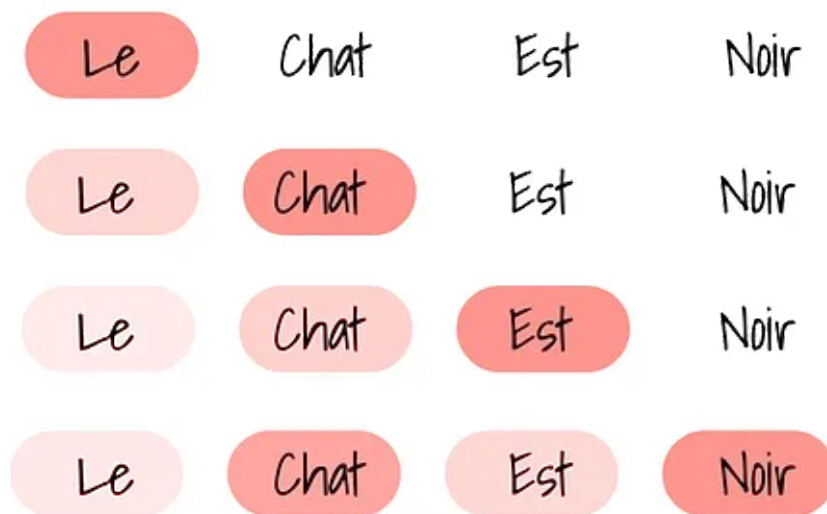
Le	Chat	Est	Noir
Le	Chat	Est	Noir
Le	Chat	Est	Noir
Le	Chat	Est	Noir

Decoder's Multi-Head Attention

Encoder's
Multi-Head Attention



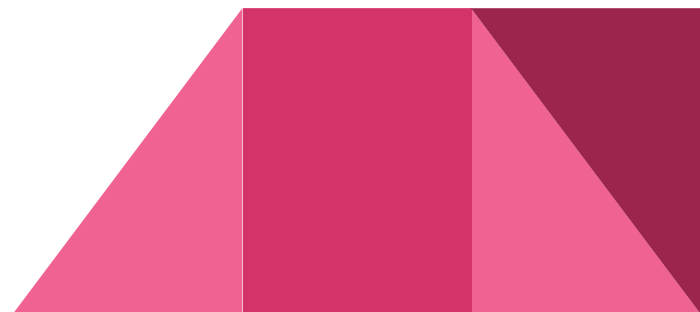
Decoder's Masked
Multi-Head Attention



Decoder's Multi-Head
Attention

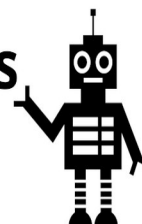
Fazendo previsões

- Transforme a frase de entrada em uma sequência de tokens
- Defina a sequência de saída inicial para o token *sos*
 - Até atingirmos o comprimento máximo ou o token *eos* ser retornado pelo modelo
- Obtenha a próxima palavra prevista.
- Obtenha o índice no vocabulário da palavra com maior probabilidade
- Concatene a próxima palavra prevista na sequência de saída



Dicas práticas

Simple Transformers



<https://github.com/ThilinaRajapakse/simpletransformers>



Transformers

<https://huggingface.co/transformers/>

Bibliografia e referências

- Russell, S. J., & Norvig, P. (2010). Artificial intelligence a modern approach. London.
- Katti Faceli, Ana Carolina Lorena, João Gama, Tiago Agostinho de Almeida e André C. P. L. F de Carvalho. Inteligência Artificial - Uma Abordagem de Aprendizado de Máquina. Terceira Edição. Editora Gen. 2021
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
- Sarzynska-Wawer, J., Wawer, A., Pawlak, A., Szymanowska, J., Stefaniak, I., Jarkiewicz, M., & Okruszek, L. (2021). Detecting formal thought disorder by deep contextualized word representations. Psychiatry Research, 304, 114135.
- VASWANI, Ashish et al. Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017. p. 6000-6010.