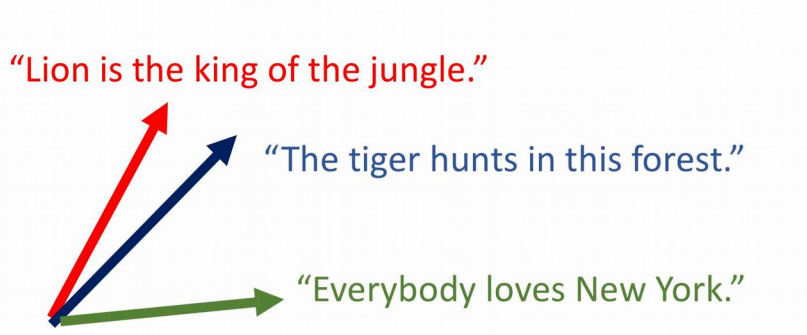
Mondat/dokumentumszintű reprezentáció

A feladat

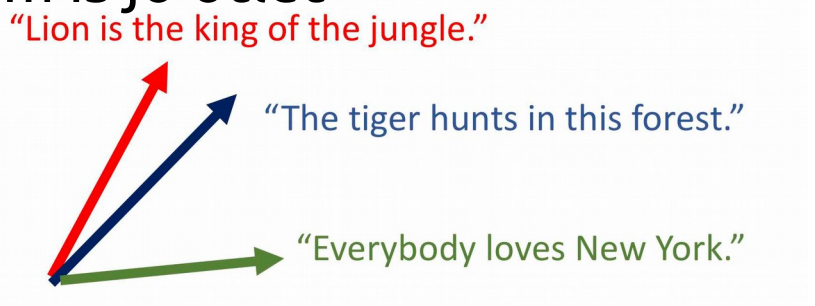
- A szószintű reprezentációinkat miként egyesíthetnénk nagyobb egységekre (pl. frázisok/mondatok/bekezdések)
- A szavak mentén történő konkatenáció nem életképes ötlet
- A legegyszerűbb (meglepően jól működő) megoldás: ?



Ábra forrása: offtheconvex.org

A feladat

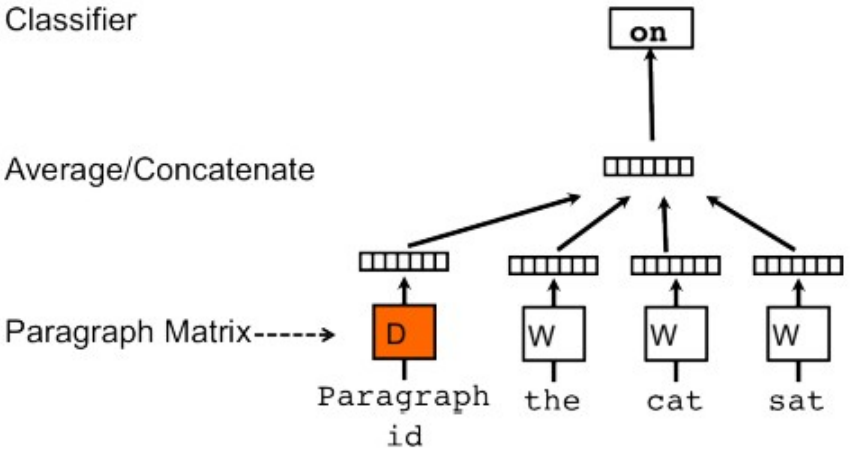
- A szószintű reprezentációinkat miként egyesíthetnénk nagyobb egységekre (pl. frázisok/mondatok/bekezdések)
- A szavak mentén történő konkatenáció nem életképes ötlet
- A legegyszerűbb (meglepően jól működő) megoldás: mondatot alkotó szavak vektorainak átlagolása
  - A koordinátánkénti szorzás/maximum is jó ötlet



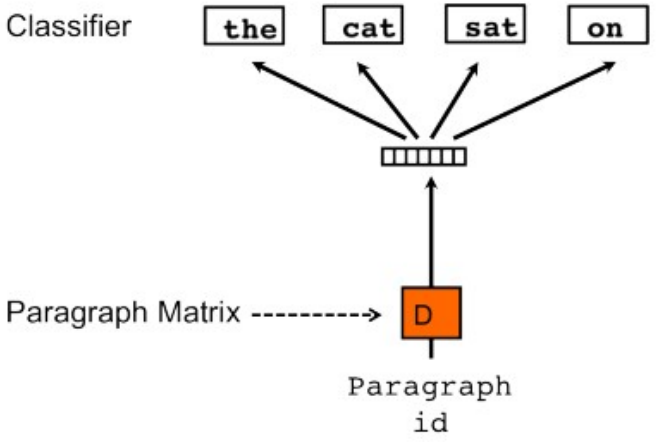
Ábra forrása: offtheconvex.org

doc2vec: Bekezdésvektorok (Le & Mikolov, 2014)

- A hagyományos CBOW/SG modellek adaptációja
  - A bekezdés azonosítóját egy virtuális szóként kezeli



Distributed Memory Model of Paragraph Vectors (PV-DM)



Paragraph vector with distributed bag of words (PV-DBOW)

doc2vec inferencia

- Minden bekezdés egyedi azonosítóval rendelkezik
  - A tanulás során csupán a tantító bekezdések azonosítóihoz fog rendelkezésünkre állni vektoros reprezentáció!
    - A hagyományos word2vec-től eltérően a bekezdésreprezentációt nem fogjuk tudni megkapni egy egyszerű lookup művelettel

doc2vec inferencia

- Minden bekezdés egyedi azonosítóval rendelkezik
  - A tanulás során csupán a tantító bekezdések azonosítóihoz fog rendelkezésünkre állni vektoros reprezentáció!
    - A hagyományos word2vec-től eltérően a bekezdésreprezentációt nem fogjuk tudni megkapni egy egyszerű lookup művelettel
  - A tanulás során nem látott bekezdések vektorait a tesztelés során tanulnunk kell
    - Bővítsük ki a D mátrixot az aktuális bekezdés vektorával, egyébként a modell minden további paraméterét fixáljuk le
    - Az új dokumentum vektorára hajtsunk végre SGD-t

doc2vec tapasztalatok

- A PV-DM modell eredményesebb a PV-DBOW-nál
  - A kettőt ugyanakkor érdemes lehet kombinálni
- PV-DM modell esetében jobb konkatenálni, mint összegezni
- A tesztelés költséges (a word2vec O(1)-éhez képest), ugyanakkor párhuzamosítható
  - kb. 1 mp/dokumentum
    - 25K átlagosan 230 szavas dokumentumon cca. 8 órás CPU-idő

Earth Mover's Distance

- Egy másik megoldás az optimális szállítási feladatra épít
  - Adott depókhoz és boltokhoz htározzuk meg, hogy melyik depóból melyik boltba mennyi kiszállítás történjen az összktg. minimalizálása mellett

$$\min_{\mathbf{T}\geq 0} \sum_{i,j=1}^{n} \mathbf{T}_{ij} c(i,j)$$

$$\text{subject to: } \sum_{j=1}^{n} \mathbf{T}_{ij} = d_i \quad \forall i \in \{1,\ldots,n\}$$

$$\sum_{i=1}^{n} \mathbf{T}_{ij} = d'_j \quad \forall j \in \{1,\ldots,n\}.$$

Earth Mover's Distance

- Egy másik megoldás az optimális szállítási feladatra épít
  - Adott depókhoz és boltokhoz htározzuk meg, hogy melyik depóból melyik boltba mennyi kiszállítás történjen az összktg. minimalizálása mellett

$$\min_{\mathbf{T} \geq 0} \sum_{i,j=1}^{n} \mathbf{T}_{ij} c(i,j)$$

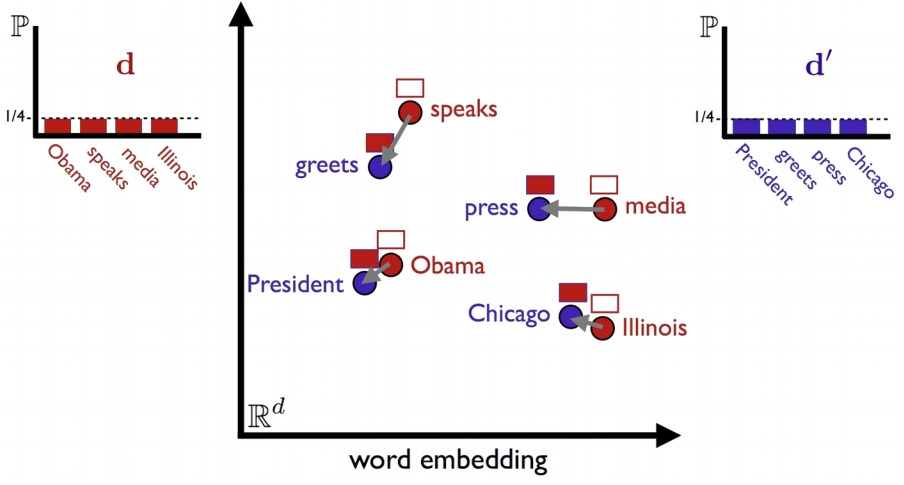
$$\text{subject to:} \sum_{j=1}^{n} \mathbf{T}_{ij} = d_i \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} \mathbf{T}_{ij} = d'_j \quad \forall j \in \{1, \ldots, n\}.$$

- T jelzi a (depó, bolt)-párok között szállított mennyiségeket
- d és d' a kínálati/keresleti értékek
- c(i,j) az i-ből j-be történő szállítás költsége

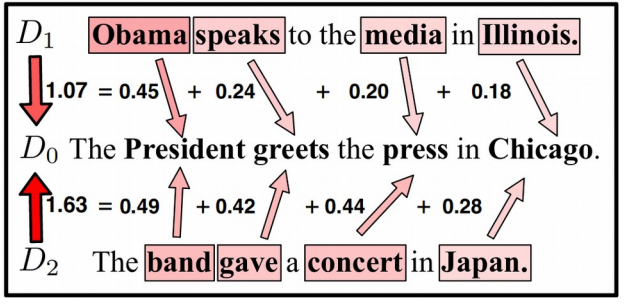
Word Mover's Distance (Kusner et al., 2015)

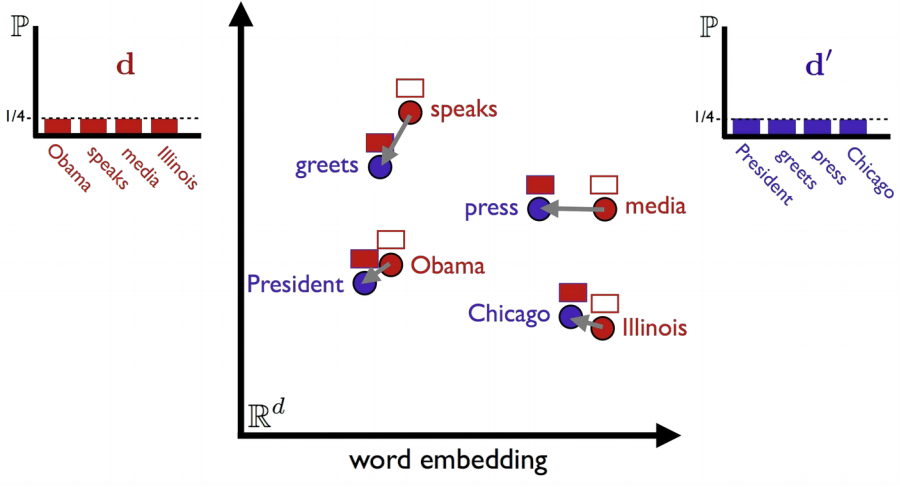
- Az optimális szállítási feladat kiszámítása p db egyedi szót tartalmazó mondatpárra költséges ($O(p^3 \log(p))$) lenne
  - A megszorító feltételek közül az egyiket eldobva egy könnyebb relaxált feladat megoldásához jutunk
    - A relaxált feladat T*-a olyan, hogy d szavait a hozzájuk legközelebb eső d'-beli szóhoz rendeljük
    - Nyilvánvalóan egy alsó korlátját kapjuk a nehéz feladatnak



Word Mover's Distance (Kusner et al., 2015)

- Az optimális szállítási feladat kiszámítása p db egyedi szót tartalmazó mondatpárra költséges ($O(p^3 \log(p))$) lenne
  - A megszorító feltételek közül az egyiket eldobva egy könnyebb relaxált feladat megoldásához jutunk
    - A relaxált feladat T*-a olyan, hogy d szavait a hozzájuk legközelebb eső d'-beli szóhoz rendeljük
    - Nyilvánvalóan egy alsó korlátját kapjuk a nehéz feladatnak





WMD korlátjának élesítése

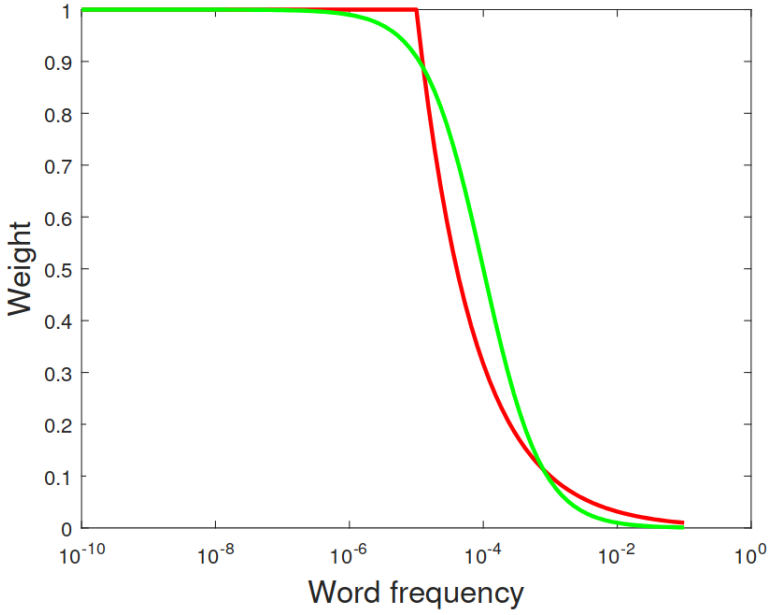
- Az egyszerű feladat megoldásával nyert alsó korlát olcsón élesíthető, ha két egyszerű feladatot oldunk meg
  - Az egyik egyszerű feladat a d-kre a másik a d’-kre vonatkozó feltételt hagyja el
  - A könnyű feladatok optimális megoldásainak maximumát véve élesíteni tudjuk az alsó korlátot

WMD korlátjának élesítése

- Az egyszerű feladat megoldásával nyert alsó korlát olcsón élesíthető, ha két egyszerű feladatot oldunk meg
  - Az egyik egyszerű feladat a d-kre a másik a d’-kre vonatkozó feltételt hagyja el
  - A könnyű feladatok optimális megoldásainak maximumát véve élesíteni tudjuk az alsó korlátot
- A relaxált távolság immár $O(p^2)$ műveletigénnyel számítható
  - Trükkökkel tovább gyorsítható az eljárás
    - Egy pontatlanabb, de gyorsabban számolható alsó korláttal történő előszűrés a leghasonlóbb dokumentumok keresése során

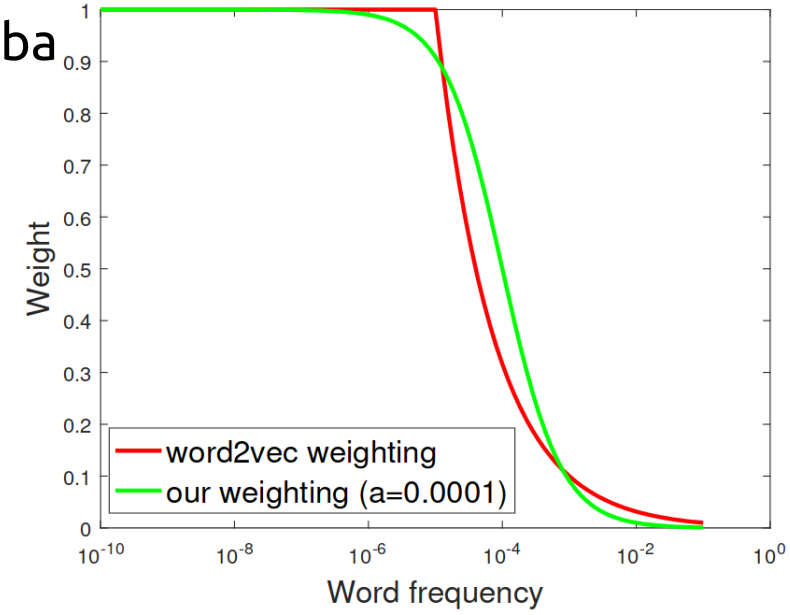
SIF: Smooth Inverse Frequency (Arora et al., 2017)

- Számoljuk ki a mondatot alkotó szavak vektorainak súlyozott átlagát
  - A túl gyakori szavak számítsanak kevésbé (vö. tf-idf)



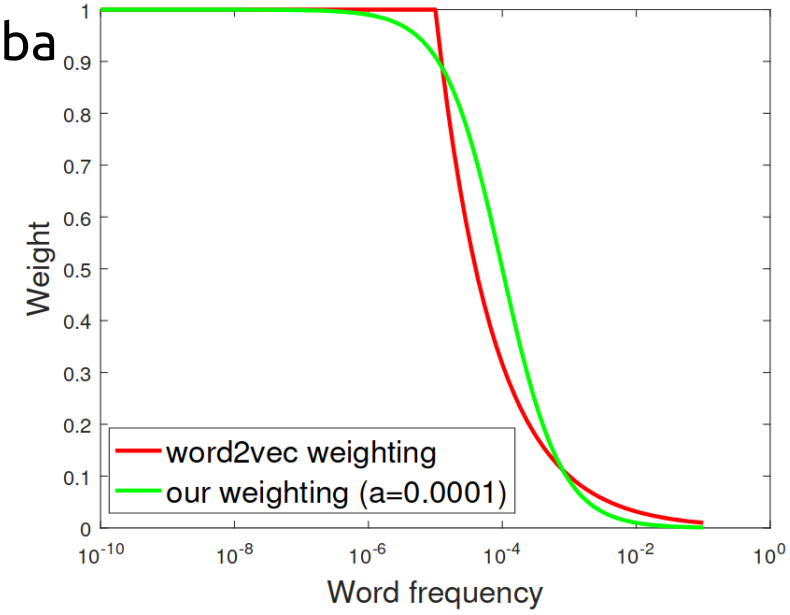
SIF: Smooth Inverse Frequency (Arora et al., 2017)

- Számoljuk ki a mondatot alkotó szavak vektorainak súlyozott átlagát
  - A túl gyakori szavak számítsanak kevésbé (vö. tf-idf)
    - w2v-féle downsampinggel is párhuzamba állítható
      - Elvetés 1/sqrt(p(w)) valószínűséggel



SIF: Smooth Inverse Frequency (Arora et al., 2017)

- Számoljuk ki a mondatot alkotó szavak vektorainak súlyozott átlagát
  - A túl gyakori szavak számítsanak kevésbé (vö. tf-idf)
    - w2v-féle downsampinggel is párhuzamba állítható
      - Elvetés 1/sqrt(p(w)) valószínűséggel
    - SIF szerint egy szó súlya legyen a/(a+p(w))



SIF algoritmus

Algorithm 1 Sentence Embedding

Input: Word embeddings $\{v_w : w \in \mathcal{V}\}$, a set of sentences $\mathcal{S}$, parameter a and estimated probabilities $\{p(w) : w \in \mathcal{V}\}$ of the words.

Output: Sentence embeddings $\{v_s : s \in \mathcal{S}\}$

for all sentence s in $\mathcal{S}$ **do**

$v_s \leftarrow \frac{1}{|s|} \sum_{w \in s} \frac{a}{a+p(w)} v_w$

end for

Form a matrix X whose columns are $\{v_s : s \in \mathcal{S}\}$, and let u be its first singular vector

for all sentence s in $\mathcal{S}$ **do**

$v_s \leftarrow v_s - uu^\top v_s$

end for

SIF algoritmus

Algorithm 1 Sentence Embedding

Input: Word embeddings $\{v_w : w \in \mathcal{V}\}$, a set of sentences $\mathcal{S}$, parameter a and estimated probabilities $\{p(w) : w \in \mathcal{V}\}$ of the words.

Output: Sentence embeddings $\{v_s : s \in \mathcal{S}\}$

1: **for all** sentence s in $\mathcal{S}$ **do**

2: $v_s \leftarrow \frac{1}{|s|} \sum_{w \in s} \frac{a}{a+p(w)} v_w$ → Mondatbeli szóvektorok súlyozása

3: **end for**

4: Form a matrix X whose columns are $\{v_s : s \in \mathcal{S}\}$, and let u be its first singular vector

5: **for all** sentence s in $\mathcal{S}$ **do**

6: $v_s \leftarrow v_s - uu^\top v_s$

7: **end for**

SIF algoritmus

Algorithm 1 Sentence Embedding

Input: Word embeddings $\{v_w : w \in \mathcal{V}\}$, a set of sentences $\mathcal{S}$, parameter a and estimated probabilities $\{p(w) : w \in \mathcal{V}\}$ of the words.

Output: Sentence embeddings $\{v_s : s \in \mathcal{S}\}$

1: **for all** sentence s in $\mathcal{S}$ **do**

2: $v_s \leftarrow \frac{1}{|s|} \sum_{w \in s} \frac{a}{a+p(w)} v_w$ → Mondatbeli szóvektorok súlyozása

3: **end for**

4: Form a matrix X whose columns are $\{v_s : s \in \mathcal{S}\}$, and let u be its first singular vector

5: **for all** sentence s in $\mathcal{S}$ **do**

6: $v_s \leftarrow v_s - uu^\top v_s$

7: **end for**

→ Túl általános információ eltávolítása

Kiértékelési lehetőségek

- Sentence Textual Similarity datasets
  - https://github.com/brmson/dataset-sts