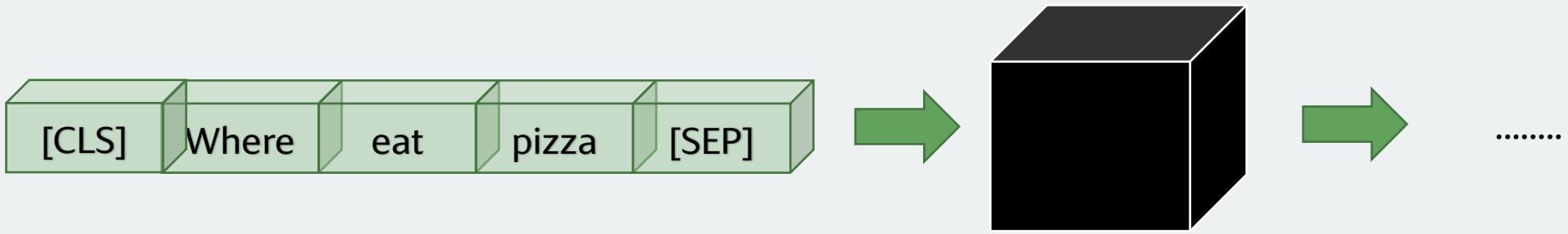
Notebook – Splade

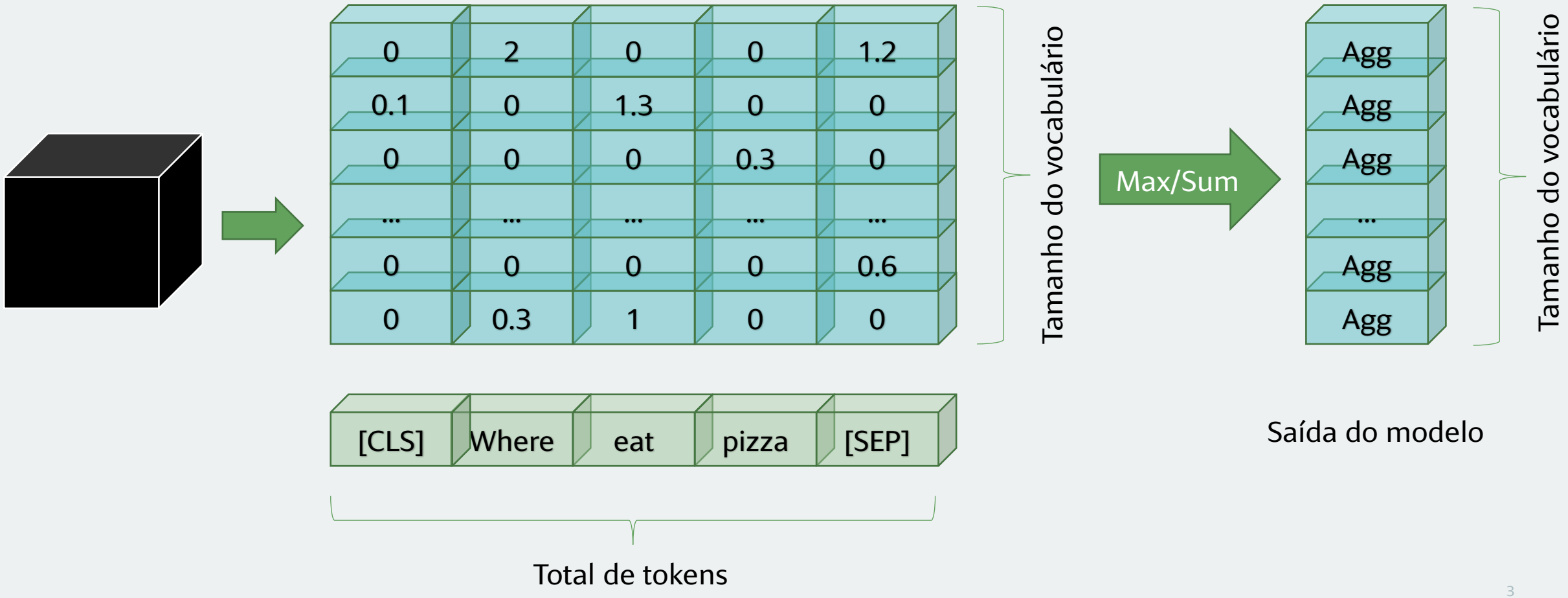
Leandro Carísio

Conceitos do exercício

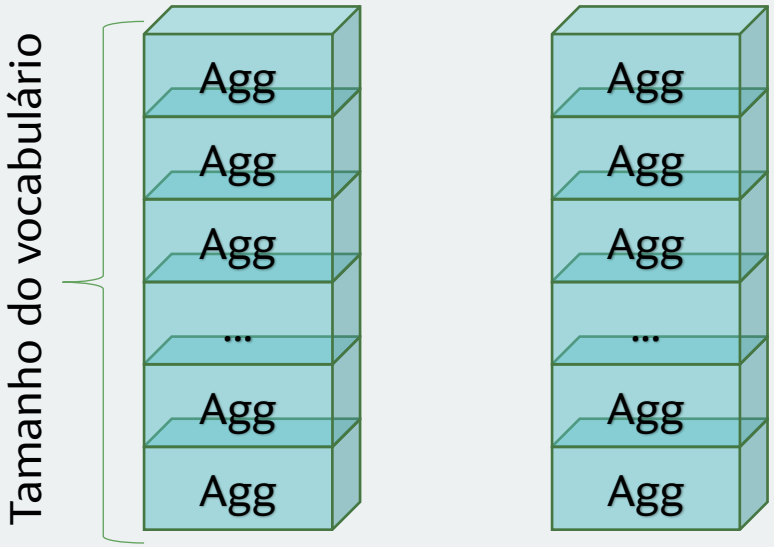


Texto de entrada

Conceitos do exercício



Conceitos do exercício



Score(d, q) = produto escalar entre d e q

No SpladeV2 os elementos do vetor da query são {0, 1}.

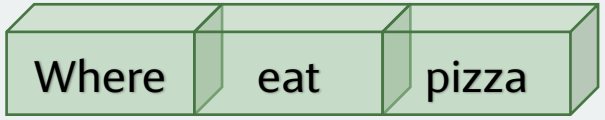
⇒ Se o corpus for pequeno, dá pra guardar a matriz de documento e resolver com uma multiplicação de matrizes.

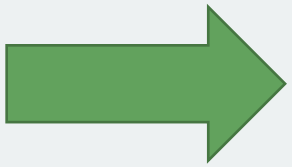
⇒ Se o corpus for grande, dá pra resolver com um índice invertido guardando como chave o token (ou token_id) e a lista de documento e score do documento

Problemas e soluções no desenvolvimento

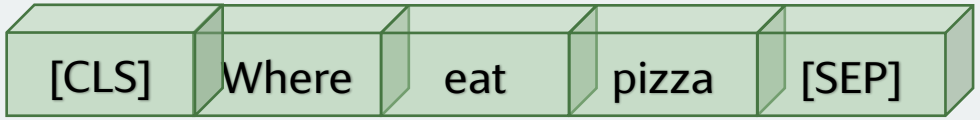
1. Resultado da primeira simulação era nDCG = 0.0000

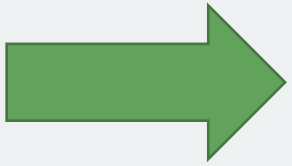
Solução: Eu estava gerando os textos sem os tokens [CLS] e [SEP]





are, him, where, food, phone, movie, store, eat, pizza



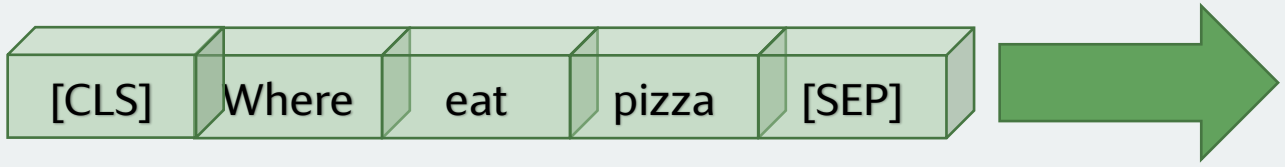


to, where, later, food, guy, places, box, mike, joe, tony, eat, restaurant, alan, cole, dick, bite, pat, package, cab, pizza, toby, domino

Problemas e soluções no desenvolvimento

2. naver/splade_v2_distil estava gerando resultado nada a ver:

Solução após ver caderno do Pedro Gabriel: Usar DistilBertTokenizer e DistilBertForMaskedLM (ou usar os AutoModels...)



BertTokenizer
BertForMaskedLM

('ব', tensor(0.0004)),
('শ', tensor(0.1958)),
('ষ', tensor(0.6900)),
('স', tensor(0.1393)),
('া', tensor(0.7554)),
('ি', tensor(0.5658)),
('ী', tensor(0.3767)),
('ো', tensor(0.7418)),
('ং', tensor(0.1557)),
('ঃ', tensor(0.6785)),
('অ', tensor(0.2900)),
('আ', tensor(0.3225)),
('ই', tensor(0.6525)),
('উ', tensor(0.2975)),

('ে', tensor(0.4884)),
('ক', tensor(0.4602)),
('চ', tensor(0.5440)),
('ট', tensor(0.1255)),
('ত', tensor(0.0173)),
('ন', tensor(0.1367)),
('ণ', tensor(0.2874)),
('প', tensor(0.4996)),
('ম', tensor(0.8778)),
('য', tensor(0.0894)),
('ল', tensor(0.3731)),
('া', tensor(0.2420)),
('ী', tensor(0.3318)),
('ু', tensor(0.4002)),

('ৃ', tensor(0.5233)),
('ৄ', tensor(0.8806)),
('ৗ', tensor(0.4128)),
('ৢ', tensor(0.2532)),
('ৣ', tensor(0.3629)),
('৯', tensor(0.8841)),
('ৰ', tensor(0.5743)),
('ৱ', tensor(0.1146)),
('৲', tensor(0.1561)),
('৳', tensor(0.0676)),
('৴', tensor(0.1247)),
('৵', tensor(0.3174)),
('৶', tensor(0.2280)),
('৷', tensor(0.1788)),
('৸', tensor(0.2981)),

Truques de código que funcionaram

Ordenar o dataset antes de indexar e usar fp16 (ideia de Marcos Piau) diminui o tempo de processamento

Resultados interessantes/inesperados

A ordem dos fatores altera o resultado:

where eat pizza: anywhere best cafe country culture eat eating famous farm favorite food habitat headquarters hotel location locations murphy pie pizza place places restaurant restaurants shop that venue visit where

pizza where eat: cafe country culture eat eating famous food habitat headquarters hotel italy location locations menu murphy pie pizza place places position restaurant restaurants venue where

Resultados interessantes/inesperados

Muitos tokens são extraídos do CLS/SEP (where eat pizza)

Com contribuição do CLS/SEP: anywhere baker best booth bowl cafe church city country culture dave diner dish domino don eat eating eden famous farm favorite find food foods garden habitat headquarters hotel hut italy joe kitchen location locations meat murphy pie pizza place places restaurant restaurants sandwich shop store that variety venue visit website where york

Sem contribuição do CLS/SEP: anywhere best cafe country culture eat eating famous farm favorite food habitat headquarters hotel location locations murphy pie pizza place places restaurant restaurants shop that venue visit where

Apesar disso, parece que o resultado final não impacta muito. Em um dos testes, nDCG@10 de 0.7276 (removendo) versus 0.7290 (não removendo).

=> Isso reduz o tamanho do índice sem muita perda de qualidade

Resultados

Método	nDCG@10
BM25 (Aula 5)	0,5956
BM25 doc. original + expansão (Aula 5)	0,6719
Implementação busca densa (Aula6)	0,3322
Resultados desta aula - naver/splade-cocondenser-ensembledistil	
max, sem contrib. SEP e CLS, f32, matriz	0,7276
max, f32, matriz	0,7290
max, f32, índice	0,7282
max, f16, índice	0,7269
max, sem contrib. SEP e CLS, f16, índice	TODO
sum, f16, índice	TODO
sum, f32, matriz	0,1931

Tópico para discussão

1. A implementação usando a função SUM como agregação deu um resultado muito ruim.

2. Calcular o score conforme proposto no SPLADEv2 piorou os resultados.

3. Parece que a melhor combinação foi a função agregação MAX (SPLADEv2) com o score calculado conforme SPLADEv1.

Faz sentido isso? Ou foi algum bug no desenvolvimento?

Obrigado

Leandro Carísio

carisio@gmail.com