Word2Vec, GloVe, FastText e Wang2Vec

...

Chrystian Arriel Amaral

Grupo de Estudos em PLN e IA - DCC/UFLA

ROTEIRO

- Revisando Word Embedding
- Word2Vec
- FastText
- Wang2Vec
- GloVe
- Demonstrações
- Praticando

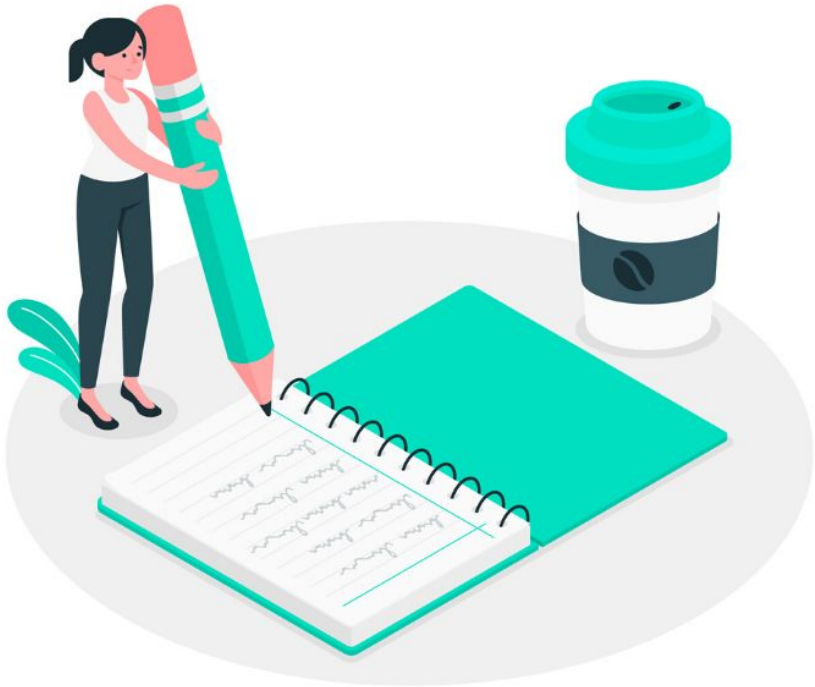


Imagem: https://hubify.com.br/blog/social-media/roteiro-de-video

Revisando Word Embedding

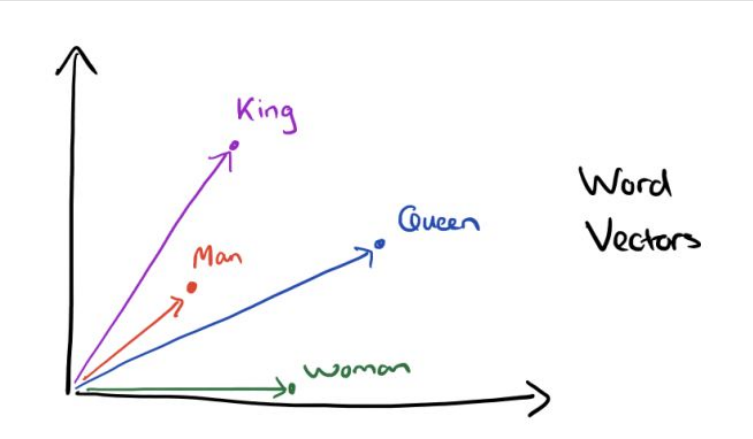


Imagem: https://becominghuman.ai/nlp-serverless-deployment-of-word-embeddings-and-retrieving-most-similar-words-using-kmeans-aws-51f129297995

“Do texto aos números”

Quando trabalhamos com PLN, normalmente temos que lidar com dados textuais, muitas vezes em grandes quantidades. Então como lidar com esse processamento gigante de texto?

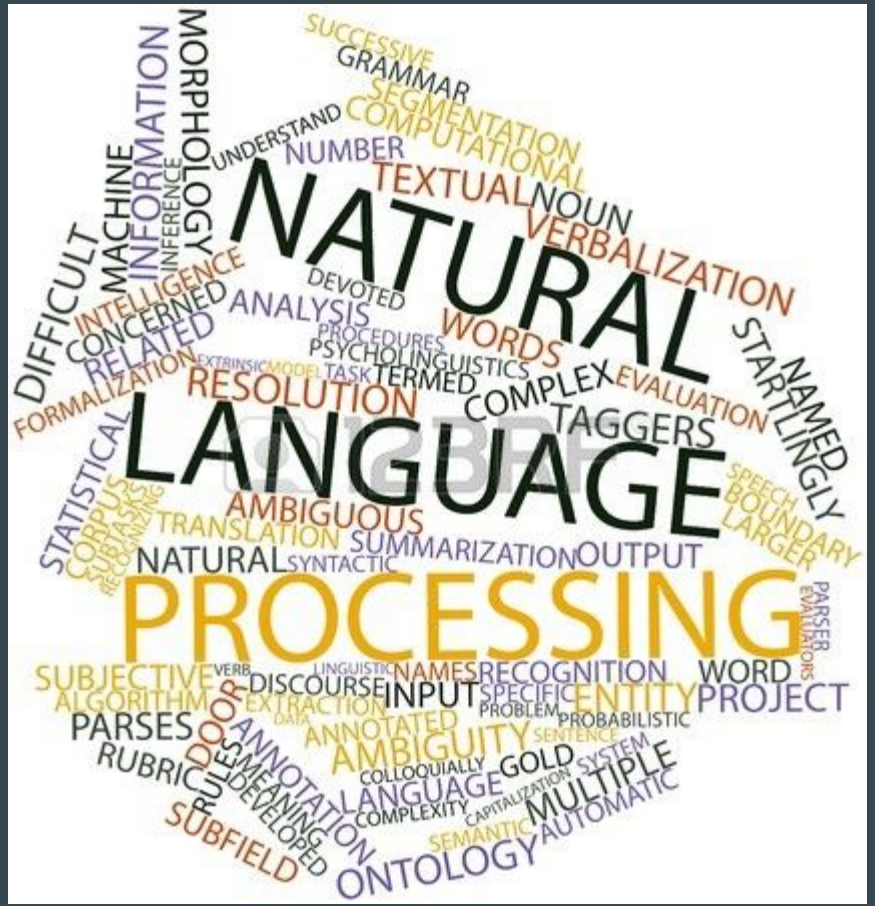


Imagem:
https://medium.com/botsbrasil/o-que-%C3%A9-o-processamento-de-linguagem-natural-49ece9371cff

“Do texto aos números”

Outra questão é o fato que computadores não lidam eficientemente bem com processamento de cadeias de caracteres (strings). Logo precisamos realizar um “tratamento” aos textos.

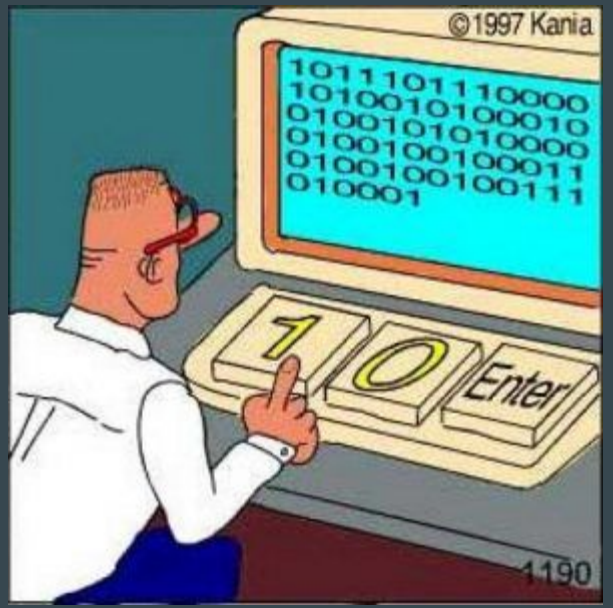
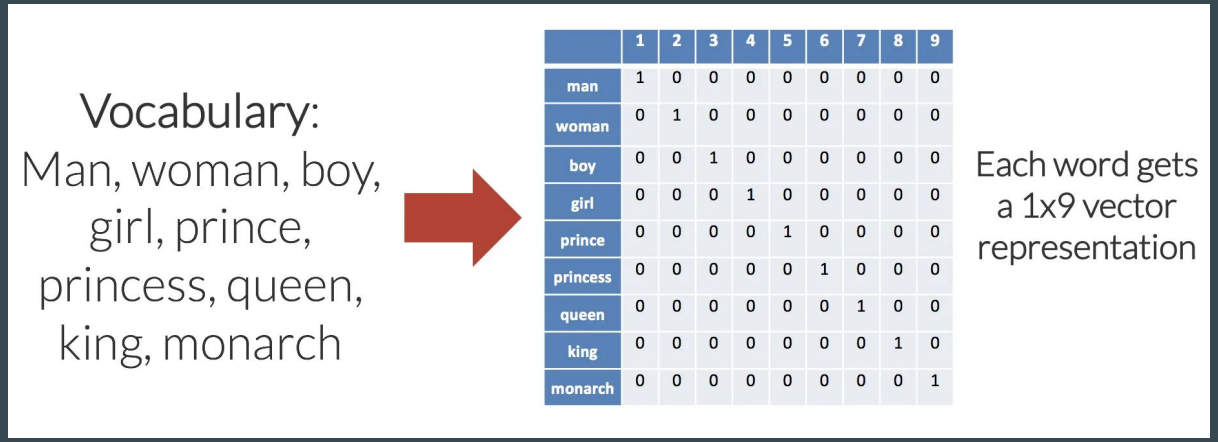


Imagem: https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/

“Do texto aos números”

Com isso temos a incorporação de palavras (Word Embeddings).

Métodos de representar textos em forma de número.



	1	2	3	4	5	6	7	8	9
man	1	0	0	0	0	0	0	0	0
woman	0	1	0	0	0	0	0	0	0
boy	0	0	1	0	0	0	0	0	0
girl	0	0	0	1	0	0	0	0	0
prince	0	0	0	0	1	0	0	0	0
princess	0	0	0	0	0	1	0	0	0
queen	0	0	0	0	0	0	1	0	0
king	0	0	0	0	0	0	0	1	0
monarch	0	0	0	0	0	0	0	0	1

Imagem: https://www.r-craft.org/r-news/get-busy-with-word-embeddings-an-introduction/

Word Embeddings

Há diferentes tipos de Word Embeddings, dos simples aos mais elaborados.
Normalmente geram um vetor numérico que representa a palavra ou o documento de palavras.

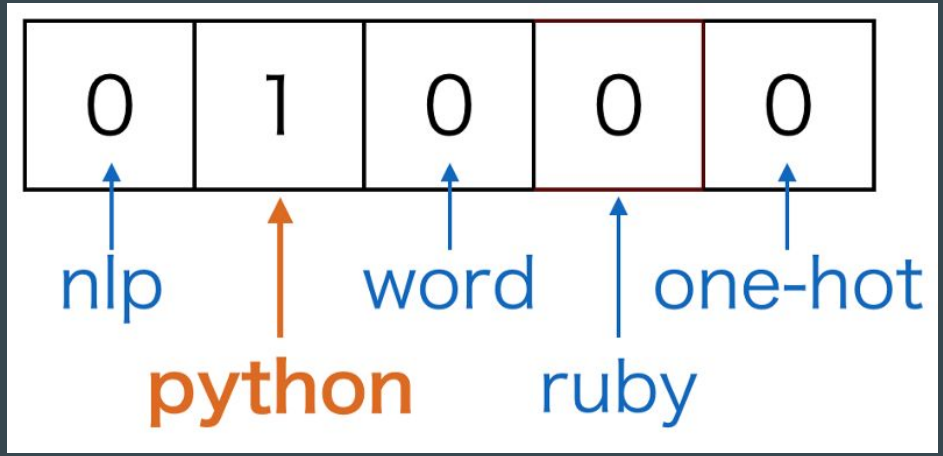


Imagem: https://icrowdnewswire.com/2017/10/16/incorporacao-de-palavras-data-science-group-iitr-medium/

Word Embeddings

Podemos classificar os tipos de word embeddings em dois grupos:

- Incorporação baseada em frequência
- Incorporação baseada em previsão

Word Embeddings baseado em frequência

São métodos mais fáceis e rápidos, funcionando com base na contagem das palavras em documentos.

Alguns exemplos são:

- Vetores de Contagem
- Vetores TF-IDF

Word Embeddings baseado em frequência

Vetores de Contagem:

Doc1 = “Take a look into the beauty of the word embedding.”

Doc2 = “The word vectorizer is the most basic word embedding.”

Dicionário dos documentos:

{'basic': 0, 'beauty': 1, 'embedding': 2, 'into': 3, 'is': 4, 'look': 5,

'most': 6, 'of': 7, 'take': 8, 'the': 9, 'vectorizer': 10, 'word': 11}

Word Embeddings baseado em frequência

Vetores de Contagem:

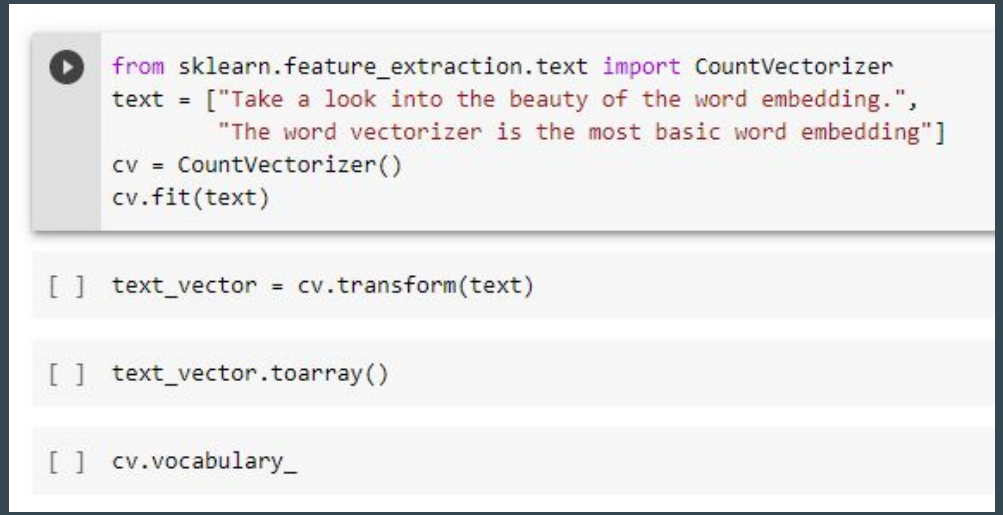
Como resultado vamos ter um vetor com a frequência de cada palavras única, em cada documento.

```
text_vector = ([[0,  1,  1,  1,  0,  1,  0,  1,  1,  2,  0,  1],
                [1,  0,  1,  0,  1,  0,  1,  0,  0,  2,  1,  2]])
```

Word Embeddings baseado em frequência

Vetores de Contagem:

Exemplo em python



```
from sklearn.feature_extraction.text import CountVectorizer
text = ["Take a look into the beauty of the word embedding.",
        "The word vectorizer is the most basic word embedding"]
cv = CountVectorizer()
cv.fit(text)
```

```
[ ] text_vector = cv.transform(text)
```

```
[ ] text_vector.toarray()
```

```
[ ] cv.vocabulary_
```

Word Embeddings baseado em frequência

Vetores de Contagem:

Uma imagem da matriz para facilitar a visualização do conceito

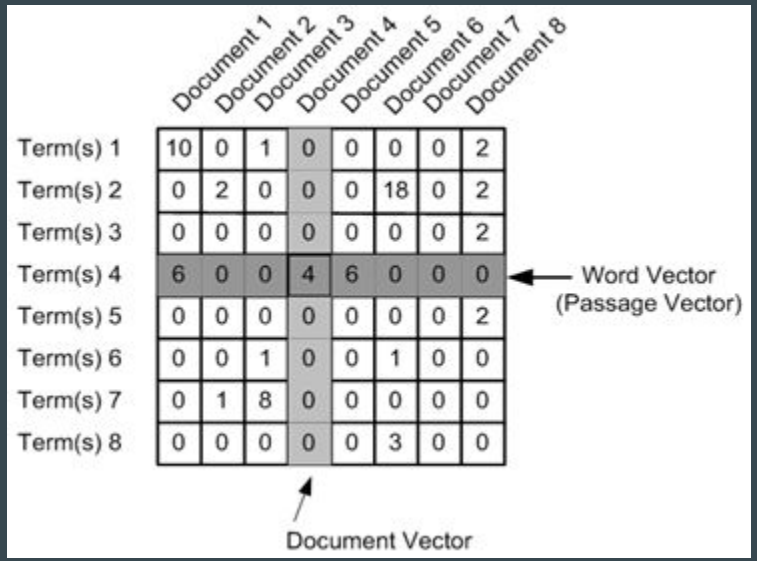


Imagem: https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/

Word Embeddings baseado em frequência

Vetores TF-IDF:

Diferente do vetor de contagem, não é considerado somente a frequência em um documento específico, mas também em todo o corpus (conjunto total dos documentos).

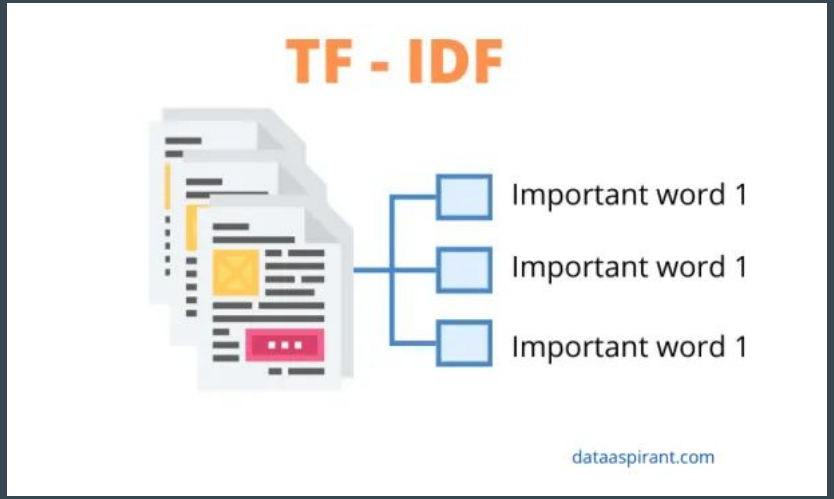


Imagem: https://dataaspirant.com/word-embedding-techniques-nlp/

Word Embeddings baseado em frequência

Vetores TF-IDF:

Palavras comuns como 'é', 'o', 'a', etc. possuiriam peso menor, dando ênfase a palavras importantes para documentos específicos.

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF

Term x within document y

$tf_{x,y}$ = frequency of x in y

df_x = number of documents containing x

N = total number of documents

Imagem: https://mropengate.blogspot.com/2016/04/tf-idf-in-r-language.html

Word Embeddings baseado em frequência

Vetores TF-IDF:

Doc1 = “Take a look into the beauty of the word embedding.”

Doc2 = “The word vectorizer is the most basic word embedding.”

Final result of Tf-Idf

	basic	beauty	embedding	into	is	look	most	of	take	the	vectorizer	word
0	0.000000	0.352728	0.250969	0.352728	0.000000	0.352728	0.000000	0.352728	0.352728	0.501938	0.000000	0.250969
1	0.341869	0.000000	0.243243	0.000000	0.341869	0.000000	0.341869	0.000000	0.000000	0.486485	0.341869	0.486485

Word Embeddings baseado em previsão

Os vetores baseados em frequência possuem o problema de serem computacionalmente muito caros, com necessidade muito grande de memória.

Para superar isso, os métodos baseados em previsão resolvem essas limitações. Eles possuem ajuda de Redes Neurais.

Exemplos de métodos baseados em previsão:

- Word2Vec
- FastText
- Wang2Vec

Word2Vec

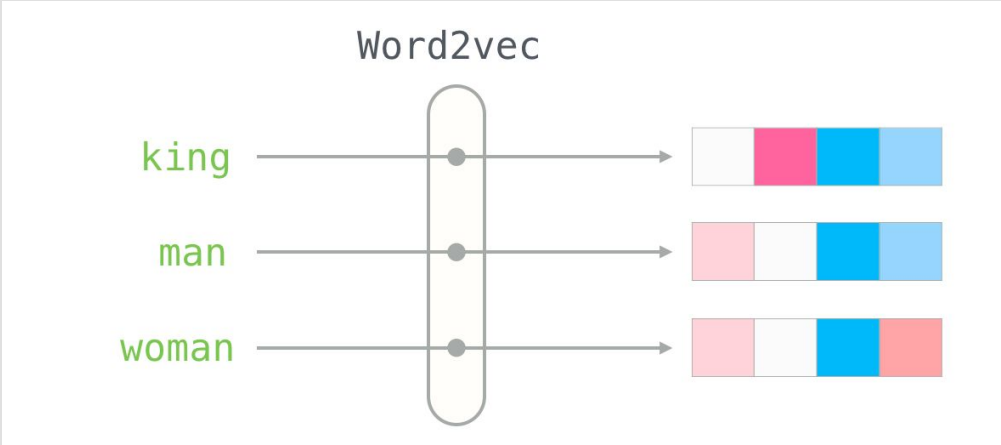
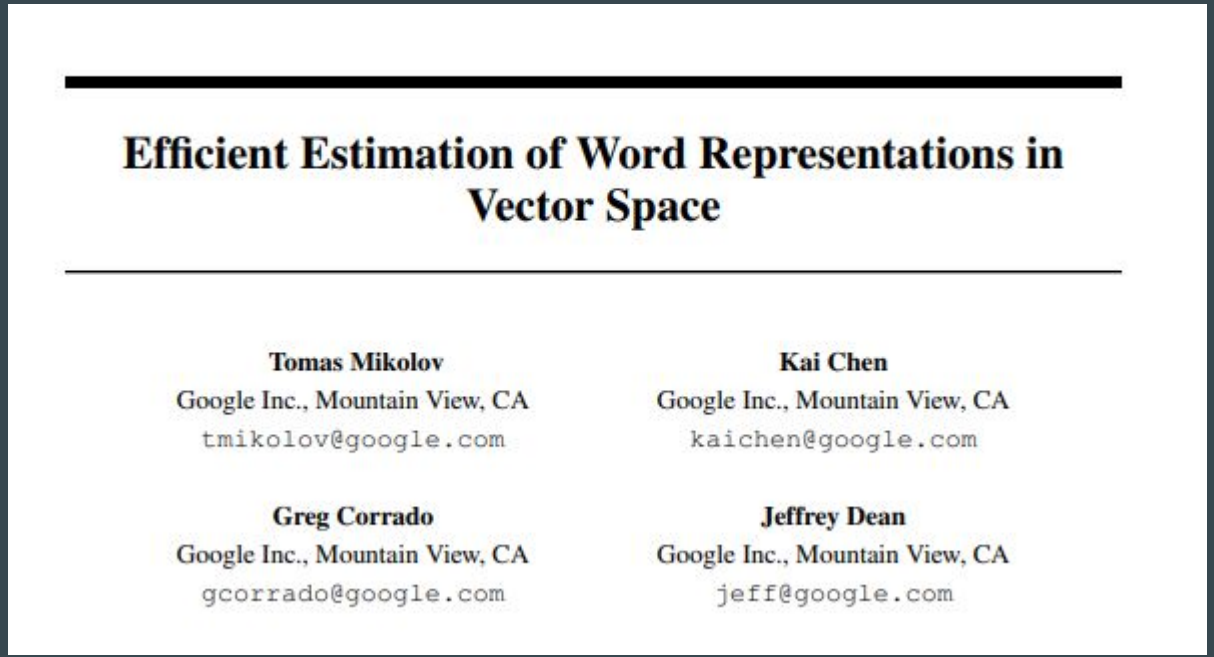


Imagem: https://blogdozouza.wordpress.com/2019/03/29/o-word2vec-ilustrado/

Word2Vec

Teve origem no artigo de Mikolov (2013a).



Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

Word2Vec

Com a combinação de duas técnicas, CBOW e Skip-Gram, as quais são redes neurais superficiais, o Word2Vec consegue gerar vetores de palavras N-dimensionais.

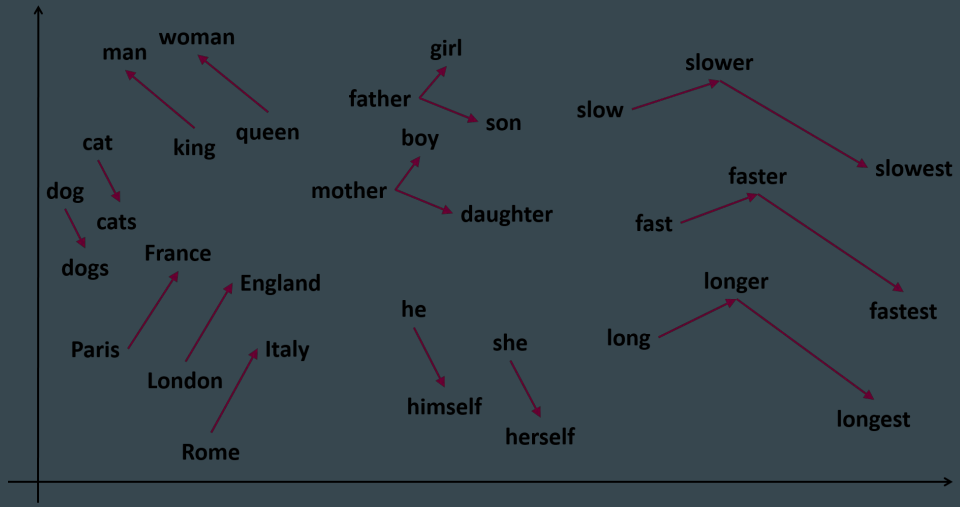


Imagem: https://towardsdatascience.com/3-basic-approaches-in-bag-of-words-which-are-better-than-word-embeddings-c2cbc7398016

Word2Vec

Além da representação de textos, é possível trabalhar com tarefas de analogias e semelhanças de palavras e coisas como “Rei - Homem + Mulher = Rainha”.

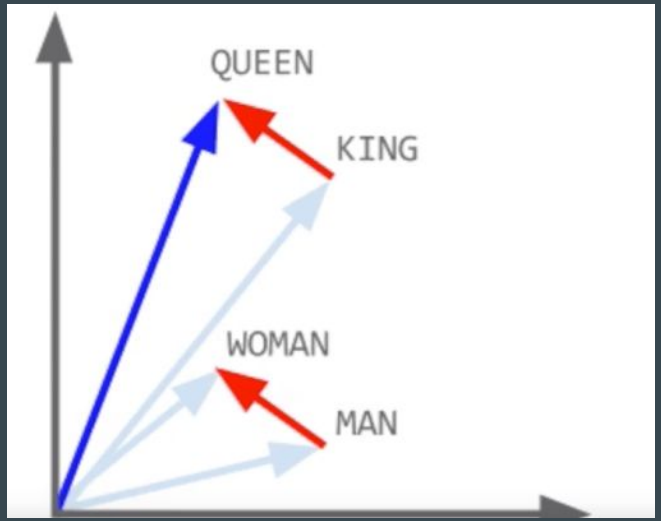


Imagem: https://medium.com/arvind-internet/applying-word2vec-on-our-catalog-data-2d74dfee419d

One-Hot

Uma matriz esparsa, onde cada posição representa uma palavra do vocabulário, sendo aquela com valor 1, representante do vetor.

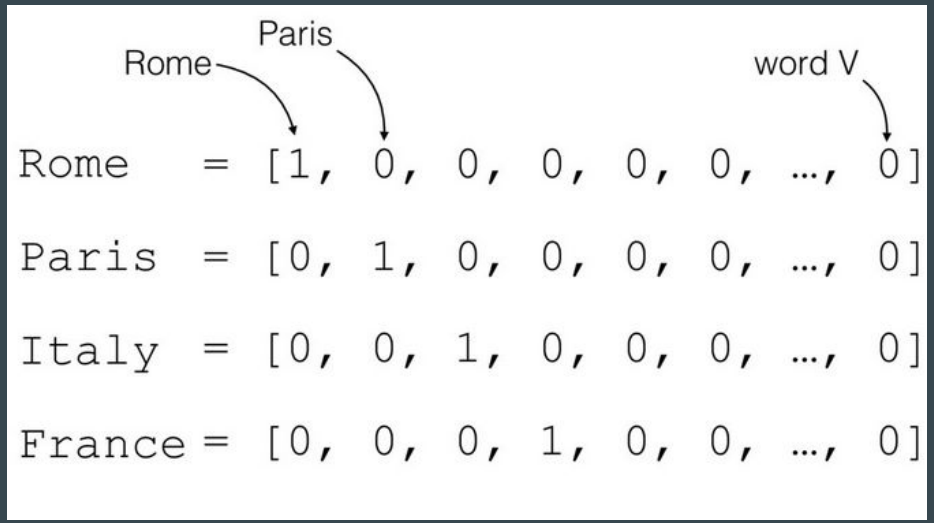


Imagem: https://medium.com/intelligentmachines/word-embedding-and-one-hot-encoding-ad17b4bbe111

Continuous Bag of Words Model (CBOW)

Busca por meio de probabilidade, uma palavra que se encaixe em um contexto. A palavra de saída é a mais provável de ser o centro da janela de contexto.

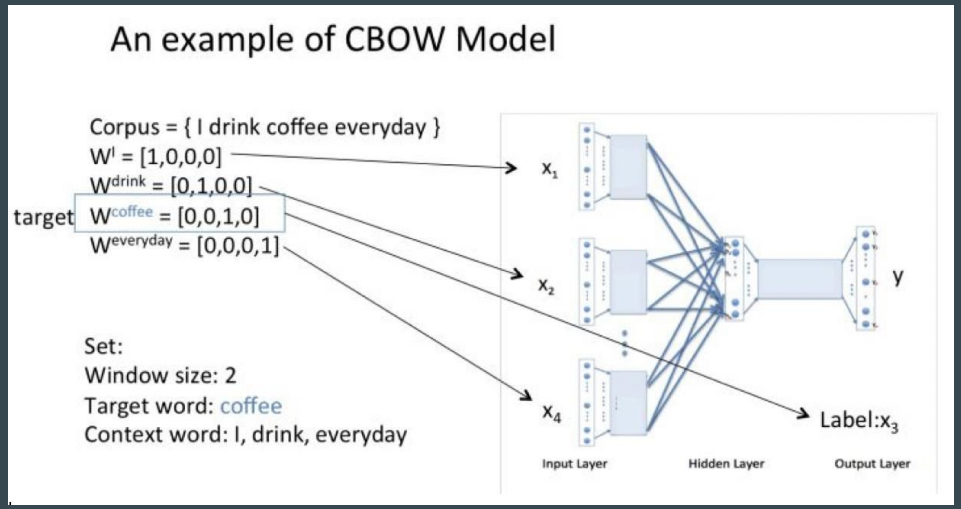


Imagem: https://www.programmersought.com/article/84944066696/

Continuous Bag of Words Model (CBOW)

Busca por meio de probabilidade, uma palavra que se encaixe em um contexto. A palavra de saída é a mais provável de ser o centro da janela de contexto.

É uma rede neural de uma camada oculta.

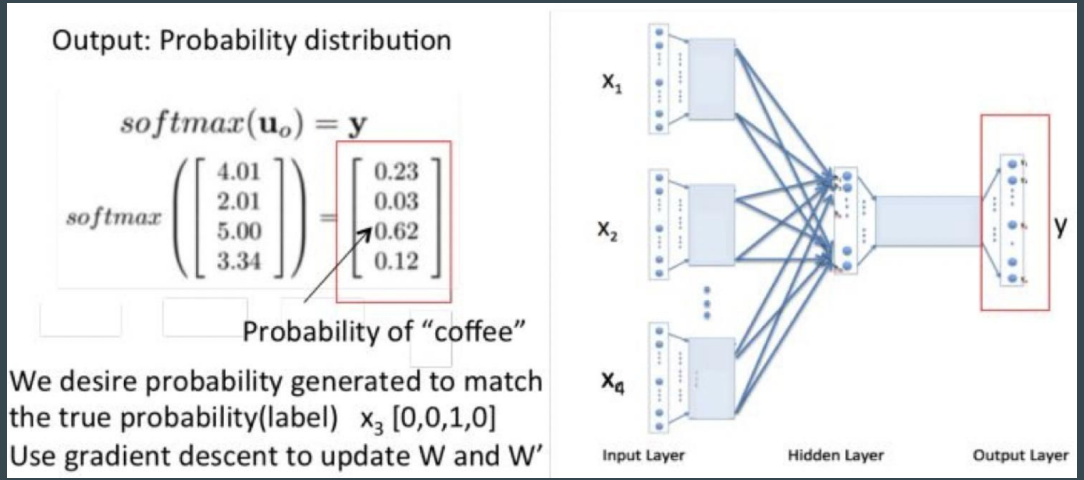


Imagem: https://www.programmersought.com/article/84944066696/

Continuous Bag of Words Model (CBOW)

Busca por meio de probabilidade, uma palavra que se encaixe em um contexto. A palavra de saída é a mais provável de ser o centro da janela de contexto.

Janela de contexto: Número de palavras antes e depois de uma palavra central.

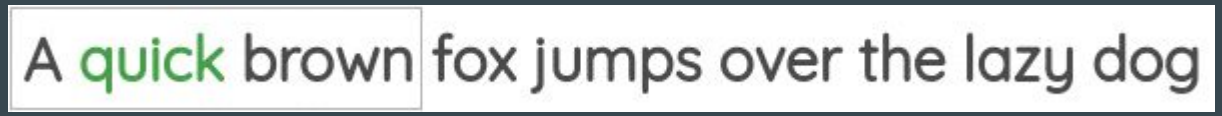


Imagem: https://amitness.com/2020/06/fasttext-embeddings/

Skip-Gram

O Skip-Gram funciona exatamente ao contrário do CBOW, sendo sua entrada uma palavra e seu resultado um contexto de palavras, as quais se encaixam com a palavra de entrada. O tamanho da janela de contexto define o tamanho da saída.

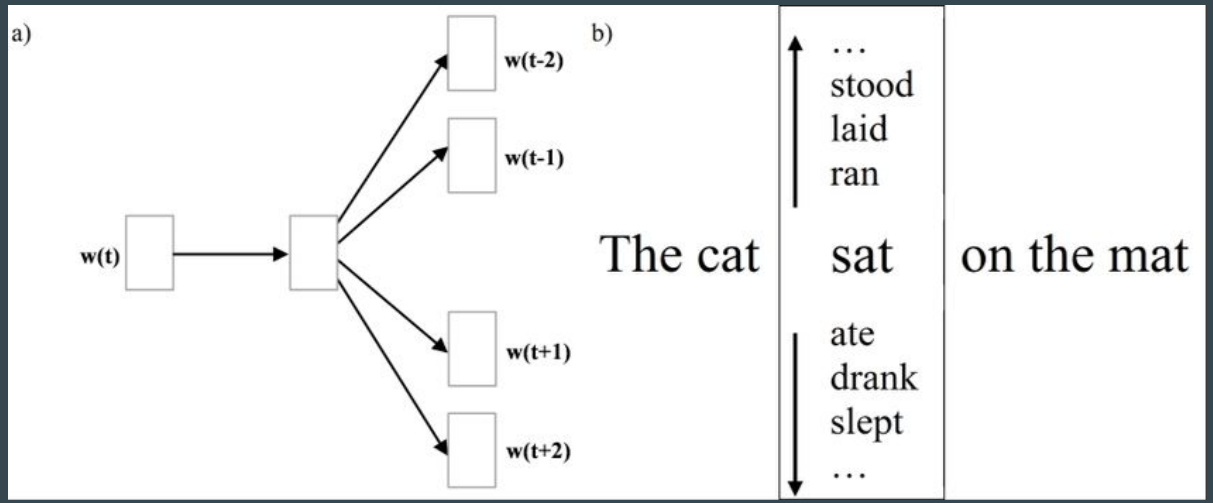


Imagem: https://towardsdatascience.com/skip-gram-nlp-context-words-prediction-algorithm-5bbf34f84e0c

Skip-Gram

O Skip-Gram funciona exatamente ao contrário do CBOW, sendo sua entrada uma palavra e seu resultado um contexto de palavras, as quais se encaixam com a palavra de entrada. O tamanho da janela de contexto define o tamanho da saída.

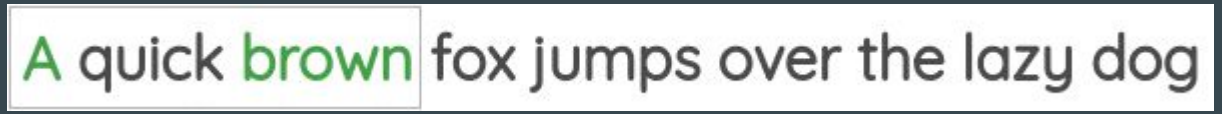


Imagem: https://amitness.com/2020/06/fasttext-embeddings/

Vetor de palavras N-Dimensional

Ao final teremos um vetor N-dimensional para cada palavra, sendo cada N um contexto, o que gera uma “nuvem de palavras” agrupadas por contextos.

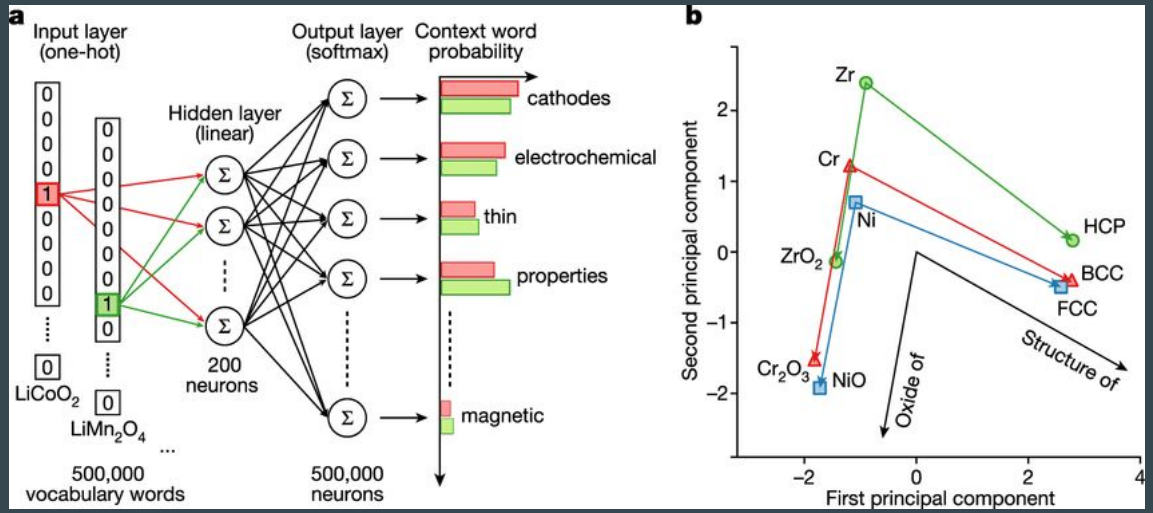


Imagem: https://www.researchgate.net/figure/Word2vec-skip-gram-and-analogies-a-Target-words-LiCoO2-and-LiMn2O4-are-represented_fig3_334209824

Analogias e semelhanças de palavras

Também fica possível realizar um cálculo de similaridade de cosseno, retornando a similaridade entre as palavras.

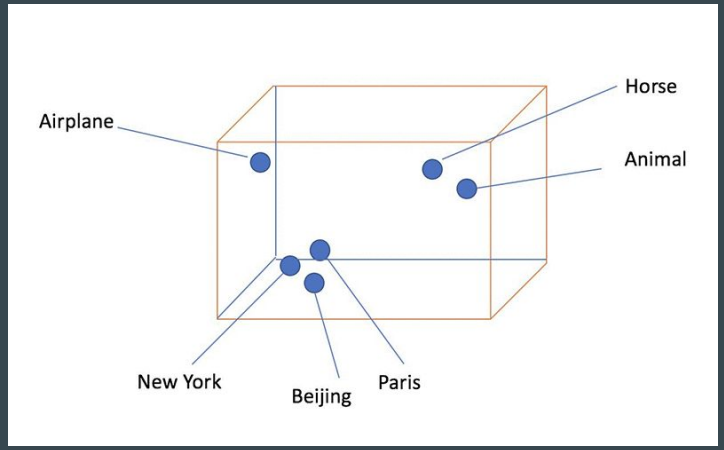


Imagem: https://br.pinterest.com/pin/540713499008933946/

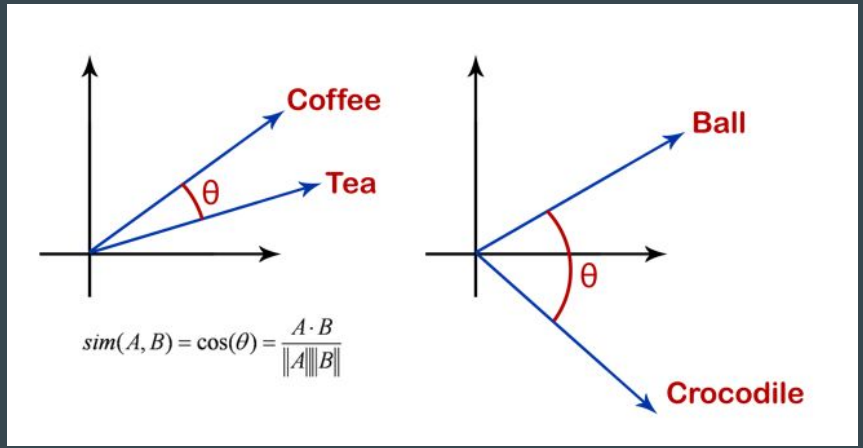
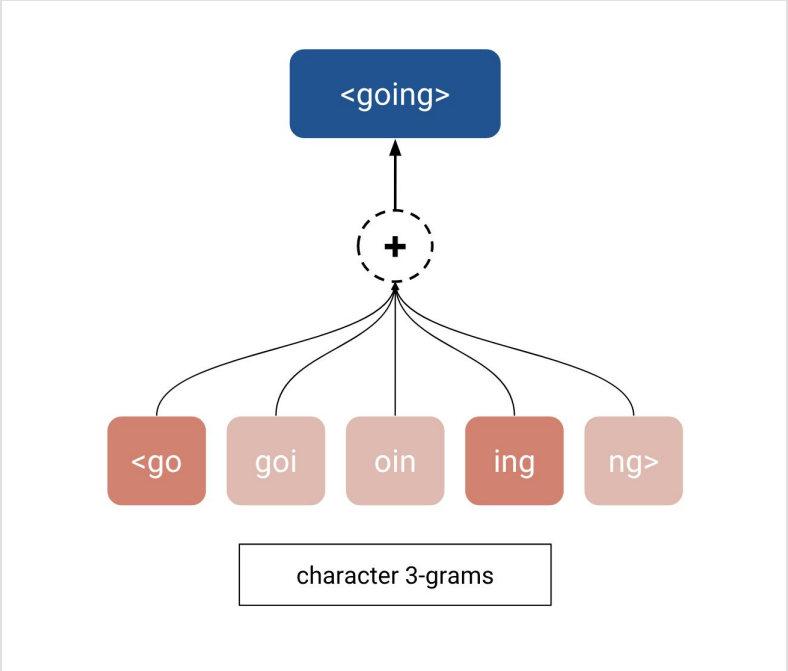


Imagem: https://ichi.pro/pt/embeddings-de-palavras-em-alto-nivel-21683637392240

FastText



Limitações do Word2Vec

Word2Vec possui limitações:

- Palavras fora do vocabulário não conseguem ser processadas;
- Em questões morfológicas, como, por exemplo, o radical, não existe relação entre as palavras.

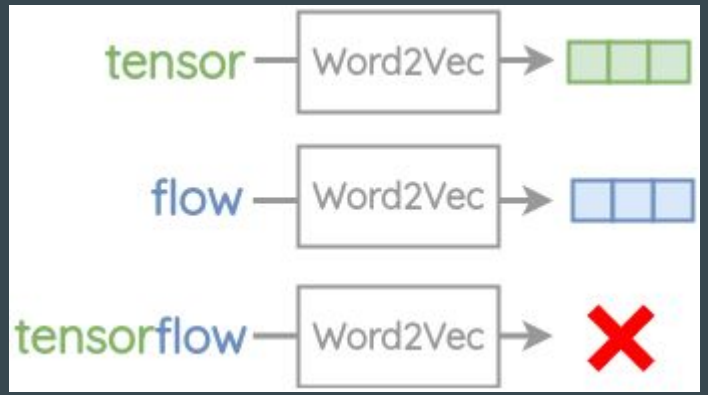
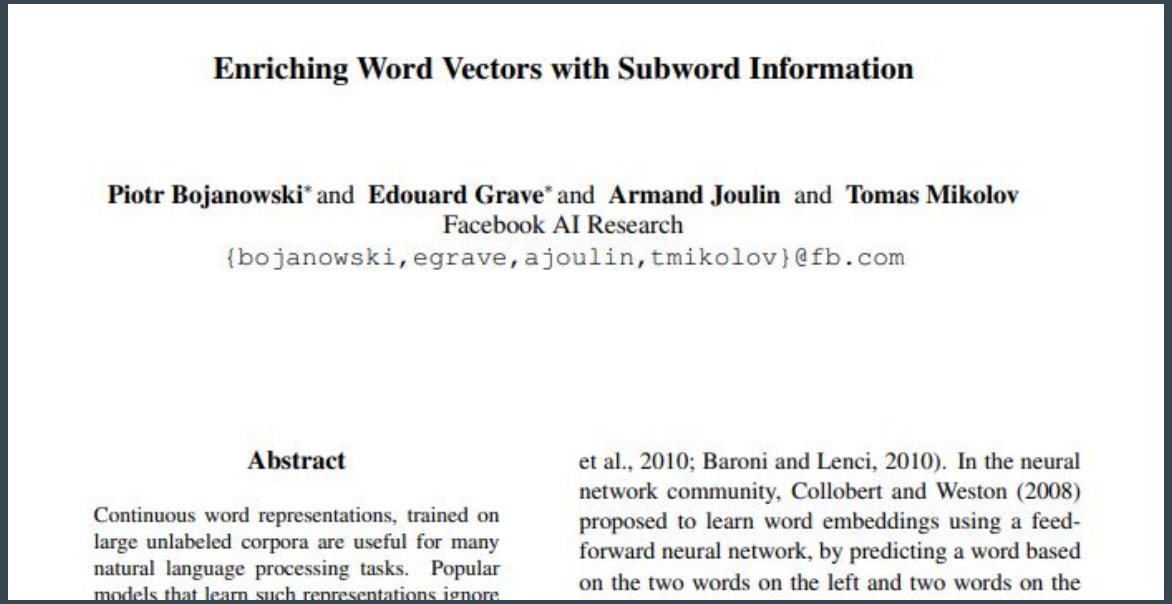


Imagem: https://amitness.com/2020/06/fasttext-embeddings/

Limitações do Word2Vec

O FastText propõe uma solução para tais problemas. Foi proposto por Bojanowski et al. (2016).



Enriching Word Vectors with Subword Information

Piotr Bojanowski* and **Edouard Grave*** and **Armand Joulin** and **Tomas Mikolov**
Facebook AI Research
{bojanowski,egrave,ajoulin,tmikolov}@fb.com

Abstract

Continuous word representations, trained on large unlabeled corpora are useful for many natural language processing tasks. Popular models that learn such representations ignore

et al., 2010; Baroni and Lenci, 2010). In the neural network community, Collobert and Weston (2008) proposed to learn word embeddings using a feed-forward neural network, by predicting a word based on the two words on the left and two words on the

FastText

O FastText gera caracteres n-grams para cada palavra.

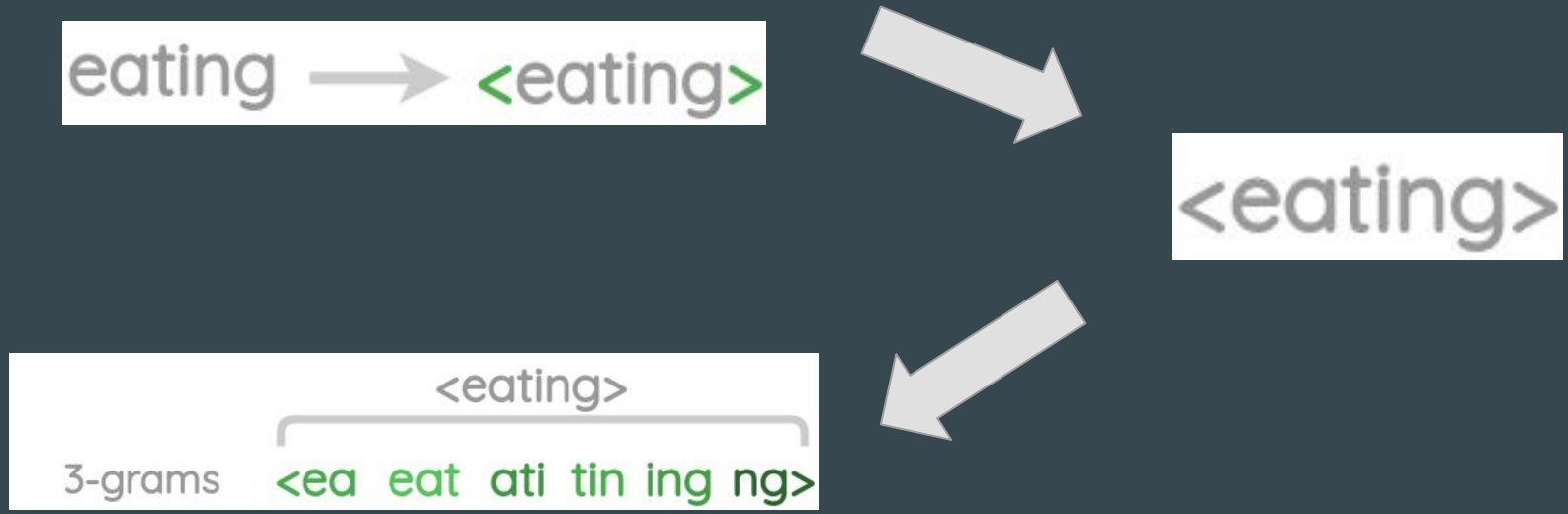


Imagem: https://amitness.com/2020/06/fasttext-embeddings/

FastText

Para n-grams exclusivos é criado um hashing, para questões de memória.

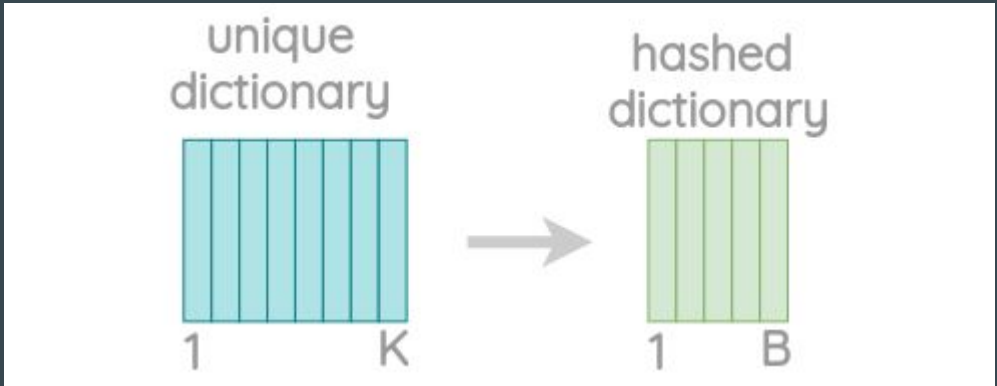


Imagem: https://amitness.com/2020/06/fasttext-embeddings/

Skip-gram com amostragem negativa

Pegando uma palavra central, é necessário prever as palavras vizinhas do contexto. Primeiro é calculado a soma dos vetores dos n-grams e da palavra completa.

Imagem: https://amitness.com/2020/06/fasttext-embeddings/

Skip-gram com amostragem negativa

Depois é pego o vetor das palavras vizinhas mais prováveis e o vetor de amostras negativas.

Imagem: https://amitness.com/2020/06/fasttext-embeddings/

Skip-gram com amostragem negativa

É pego o produto escalar das palavras de contexto e a palavra central, gerado uma pontuação com uma função SGD e por fim atualizado os vetores. Aproximando das palavras de contexto e distanciando das amostras negativas.

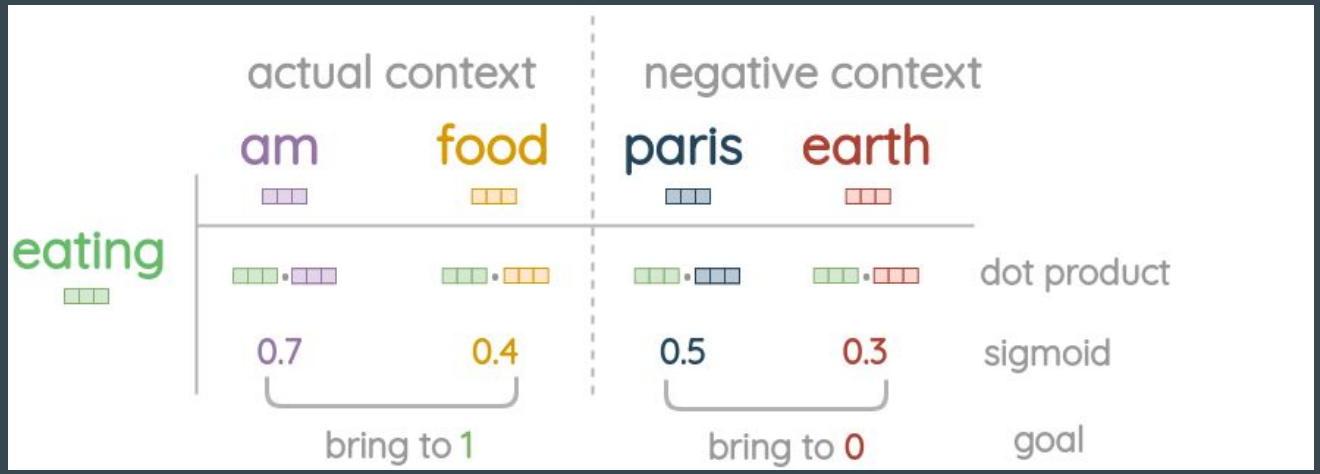


Imagem: https://amitness.com/2020/06/fasttext-embeddings/

Vantagens sobre o Word2Vec

O FastText é uma evolução do Word2Vec, pois possui a mesma arquitetura com o skip-gram e CBOW.

Têm melhores desempenhos para análises sintáticas em idiomas “ricos”.

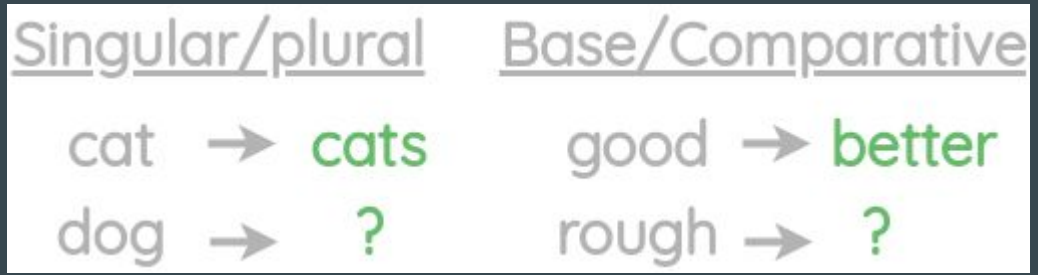


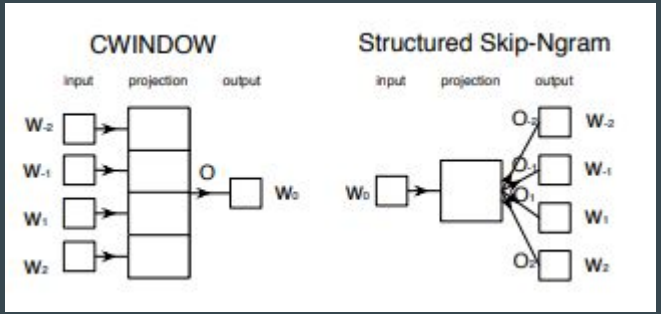
Imagem: https://amitness.com/2020/06/fasttext-embeddings/

Wang2Vec

Wang2Vec

É uma variante do Word2Vec, que visa suprir a falta de ordem das palavras no Word2Vec.

Com base no CBOW e no Skip-gram, o método cria duas novas estruturas: CWINDOW e Structured Skip-Ngram.



GloVe

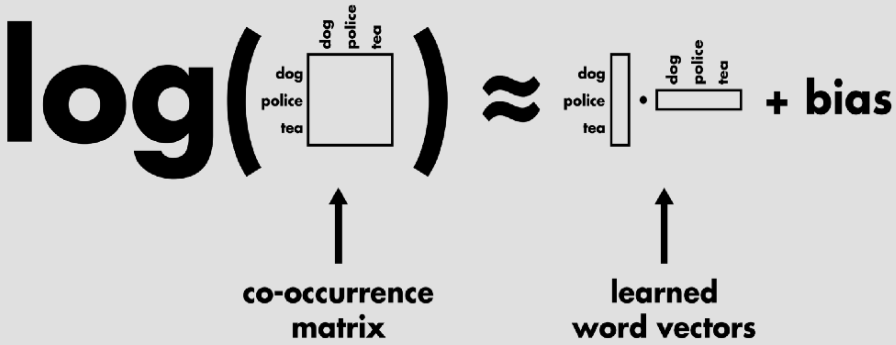


Imagem: https://laptrinhx.com/emnlp-what-is-glove-part-v-3842139123/

Matriz de Coocorrência

Matrizes que marcam a frequência que duas palavras estão em um mesmo contexto.

Exemplo: “the cat sat on the mat”

	the	cat	sat	on	mat
the	0	1	0	1	1
cat	1	0	1	0	0
sat	0	1	0	1	0
on	1	0	1	0	0
mat	1	0	0	0	0

Imagem: https://towardsdatascience.com/light-on-math-ml-intuitive-guide-to-understanding-glove-embeddings-b13b4f19c010

Matriz de Coocorrência

A métrica para obter uma semelhança semântica entre as palavras, utiliza três palavras por vez.

P_{ik}/P_{jk}, onde $P_{ik} = X_{ik}/X_i$

No exemplo i = 'ice' e j = 'steam'

Probability and Ratio	k = *solid*	k = *gas*	k = *water*	k = *fashion*
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Imagem: https://towardsdatascience.com/light-on-math-ml-intuitive-guide-to-understanding-glove-embeddings-b13b4f19c010

GloVe

O GloVe são “vetores globais”, o qual o método captura estatísticas globais e locais de um corpus.

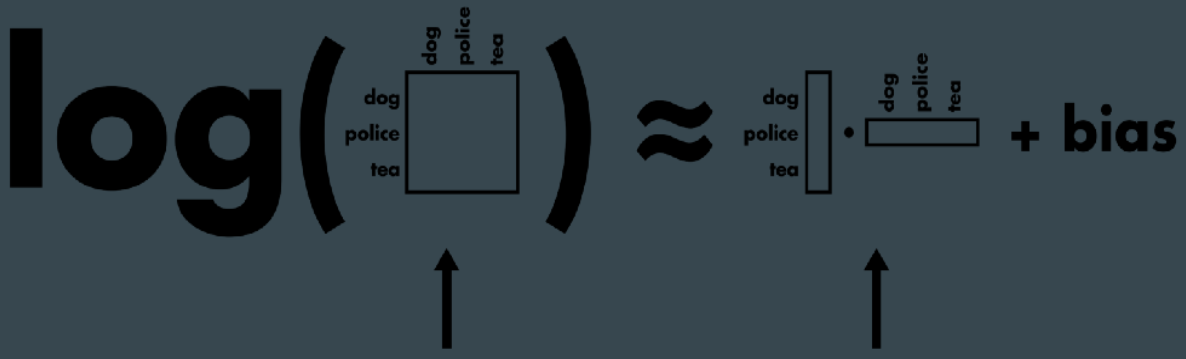


Imagem: https://laptrinhx.com/emnlp-what-is-glove-part-v-3842139123/

GloVe

O GloVe parte da combinação de métodos de contagem e de previsão. Por meio das matrizes de coocorrência, o método constrói sua função de perda.

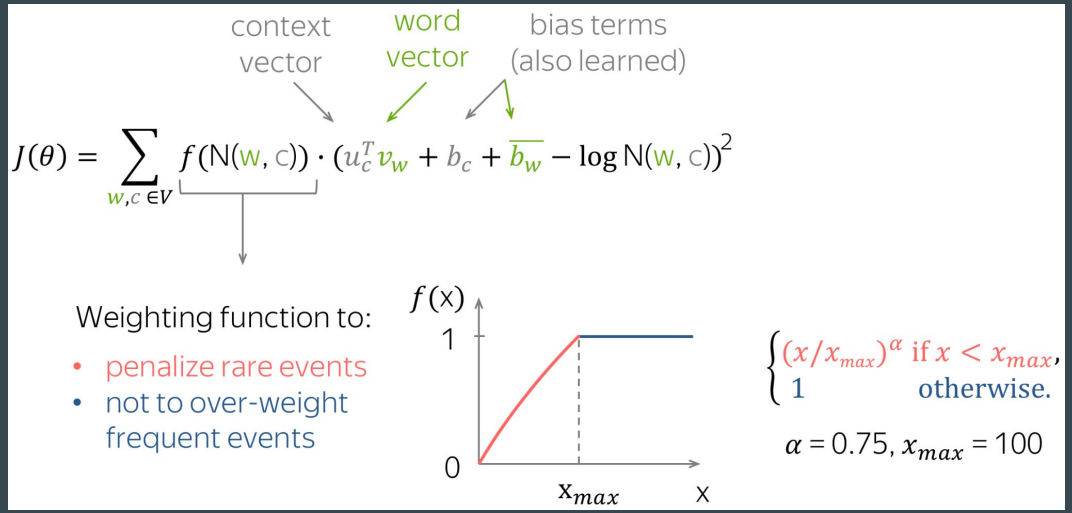


Imagem: https://lena-voita.github.io/nlp_course/word_embeddings.html#w2v_negative_sampling

GloVe

O GloVe também possui os vetores para palavras centrais e contextos. Logo é possível calcular distâncias entre eles, igualmente aos métodos de predição.

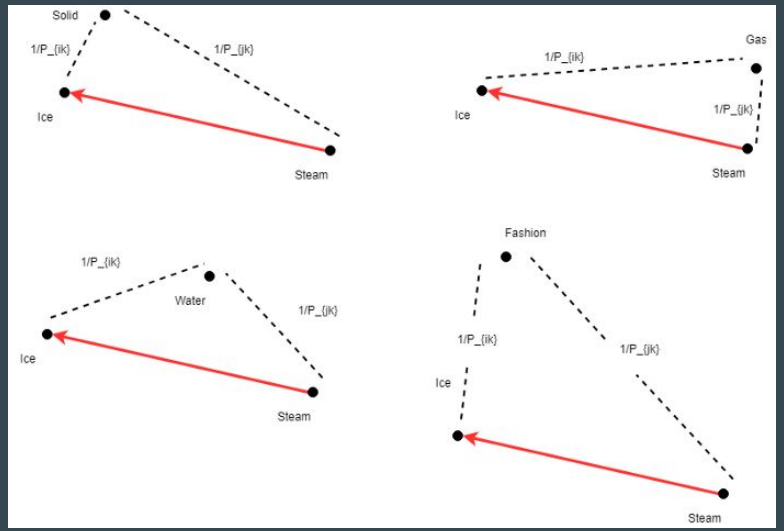


Imagem: https://lena-voita.github.io/nlp_course/word_embeddings.html#w2v_negative_sampling

Demonstrações

Telecurso 2000 de Word Embeddings

Link: https://colab.research.google.com/drive/1EC315rg4BZG8PBTAvhaPyEnX16zTyee1?usp=sharing

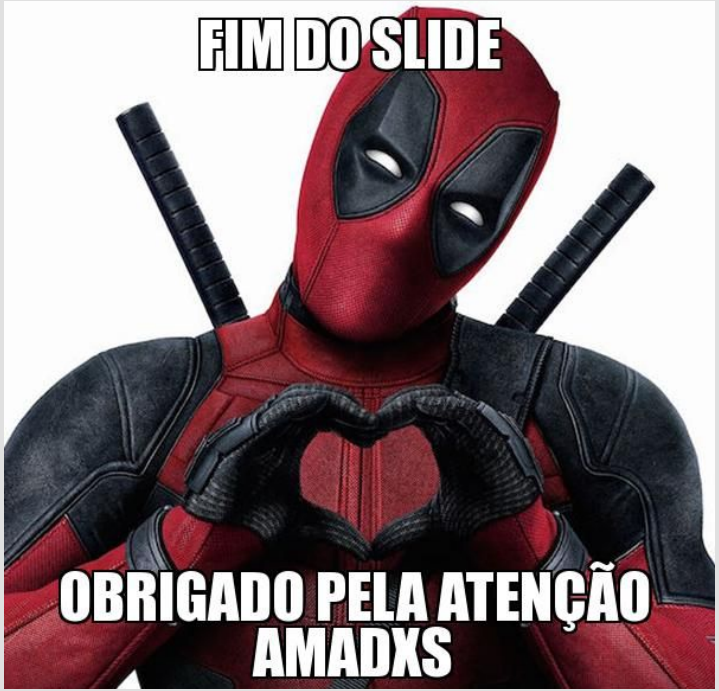
Imagem: https://tenor.com/view/telecurso-2000-tv-globo-aula-gif-9619950

Praticando

Prática

- Implementar texto culturais de diferentes domínios da sua região e ver os resultados dos modelos pré-treinados.
- Como prática, fica a busca e implementação do método Wang2Vec e criação de análises de analogia.

Informações do Material



Sobre o material

Material produzido por Chrystian Arriel Amaral, aluno de Ciência da Computação do DCC/UFLA e membro do Grupo de Estudos em PLN + IA.

Orientação dos professores Erick Galani Maziero e Paula Christina Fiqueira Cardoso, do DCC/UFLA.

Bibliografia - Artigos

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. ***Efficient Estimation of Word Representations in Vector Space***. In ICLRWorkshop Papers.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2016). ***Enriching Word Vectors with Subword Information***.

Pennington, J., Socher, R., and Manning, C. D. (2014). ***Glove: Global vectors for word representation***. Proceedings of the 2014 Conference on Empiricial Methods in Natural Language Processing (EMNLP-2014).

Ling, W., Dyer, C., Black, A., and Trancoso, I. (2015). ***Two/too simple adaptations of word2vec for syntax problems***. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics.

Bibliografia - Artigos

NSS. "***An Intuitive Understanding of Word Embeddings: From Count Vectors to Word2Vec***". Analytics Vidhya, 4 de junho de 2017. Endereço eletrônico: https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/

Mohammed Zeeshan Mulla. "***Word Embeddings in Natural Language Processing | NLP***". Medium, 28 de junho de 2020. Endereço eletrônico: https://medium.com/@zeeshanmulla/word-embeddings-in-natural-language-processing-nlp-5be7d6fb1d73

Christian S. Perone. "***Introdução aos Word Embeddings***". Canal Machine Learning Porto Alegre, na plataforma YouTube, em 2 de fevereiro de 2017. Endereço eletrônico: https://www.youtube.com/watch?v=EVMIR6siWbI

Alex Souza. "***O WORD2VEC ILUSTRADO***". Blog do Zouza, 29 de março de 2019. Endereço eletrônico: https://blogdozouza.wordpress.com/2019/03/29/o-word2vec-ilustrado/

Natasha Latysheva. "***Why do we use word embeddings in NLP?***". Towards Data Science, 10 de setembro de 2019. Endereço eletrônico: https://towardsdatascience.com/why-do-we-use-embeddings-in-nlp-2f20e1b632d2

Halyna Oliinyk. "***Word embeddings: exploration, explanation, and exploitation (with code in Python)***". Towards Data Science, 3 de dezembro de 2017. Endereço eletrônico: https://towardsdatascience.com/word-embeddings-exploration-explanation-and-exploitation-with-code-in-python-5dac99d5d795

Bibliografia - Artigos

Ticiana Linhares. Apresentação "***Resolvendo problemas com Processamento de Linguagem Natural***". Canal Insight Lab, na plataforma YouTube. Endereço eletrônico: https://www.youtube.com/watch?v=acPnPuKe3ik

Usman Malik. "***Python for NLP: Working with Facebook FastText Library***". Stack Abuse. Endereço eletrônico: https://stackabuse.com/python-for-nlp-working-with-facebook-fasttext-library/

Kyubyong Park. "***Pre-trained word vectors of 30+ languages***". Repositorio do GitLab. Endereço eletrônico: https://github.com/Kyubyong/wordvectors

Sunil Ray. "***Understanding and coding Neural Networks From Scratch in Python and R***". Analytics Vidhya, 4 de julho de 2020. Endereço eletrônico: https://www.analyticsvidhya.com/blog/2020/07/neural-networks-from-scratch-in-python-and-r/

Sharmila Polamuri. "***MOST POPULAR WORD EMBEDDING TECHNIQUES IN NLP***". Dataaspirant, 18 de agosto de 2020. Endereço eletrônico: https://dataaspirant.com/word-embedding-techniques-nlp/

Saurabh Pal. "***Implementing Word2Vec in Tensorflow***". Medium, 22 de junho de 2019. Endereço eletrônico: https://medium.com/analytics-vidhya/implementing-word2vec-in-tensorflow-44f93cf2665f

Bibliografia - Artigos

Amit Chaudhary. "***A Visual Guide to FastText Word Embeddings***". Blog Amit Chaudhary, 21 de junho de 2020. Endereço eletrônico: ***https://amitness.com/2020/06/fasttext-embeddings/***

Thushan Ganegedara. "***Intuitive Guide to Understanding GloVe Embeddings***". Towards Data Science, 5 de maio de 2019. Endereço eletrônico: https://towardsdatascience.com/light-on-math-ml-intuitive-guide-to-understanding-glove-embeddings-b13b4f19c010