Word 2 vec

CE-299

Matheus Mota e Reuben Katz

Agenda

1 **Contexto**

2 **Quem Usa/ Introdução**

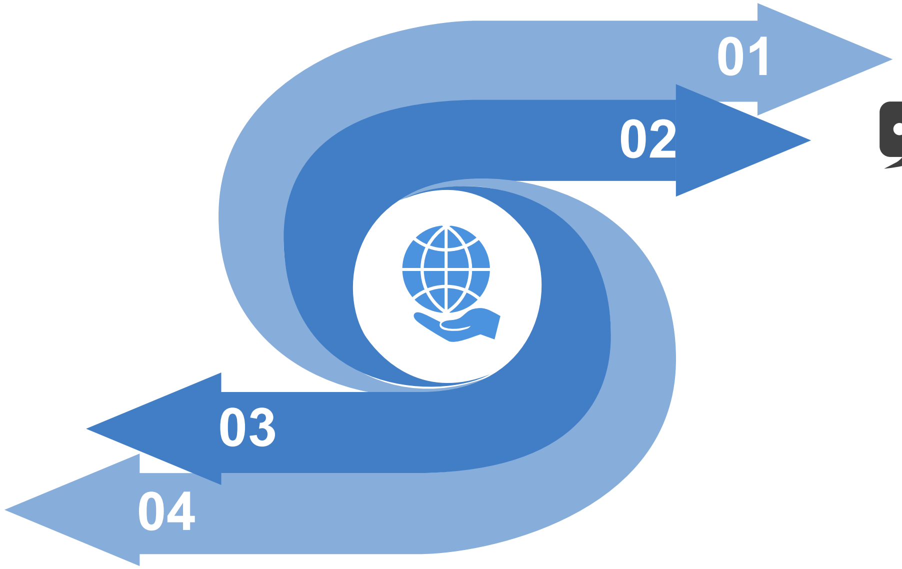
3 **Matemática**

4 **Código**

5 **Dúvidas**

Word Embedding

Natural Language Processing (NLP)



Hipótese Dstribucional

Palavras determinados pelo seu contexto.

Procura do Contexto de uma palavra

Utilizações do Word 2 vec

Área Médica

Tentar mapear medicamentos e relacionar diferentes remédios

Conseguir respostas de formulários

Conseguir identificar, para uma grandes quantidade de dados, avaliações sobre produtos/serviços

Tradução

Tradução de palavras através da identificação de seu contexto

Recomendação de música ou vídeo

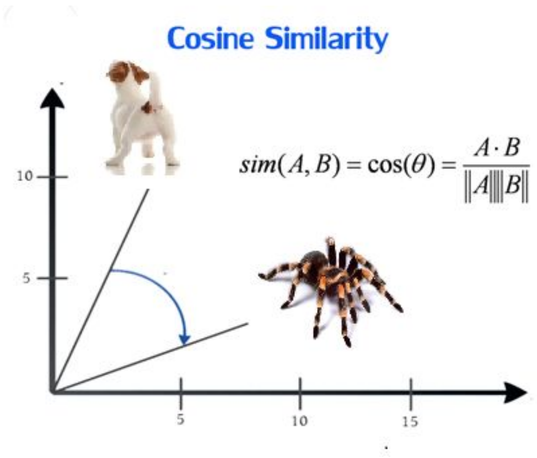
A recomendação de músicas, playlists, etc, é feita por meio de word2vec, onde encontra-se o contexto as letras da música.

Métodos existentes

Negative Sampling
Método otimizado.

Hierarchical softmax
Método otimizado

Skip-gram (SG)
- Método original;
- Insere-se um contexto e cria-se várias distribuições de probabilidade distintas.
- Melhor para um dataset de treino pequeno e é bom para palavras raras.



Conitnuos bag of words (CBOW)
- Método original;
- Pega uma palavra e tenta entender seu contexto;
- Muito mais rápido;
- Melhor para palavras mais frequentes;

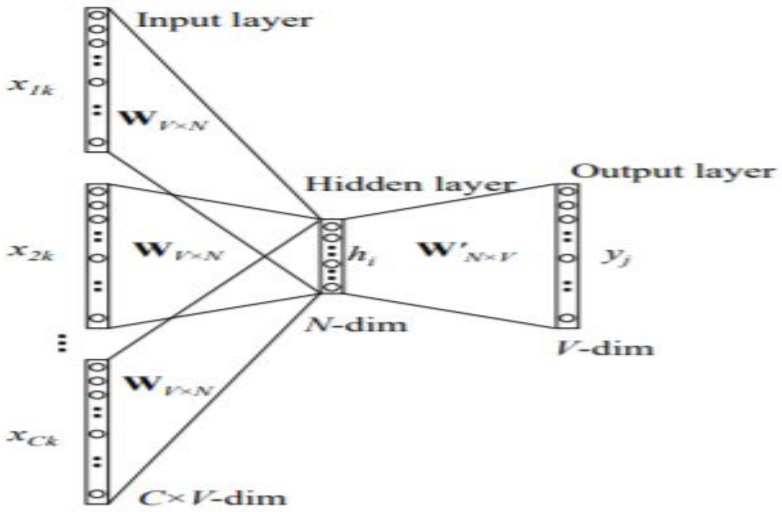


Figure 2: Continuous bag-of-word model

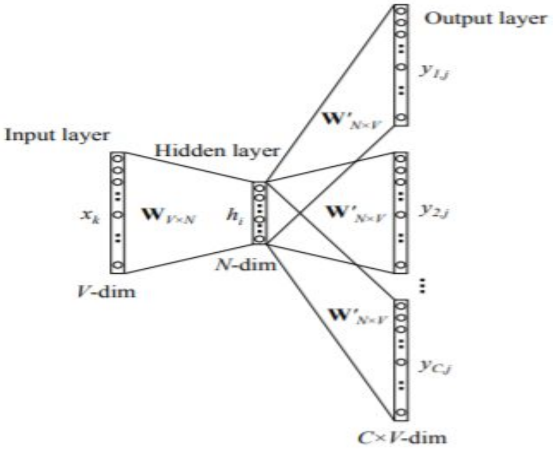


Figure 3: The skip-gram model.

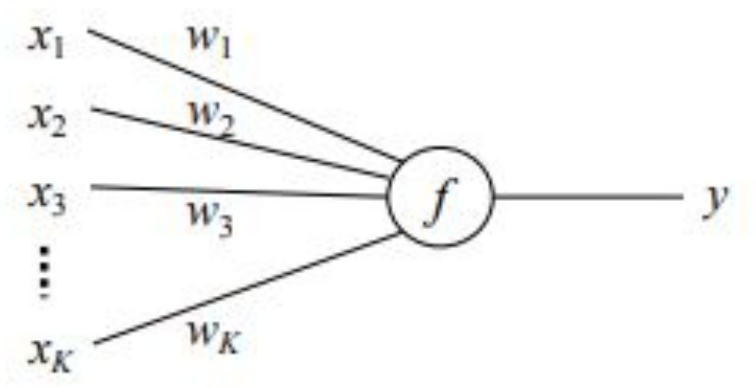
yesterday was a [...] day.
1)Beautiful, nice => Boas palavras
2)Delightful => palavra ruim, por ser menos provável

Dado delightful, dirá que essa palavra provavelmente encaixa na frase yesterday was a [...] day

AI

Back Propagation Basics

Algoritmo de aprendizado para uma unidade



A figura representa um neurônio, única camada.

$\{x_1, \cdots, x_K\}$ Valores de entrada

$\{w_1, \cdots, w_K\}$ Valores de saída

A unidade das palavras segue:

$$y = f(u),$$

Onde u é um número escalar, no qual o novo imput no neurônio é dado por:

$$u = \sum_{i=0}^{K} w_i x_i.$$

$$u = \mathbf{w}^T \mathbf{x}$$

Back Propagation Basics

O primeiro exemplo de escolha de f(u) é sendo a função de unidade.

$$f(u) = \begin{cases} 1 & \text{if } u > 0 \\ 0 & \text{otherwise} \end{cases}$$

Um neurônio com esta função de ligação é chamado de perceptron. O algoritmo de aprendizado para um perceptron é um algoritmo de perceptron.

A equação é definida como:

$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - \eta \cdot (y - t) \cdot \mathbf{x}$$

Onde t é o rótulo e n é a taxa de aprendizagem.

Para um segundo exemplo escolhemos f(u) como a função logística(sigmóide), definida como:

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

A função logística tem as seguintes propriedades:

1) A saída está entre 0 e 1
2) Ao contrário da função da unidade, a função é suave e diferenciável, possibilitando a derivação mais fácil.

$$\sigma(-u) = 1 - \sigma(u)$$

$$\frac{d\sigma(u)}{du} = \sigma(u)\sigma(-u)$$

Back Propagation Basics

Nós usamos o gradiente estocástico para o modelo de algoritmo de aprendizagem. Na ordem de derivada da equação de atualização, nós definimos a função erro.

Para seguir o objetivo da função, utilizaremos a função erro como:

$$E = \frac{1}{2}(t - y)^2$$

Derivando E em relação à wi, temos:

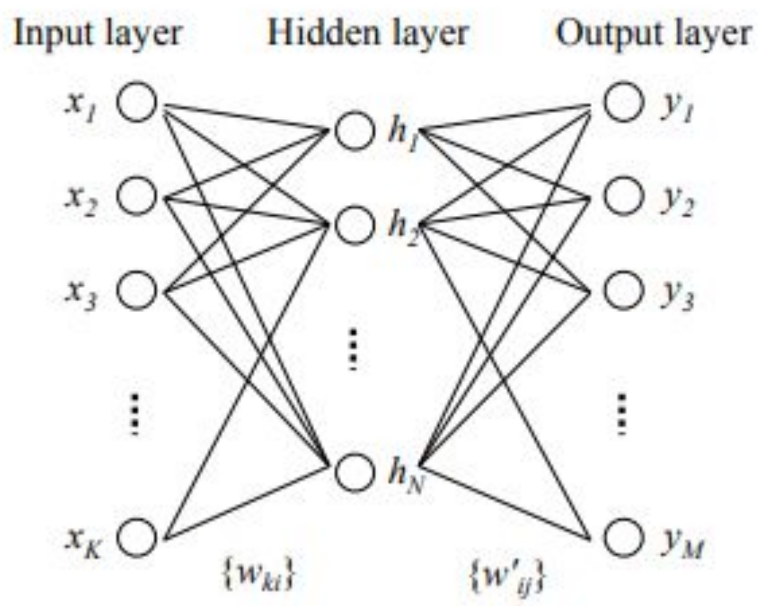
$$\begin{aligned} \frac{\partial E}{\partial w_i} &= \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial u} \cdot \frac{\partial u}{\partial w_i} \\ &= (y - t) \cdot y(1 - y) \cdot x_i \end{aligned}$$

Uma vez que temos a derivada, nós aplicaremos no gradiente estocástico descendente.

$$\mathbf{w}^{(\mathrm{new})} = \mathbf{w}^{(\mathrm{old})} - \eta \cdot (y - t) \cdot y(1 - y) \cdot \mathbf{x}.$$

Back Propagation Basics

Para múltiplas redes neurais, temos:



$$\{x_k\} = \{x_1, \cdots, x_K\}$$

Conjunto com a camada de entrada

$$\{h_i\} = \{h_1, \cdots, h_N\}$$

Conjunto com a camada oculta

$$\{y_j\} = \{y_1, \cdots, y_M\}$$

Conjunto com a camada de saída

Continuous Bag-of-Word Model

Computando a unidade com a função logística, temos a hi com sendo o valor de saída da camada oculta.

$$h_i = \sigma(u_i) = \sigma\left(\sum_{k=1}^{K} w_{ki} x_k\right).$$

De modo similar para yi na camada de saída, definimos como:

$$y_j = \sigma(u'_j) = \sigma\left(\sum_{i=1}^{N} w'_{ij} h_i\right).$$

Usando a soma quadrática da função erro dada por:

$$E(\mathbf{x}, \mathbf{t}, \mathbf{W}, \mathbf{W}') = \frac{1}{2}\sum_{j=1}^{M}(y_j - t_j)^2,$$

$\mathbf{W} = \{w_{ki}\}$, a $K \times N$ Matriz de peso da camada oculta(entrada -> oculta)

$\mathbf{W}' = \{w'_{ij}\}$, a $N \times M$ Matriz de peso da camada oculta(oculta -> saída)

$\mathbf{t} = \{t_1, \cdots, t_M\}$ Vetor M dimensional para os melhores neurônios.

Continuous Bag-of-Word Model

O resultado final da atualização seria:

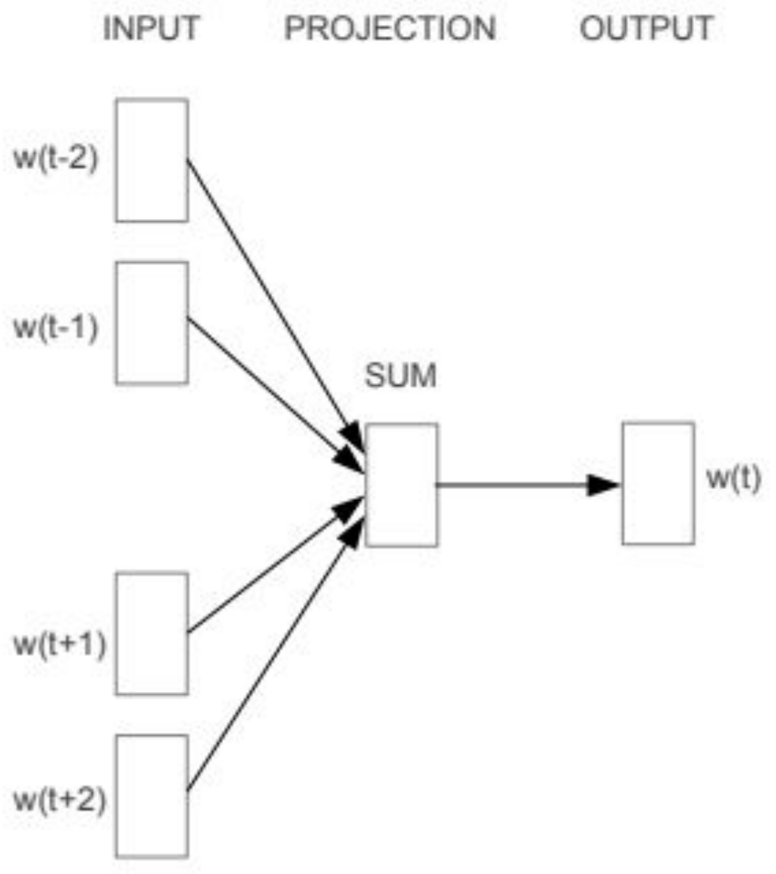
$$w_{ki}^{(\text{new})} = w_{ki}^{(\text{old})} - \eta \cdot \mathrm{EI}_i \cdot x_k.$$

Continuous Bag-of-Word Model

A arquitetura do modelo CBOW tenta prever a palavra de destino atua com base nas palavras de contexto de origem. Considerando uma frase simples,

“the quick brown fox jumps over the lazy dog”

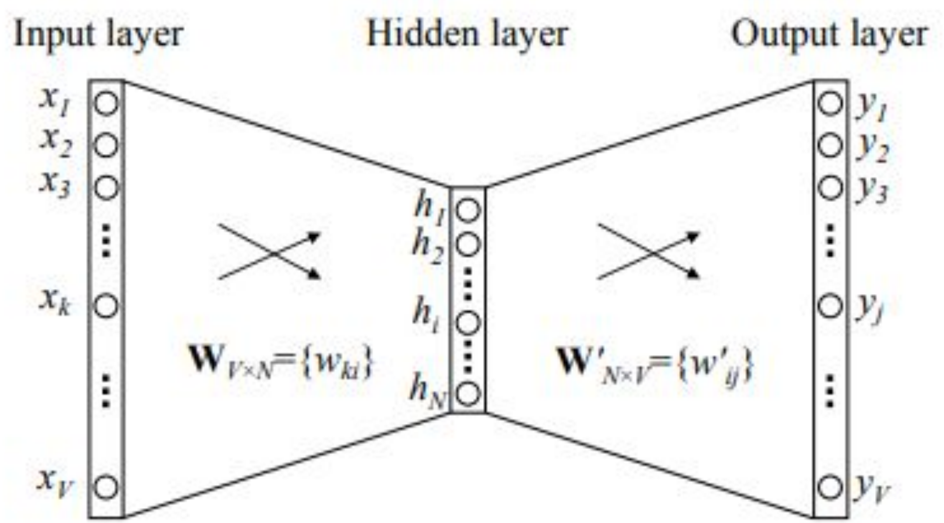
pode ser pares de (context_window, target_word) onde, se considerarmos uma janela de contexto de tamanho 2, temos exemplos como ([quick, fox], brown), [the, brown], quick), ([the, dog], lazy) e assim por diante.
Portanto, o modelo tenta prever a palavra-alvo com base nas palavras context_window.



CBOW

Continuous Bag-of-Word Model

Usaremos o modelo simplificado de rede neural para a definição simplificada do contexto.



As unidades nas camadas adjacentes estão totalmente conectadas. A entrada do vetor codificado é do tipo one-hot, o que significa que para um determinado contexto de entrada de palavras, apenas um dos elementos do vetor terá valor 1 e os outros terão valor zero.

W -> Pesos da camada de entrada(V x N)

$$\mathbf{h} = \mathbf{W}^T\mathbf{x} = \mathbf{W}^T_{(k,\cdot)} := \mathbf{v}^T_{w_I},$$

VwI -> representação do vetor de palavras de entrada.

Continuous Bag-of-Word Model

Da camada oculta à camada de saída, há uma matriz de pesos denominada W'(N x V). Usando esses pesos, podemos calcular uma pontuação uj para cada palavra do vocabulário.

$$u_j = {\mathbf{v}'_{w_j}}^T \mathbf{h},$$

V'wj é a j-ésima coluna da matriz W'.

Então podemos usar o softmax, aplicado à um log para deixar a transformação linear, para obter a distribuição posterior das palavras. Elas são um grupo multinomial distribuído.

$$p(w_j|w_I) = y_j = \frac{\exp(u_j)}{\sum_{j'=1}^{V} \exp(u_{j'})},$$

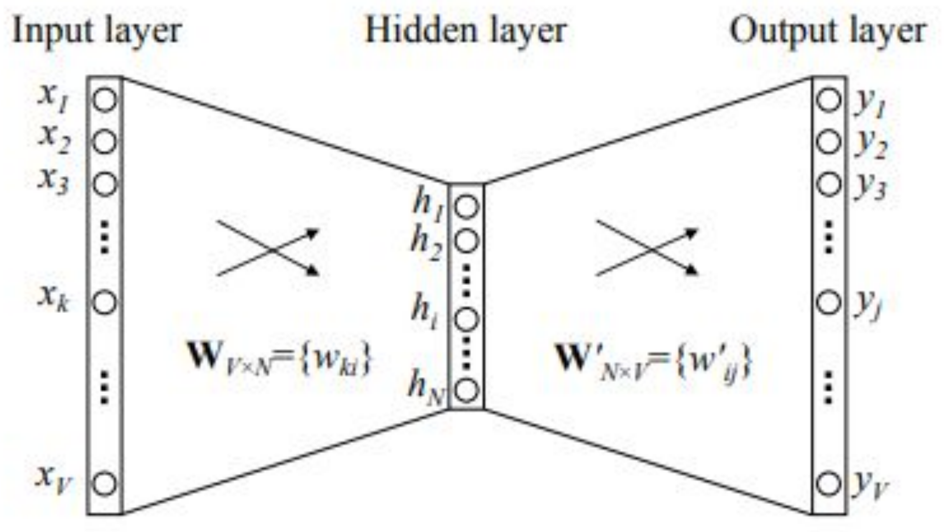
yj -> saída da j-ésima unidade da camada de saída.

Substituindo uj em yj, temos:

$$p(w_j|w_I) = \frac{\exp\left({\mathbf{v}'_{w_j}}^T \mathbf{v}_{w_I}\right)}{\sum_{j'=1}^{V} \exp\left({\mathbf{v}'_{w_{j'}}}^T \mathbf{v}_{w_I}\right)}$$

Vw e V'w são duas representações para para a palavra w.

Continuous Bag-of-Word Model



Poderíamos adotar que:

Vw -> vetor de entrada
V’w -> vetor de saída

Para uma mesma palavra w.

Continuous Bag-of-Word Model

Para a equação da etapa:

Camada Oculta -> Pesos de saída

Vamos derivar a equação de atualização de peso para este modelo.

Os resultados obtidos são parte da teoria da retroprogramação.

O objetivo do treinamento é maximizar a condição de probabilidade da palavra real de saída, w0, dado uma entrada de contextos WI em relação aos pesos.

$$\begin{aligned} \max p(w_O|w_I) &= \max y_{j^*} \\ &= \max \log y_{j^*} \\ &= u_{j^*} - \log \sum_{j'=1}^{V} \exp(u_{j'}) := -E, \end{aligned}$$

$$E = -\log p(w_O|w_I)$$

E é a função de perda, e j* é o índice da palavra atual de saída da camada de saída.

Continuous Bag-of-Word Model

Vamos derivar a equação de atualização dos pesos entre as camadas ocultas e de saída.

Faremos a derivada de E em relação ao input líquido da j-ésima unidade.

$$\frac{\partial E}{\partial u_j} = y_j - t_j := e_j$$

Onde tj = 1 quando a j-ésima unidade é a palavra atual de saída, caso contrário tj = 0.

Essa derivada já representa o erro ej da camada de saída.

Em seguida, tomamos a derivada em relação à W’ij para obter o gradiente de:

Camada oculta -> Pesos de saída

$$\frac{\partial E}{\partial w'_{ij}} = \frac{\partial E}{\partial u_j} \cdot \frac{\partial u_j}{\partial w'_{ij}} = e_j \cdot h_i$$

Portanto, usando a descida do gradiente estocástico, obtemos a equação de atualização de peso para:

Camada oculta -> Pesos de saída

Continuous Bag-of-Word Model

$$w'_{ij}{}^{(\text{new})} = w'_{ij}{}^{(\text{old})} - \eta \cdot e_j \cdot h_i.$$

$$\mathbf{v}'_{w_j}{}^{(\text{new})} = \mathbf{v}'_{w_j}{}^{(\text{old})} - \eta \cdot e_j \cdot \mathbf{h} \qquad \text{for } j = 1, 2, \cdots, V.$$

Onde n>0 é a taxa de aprendizado.

Continuous Bag-of-Word Model

Para a equação da etapa:

Entrada -> Pesos ocultos

Tendo obtido as equações de atualização para w0, agora podemos passar para w. Tomamos a derivada de E na saída da camada oculta, obtendo

$$\frac{\partial E}{\partial h_i} = \sum_{j=1}^{V} \frac{\partial E}{\partial u_j} \cdot \frac{\partial u_j}{\partial h_i} = \sum_{j=1}^{V} e_j \cdot w'_{ij} := \mathrm{EH}_i$$

Lembrando que hi é a combinação linear das palavras com os pesos,

$$h_i = \sum_{k=1}^{V} x_k \cdot w_{ki}$$

Pegaremos a derivada de E em W

$$\frac{\partial E}{\partial w_{ki}} = \frac{\partial E}{\partial h_i} \cdot \frac{\partial h_i}{\partial w_{ki}} = \mathrm{EH}_i \cdot x_k$$

Isso é equivalente para o produto dos tensores x e EH.

$$\frac{\partial E}{\partial \mathbf{W}} = \mathbf{x} \otimes \mathrm{EH} = \mathbf{x}\mathrm{EH}^T$$

Continuous Bag-of-Word Model

A partir do qual obtemos uma matriz V x N. Como apenas uma componente de x é diferente de zero, apenas uma linha da derivada de E em relação à W é diferente de zero e o valor dessa linha é EH transposta.

Nós obtemos a equação de atualização como:

$$\mathbf{v}_{w_I}^{(\text{new})} = \mathbf{v}_{w_I}^{(\text{old})} - \eta \mathrm{EH}^T$$

Continuous Bag-of-Word Model

Para múltiplos contextos de palavras, há uma generalização da teoria.

Ao computar a saída da camada oculta, em vez de copiar o vetor de entrada do contexto da palavra de entrada, o modelo CBOW leva a média dos vetores das palavras do contexto de entrada e usa o produto da equação (camada de entrada -> matriz de pesos ocultos) e o vetor médio como saída.

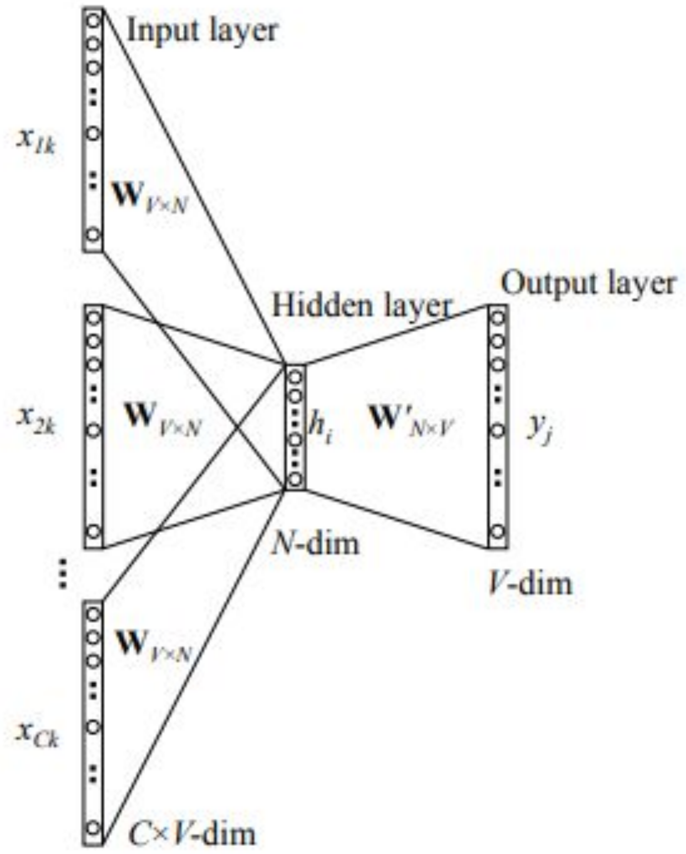
$$
\begin{aligned}
\mathbf{h} &= \frac{1}{C}\mathbf{W}^T(\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_C) \\
&= \frac{1}{C}(\mathbf{v}_{w_1} + \mathbf{v}_{w_2} + \cdots + \mathbf{v}_{w_C})^T
\end{aligned}
$$

C -> número de palavras no contexto

A função perda é:

$$
\begin{aligned}
E &= = -\log p(w_O | w_{I,1}, \cdots, w_{I,C}) \\
&= -u_{j^*} + \log \sum_{j'=1}^{V} \exp(u_{j'}) \\
&= -\mathbf{v}'_{w_O}{}^T \cdot \mathbf{h} + \log \sum_{j'=1}^{V} \exp(\mathbf{v}'_{w_j}{}^T \cdot \mathbf{h})
\end{aligned}
$$

Continuous Bag-of-Word Model



A equação de atualização (pesos ocultos -> saída) permanece a mesma para o modelo do contexto de uma palavra:

$$\mathbf{v}'_{w_j}{}^{(\text{new})} = \mathbf{v}'_{w_j}{}^{(\text{old})} - \eta \cdot e_j \cdot \mathbf{h} \qquad \text{for } j = 1, 2, \cdots, V.$$

Precisamos aplicar isso à todos os elementos da matriz de (pesos de saída -> camada oculta) para cada instância do treinamento.

Continuous Bag-of-Word Model

A equação de atualização para (entrada -> pesos ocultos) é semelhante a:

$$\mathbf{v}_{w_I}^{(\text{new})} = \mathbf{v}_{w_I}^{(\text{old})} - \eta \mathrm{EH}^T$$

Exeto que agora é necessário aplicar a equação para cada palavra no contexto.

Sendo cada componente definida como:

$$\mathrm{EH}_i = \sum_{j=1}^{V} \mathrm{EI}_j \cdot w'_{ij}.$$

Continuous Bag-of-Word Model

A simulação do modelo matemático apresentado é mostrado no site:

https://ronxin.github.io/wevi/

AI

Artificial Intelligence

Jupyter Notebook

Hand's

on!

Referências

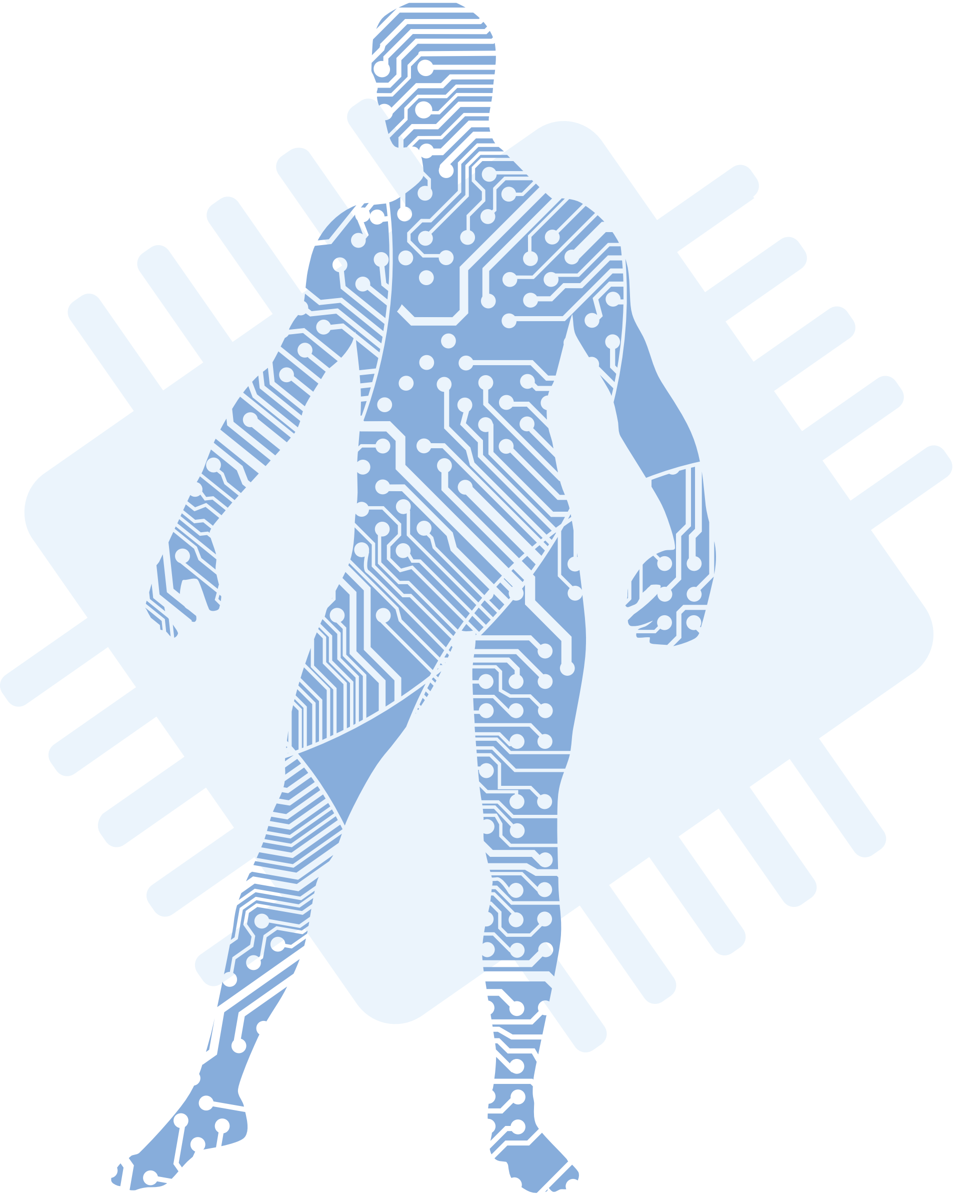
01 https://arxiv.org/abs/1301.3781 (paper original da google)

02 https://arxiv.org/pdf/1310.4546.pdf (paper original da google extensão)

03 https://www.deeplearningweekly.com/blog/demystifying-word2vec/ (problemas similares)

04 https://arxiv.org/ftp/arxiv/papers/1502/1502.03682.pdf (paper aplicação word 2 vec área médica)

05 https://arxiv.org/pdf/1411.2738.pdf (matemática por trás do word 2 vec)



Obrigado

Perguntas