

Word Embeddings - gęste wektory reprezentujące znaczenie słów

mgr inż. Dawid Wisniewski - Przetwarzanie języka naturalnego

22 March 2020

Na poprzednich laboratoriach (lab1 i lab2) poznaliśmy wektorowe reprezentacje **dokumentów**, gdzie każda pozycja w wektorze reprezentowała inne słowo, lub sekwencję słów/znaków (n-gram).

Jak widzieliśmy, reprezentacja ta dawała dość dobre rezultaty w problemach klasyfikacji. Niestety, taka reprezentacja posiada bardzo ograniczoną wiedzę o świecie.

Weźmy jako przykład dwa dokumenty:

- Dokument1: Ala ma kota
- Dokument2: Alicja posiada kotka

Reprezentacje, które poznaliśmy nie zawierają informacji o znaczeniu słów, zliczają one jedynie ile razy dany ciąg znaków pojawił się w tekście i zapisują odpowiednią informację w wektorze liczb.

Jeśli posiadamy słownik utworzony z powyższych dokumentów: {Ala, ma, kota, Alicja, posiada, kotka}, to reprezentacje bag-of-words tych dokumentów wyglądają następująco:

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

Nie da się z tych wektorów wywnioskować, że kotek, to coś podobnego do kota, Alicja to duża Ala, a 'ma' i 'posiada' to synonimy.

Jak zatem reprezentować słowa, aby zawrzeć w tej reprezentacji informacje o znaczeniu?

Brytyjski lingwista John R. Firth w 1957 roku wypowiedział zdanie, które miało wielki wpływ na NLP:

On powiedział

You shall know a word by the company it keeps

I rzeczywiście, znaczenie słowa wypływa z kontekstu w jakim to słowo występuje. Ile to razy czytając tekst w obcym języku byliśmy w stanie odkryć znaczenie nieznanego nam słowa na podstawie słów okalających nieznaną wyraz?

Przykład

Kupiłem XXX, ma dobrą kartę graficzną i idealnie mieści się do plecaka.

Z bardzo dużym prawdopodobieństwem, XXX to laptop.

Cwane, ale jak zrobić z tego wektory?

1. Weźmy bardzo duży zbiór dokumentów, im więcej tym lepiej.
2. Ustalmy sobie rozmiar okienka, np. 4 tokeny w lewo i 4 w prawo od aktualnego słowa (tokenu).
3. Przeczytajmy te dokumenty token po tokenie, sprawdzając z czym (z jakimi tokenami) dany token współwystępował wewnątrz okienka.

Zapiszmy informacje o współwystąpieniach za pomocą macierzy współwystąpień.

Otrzymamy bardzo dużą macierz. Ponieważ dowolne słowo, w teorii może współwystępować z dowolnym innym słowem, rozmiar tej macierzy to: **ilosc unikalnych tokenów we wszystkich dokumentach razem do kwadratu** (czyt. dużo więcej niż byśmy sobie tego życzyli).

Jeśli różnych tokenów we wszystkich dokumentach jest 20000, macierz ta, zaimplementowana naiwnie (nie jako sparse matrix), będzie prawdopodobnie rozmiaru $20000 * 20000 * \text{ilość bajtów do przechowania potencjalnie dużych liczb}$ (załóżmy 4 bajty), czyli około 1.5 GB.

Jednak wiersze (bądź kolumny przed podjęciem kolejnych kroków) będą miały 2 ważne cechy:

- Pojedynczy wektor będzie bardzo długi
- Ale będzie miał w sobie informację o znaczeniu, gdyż zawrze informacje o kontekście w jakim słowo było widziane w korpusie

Kiedy porównamy reprezentacje wektorowe słów kot i kotek, prawdopodobnie często na tych samych pozycjach będą występować elementy niezerowe i ich wartości będą pewnie dość podobne.

Np i kot i kotek współwystępować będą często ze słowami: kuweta, pazury, karma (duże wartości niezerowe w kolumnach powiązanych z wymienionymi słowami) i oba bardzo rzadko (jeśli w ogóle) współwystępować będą ze słowami: DiscoPollo, Wiktuał czy Koronawirus.

Używając odpowiedniej miary możemy zmierzyć to podobieństwo. Bardzo często używa się tzw. podobieństwa kosinusowego, która reprezentuje kosinus kąta pomiędzy dwoma wektorami.

$$\text{similarity}(\vec{v1}, \vec{v2}) = \frac{\vec{v1} \cdot \vec{v2}}{\|\vec{v1}\| \|\vec{v2}\|}$$

Innymi słowy: podobieństwo kosinusowe to ułamek; dzielimy sumę iloczynów odpowiadających sobie elementów wektora przez iloczyn długości (euklidesowych) tych wektorów.

Tam, gdzie dwa wektory na tej samej pozycji będą miały wartości niezerowe o tym samym znaku, tam będzie wyższa wartość miary kosinusowej. Im więcej takich sytuacji tym wyższe podobieństwo.

Redukcja wymiarowości.

Okazuje się, że tak długie wektory, które otrzymaliśmy możemy „kompresować”, tj. zmniejszyć ich długość zachowując tylko najważniejsze informacje zakodowane w pierwotnym wektorze. Istnieje wiele sposobów na redukcję wymiarowości, dobrym źródłem informacji jest np.

(<https://pbiecek.github.io/NaPrzelajDataMiningR/part-2.html> lub <http://www.cs.put.poznan.pl/jstefanowski/gi/redukcjawymiarowPCA.pdf>).

Wtedy, znaczenie każdego słowa można reprezentować za pomocą zadanej liczby elementów w wektorze (np. 300), gdzie wektor 300-elementowy jest skompresowaną reprezentacją pierwotnego, wynikającego z macierzy współwystąpień.

Word embeddings

Word embeddings to reprezentacje wektorowe znaczeń słów, gdzie wektory są gęste (mają bardzo mało elementów zerowych) i o ograniczonej długości (najczęściej z zakresu 50 - 1000 elementów).

Okazuje się, że word embeddings mają ciekawe własności, gdyż kodują nie tylko znaczenie jako takie, ale za pomocą dodawania i odejmowania wektorów, możemy wyznaczyć analogie (patrz jedno z zadań na dzisiejszy tydzień).

Podjęcia oparte o sieci neuronowe.

W ostatniej dekadzie pojawiły się dużo wydajniejsze metody generowania embeddingów niż przedstawiona na wcześniejszych slajdach. O ile ogólna zasada działania pozostaje podobna, tak z użyciem sieci neuronowych udało się zmniejszyć wymagania pamięciowe i czas na generowanie dobrych reprezentacji słów. Nadal jednak wcześniej na slajdach przedstawiona historia jest najprostszym wprowadzeniem do idei word embeddingów.

Najbardziej popularne metody wyznaczania embeddingów to GloVe i Word2Vec

(https://www.cs.put.poznan.pl/alawrynowicz/PJN_4-2019.pdf,
<https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf>)

W internecie znajdziemy zbiory już wygenerowanych embeddingów, np. na stronie: <https://nlp.stanford.edu/projects/glove/> znajdziemy różnej długości wektory stworzone za pomocą GloVe.

Ale chwila, mamy reprezentacje pojedynczych słów, jak to wykorzystać do klasyfikacji dokumentów, gdzie każdy dokument powinien być wektorem o równej długości?

Mając sekwencję słów(tokenów) w dokumencie, gdzie dla każdego słowa(tokenu) posiadamy embedding, zawsze tych samych rozmiarów, (np. wektor 300 liczb), możemy cały dokument reprezentować jako środek ciężkości wszystkich tych embeddingów:

1. Dla danego dokumentu wykonaj tokenizację, zbierz embeddingi przypisane poszczególnym tokenom (np. z pretrenowanego zbioru embeddingów z <https://nlp.stanford.edu/projects/glove/>) iterując token po tokenie w dokumencie
2. Mając zbiór tokenów i przypisanych do nich embeddingów (każdy o zadanej długości, np 300), wyznacz wektor o długości 300, reprezentujący cały dokument, gdzie:
 - 2.1 na 1 pozycji wektora zapisujemy średnią arytmetyczną wszystkich pierwszych pozycji embeddingów dla tokenów z dokumentu
 - 2.2 na 2 pozycji wektora zapisujemy średnią arytmetyczną wszystkich drugich pozycji embeddingów dla tokenów z dokumentu
 - 2.3 ...i tak dalej, aż do 300 elementu

Okazuje się, że tak prosta metoda, gdzie znaczenie dokumentu to "średnießnaczenie zawartych w nich słów, daje często bardzo dobre rezultaty! Oczywiście nie jest to jedyna metoda (dobry przegląd znajdziecie pod adresem:

<https://towardsdatascience.com/document-embedding-techniques-fed3e7a6a25d>), ale idealna w swojej prostocie do wprowadzenia w problematykę i często wykorzystywana w praktyce z uwagi na prostotę implementacji.

(BONUS): **Odległość Levenshteina**: na naszych laboratoriach wykorzystywać będziemy też odległość Levenshteina do automatycznego poprawiania literówek w tekście. O ile sama metoda wyznaczania odległości edycyjnej (często używana alternatywna nazwa dla odległości Levenshteina) jest już zaimplementowana w wielu narzędziach, o tyle warto zapoznać się z jej działaniem. Pod adresem: https://pl.wikipedia.org/wiki/Odleg%C5%82o%C5%9B%C4%87_Levenshteina znajdziecie Państwo przystępny opis.

Dla przypomnienia – na portalu Piazza:

<https://piazza.com/class/k7qu9e2ort96ru> znajduje się kurs PJN, gdzie wysyłamy komentarze do zadań i bieżące komunikaty, warto mieć tam konto, aby wszelkie bieżące wiadomości dotarły do Państwa jak najszybciej.