



Distributed word embeddings

A softmax függvény

- Bináris osztályozás szigmoid függvénnyel

$$\sigma(z) = \frac{1}{1 + \exp^{-z}}$$

- $z = w^T x$ mint normalizálatlan valószínűség

- Alternatíván legyen $z_1 = w_1^T x$ és $z_2 = w_2^T x$ a pozitív és negatív osztályba tartozás normalizálatlan valószínűségei

$$\text{softmax}(z) = \left[\frac{\exp^{z_1}}{\exp^{z_1} + \exp^{z_2}}, \frac{\exp^{z_2}}{\exp^{z_1} + \exp^{z_2}} \right]$$

- $z = z_1 - z_2$ választással a logisztikus regresszió is ezt csinálja

Szoftmax példa

•A szoftmax függvény segítségével $(-\infty, \infty)$ intervallumból jövő értékekből eloszlást gyárthatunk

–Oszályozást végző neurális hálók gyakori összetevője

$[-2, 3, 0] \rightarrow [0.14 \quad 20.09 \quad 1.00]$

$[-1, 4, 1] \rightarrow [0.37 \quad 54.60 \quad 2.72]$

$[1, 2, 2.3] \rightarrow [2.72 \quad 7.39 \quad 9.97]$

Szoftmax példa

•A szoftmax függvény segítségével $(-\infty, \infty)$ intervallumból jövő értékekből eloszlást gyárthatunk

–Oszályozást végző neurális hálók gyakori összetevője

$[-2, 3, 0] \rightarrow [0.14 \quad 20.09 \quad 1.00] \rightarrow [0.006 \quad 0.946 \quad 0.048]$

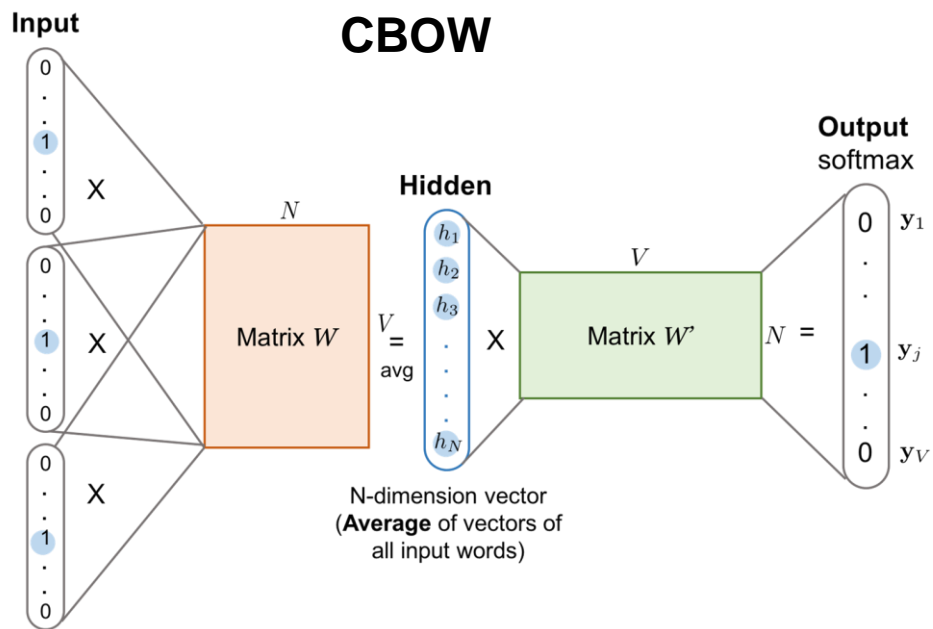
$[-1, 4, 1] \rightarrow [0.37 \quad 54.60 \quad 2.72] \rightarrow [0.006 \quad 0.946 \quad 0.048]$

$[-1, 2, 2.3] \rightarrow [2.72 \quad 7.39 \quad 9.97] \rightarrow [0.189 \quad 0.377 \quad 0.434]$

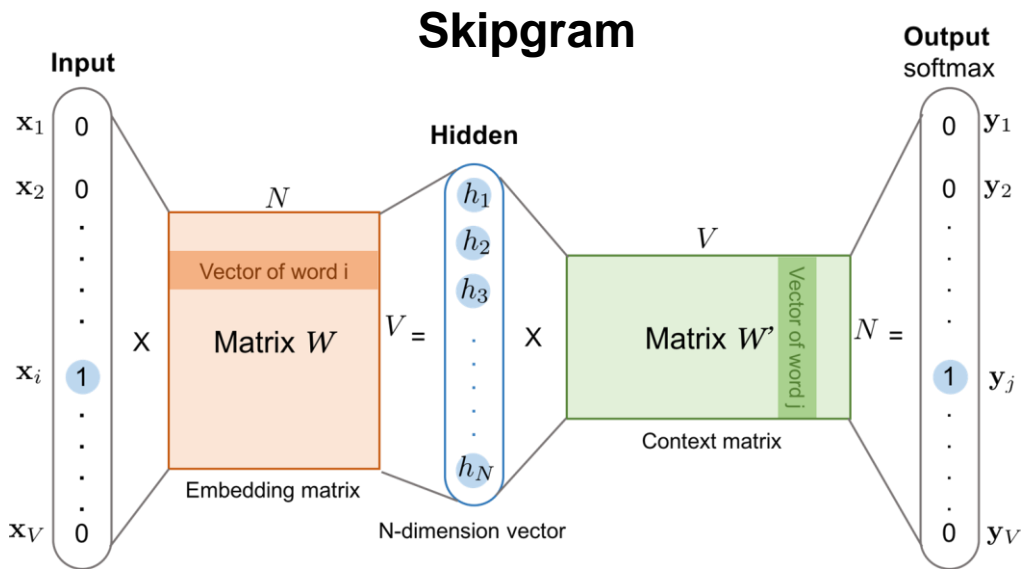
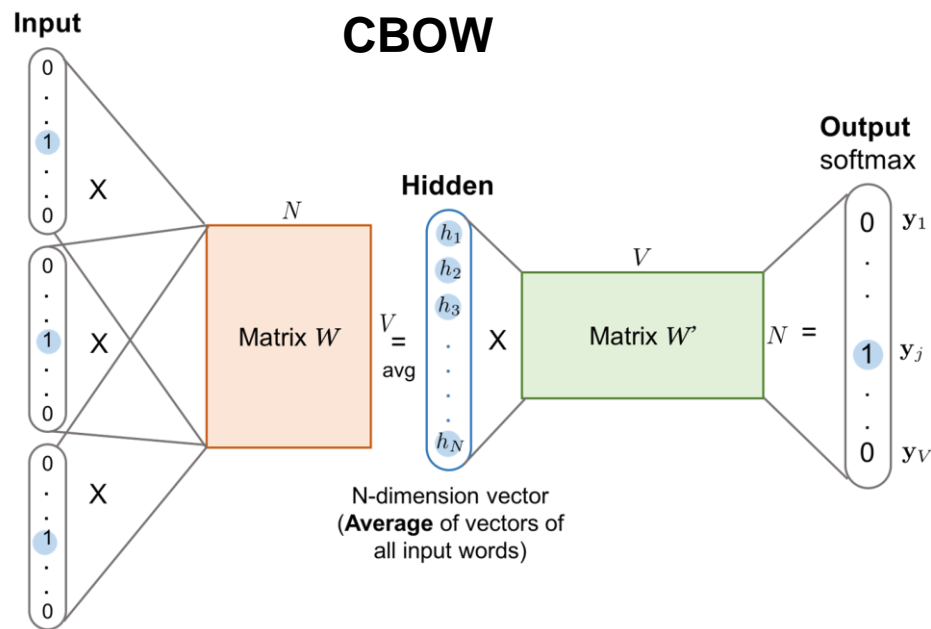
word2vec (Mikolov et al., 2013)

- Algoritmuscsalád több (kritikus) hiperparaméterrel
 - Alapcélja: olyan prediktív modellt tanulni, ami képes minél pontosabban megbecsülni, hogy ha egy szövegrészből kitakarunk egy/több szót, akkor mi/mik volt/voltak az/azok
- Minden szóhoz rendeljünk egy kontextus és output reprezentációt (egy-egy N dimenziós vektort)
 - Predikcióinkat a kontextus és output vektorok pontszorzatain alkalmazott softmax függvénnnyel hozzuk meg

Continuous bag of words (CBOW) vs. Skipgram



Continuous bag of words (CBOW) vs. Skipgram



A predikciós mechanizmus

•Egy-egy kontextusablak viszonyában regisztráljuk a predikció kapcsán jelentkező hibát, és frissítjük a szóreprzentációkat

$$p(o_i | c_j) = \text{softmax}(\mathbf{1}_j^T W W')$$

– $\mathbf{1}_j = [0 \ 0 \ 0 \ \dots \ 0 \ 0 \ 1 \ 0 \ 0 \ \dots \ 0]$ alakú ún. one-hot vektor

j. pozíció

– W és W' paraméterek függetlenek egymástól (Miért?)

–Kezdetben random értékeket tartalmaznak, SGD-vel frissítjük őket a tanulás során

A frissítési szabály

- Szükségünk van a predikció hibájának gradiensére
 - A predikciós hiba az elvárt szó előrejelzésének negált log valószínűsége

$$\ell = -\log \left(p(o_i | c_j) \right) = -\log \frac{e^{w_j^T w'_i}}{\sum_{k=1}^{|V|} e^{w_j^T w'_k}} = ?$$

- Mi lesz a hibatag gradiense? Hogy lehet értelmezni?
 - <https://stats.stackexchange.com/questions/253244/gradients-for-skipgram-word2vec>

A veszteség gradiense


$$\ell = -\log\left(p(o_i | c_j)\right) = -\log \frac{e^{w_j^T w'_i}}{\sum_{k=1}^{|V|} e^{w_j^T w'_k}} = -w_j^T w'_i + \log \sum_{k=1}^{|V|} e^{w_j^T w'_k}$$

$$\nabla_{w_j} \ell = -w'_i + \sum_{k=1}^{|V|} p(o_k | c_j) w'_k$$

Az elvi modell problémái

- A célfüggvény gradiensét nagyon költséges kiszámolni
 - Egyetlen frissítés alkalmával a teljes szótár (V) fölötti összegzést igényel
- A probléma javítható hierarchikus szoftmax vagy negatív mintavételezés alkalmazásával

Hierarchikus softmax

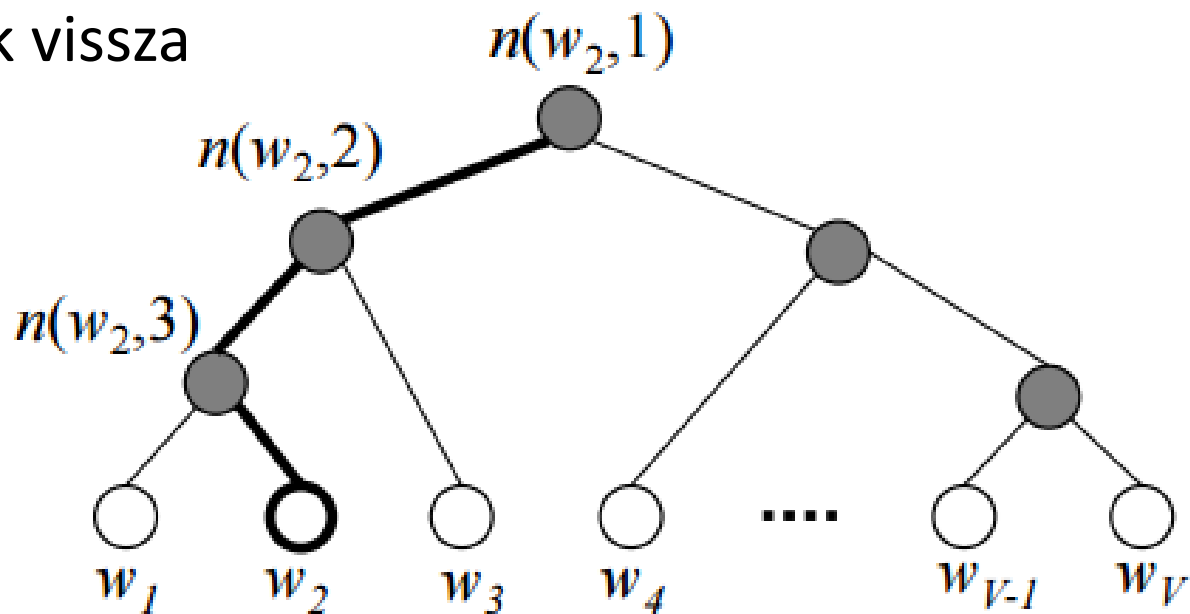
- A predikációs mechanizmust (W') cseréljük le egy bináris fára
 - N csúcsú bináris fa várható magassága $\log(N)$ 
 - A bináris fa minden csúcsa egy-egy döntést hoz
 - Egy outputra vonatkozó predikció legyen a fában hozzá való eljutás során hozott döntések valószínűségeinek szorzata
- 1 db $|V|$ kimenetelű multinomiális eloszlás helyett hozzunk $\log(|V|)$ bináris döntést

Hierarchikus softmax illusztrációja

- Minden csúcs egy-egy mini osztályozó

 - Balra vagy jobbra tovább?

- A hibát így propagáljuk vissza



Negatív mintavételezés

- A teljes szótár feletti predikció helyett hozzunk néhány egyszerű bináris döntést (valid/invalid kontextusról van-e szó?)

- Bináris döntést hozni sokkal olcsóbb, mint egy $|V|$ kimenetelűt

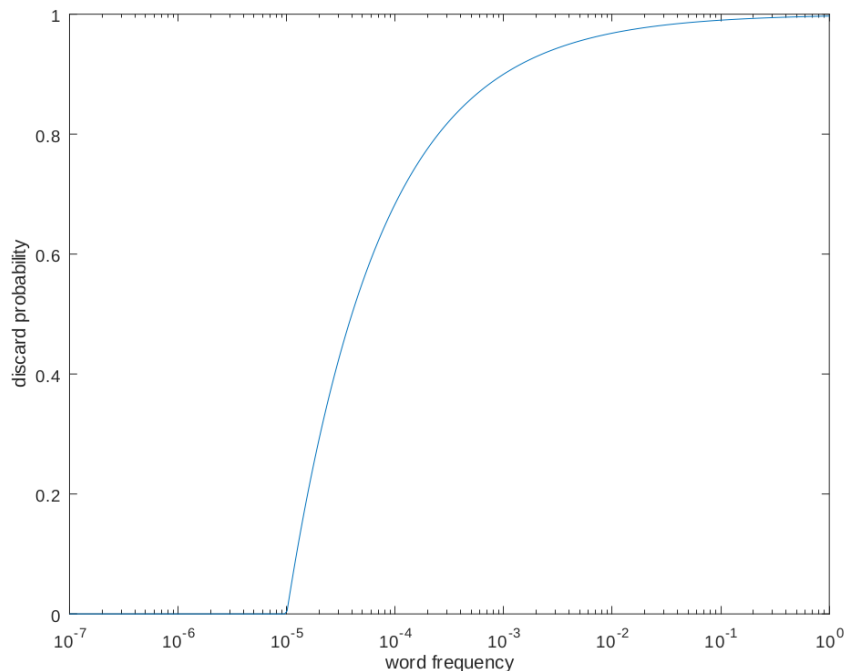
$$l = -\log(p(Y = 1|o_i, c_j)) - \sum_{\{k=1, o_k \sim Q\}}^K \log(p(y = 0|o_k, c_j))$$

- Q a szótár elemei fölötti gyakorisági eloszlás

- Q meghatározása során a szavak gyakoriságát emeljük egy 1-nél kisebb hatványra (pl. 0.75)

További trükkök

- Adaptív ablakméret (távolabbi szomszédok alulsúlyozása)
- Gyakori szavak alulmintavételezése $1 - \sqrt{t/f(w)}$ valószínűséggel
 - t egy hiperparaméter ($t \approx 1e-5$)
 - $f(w)$ a szó gyakorisága
- Gyakori mintázatok (pl. *New_York*) beazonosítása
 - Történhet pl. PPMI segítségével
- Stb, stb...

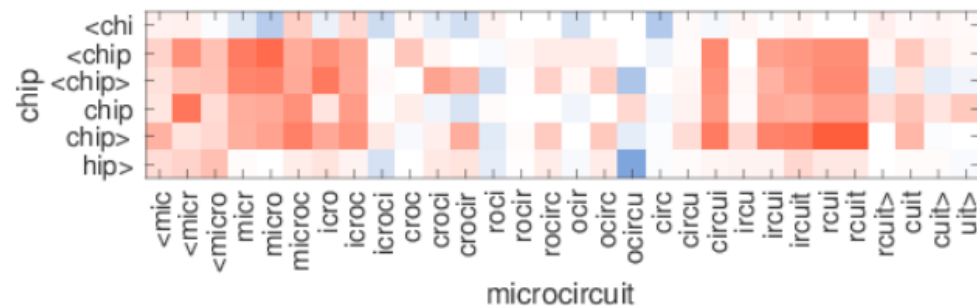
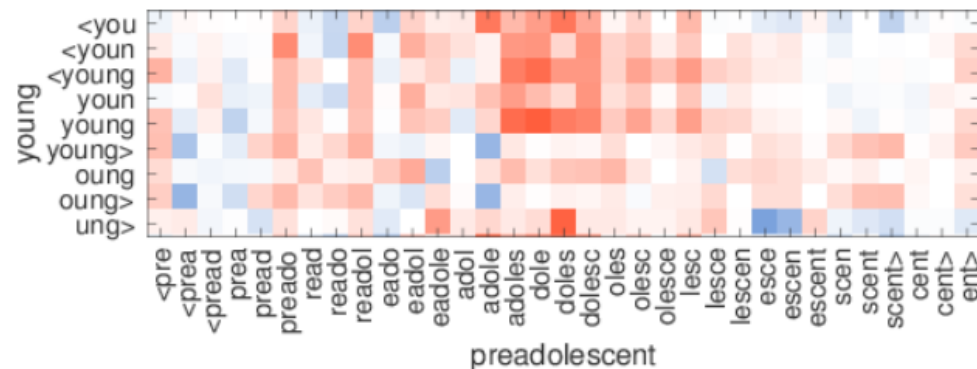


Egyéb megoldások – fasttext

- A szavak helyett gondolkozzunk szótöredékekben
 - Hasznos lehet morfológiailag változatos nyelvek esetében
 - Lényegében minden karakterlánchoz fogunk tudni reprezentációt adni a szótöredék–reprezentációkra alapozva
- A szavakhoz tartozó vektoros reprezentációra tekintsünk az azokat alkotó szótöredékek vektoros reprezentációinak összegeként
- Előtanított vektorok elérhetők 150+ nyelvre (fasttext.cc)

fasttext (Bojanowski et al., 2017)

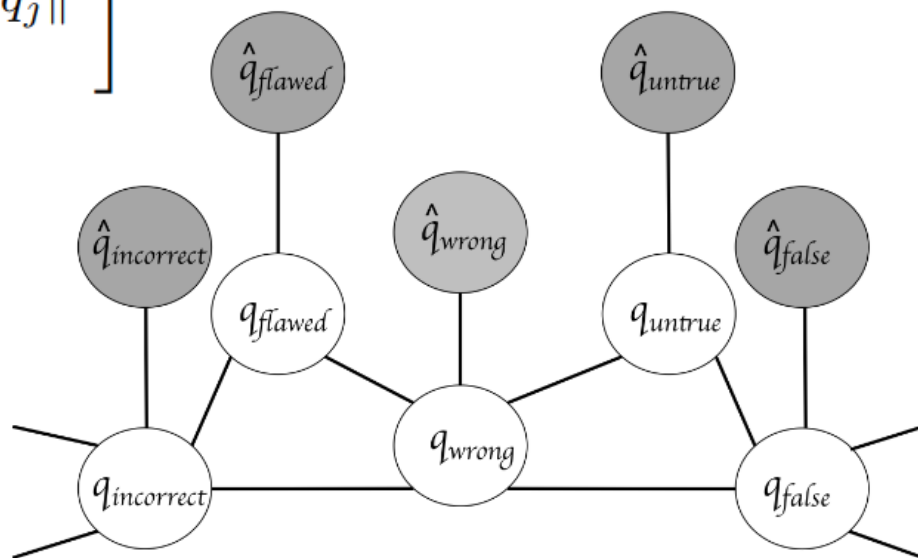
- Lényegében a skipgram modell kiterjesztése input/output karakter n-gramokkal
- Az ábrán egy szópárra nézve, az azt alkotó töredékpárok reprezentációinak hasonlósága áll
 - Az x tengelyen mindkét esetben egy OOV szó áll



Retrofitting (Faruqui et al., 2015)

•Tanuljunk egy kezdeti szóreprzentációt \hat{q} , majd ezt finomítsuk tudásbázisokra támaszkodva

$$\Psi(Q) = \sum_{i=1}^n \left[\alpha_i \|q_i - \hat{q}_i\|^2 + \sum_{(i,j) \in E} \beta_{ij} \|q_i - q_j\|^2 \right]$$

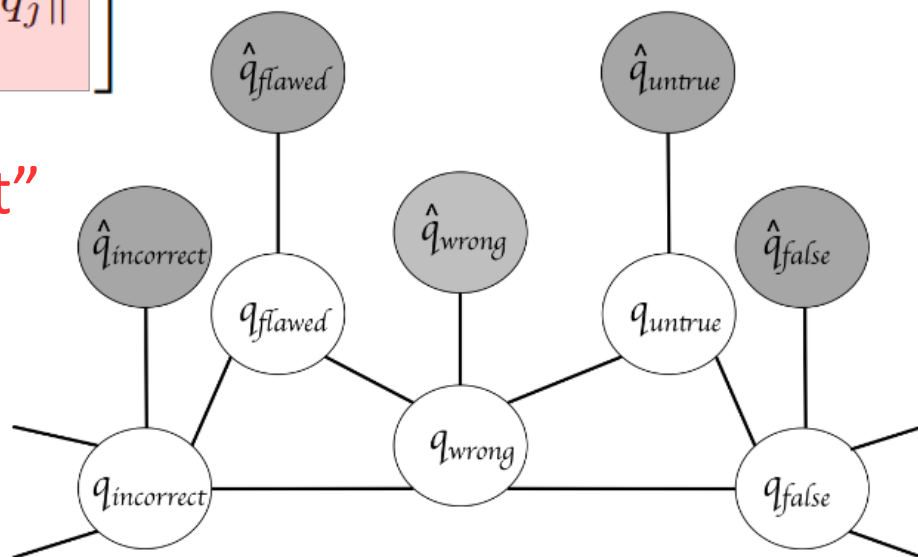


Retrofitting (Faruqui et al., 2015)

•Tanuljunk egy kezdeti szóreprzentációt \hat{q} , majd ezt finomítsuk tudásbázisokra támaszkodva

$$\Psi(Q) = \sum_{i=1}^n \left[\alpha_i \|q_i - \hat{q}_i\|^2 + \sum_{(i,j) \in E} \beta_{ij} \|q_i - q_j\|^2 \right]$$

–Azaz a tudásbázisban “összekötött”
szavak viselkedjenek hasonlóan,
miközben az eredetileg tanult
reprzentációjuktól se térjenek el
túlzottan



Explicit Retrofitting (Glavaš & Vulić, 2018)

- A Faruqui-féle retrofitting problémája, hogy csak a tudásbázisban található tudás integrálására képes
- Az explicit retrofitting megpróbálja a tudásbázisban tárolt tudást absztrahálni
 - Egy olyan neurális modellt tanul, ami utólagosan minden vektor specializálását el tudja végezni, nem csak a tudásbázisban lévőkét

Prior tudás integrálása (Song et al., 2017)

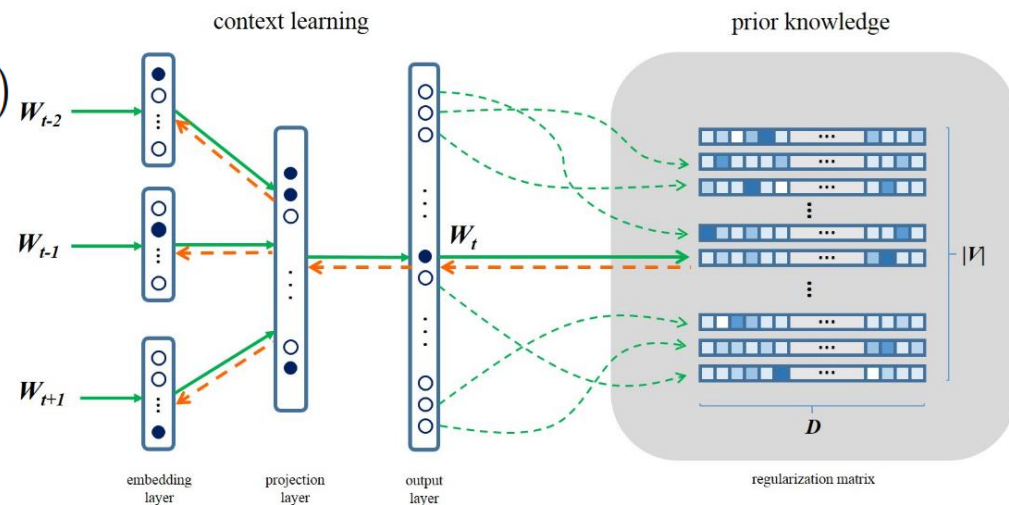
- A kontextusszó kilétén túl, valamilyen egyéb ismerv előrejelzésére is képessé akarjuk tenni a modellünket
 - A prior tudás jöhet tudásbázisból, de automatizmus által is előállhat (LDA=látens Dirichlet allokáció, egy topikmodell)

Prior tudás integrálása (Song et al., 2017)

•A kontextusszó kilétén túl, valamilyen egyéb ismérv előrejelzésére is képessé akarjuk tenni a modellünket

–A prior tudás jöhet tudásbázisból, de automatizmus által is előállhat (LDA=látens Dirichlet allokáció, egy topikmodell)

$$\mathcal{L} = \frac{1}{|V|} \sum_{i=1}^{|V|} \log p(w_i, \psi(w_i) \mid \sum_{0 < |j| \leq c} v_{i+j})$$



Glove (Pennington et al., 2014)

- Egyszerre globális és lokális
 - A korpusz globális statisztikáit próbálja a modell rekonstruálni
 - A paraméterek frissítése sztochasztikusan történik

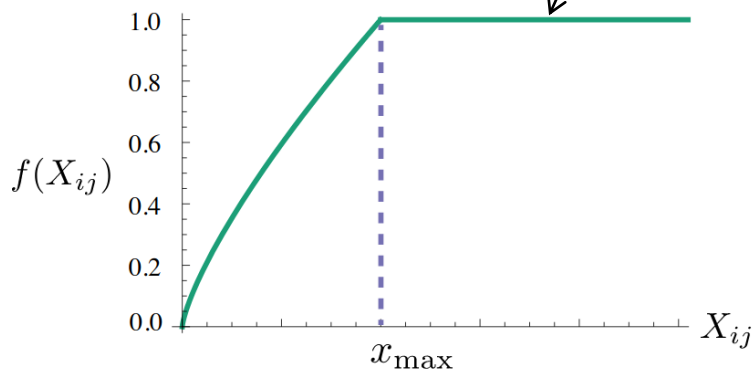
$$J = \sum_{i,j}^{|V|} f(X_{ij}) (w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2$$

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k ice)/P(k steam)$	8.9	8.5×10^{-2}	1.36	0.96

Glove (Pennington et al., 2014)

- Egyszerre globális és lokális
 - A korpusz globális statisztikáit próbálja a modell rekonstruálni
 - A paraméterek frissítése sztochasztikusan történik

$$J = \sum_{i,j}^{|V|} f(X_{ij}) (w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2$$



↘
Globális együttlőfordulási gyakoriság



Szórepresentációk kiértékelése

Szóhasonlóság

- Adott szópárok adatbázisa emberek által hozzárendelt hasonlósági értékekkel (similarity vs. relatedness)
- A szóreprzentációink alapján próbáljuk rekonstruálni az emberek által adott hasonlósági pontszámokat
 - Általában a szópárok vektorainak koszinusz hasonlóságának és az emberi hasonlóságok korrelációs együtthatóját nézzük

		emberi	automatikus
kutya	eb	9.9	9.1
kutya	hajó	1.1	1.7
...
ló	ménes	8	6.7

→ $\rho=0.62$

Lexikális következtetés

•(a,b) szópár vonatkozásában a reprezentációk alapján próbáljuk meg eldönteni, hogy a-ból következik-e b

–Másképpen b hipernímája-e a-nak (általánosabb fogalom-e nála)

•kutya \rightarrow emlős

•kutya \nrightarrow hüllő

•emlős \nrightarrow kutya

–(Levy et al., 2015) megmutatta, hogy a felügyelt módszerek hajlamosak a tanulni kívánt reláció helyett prototipikus hipernímákat tanulni csupán

Szóanalógiák

- $A:B :: C:D$ alakú szó párok gyűjteménye

- Milyen gyakran tudjuk eltalálni azt, hogy adott A , B és C -hez mi a legjobban passzoló D szó

- $A-B \approx C-D$, vagyis melyik D -nek a $(C-A+B)$ -vel vett hasonlósága a maximális

- (Levy & Goldberg, 2014): a $(C-A+B)$ -t könnyen dominálni tudja egyetlen vektor, így keressük inkább azon D -t amire $\frac{\cos(D, C) * \cos(D, B)}{\cos(D, A)}$ maximális

Szóanalógiás adatbázisok

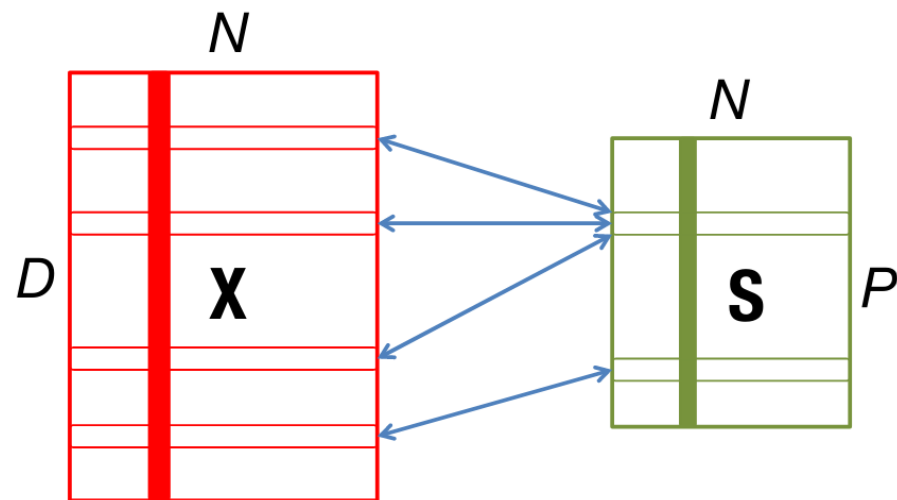
- Szintaktikus párok (walk:walked :: bake:baked)
- Szemantikus párok (Spain:Madrid :: France:Paris)
- Angolra pl. Google analogy test set vagy Better Analogy Test Set (BATS)
- Magyarra is létezik kiértékelő adatbázis
<http://corpus.nytud.hu/efnilex-vect/>

Kiértékelésorientált utófeldolgozás (Artetxe et al., 2018)

- A szóreprzentációk változatos információkat tárolnak, amelyek a felszínre hozhatók
- Ha X a beágyazásmátrix, akkor XX^T a szópárok (elsőfokú) hasonlóságát adja meg, $(XX^T)^n$ pedig az n -edfokút
 - $(XX^T)^n = X(X^TX)^{n-1}X$, ahol X^TX felbontható $Q\Lambda Q^T$ alakban
 - A transzformált beágyazásmátrixunk legyen $XQ\Lambda^{1/2}$

QVEC (Tsvetkov et al., 2015)

- Korrelációalapú metrika, ami azt nézi, hogy a szóvektorok dimenziói mennyire feleltethetők meg dolgok valós életbeli tulajdonságainak
- X mátrix a szóbeágyazásokat, S pedig a szemantikus tudást tartalmazó mátrix

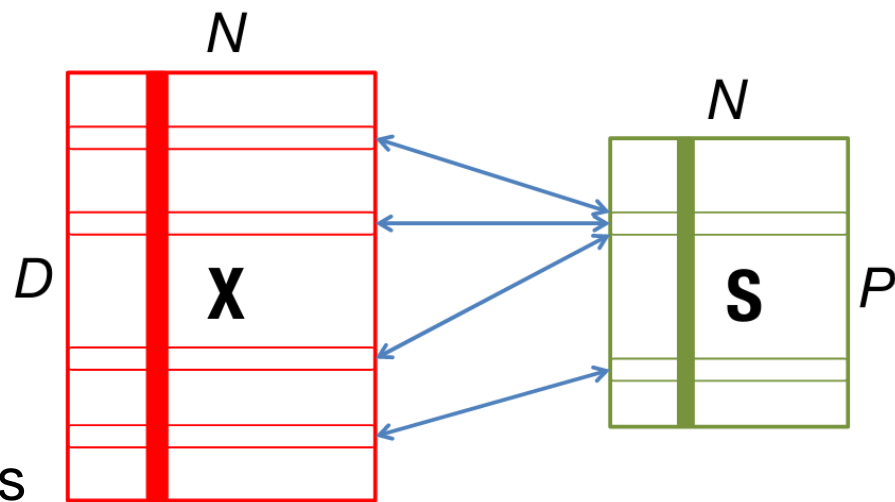


QVEC (Tsvetkov et al., 2015)

- Korrelációalapú metrika, ami azt nézi, hogy a szóvektorok dimenziói mennyire feleltethetők meg dolgok valós életbeli tulajdonságainak
- X mátrix a szóbeágyazásokat, S pedig a szemantikus tudást tartalmazó mátrix
 - S valamilyen tudásbázis alapján (pl. WordNet/SemCor) állítható elő

$$\text{QVEC} = \max_{\mathbf{A} | \sum_j a_{ij} \leq 1} \sum_{i=1}^D \sum_{j=1}^P r(\mathbf{x}_i, \mathbf{s}_j) \times a_{ij}$$

korreláció hozzárendelés



Alkalmazásban való kiértékelés

- Intrinzikus kiértékelésnél jóval költségesebb, viszont végső soron ez érdekel bennünket
- Sajnos az intrinzikus kiértékelés gyakran nem korrelál az alkalmazásbeli kiértékeléssel
- Kérdés, hogy a kapott eredmény mennyire köszönhető a szóreprezentációnak, illetve mennyiben az arra építő modellnek
- Intrinsic/extrinsic kiértékeléstől függetlenül, általában érdemes több benchmark adatbázison is mérjünk

Lehetséges külső kiértékelés: struktúratanulás

•Szekvenciaelemzés

- Part-of-Speech tagging: mi az egyes szavak szófaja?
- Named Entity Recognition: hol és milyen típusú tulajdonnév található a szövegben

•A feltorlódó névelemeket el akarjuk tudni határolni egymástól

- Chunkolás: sekély mondattani elemzés (főnévi/igei/melléknévi csoportok egymástól való elkülönítése)

–...

•Egyéb struktúrák tanulása, pl. konstituens vagy dependenciaelemzés

További alkalmazási lehetőségek

- Gépi fordítás
- Kérdésmegválaszolás
- Kivonatolás
- Dokumentumosztályozás