Επεξεργασία Φυσικής Γλώσσας & Μηχανική Μάθηση

Αναπαραστάσεις λέξεων

Κάτια Κερμανίδου
kerman@ionio.gr

Διανύσματα λέξεων: One-hot vectors

	abandon	ability	able	…	ants	…	zone
ants	0	0	0	0	1	0	0

- Ο αριθμός των στηλών είναι ο αριθμός των λέξεων στο λεξικό της γλώσσας μου
  - Της τάξης των εκατοντάδων χιλιάδων
- Πώς θα αποδώσω την σημασιολογική ομοιότητα ανάμεσα στο «Seattle motel» και στο «Seattle hotel»;
- τα παρακάτω διανύσματα είναι ορθοκανονικά – άπειρη απόσταση

$$motel = [0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0]$$
$$hotel = [0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0]$$

- Λύση: έμμεση απεικόνιση της ομοιότητας στα διανύσματα, λαμβάνοντας υπόψη το συμφραστικό περιβάλλον που εμφανίζεται η λέξη

Κατανεμημένα διανύσματα λέξεων (Distributional word vectors)

"You shall know a word by the company it keeps" (J. R. Firth, 1957)

government debt problems turning into banking crises as has happened in

saying that Europe needs unified banking regulation to replace the hodgepodge

↖ These words will represent *banking* ↗

Μια από τις σημαντικότερες ιδέες της σύγχρονης στατιστικής ΕΦΓ

Word Representations μέσω των κειμένων που οι λέξεις εμφανίζονται (1/2)

	doc1	doc2	doc3	doc4	…
Λέξη1	0	0	0	1	
Λέξη2	1	0	1	1	
Λέξη3	1	0	0	0	
Λέξη4	0	1	0	0	
…					

Term-document incidence matrix

	doc1	doc2	doc3	doc4	…
Λέξη1	0	0	0	3	
Λέξη2	4	0	17	7	
Λέξη3	9	0	0	0	
Λέξη4	0	6	0	0	
…					

Term-document frequency matrix

Word Representations μέσω των κειμένων που οι λέξεις εμφανίζονται (2/2)

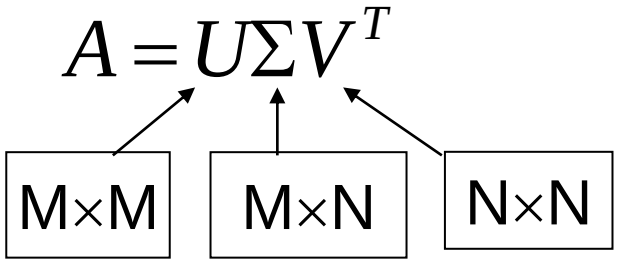
- tf: η συχνότητα εμφάνισης της λέξης στο κείμενο (term frequency)
- df: ο αριθμός των εγγράφων στα οποία εμφανίζεται η λέξη (document frequency)
- idf: $\log_{10}$ (N/df), όπου N ο συνολικός αριθμός εγγράφων (inverse document frequency)
- tfidf = tf * idf

	doc1	doc2	doc3	doc4	…
Λέξη1	0	0	0	1.02	
Λέξη2	4.11	0	1.65	2.36	
Λέξη3	2.89	0	0	0	
Λέξη4	0	1.08	0	0	
…					

Term-document tfidf matrix

Word Representations μέσω των κειμένων που οι λέξεις εμφανίζονται: Latent Semantic Indexing

- Singular Value Decomposition
- Ένας πίνακας A (MxN) μπορεί να διαχωριστεί σε 3 πίνακες ως εξής:



- Οι στήλες του U είναι τα ιδιοδιανύσματα του πίνακα $\boldsymbol{AA^T}$.
- Οι στήλες του V είναι τα ιδιοδιανύσματα του πίνακα $\boldsymbol{A^TA}$.
- Ο Σ είναι διαγώνιος πίνακας, με στοιχεία της διαγωνίου τα $\sigma_i' = \sqrt{\lambda_i}$ όπου $\lambda_1 \ldots \lambda_r$, οι ιδιοτιμές του $\boldsymbol{AA^T}$ (ή του $\boldsymbol{A^TA}$ –
- ίδιες είναι)
- $\boldsymbol{\sigma_i}$: ιδιάζουσες τιμές **(singular values)**
- **Αν στον υπολογισμό του παραπάνω γινομένου κρατήσω μόνο τις k μεγαλύτερες ιδιάζουσες και μηδενίσω τις υπόλοιπες, τότε υπολογίζω τον Ak, μια προσέγγιση του A, τάξης k**

Παράδειγμα

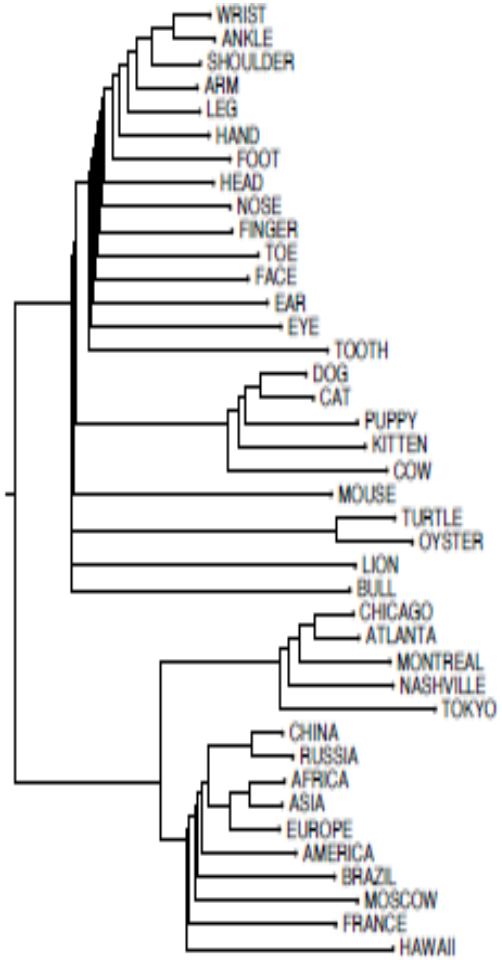
C	d_1	d_2	d_3	d_4	d_5	d_6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
wood	1	0	0	1	1	0
tree	0	0	0	1	0	1

C_2	d_1	d_2	d_3	d_4	d_5	d_6
ship	0.85	0.52	0.28	0.13	0.21	−0.08
boat	0.36	0.36	0.16	−0.20	−0.02	−0.18
ocean	1.01	0.72	0.36	−0.04	0.16	−0.21
wood	0.97	0.12	0.20	1.03	0.62	0.41
tree	0.12	−0.39	−0.08	0.90	0.41	0.49

Similarity of d_2 and d_3 in the original space: 0.

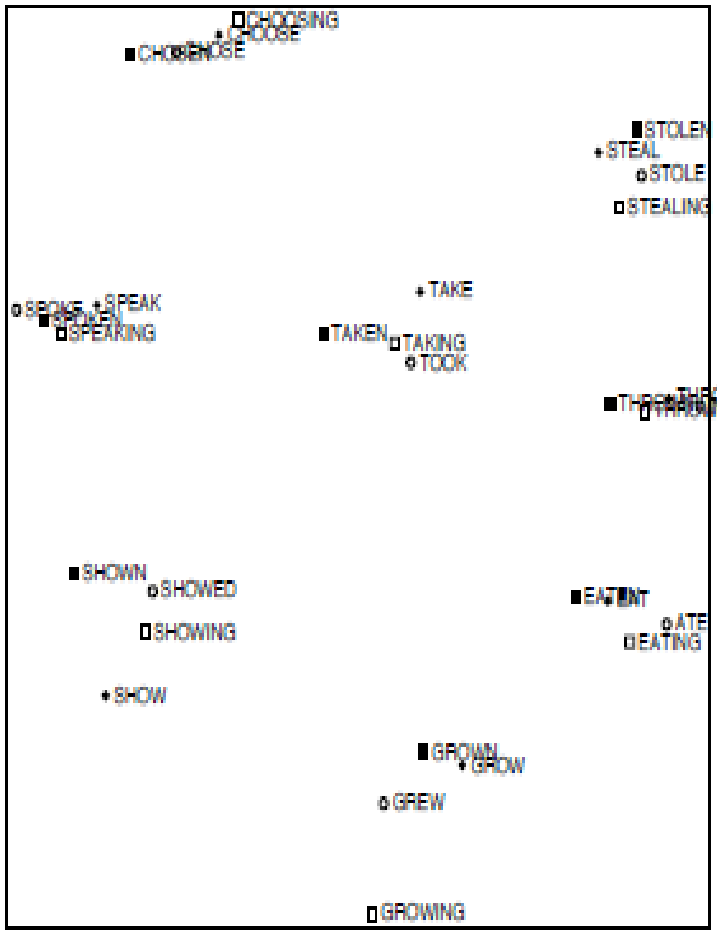
Similarity of d_2 and d_3 in the reduced space: $0.52 * 0.28 + 0.36 * 0.16 + 0.72 * 0.36 + 0.12 * 0.20 + -0.39 * -0.08 \approx 0.52$

Ενδιαφέροντα αποτελέσματα



An Improved Model of Semantic Similarity Based on Lexical Co-Occurrence
Rohde et al. 2005

- Σημασιολογικά



of Semantic Similarity Based on Lexical Co-Occurrence

- Συντακτικά

Προβλήματα με το LSI

- Πολύ μεγάλη υπολογιστική πολυπλοκότητα
- Δύσκολα ενσωματώνονται καινούριες λέξεις ή έγγραφα
- Λύση:
  - Αυτόματη μάθηση διανυσμάτων λέξεων χαμηλής διαστατικότητας
  - **Word2vec** (Mikolov et al. 2013)
    - Ρηχά μοντέλα που χρησιμοποιούνται για την παραγωγή word embeddings

Word Representations μέσω των λέξεων που τις περιβάλλουν σε ένα παράθυρο συμφραζομένων: Window-based co-occurrence matrix

Example corpus:

- I like deep learning.
- I like NLP.
- I enjoy flying.

το «like» συνεμφανίζεται με το «deep» σε παράθυρο συμφραζομένων [-1, +1] μια φορά

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Word Representations μέσω των λέξεων που τις περιβάλλουν σε ένα παράθυρο συμφραζομένων: Window-based co-occurrence matrix

- Οι πίνακες αυτοί
  - Αυξάνουν πολύ σε διαστατικότητα καθώς αυξάνεται το λεξικό
  - Απαιτούν μεγάλους πόρους για αποθήκευση
  - Έχουν προβλήματα πολύ αραιών εγγραφών (sparsity)
- Λύση
  - Κρατάμε μόνο την σημαντική πληροφορία μέσω ενός μικρού αριθμού διαστάσεων (25-1000 διαστάσεις)
    - Word embeddings
  - Οπότε κάθε λέξη αναπαρίσταται σαν ένα πυκνό διάνυσμα

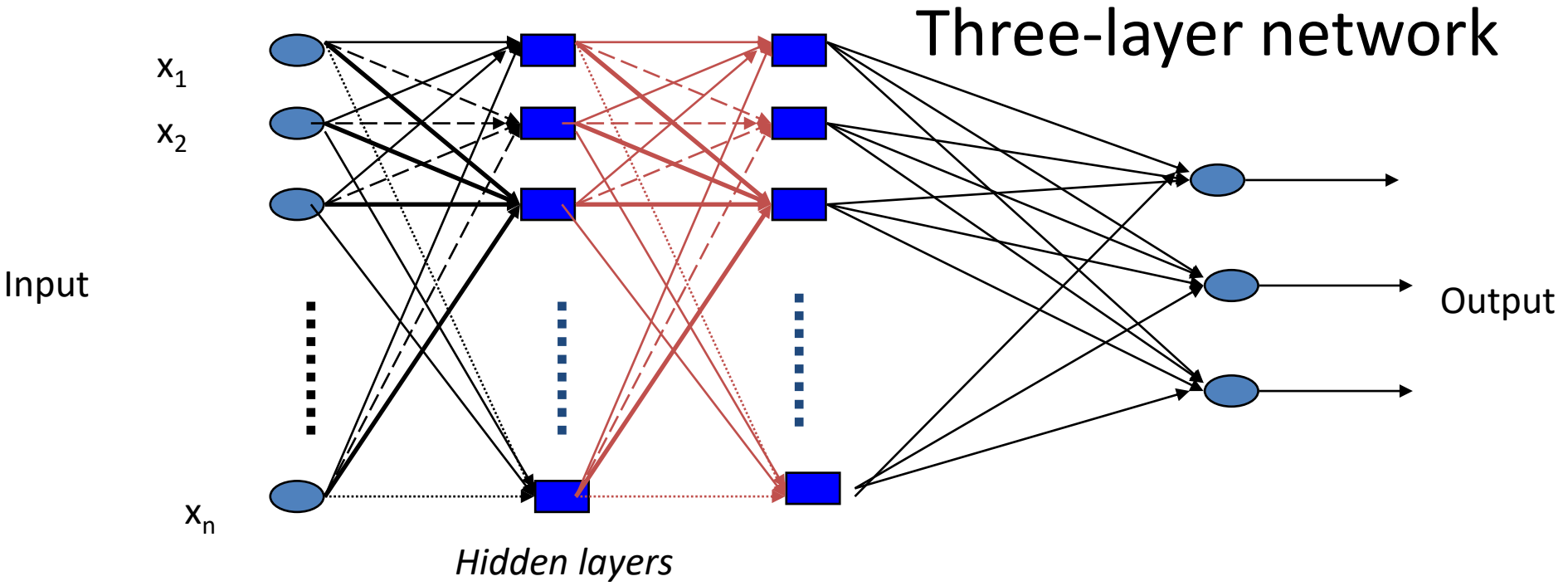
Word embeddings

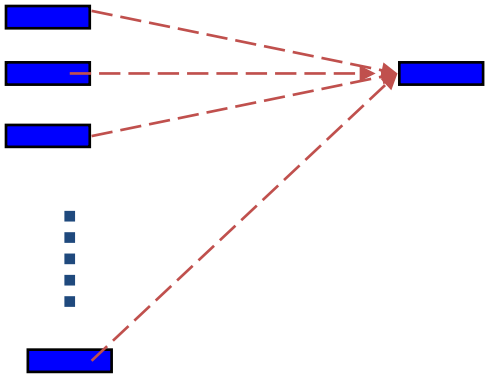
Δημιουργείται ένα πυκνό διάνυσμα για κάθε λέξη

Το διανύσματα δυο λέξεων που εμφανίζονται σε όμοια περιβάλλοντα είναι όμοια (κοντά το ένα στο άλλο στον διανυσματικό χώρο)

$$banking = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$$

Ανοίγει παρένθεση: Ρηχά Νευρωνικά Δίκτυα (Feed-forward neural networks)





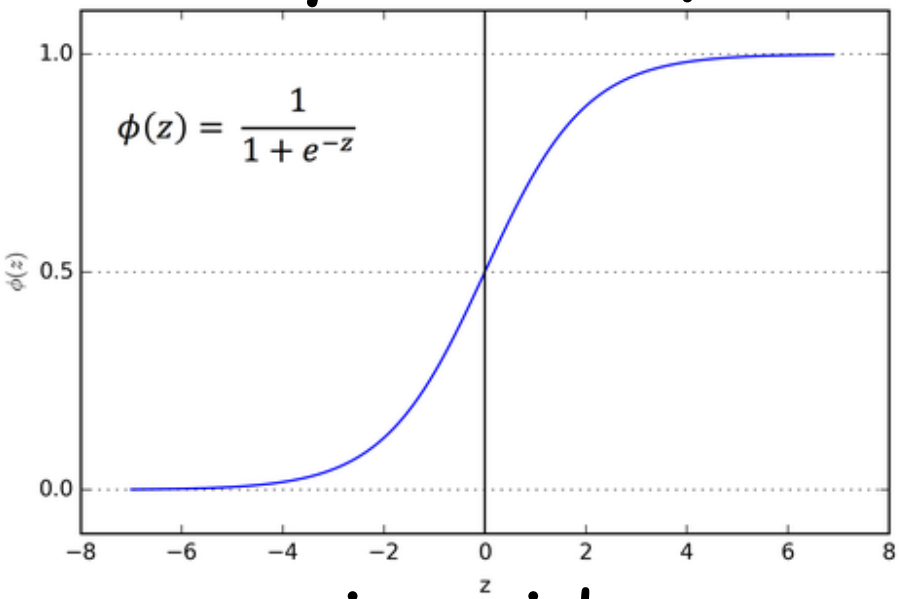
Activation function

$$y_i = f\left(\sum_{j=1}^{m} w_{ij} x_j + b_i\right)$$

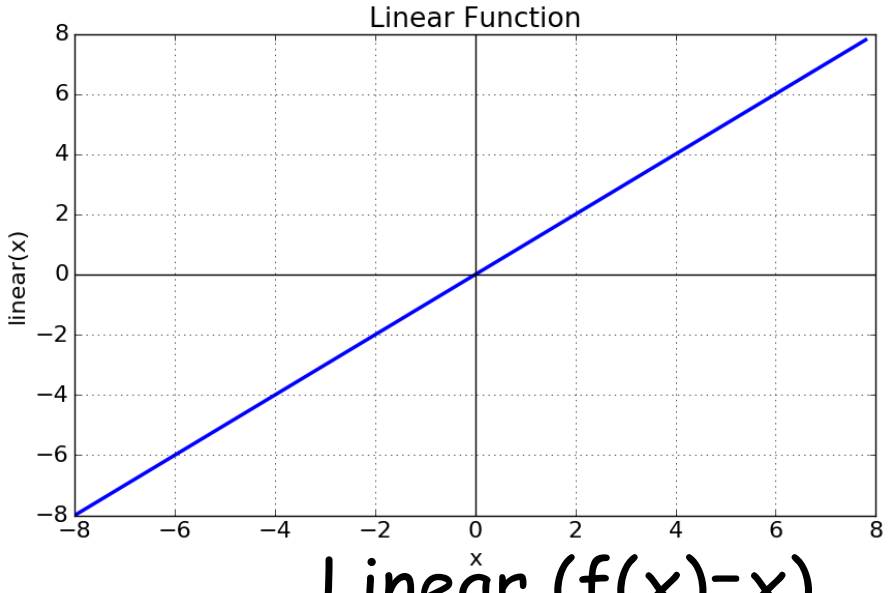
Παρένθεση: Feed-forward Neural Network (συνέχεια)

- Σε ένα FFNN, σε κάθε κρυφό κόμβο
  - Υπολογίζεται το άθροισμα των σταθμισμένων με βάρη εισόδων του κόμβου
    - Τα βάρη αυτά είναι οι παράμετροι που πρέπει το δίκτυο να μάθει κατά την εκπαίδευσή του
  - Εφαρμόζεται στο άθροισμα αυτό μια συνάρτηση ενεργοποίησης (activation function)
  - Η συνάρτηση αυτή μπορεί να είναι γραμμική ή μη γραμμική

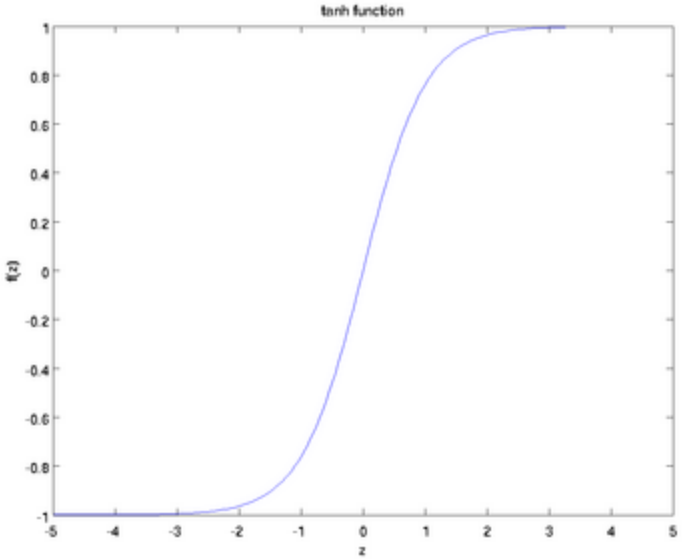
Παρένθεση: Activation functions



sigmoid



Linear (f(x)=x)



$$f(x) = tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

Κλείνει η παρένθεση: Back propagation

- Αρχικά τα βάση παίρνουν τυχαίες τιμές
- Υπολογίζονται οι έξοδοι του τελευταίου επιπέδου
- Υπολογίζεται η διαφορά (σφάλμα Ε) αυτών των εξόδων από τις επιθυμητές εξόδους
- Παίρνουμε την παράγωγο του σφάλματος
- Τα καινούρια βάρη υπολογίζονται από το τελευταίο επίπεδο προς τα πίσω από τα προηγούμενα βάρη, βάσει της σχέσης

$$\theta^{t+1} = \theta^t - \alpha \frac{\partial E(X, \theta^t)}{\partial \theta}$$

- όπου α ένας παράγοντας που ονομάζεται learning rate
- μικρό α: κάθε εποχή (επανάληψη) προκαλεί μικρή μεταβολή στα βάρη – χρειάζονται περισσότερες επαναλήψεις (υπερβολικά μικρό α μπορεί να «κολλήσει» την διαδικασία)
- μεγάλο α: μεγάλες μεταβολές στα βάρη, λιγότερες επαναλήψεις (υπερβολικά μεγάλο α μπορεί να φέρει πολύ γρήγορα σύγκλιση σε τοπικά βέλτιστα – suboptimal solution)

Word Representations μέσω των λέξεων που τις περιβάλλουν σε ένα παράθυρο συμφραζομένων: word2vec

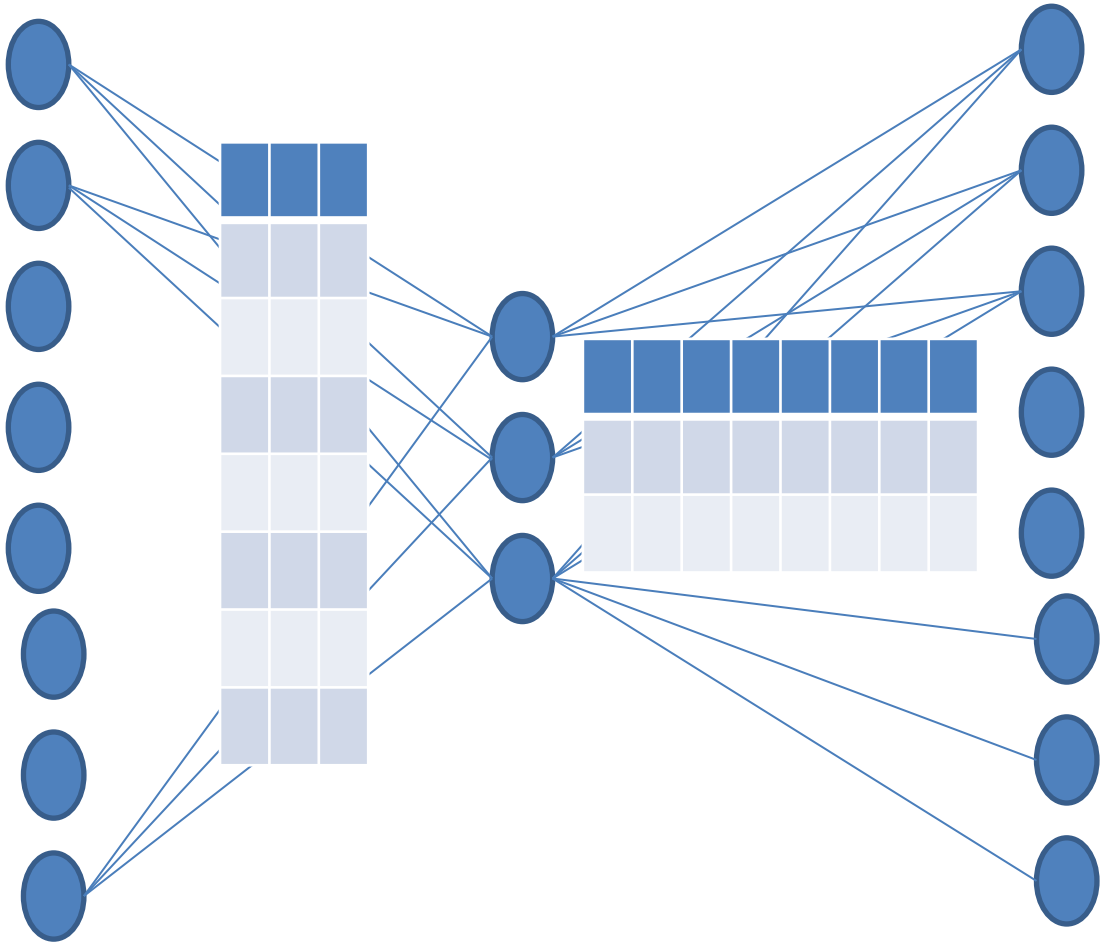
- Έστω το σώμα κειμένων
- the dog saw a cat. the dog chased the cat. the cat climbed a tree.
- Το λεξικό έχει 8 λέξεις: a, cat, chased, climbed, dog, saw, the, tree

	1-hot vectors
a	[1,0,0,0,0,0,0,0]
cat	[0,1,0,0,0,0,0,0]
chased	[0,0,1,0,0,0,0,0]
…	

word2vec: Αρχιτεκτονική ΝΔ

- Ένα νευρωνικό δίκτυο θα έχει σαν είσοδο 8 νευρώνες (όσο το μέγεθος του λεξικού)
- Στην έξοδο θα έχει επίσης 8 νευρώνες (όσο το μέγεθος του λεξικού)
- Θα έχει ένα κρυφό επίπεδο, με τόσους νευρώνες όση και η διάσταση που έχω επιλέξει για τα διανύσματα των λέξεων, έστω 3
- Όλοι οι κρυφοί νευρώνες είναι γραμμικοί.
- Επομένως τα βάρη από το input layer στο κρυφό επίπεδο θα είναι ένας πίνακας WI 8*3, ενώ
- τα βάρη από το κρυφό επίπεδο στο επίπεδο εξόδου θα είναι ένας πίνακας WO 3*8
- Αρχικά οι πίνακες αυτοί παίρνουν μικρές τυχαίες τιμές

Αρχιτεκτονική ΝΔ



Έστω οι αρχικοί WI και WO

WI =

-0.094491	-0.443977	0.313917
-0.490796	-0.229903	0.065460
0.072921	0.172246	-0.357751
0.104514	-0.463000	0.079367
-0.226080	-0.154659	-0.038422
0.406115	-0.192794	-0.441992
0.181755	0.088268	0.277574
-0.055334	0.491792	0.263102

Wo =

0.023074	0.479901	0.432148	0.375480	-0.364732	-0.119840	0.266070	-0.351000
-0.368008	0.424778	-0.257104	-0.148817	0.033922	0.353874	-0.144942	0.130904
0.422434	0.364503	0.467865	-0.020302	-0.423890	-0.438777	0.268529	-0.446787

word2vec: Εκπαίδευση

- Εισάγω στην είσοδο την λέξη cat, δηλ το [0 1 0 0 0 0 0 0]
- Οι έξοδοι των κόμβων του κρυφού επιπέδου θα είναι
- Ht = [0 1 0 0 0 0 0 0] * WI = [-0.490796 -0.229903 0.065460]
- Οι έξοδοι των κόμβων του επιπέδου εξόδου είναι
- Ht * WO = [0.100934 -0.309331 -0.122361 -0.151399 0.143463 -0.051262 -0.079686 0.112928]
- Κάθε ένας κόμβος του επιπέδου εξόδου θα παράξει στην έξοδό του την πιθανότητα μια από τις 8 λέξεις του λεξικού να είναι στο συμφραστικό περιβάλλον της λέξης cat.
- Το άθροισμα αυτών των πιθανοτήτων θα πρέπει να είναι 1.
- Έτσι η δουλειά το επιπέδου εξόδου είναι να κανονικοποιήσει το παραπάνω διάνυσμα, ώστε οι τιμές του να αθροίζουν στο 1.
- Η κανονικοποίηση γίνεται με την συνάρτηση softmax
- Πχ για τον 1ο νευρώνα
- $y1_{\text{κανονικοποιημένη}} = e^{y1}/(e^{y1}+e^{y2}+...+e^{y8})$
- Οπότε το παραπάνω διάνυσμα, ματά την κανονικοποίηση είναι
- [0.143073 0.094925 0.114441 **0.111166** 0.149289 0.122874 0.119431 0.144800]
- Το 0,143 είναι η πιθανότητα η λέξη a να βρεθεί στο περιβάλλον της cat
- Το 0,111 είναι η πιθανότητα η λέξη climbed να βρεθεί στο περιβάλλον της cat

word2vec: Εκπαίδευση

- Πόσο σφάλμα κάνω στον υπολογισμό της πιθανότητας του climbed;
- Αφαιρώ το διάνυσμα [0.143073 0.094925 0.114441 **0.111166** 0.149289 0.122874 0.119431 0.144800]
- Από το διάνυσμα στόχο [0 0 0 1 0 0 0 0]
- Με το σφάλμα γνωστό, μπορώ να ξαναυπολογίσω τους πίνακες WI και WO με backpropagation
- Η εκπαίδευση συνεχίζεται για όλα τα ζεύγη λέξεων εισόδου/λέξεων συμφραστικού περιβάλλοντος του σώματος εκπαίδευσης
- Ο τελικός πίνακας Ht (1X3) που προκύπτει για κάθε λέξη είναι το διάνυσμα αναπαράστασης της λέξης

Πηγές

- https://medium.com/@zafaralibagh6/a-simple-word2vec-tutorial-61e64e38a6a1