NGramy w klasyfikacji i generowaniu tekstu

mgr inż. Dawid Wisniewski - Przetwarzanie języka naturalnego

15 March 2020

Laboratoria 3 i

Na poprzednich laboratoriach tworzyliśmy reprezentację wektorową dokumentów, w której każdej pozycji wektora przypisana była obecność pojedynczego słowa. Pod tą pozycją zapisywaliśmy ile razy dane słowo się pojawiło w naszym dokumencie (reprezentacja ta nazywa się bag of words – https://machinelearningmastery.com/gentle-introduction-bag-words-model/).

Bag of words jednak nie jest idealne, gdyż nie bierze pod uwagę kolejności następowania po sobie słów w zdaniu. *The cat likes Mike* i *Mike likes the cat* mimo innego znaczenia będą jako wektory Bag of words reprezentowane identycznie (!).

Laboratoria 3 ii

Jak zatem żyć panie prezydencie?

Stwórzmy wektor, w którym nie będziemy reprezentować pojedynczych słów(tokenów) w jednej kolumnie, a n następujących po sobie słów(tokenów).

Definicja

N-gram - sekwencja n następujących po sobie elementów (tokenów/znaków).

Często krótkie n-gramy mają swoje nazwy:
1-gram: unigram (te już używaliśmy - to pojedyncze słowa
2-gram: bigram
3-gram: trigram

Laboratoria 3 iii

Reprezentując n-tki słów, przemycimy trochę informacji o znaczeniu zdania.

- Dokument1: the cat likes Mike.
- Dokument2: Mike likes the cat.
- Dokument3: the cat likes milk.

Stokenizujmy dokumenty i wygenerujmy z nich wszystkie 2-gramy tokenów (2-gram reprezentowany wewnątrz [ ]):

- Dokument1: [the cat], [cat likes], [likes Mike], [Mike .]
- Dokument2: [Mike likes], [likes the], [the cat], [cat .]
- Dokument3: [the cat], [cat likes], [likes milk], [milk .]

Laboratoria 3 iv

Zacznijmy od stworzenia słownika, czyli zbioru wszystkich widzianych w dokumentach n-gramów (tutaj 2-gramów) zachowując tylko unikalne wpisy:

Vocabulary = {[the cat], [cat likes], [likes Mike], [Mike .], [Mike likes], [likes the], [cat .], [likes milk], [milk .]}

Zwróć uwagę jak jest rozmiar słownika opartego o bigramy w stosunku do słownika opartego o pojedyncze tokeny (unigramy)

A następnie osadźmy każdy dokument w przestrzeni wektorowej, gdzie i-ta pozycja w wektorze odpowiada i-temu wpisowi ze słownika. Pod i-tą pozycją w wektorze zapiszmy ile razy dany n-gram objawił się w naszym dokumencie.

Laboratoria 3 v

- Dokument1: [1, 1, 1, 1, 0, 0, 0, 0, 0]
- Dokument2: [0, 0, 0, 0, 1, 1, 1, 0, 0]
- Dokument3: [1, 1, 0, 0, 0, 0, 0, 1, 1]

Z użyciem 2-gramów (bigramów) dokumenty 1 i 2 są reprezentowane za pomocą innych wektorów (w przeciwieństwie do unigramów)!. W ten sposób przemycamy trochę informacji o znaczniu i gramatyce.

Do stworzenia wektora cech możemy użyć pewnego przedziału n-gramów, np. wszystkich unigramów i bigramów. Dla zadanych wyżej dokumentów słownik w tej sytuacji będzie sumą słowników dla unigramów i bigramów.

Vocabulary = {[the], [cat], [likes], [Mike], [.], [milk], [the cat], [cat likes], [likes Mike], [Mike .], [Mike likes], [likes the], [cat .], [likes milk], [milk .]}

Laboratoria 3 vi

NGramy w generowaniu tekstów.

Czy marzyłeś/marzyłaś o własnej fabryce fake newsów?

Czy chciałeś/chciałaś dołączyć do internetowych specjalistów od wszystkiego, generując mądre, dziedzinowe treści niskim kosztem?

Już dziś, dzięki przedmiotowi PJN, będziesz mógł/mogła osiągnąć to jednym prostem sposobem!

Poznaj NGramowy model języka - może-nie-najlepszy-ale-prosty-do-wytłumaczenia model języka (zapoznajcie się z wstępem do rozdziału 3 i sekcją 3.1 z https://web.stanford.edu/~jurafsky/slp3/3.pdf).

Definicja

Model języka - Funkcja przypisująca sekwencji tokenów (zdaniom) prawdopodobieństwa. $P(W) = P(w_1, w_2, \ldots, w_n)$, gdzie w_i to i-ty token (słowo).

Może być użyty do wyznaczania prawdopodobieństwa słowa (tokenu) następnego względem poprzedzającej, zdefiniowanej sekwencji słów (tokenów): $P(w_4|w_1, w_2, w_3)$

Laboratoria 3 viii

Wyobraźmy sobie, że mamy dokument zaczynający się od słów **Ala ma** i chcemy wygenerować token po tokenie kolejne słowa(tokeny) w sposób automatyczny.

Zbierzmy (najlepiej jak największy) zbiór danych zawierających zdania i przyjrzyjmy się czy sekwencja **Ala ma** gdzieś w nich występuje:

- **Ala ma** kota i psa.
- **Ala ma** koronawirusa.
- **Ala ma** kota i poprawkę z MP
- **Ala ma** kota syjamskiego.

Laboratoria 3 ix

W naszym hipotetycznym korpusie (zbiorze) danych mamy 4 wystąpienia sekwencji **Ala ma**. Czy możemy je jakoś wykorzystać do stworzenia kontynuacji naszego zdania?

Pewnie! wiedząc, że spośród wszystkich (4) wystąpień sekwencji **Ala ma** mamy: 3 razy słowo **kota** jako kontynuację i raz **koronawirusa** możemy poprzez zwykłe zliczanie stwierdzić, że prawdopodobieństwo słowa **kota** będącego kontynuacją wynosi 0.75 (zobacz fragment książki z slajdu 6).

$$P(kota|Ala, ma) = \frac{Count(Ala,ma,kota)}{Count(Ala,ma)} = 3/4$$

Możemy zatem wylosować spośród zadanych dwóch słów kota i koronawirusa kontynuację, zgodnie z wyliczonymi prawdopodobieństwami.

Następnie, zakładając, że wybraliśmo słowo **kota**, moglibyśmy stworzyć kontynuację, w sposób analogiczny zliczając ile razy jaki token pojawił się po sekwencji **Ala ma kota**.

Jednak dla długich sekwencji będziemy mieli problem - jeśli nasza wygenerowana sekwencja będzie miała np. 10 tokenów **Ala ma pięknego, szarego, amerykańskiego kota wabiącego się Michał** - jest bardzo mała szansa, że taka sekwencja pojawiła się gdziekolwiek, a nawet jeśli się pojawiła, to pewnie w bardzo niewielu kontekstach zawężając nam znacznie możliwości wyboru kontynuacji.

Z tego powodu korzystamy z założenia Markowa (Markov assumption), które mówi nam, że dobrym przybliżeniem w procesie wyboru kontynuacji może być ograniczenie się do n-ostatnich elementów sekwencji, tj. zamiast brać pod uwagę wszystkie 10 słów jak w przykładzie, weźmy tylko ostatnie np. 2 i poszukajmy w tekście kontynuacji dla tych dwóch słów (tokenów). Zwiększy to znacząco ilość możliwych kontynuacji.

$P(w_7|w_1, w_2, w_3, w_4, w_5, w_6) \approx P(w_7|w_5, w_6)$

Laboratoria 3 xii

Zatem w przykładzie ze slajdu 8, zakładając, że odnosimy się do ostatnich 2 tokenów, szukając kontynuacji dla **Ala ma kota** weźmy pod uwagę ostatnie 2 tokeny **ma kota** i sprawdźmy jakie kontynuacje w zbiorze zdań znaleźliśmy:

- Ala **ma kota** i psa.
- Ala **ma kota** i poprawkę z MP
- Ala **ma kota** syjamskiego.
- Piotr **ma kota** i owcę.

Widzmy zatem, że słowo **i** następuje w 3/4 przypadków po **ma kota**, a w 1/4 przypadków znaleźliśmy słowo **syjamskiego**.

Laboratoria 3 xiii

Znów, wybieramy losowo kontynuację zgodnie z wyliczonym rozkładem prawdopodobieństwa i znów powtarzamy proces, kończąc go np. po wygenerowaniu kropki.