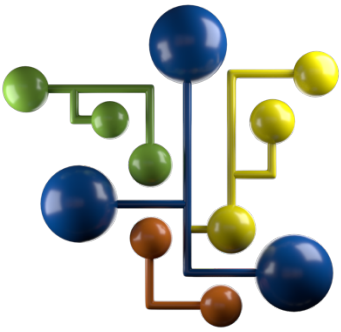


Data Science Academy

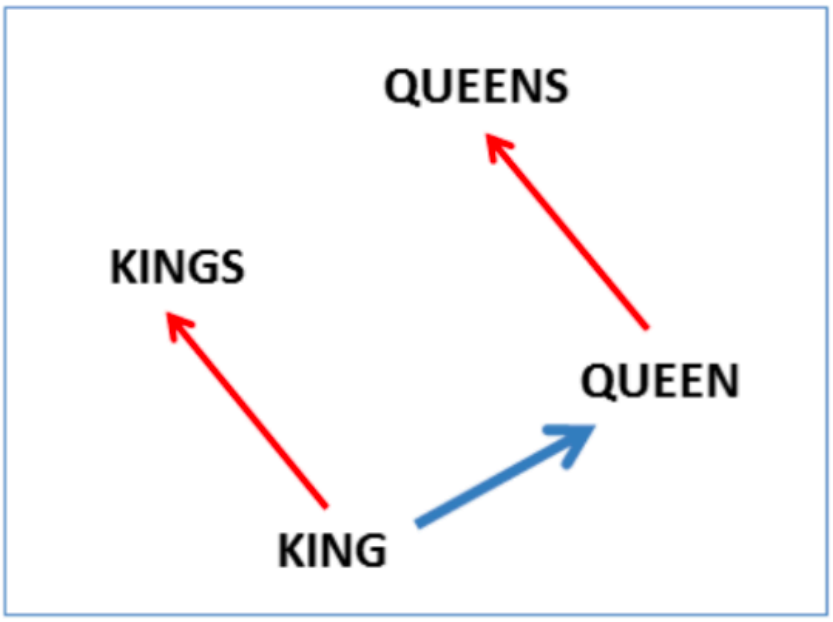
www.datascienceacademy.com.br

Processamento de Linguagem Natural

Compreendendo o Word2vec

Aqui é onde a diversão começa! O Word2Vec é o primeiro “Neural Embedding Model” (ou pelo menos o primeiro que ganhou popularidade em 2013) e ainda é utilizado pela maioria dos pesquisadores. Doc2Vec, seu filho, é o modelo mais popular para representação de parágrafos, que foi inspirado no Word2Vec.

Word2Vec é um método estatístico para aprender eficientemente um Word Embedding independente, a partir de um corpus de texto. Foi desenvolvido por Tomas Mikolov, et al. no Google em 2013 como uma resposta para tornar o treinamento de embeddings baseados em redes neurais mais eficiente e, desde então, tornou-se o padrão de fato para o desenvolvimento de word embeddings pré-treinadas.



Em primeiro lugar, é importante entender que Word2Vec não é um único algoritmo monolítico. De fato, o Word2Vec contém dois modelos distintos (CBOW e Skip-gram), cada um com dois métodos de treinamento diferentes (com / sem amostragem negativa) e outras variações (por exemplo, softmax hierárquico), que equivalem a pequenos "espaços" de algoritmos. Além disso, ele também contém um pipeline de pré-processamento aprimorado, cujos efeitos sobre o desempenho geral foram revisados adequadamente.

Outra coisa que é importante entender, é que Word2Vec não é aprendizagem profunda; tanto o CBOW quanto o Skip-gram são modelos neurais "superficiais". Tomas Mikolov ainda disse que toda a ideia por trás do Word2Vec é demonstrar que você pode obter melhores representações de palavras e a capacidade de aprender com conjuntos de dados muito maiores. Mas Word2Vec é o primeiro passo para conhecer modelos mais avançados e realmente profundos.

A técnica Word2Vec baseia-se em uma arquitetura totalmente conectada de feed-forward. Vamos começar com uma frase simples como:

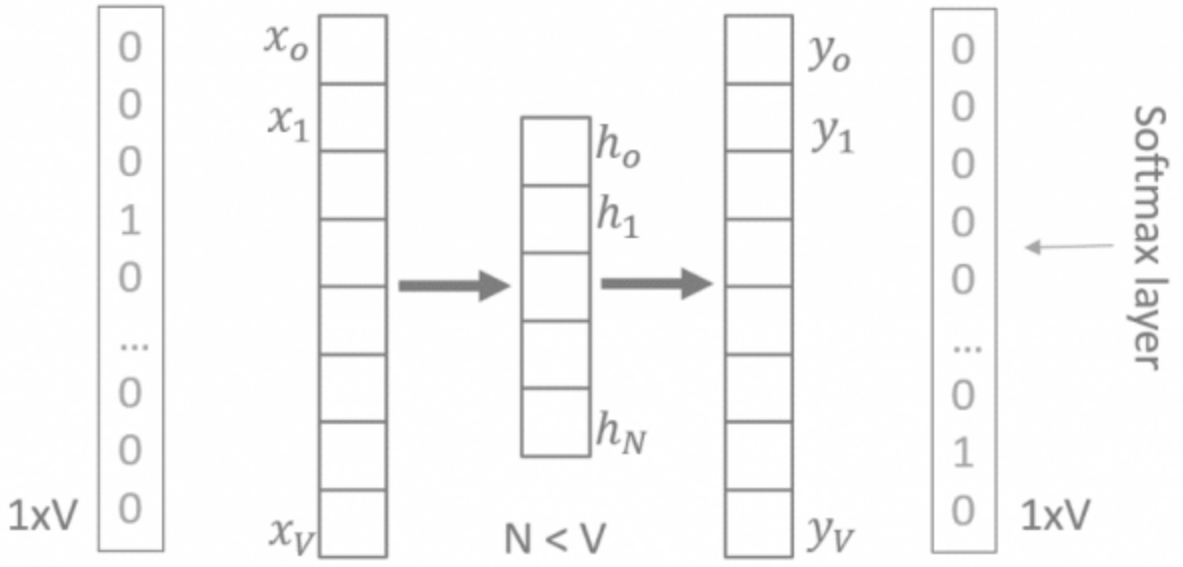
"**the quick brown fox jumped over the lazy dog**"

e vamos considerar o contexto palavra por palavra. Por exemplo, a palavra "fox" é cercada por várias outras palavras; esse é o seu contexto. Se usarmos um contexto de avanço (forward) de tamanho 3, a palavra "fox" depende do contexto "the quick brown"; a palavra "jumped" do contexto "quick brown fox" e assim por diante. Se usarmos um contexto de retrocesso (backward) de tamanho 3, a palavra "fox" depende do contexto "jumped over the"; a palavra "jumped" do contexto "over the lazy" e assim por diante. Se usarmos um contexto central de tamanho 3, o contexto "fox" é "quick brown jump"; o contexto "jumped" é "brown fox" e assim por diante. O tipo mais comumente usado é um contexto de avanço.

Dado um contexto e uma palavra relacionada a esse contexto, enfrentamos dois problemas possíveis:

a. a partir desse contexto, predizer a palavra alvo (Continuous Bag of Words ou abordagem CBOW)
b. da palavra alvo, prever o contexto de onde veio (abordagem Skip-gram)

Vamos nos limitar ao contexto C = 1. Se usarmos uma rede neural totalmente conectada, com uma camada oculta, acabaremos com uma arquitetura como a da figura abaixo para ambas as abordagens.



Os padrões de entrada e saída são vetores one-hot únicos, com dimensão 1xV, onde V é o tamanho do vocabulário. No caso da estratégia CBOW, o contexto one-Hot encoded alimenta a entrada e a palavra one-hot encoded é prevista pela camada de saída. No caso da estratégia Skip-gram, a palavra faltante one-hot encoded alimenta a entrada, enquanto a camada de saída tenta reproduzir o contexto de uma palavra com um vetor one-hot. O número de neurônios ocultos é N, com N < V.

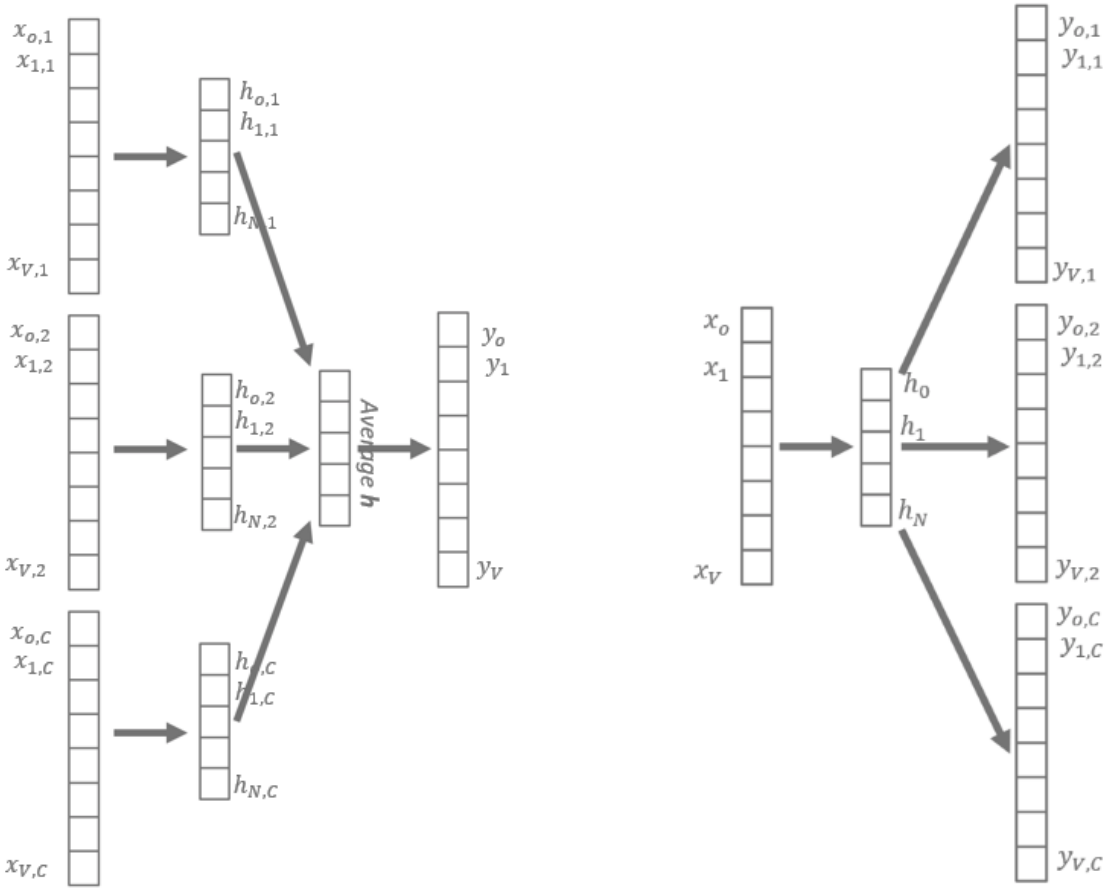
Note que as camadas de entrada e saída têm ambas as dimensões 1xV, onde V é o tamanho do vocabulário, uma vez que ambos representam o vetor one-hot encoding de uma palavra. Observe também que a camada oculta tem menos unidades (N) do que a camada de entrada (V). Então, se eu representar a palavra de entrada com as saídas do neurônio oculto, em vez de com a codificação original, eu já reduzo o tamanho do vetor da palavra, esperançosamente mantendo o suficiente da informação original. As saídas ocultas dos neurônios fornecem a word embedding. Essa representação de palavras, sendo muito mais compacta do que uma codificação simples, produz uma representação muito menos esparsa do espaço do documento. É simples e genial.

Uma função de ativação softmax é usada na camada de saída e a seguinte função de erro é adotada durante o treinamento:

$$E = -log(p(w_o|w_i))$$

onde w0 é a palavra de saída e wi é a palavra de entrada. Ao mesmo tempo, para reduzir o esforço computacional, uma função de ativação linear é usada para os neurônios ocultos e os mesmos pesos são usados (CBOW) ou todas as saídas (Skip-gram).

Se nos movermos para um contexto maior, por exemplo, com tamanho C = 3, precisamos alterar um pouco a estrutura da rede:



Para a abordagem CBOW, precisamos de camadas de entrada C de tamanho V para coletar vetores one-hot encoded das palavras. A camada oculta correspondente fornece então encadeamentos da palavra C, cada um de tamanho N. Para resumir os envios C, uma camada intermediária é adicionada para calcular o valor médio. A camada de saída tenta produzir a representação codificada onde-hot encoded da palavra de destino, com as mesmas funções de ativação e a mesma função de erro que a arquitetura de rede V-N-V na figura com C = 1.

Amostragem Negativa (Negative Sampling)

Às vezes, além dos exemplos positivos clássicos no conjunto de treinamento, exemplos negativos são fornecidos. Exemplos negativos são entradas erradas para as quais nenhuma saída pode ser determinada. Praticamente, os exemplos negativos devem produzir um vetor codificado com um único componente de todos os 0s. A literatura relata que o uso de exemplos negativos pode ajudar na precisão, mesmo que uma explicação clara do motivo ainda não tenha sido fornecida.

Em geral, a abordagem CBOW já é uma melhoria no que diz respeito ao cálculo de uma matriz de co-ocorrência, uma vez que requer menos memória. No entanto, os tempos de treinamento para a abordagem CBOW podem ser muito longos. A abordagem Skip-gram aproveita a amostragem negativa e permite a diversificação semântica de uma palavra. Tudo isso em conjunto, forma o Word2vec.

Referência:

Distributed Representations of Sentences and Documents – Tomas Mikolov
https://cs.stanford.edu/~quoc le/paragraph_vector.pdf