10707
Deep Learning: Spring 2020

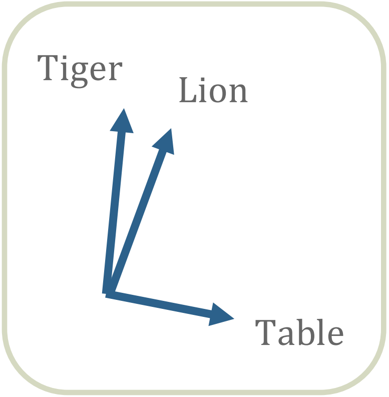
Andrej Risteski

Machine Learning Department

Lecture 20:
Word embeddings, under the hood

Word embeddings

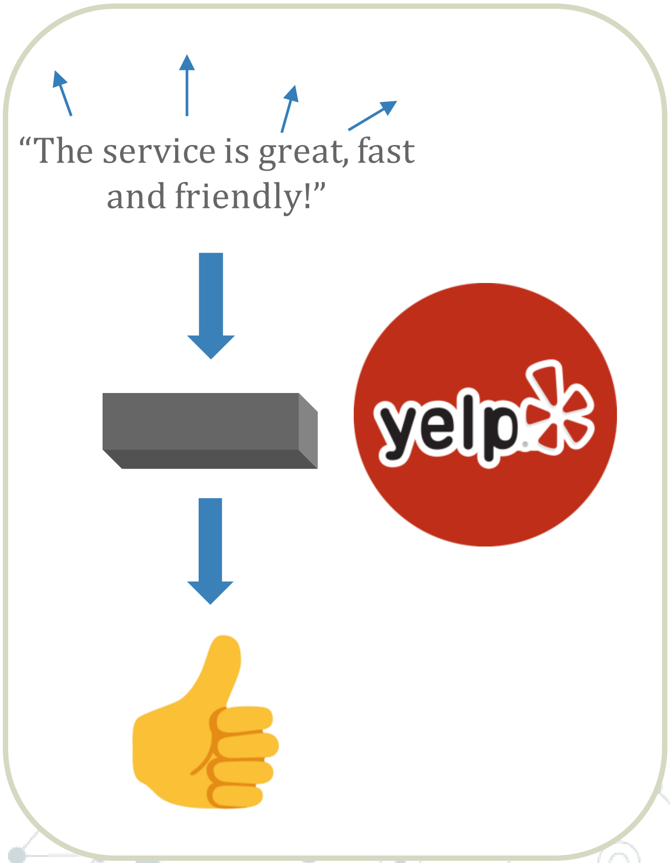
Semantically meaningful **vector representations** of words



Example: Inner product (possibly scaled, i.e. cosine similarity) correlates with word similarity.

Word embeddings

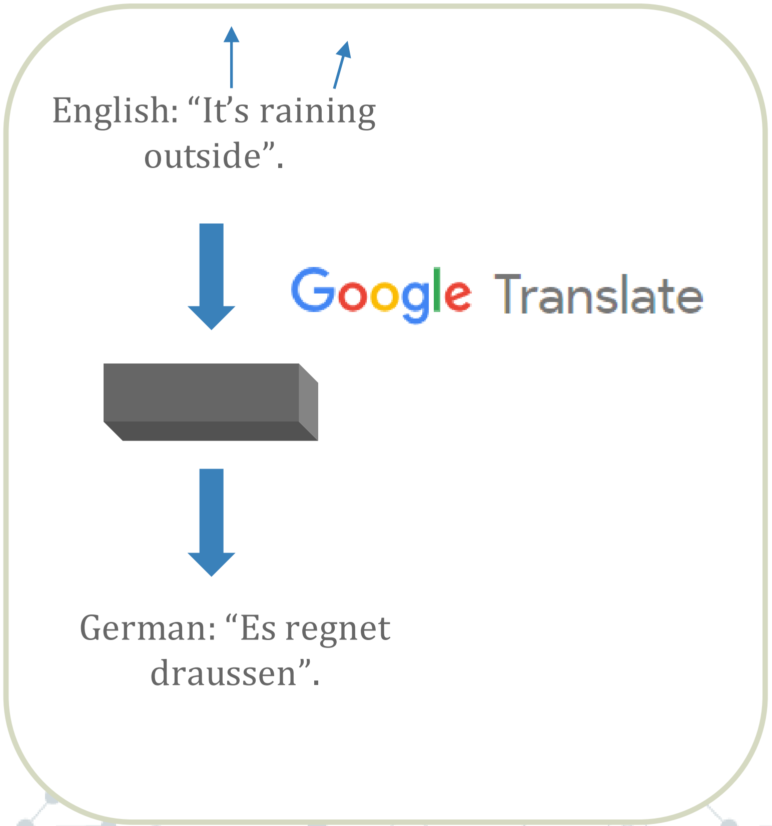
Semantically meaningful **vector representations** of words



Example: Can use embeddings to do sentiment classification by training a simple (e.g. linear) classifier

Word embeddings

Semantically meaningful **vector representations** of words



English: "It's raining outside".
Google Translate
German: "Es regnet draussen".

Example: Can train a "simple" network that if fed word embeddings for two languages, can effectively translate.

Word embeddings via predictive learning

Basic task: predict the next word, given a few previous ones.

I	am	running	a	little	????

Late:	0.9
Early:	0.05
Tired:	0.04
Table:	0.01

In other words, optimize for

$$\max_{\theta} \sum_{t} \log \; p_{\theta}(x_t | x_{t-1}, x_{t-2}, \dots, x_{t-L})$$

Word embeddings via predictive learning

Basic task: predict the next word, given a few previous ones.

$$\max_{\theta} \sum_{t} \log p_{\theta}(x_t | x_{t-1}, x_{t-2}, \dots, x_{t-L})$$

Inspired by classical assumptions in NLP that the underlying distribution is Markov – that is, x_t only depends on the previous few words.

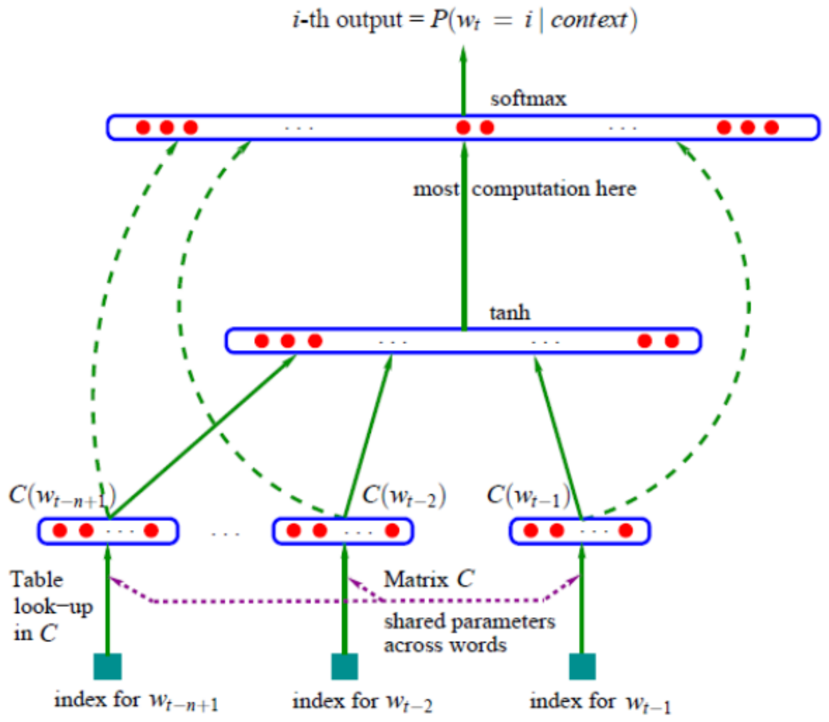
(Of course, this is violated if you wish to model long texts like paragraphs/books.)

The main problem: The trivial way of parametrizing $p_{\theta}(x_t | x_{t-1}, x_{t-2}, \dots, x_{t-L})$ is a “lookup table” with V^L entries.

Word embeddings via predictive learning

Basic task: predict the next word, given a few previous ones.

$$\max_{\theta} \sum_{t} \log p_{\theta}(x_t | x_{t-1}, x_{t-2}, \dots, x_{t-L})$$



[*Bengio-Ducharme-Vincent-Janvin '03*]: A neural parametrization of the above probabilities.

Main ingredients:

Embeddings: A word embedding $C(w)$ for all words w in dictionary.

Non-linear transforms: Potentially deep network taking as inputs i, $C(x_{t-1}), C(x_{t-2}), \dots, C(x_{t-L})$, and outputting some vector o. Can be recurrent net too.

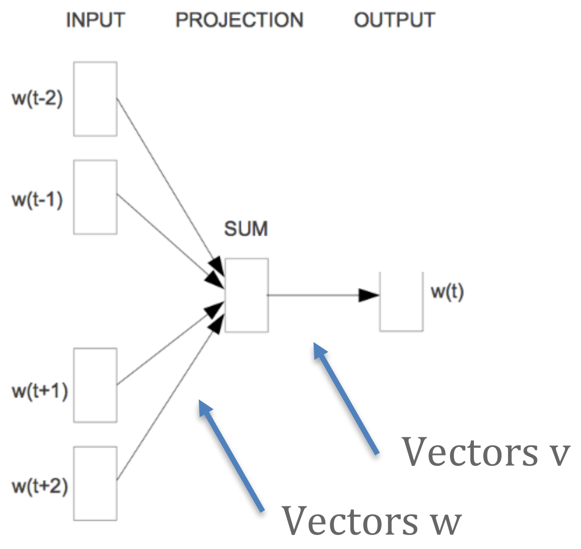
Softmax: Softmax distribution for x_t with parameters given by o.

Word embeddings via predictive learning

Related: predict *middle* word in a sentence, given *surrounding* ones

$$\max_{\theta} \sum_{t} \log p_{\theta}(x_t | x_{t-L}, \dots, x_{t-1}, x_{t+1}, \dots, x_{t+L})$$

CBOW (Continuous Bag of Words): proposed by *Mikolov et al.* '13



Parametrization is chosen s.t.

$$p_{\theta}(x_t | x_{t-L}, \dots, x_{t-1}, x_{t+1}, \dots, x_{t+L}) \propto$$

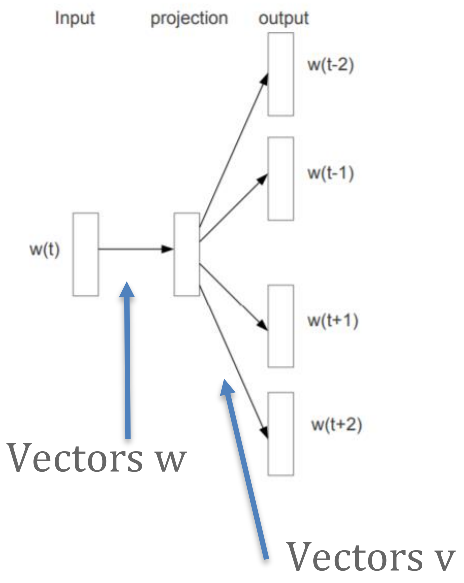
$$\exp\left(v_{x_t}, \sum_{i=t-L}^{t+L} w_{t_i} \right)$$

Word embeddings via predictive learning

Related: predict surrounding words, *given* middle word

$$\max_{\theta} \sum_{t} \sum_{i=t-L, i\neq t}^{t+L} \log\ p_\theta(x_i | x_t)$$

Skip-Gram: (also) proposed by *Mikolov et al.* '13



Parametrization is s.t. $p_\theta(x_i | x_t) \propto \exp(v_{x_i}, w_{x_t})$

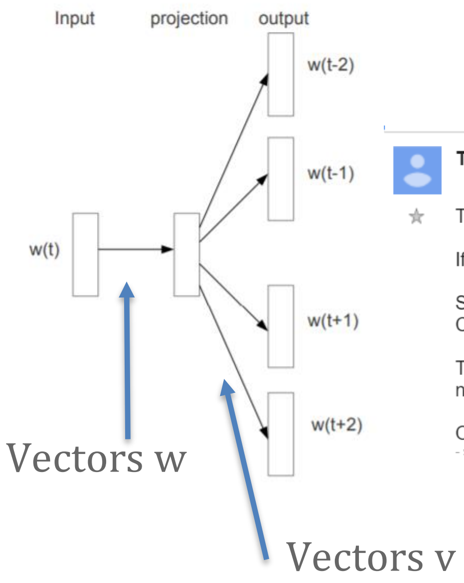
In practice, lots of other tricks are tacked on to deal with the slowest part of training: the softmax distribution (partition function sums over entire vocabulary).
Common ones are *negative sampling, hierarchical softmax*, etc.

Word embeddings via predictive learning

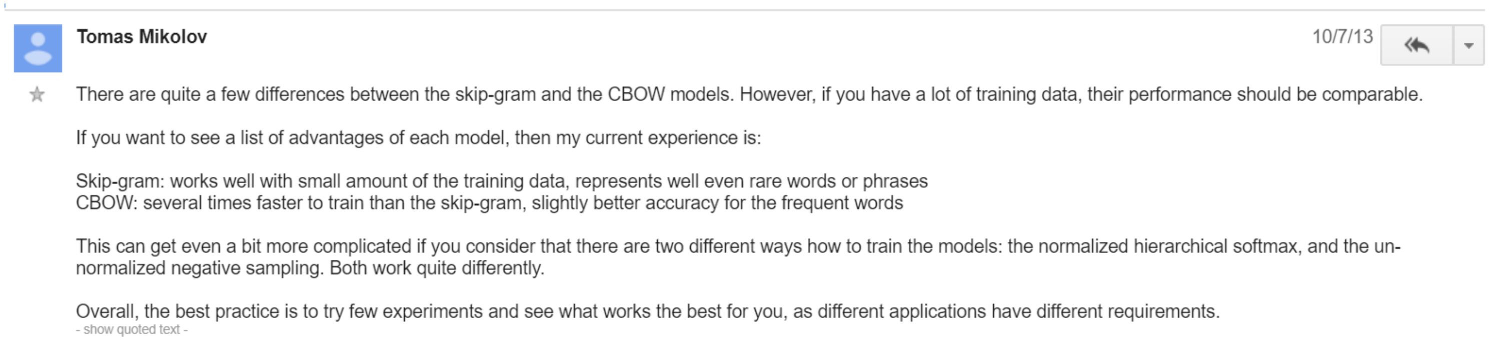
Related: predict surrounding words, *given* middle word

$$\max_{\theta} \sum_{t} \sum_{i=t-L, i\neq t}^{t+L} \log \; p_\theta(x_i | x_t)$$

Skip-Gram: (also) proposed by *Mikolov et al.* '13



Parametrization is s.t. $p_\theta(x_i | x_t) \propto \exp(v_{x_i}, w_{x_t})$



Tomas Mikolov 10/7/13

There are quite a few differences between the skip-gram and the CBOW models. However, if you have a lot of training data, their performance should be comparable.

If you want to see a list of advantages of each model, then my current experience is:

Skip-gram: works well with small amount of the training data, represents well even rare words or phrases
CBOW: several times faster to train than the skip-gram, slightly better accuracy for the frequent words

This can get even a bit more complicated if you consider that there are two different ways how to train the models: the normalized hierarchical softmax, and the un-normalized negative sampling. Both work quite differently.

Overall, the best practice is to try few experiments and see what works the best for you, as different applications have different requirements.
- show quoted text -

Semantic similarity

Observation: similar words tend to have larger (renormalized) inner products (also called cosine similarity).

Precisely, if we look at the word embeddings for words i,j

$\left\langle \frac{w_i}{||w_i||}, \frac{w_j}{||w_j||} \right\rangle = \cos(w_i, w_j)$ tends to be larger for similar words i,j

Example: the nearest neighbors to "Frog" look like

0. *frog*
1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

3. litoria

4. leptodactylidae

5. rana

7. eleutherodactylus

To solve semantic similarity query like "which is the most similar word to", output the word with the highest cosine similarity.

Explaining similarities

Part 1: The distributional hypothesis of language:

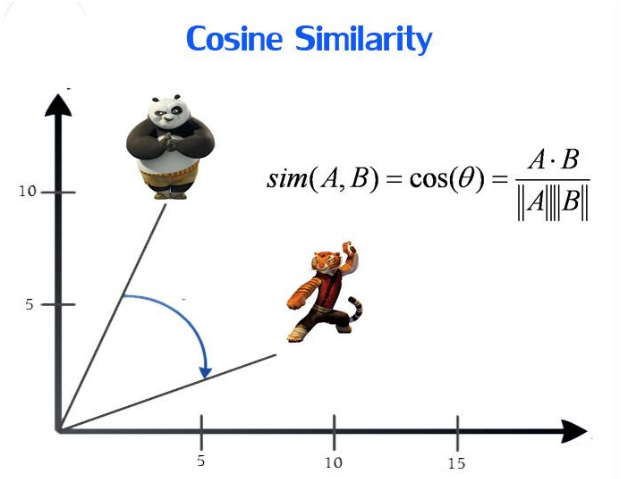
"You shall know a word by the company it keeps."
[Harris'54], [Firth'57])

(Ludwig Wittgenstein)

$M_{w,c}$ = Pr[words w, c co-occur in window of size 5]

Row w is natural *word embedding* for word w.
[HAL; Lund & Burgess'96].

Empirical observation:
Cosine similarity correlates with human ratings of similarity.



Explaining similarities

Follow-up idea: find a "low-dimensional" approximation (both for computational efficiency and "denoising")

$M_{w,c}$ = Pr[words w, c co-occur in window of size 5]

Matrix of inner products is MM^T – so we try to find matrix of embeddings $\widetilde{M} \in \mathbb{R}^{N\times k}$, ($N$ = size of vocabulary, k = dim. of embeddings), s.t. $\widetilde{M}\,\widetilde{M}^T \approx MM^T$, i.e. the similarities are preserved.

Obvious choice: SVD!

Recall: Rank-k SVD of a matrix M gives matrices $U_k, V_k \in \mathbb{R}^{n\times k}, \Sigma_k \in \mathbb{R}^{k\times k}$ s.t. $\|M - U_k\Sigma_k V_k^T\|_F$ is minimized.

LSA [Landauer-Dumais'97]: Semantic vectors via rows of $U_k\Sigma_k$ of rank-k SVD of co-occurence matrix. (say k=300)

COALS [Rohde et al'05]: Before SVD take square root of each entry of co-occurrence matrix. (mysterious transformation – "range squishing")

Explaining similarities

Part 2: SkipGram/CBOW implicitly try to do (weird) dimension reduction

Rewriting the SkipGram objective, we have:

$$\max_{\theta} \sum_{t} \sum_{i=t-L, i \neq t}^{t+L} \log \; p_{\theta}(x_i | x_t)$$

$$= \max_{\theta} \sum_{i,j} f_{ij} \log \frac{\exp(\langle v_i, w_j \rangle)}{\sum_{j'} \exp(\langle v_i, w_{j'} \rangle)}$$

Number of times word i appears
within distance 5 of word j in corpus

Try to maximize inner products for v_i, w_j for words i,j that tend to co-occur more!

Explaining similarities

Part 2: SkipGram/CBOW implicitly try to do (weird) dimension reduction

Levy-Goldberg '13 show that a natural modification of the SkipGram objective (SGNS) results in word vectors satisfying

$$\langle v_i, w_j \rangle \approx PMI(i,j) = \frac{\log P(i,j)}{\log P(i) \log P(j)}$$

Pointwise mutual information, captures co-occurrence correlation b/w i,j

(Normalized) frequency of seeing i,j in a local window

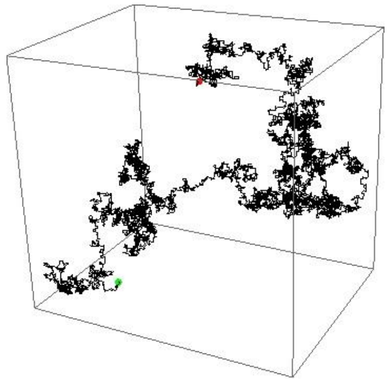
They also show above correspondence holds provably, if word vectors are allowed to be very high dimensional (dimension of vocabulary or more).

But, in practice, good word embeddings are ~300 dimensional, even with corpora as large as all of Wikipedia. ***Why???***

A generative model for language with semantics [Arora, Li, Liang, Ma, Risteski'15]

Corpus is generated sequentially, t-th word produced at time t.

Process driven by **discourse vector** doing a random walk in a d-dim. "discourse space"(d << N = vocab size).



Parameters:

Each word has (time invariant) latent vector v_w

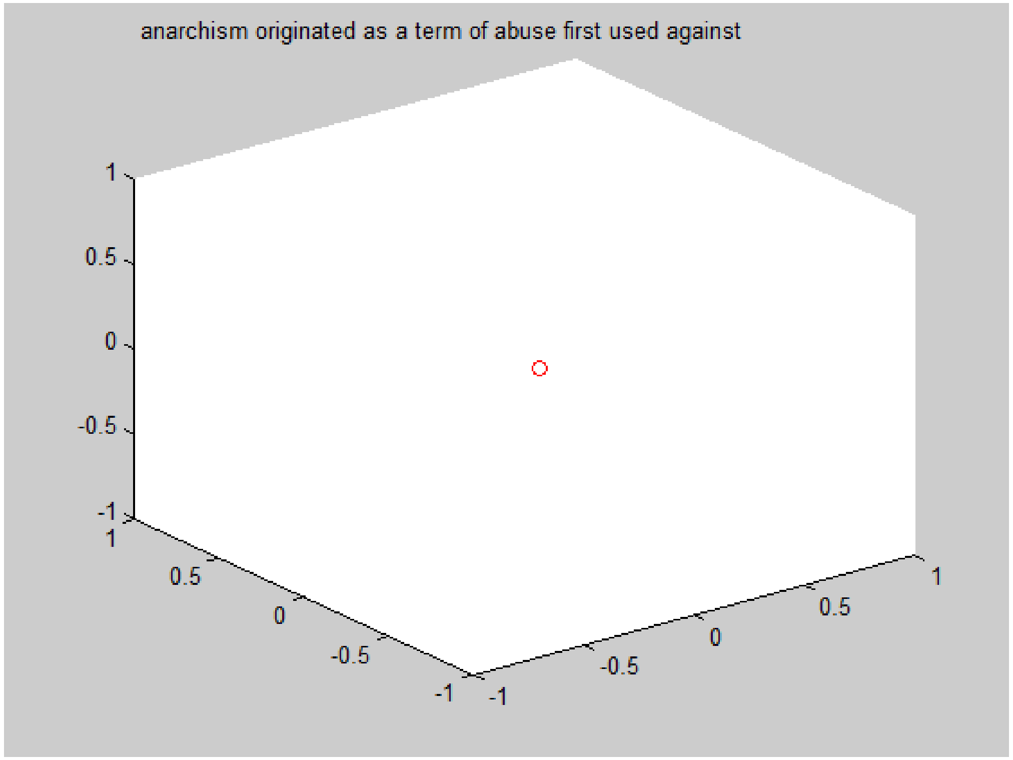
If discourse vector at time t is c_t,

$$\Pr[w \text{ is output}] \propto \exp(v_w \cdot c_t)$$

(Similar to "loglinear topic model" of Mnih-Hinton'07)

Generative model (illustration)

Discourse vector c_t does random walk; $\Pr[w \text{ is output}] \propto \exp(v_w \cdot c_t)$ if context vector at time t is c_t



Assumptions on model parameters

(i) Discourse space: unit sphere; c_t does *slow random walk* – drift

$||c_t - c_{t-1}||_1 \ll \frac{1}{\sqrt{d}}$ with high probability.

Moreover, stationary distribution of random walk is uniform over unit sphere (denoted as $\mathcal{C}$) , and $c_t \sim \mathcal{C}$ for all t (i.e. walk is started from stationary distribution)

(ii) v_w ‘s **spatially isotropic**; drawn from $s\, N(0, I_d)$, s scalar rand. var. Details of s’s distribution are not super important: only that it’s expectation is $\Theta(1)$ and it’s bounded w.p. 1 by some constant.

Hence, $||v_w|| = \theta(\sqrt{d})$ with high probability.

Co-occurrences capture PMI

Let P(w,w') be the probability that w, w' appear consecutively.

Thm:

$$\log p(w,w') = \frac{1}{2d}\|v_w + v_{w'}\|^2 - 2\log Z \pm \epsilon$$

$$\log p(w) = \frac{1}{2d}\|v_w\|_2^2 - \log Z \pm \epsilon$$

$$\log(\mathrm{PMI}(w,w')) = \frac{1}{2d}\langle v_w, v_{w'}\rangle \pm 3\epsilon$$

$$\epsilon = o(1)$$

(Norm of word vec determines frequency; spatial orientation "meaning")

Hence, low dimensional vectors approximately recover PMI!

Note: $||v_w|| = \theta(\sqrt{d})$ whp, so $\frac{1}{2d}||v_w + v_{w'}||^2 = \Theta(1)$, so error is low order.

Note: The introduction of the scalar s allows word probability to vary.

Main proof ingredients

Recall, $\Pr[w \text{ is output}] \propto \exp(v_w \cdot c_t)$

$$= \frac{\exp(v_w \cdot c_t)}{Z_{c_t}}$$

where $Z_c = \sum_w \exp(v_w \cdot c)$ is the partition function

Main idea 1: Z_c is constant within (1± o(1)) for almost all c whp, in other words $(1 - o(1))Z \leq Z_c \leq (1 + o(1))Z$

This crucially uses the spatial isotropy of the word embeddings.

Main idea 2: Context changes little from t to t+1.

Proof sketch

Thm: $\log p(w,w') = \frac{1}{2d}\|v_w + v_{w'}\|^2 - 2\log Z \pm \epsilon$

Pf. Marginalizing, we get:

$$p(w,w') = \int_{c,c'} p(w|c)p(w'|c')p(c,c')dcdc' =$$

$$\int_{c,c'} \frac{\exp(\langle v_w, c\rangle)}{Z_c}\frac{\exp(\langle v_{w'}, c'\rangle)}{Z_{c'}} p(c)p(c'|c)dcdc'$$

c, c' = successive context vectors

Recall: $(1-o(1))Z \leq Z_c \leq (1+o(1))Z$

Lemma: If context drifts slowly then can treat c ≈ c' :

$$\int_{c'} \frac{\exp(\langle v_{w'}, c'\rangle)}{Z_{c'}} p(c)p(c'|c)dc' = (1 \pm o(1))\exp(\langle v_{w'}, c\rangle)$$

Proof sketch (contd.)

Thus, $\int_{c,c'} \frac{\exp(\langle v_w, c\rangle)}{Z_c} \frac{\exp(\langle v_{w'}, c'\rangle)}{Z_{c'}} p(c)p(c'|c)dcdc' =$

$$\frac{(1 \pm o(1))^2}{Z^2} \int_c \exp(\langle v_w + v'_w, c\rangle)p(c)dc$$

However, p(c) is a product distribution, so the above expression factors:

$$\int_c \exp(\langle v_w + v'_w, c\rangle)p(c)dc = \mathbb{E}_c \exp(\langle v_w + v'_w, c\rangle) = \Pi_{i=1}^d \mathbb{E}_{c_i} \exp(v_w + v'_w)_i c_i$$

Distribution of c_i is approximately Gaussian w/ variance 1/d, and for X a Gaussian with variance 1/d, we have

$$\mathbb{E}_X[\exp(aX)] = \frac{\sqrt{d}}{\sqrt{2\pi}} \int_x e^{ax} e^{-d\,x^2} dx = \frac{\sqrt{d}}{\sqrt{2\pi}} \int_x e^{-\left(dx - \frac{a}{2\sqrt{d}}\right)^2} e^{\frac{a^2}{4d}} = e^{\frac{a^2}{4d}}$$

Proof sketch (contd.)

Thus, $\int_{c,c'} \frac{\exp(\langle v_w, c\rangle)}{Z_c} \frac{\exp(\langle v_{w'}, c'\rangle)}{Z_{c'}} p(c)p(c'|c)dcdc' =$

$$\frac{(1 \pm o(1))^2}{Z^2} \int_c \exp(\langle v_w + v'_w, c\rangle)p(c)dc$$

However, p(c) is a product distribution, so the above expression factors:

$$\int_c \exp(\langle v_w + v'_w, c\rangle)p(c)dc = \mathbb{E}_c \exp(\langle v_w + v'_w, c\rangle) = \Pi_{i=1}^d \mathbb{E}_{c_i} \exp(v_w + v'_w)_i c_i$$

$$\approx \exp\left(\frac{1}{4d} ||v_w + v'_w||^2\right)$$

Explaining similarities

We know co-occurrences under model approximately match the PMI similarity measure.

Next, we make a connection to CBOW. Recall in CBOW:

$$p_\theta(x_t | x_{t-L}, \dots, x_{t-1}, x_{t+1}, \dots, x_{t+L}) \propto \exp\left(v_{x_t}, \sum_{i=t-L}^{t+L} w_{t_i} \right)$$

Consider a simplified version of our model, where for the window $(x_{t-L}, \dots, x_{t+L})$ the discourse c does not change at all. Namely:

Sample c $\sim C$, then sample $(x_{t-L}, \dots, x_{t+L}) \sim \Pi_{i=t-L}^{t+L} \exp(\langle v_{x_i}, c \rangle) / Z_c = \frac{\exp(\langle \sum_i v_{x_i}, c \rangle)}{Z_c^L}$

Let's assume $Z_c = Z$ for simplicity as well.

Then, the maximum likelihood estimate for c is:

$$\max_c p(c | x_{t-L}, \dots, x_{t+L}) = \max_c p(c, x_{t-L}, \dots, x_{t+L}) = \max_c p(x_{t-L}, \dots, x_{t+L} | c)$$

Bayes rule

Since p(c) is uniform

Explaining similarities

We know co-occurrences under model approximately match the PMI similarity measure.

Next, we make a connection to CBOW. Recall in CBOW:

$$p_\theta(x_t | x_{t-L}, \dots, x_{t-1}, x_{t+1}, \dots, x_{t+L}) \propto \exp\left(v_{x_t}, \sum_{i=t-L}^{t+L} w_{t_i} \right)$$

Consider a simplified version of our model, where for the window $(x_{t-L}, \dots, x_{t+L})$ the discourse c does not change at all. Namely:

Sample c $\sim C$, then sample $(x_{t-L}, \dots, x_{t+L}) \sim \Pi_{i=t-L}^{t+L} \exp(\langle v_{x_i}, c \rangle) / Z_c = \dfrac{\exp(\langle \sum_i v_{x_i}, c \rangle)}{Z_c^L}$

Let's assume $Z_c = Z$ for simplicity as well.

Then, the maximum likelihood estimate for c is:

$$\begin{aligned} \max_c p(c | x_{t-L}, \dots, x_{t+L}) = \max_c p(c, x_{t-L}, \dots, x_{t+L}) &= \max_c p(x_{t-L}, \dots, x_{t+L} | c) \\ &= \max_c \exp(\langle \textstyle\sum_i v_{x_i}, c \rangle) \end{aligned}$$

Clearly, the solution to this maximization problem is $\dfrac{\sum_i v_{x_i}}{\left\| \sum_i v_{x_i} \right\|}$

Explaining similarities

We know co-occurrences under model approximately match the PMI similarity measure.

Next, we make a connection to CBOW. Recall in CBOW:

$$p_\theta(x_t | x_{t-L}, \dots, x_{t-1}, x_{t+1}, \dots, x_{t+L}) \propto \exp\left(v_{x_t}, \sum_{i=t-L}^{t+L} w_{t_i} \right)$$

Consider a simplified version of our model, where for the window $(x_{t-L}, \dots, x_{t+L})$ the discourse c does not change at all. Namely:

Sample c $\sim C$, then sample $(x_{t-L}, \dots, x_{t+L}) \sim \Pi_{i=t-L}^{t+L} \exp(\langle v_{x_i}, c \rangle) / Z_c = \frac{\exp(\langle \sum_i v_{x_i}, c \rangle)}{Z_c^L}$

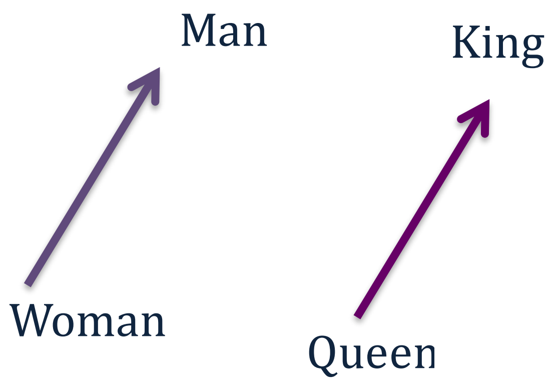
Let's assume $Z_c = Z$ for simplicity as well.

Then, the maximum likelihood estimate for c is $\frac{\sum_i v_{x_i}}{\left\| \sum_i v_{x_i} \right\|}$

Hence, CBOW effectively tries to estimate the discourse, and maximize the probability of v_{x_t}, given this estimate.

Analogies

Observation: You can solve *analogy* queries by linear algebra.



Precisely, *w* = queen will be the solution to:

$$\operatorname{argmin}_w \|v_w - v_{\text{king}} - (v_{\text{woman}} - v_{\text{man}})\|^2$$

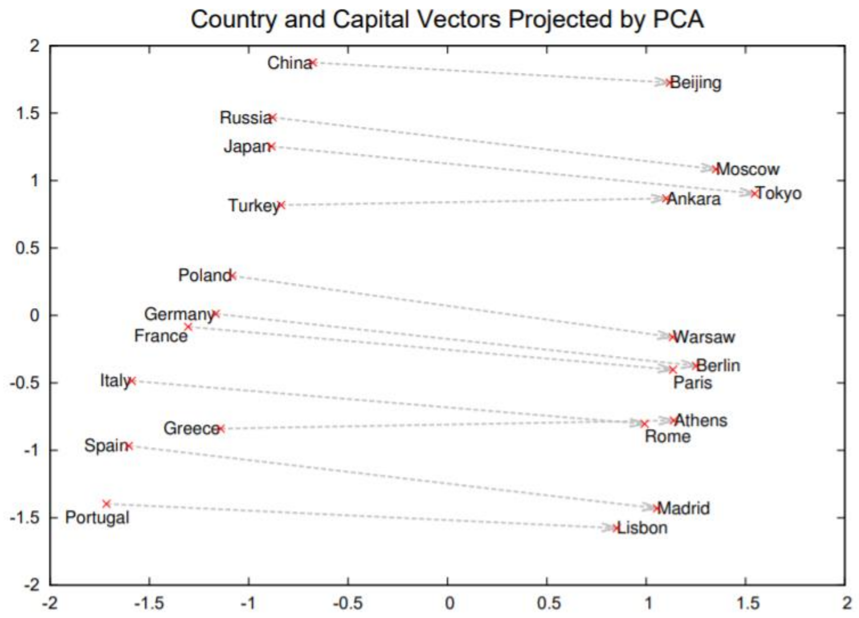


Figure 2: Two-dimensional PCA projection of the 1000-dimensional Skip-gram vectors of countries and their capital cities. The figure illustrates ability of the model to automatically organize concepts and learn implicitly the relationships between them, as during the training we did not provide any supervised information about what a capital city means.

Co-occurrence + log → high-dim. analogy solution [Levy-Goldberg'13]

The solution w should satisfy -- for most words c

$$\frac{P(\chi|\text{man})}{P(\chi|\text{woman})} \approx \frac{P(\chi|\text{king})}{P(\chi|W)}$$

e.g. c = dress → both sides are ≪ 1

c = John → both sides are ≪ 1

c unrelated to gender → both sides are 1.

Suggests: to solve analogy, maybe find word w that minimizes

$$\sum_{\chi} \left(\log \left(\frac{P(\chi \mid \text{man})}{P(\chi \mid \text{woman})} \right) - \log \left(\frac{P(\chi \mid \text{King})}{P(\chi \mid \text{W})} \right) \right)^2$$

Co-occurrence + log → high-dim. analogy solution [Levy-Goldberg'13]

Soln: w minimizing $\sum_{\chi}\left(\log\left(\frac{P(\chi \mid \text{man})}{P(\chi \mid \text{woman})}\right) - \log\left(\frac{P(\chi \mid \text{King})}{P(\chi \mid \text{W})}\right)\right)^2$ (*)

Word embedding: $\text{PMI}(w,\chi) = \log\left(\frac{P(w,\chi)}{P(w)P(\chi)}\right)$

Then $\|v_W - v_{\text{king}} - (v_{\text{woman}} - v_{\text{man}})\|^2$ = (*)

$$\begin{array}{cc} & v_w \\ c \rightarrow & \begin{bmatrix} \\ \text{PMI}(w, c) \\ \\ \end{bmatrix} \end{array}$$

But, success of explicit vectors on analogy tasks not very good.

Empirically one seems to find 300-dim embeddings s.t. $\langle v_w, v_c \rangle \approx$ PMI(w, c). [Pennington et al.'14]

However, error in '≈' up to **17%**, and 4 such inner products are used!

Analogies using low-dimensional vectors in log-linear generative model

We will use our model to explain why low-dimensional vectors actually **improve** analogy solving.

The Levy-Goldberg '13 intuition implies: $\frac{P(\chi \mid \text{king})}{P(\chi \mid \text{queen})} \approx \frac{P(\chi \mid \text{man})}{P(\chi \mid \text{woman})}$

Implicitly: the ratio $\frac{P(\chi|a)}{P(\chi|b)}$ is a function of **semantic relation R** only:

Noise

$$\frac{P(\chi|a)}{P(\chi|b)} = \nu_R(\chi)\xi_{a,b,R}(\chi)$$ Taking logs:

$$\boxed{\log(P(\chi|a)) - \log(P(\chi|b)) = \log(\nu_R(\chi)) + \tilde{n}_{a,b,R}(\chi)}$$

So, for explicit vectors $\tilde{v}_w$, $\tilde{v}_a - \tilde{v}_b = \tilde{\mu}_R + \tilde{n}_{a,b,R}$

$\tilde{\mu}_R = \log(\nu_R)$

$$\tilde{v}_w(\chi) = \log(P(w,\chi)/(P(w)P(\chi)))$$

Error reduction using low-dimensional vectors

$$\boxed{\log(P(\chi|a)) - \log(P(\chi|b)) = \log(\nu_R(\chi)) + \tilde{n}_{a,b,R}(\chi)}$$

$$\tilde{\mu}_R = \log(\nu_R)$$

Hence, want to show: $v_a - v_b = \mu_R + n_{a,b,R}$, $\|n_{a,b,R}\| \ll \|\tilde{n}_{a,b,R}\|$

Let V be matrix with rows v_χ for all words χ. Let $\epsilon_{a,b}$ be the error in the PMI approximation from main thm for generative model.

By main Thm, $\log(P(\chi|a) - \log(P(\chi|b) = \frac{1}{d}\langle v_\chi, v_a - v_b\rangle + \epsilon_{a,b}(\chi)$

$$= \frac{1}{d} V(v_a - v_b) + \epsilon_{a,b}(\chi)$$

Rewriting, we get $\boxed{V(v_a - v_b) = d\tilde{\mu}_R + d(\tilde{n}_{a,b,R}(\chi) - \epsilon_{a,b}(\chi))}$

Error reduction using low-dimensional vectors

Rewriting, we get $V(v_a - v_b) = d\,\tilde{\mu}_R + d(\tilde{n}_{a,b,R}(\chi) - \epsilon_{a,b}(\chi))$

$$\Updownarrow$$

Linear regression with design matrix V !!

$$X\beta = y + \rho$$

"Solving for" $v_a - v_b$, we get $v_a - v_b = V^{\dagger}(d\,\tilde{\mu}_R + d(\tilde{n}_{a,b,R}(\chi) - \epsilon_{a,b}(\chi)))$

If we have V that is well-conditioned, you can denoise a lot more.

For example, if $V \in \mathbb{R}^{N\times d}$ has entries that are random Gaussian with variance 1/N, $\sigma_{\min}(V) = \Omega\left(\sqrt{\frac{N}{d}}\right)$, so $\sigma_{\max}(V^{\dagger}) = O\left(\sqrt{\frac{d}{N}}\right)$

Hence, $||V^{\dagger}d(\tilde{n}_{a,b,R}(\chi) - \epsilon_{a,b}(\chi))||_2 \le d\sqrt{\frac{d}{N}}\,||\,\tilde{n}_{a,b,R}(\chi) - \epsilon_{a,b}(\chi))||_2$

If $d \ll N$, this can be driven to be very small. Other "isotropy" kind of conditions on V suffice (need not be fully random).

Error reduction using low-dimensional vectors

Rewriting, we get $V(v_a - v_b) = d\,\tilde{\mu}_R + d(\tilde{n}_{a,b,R}(\chi) - \epsilon_{a,b}(\chi))$

$$\Updownarrow$$

Linear regression with design matrix V !!

$$X\beta = y + \rho$$

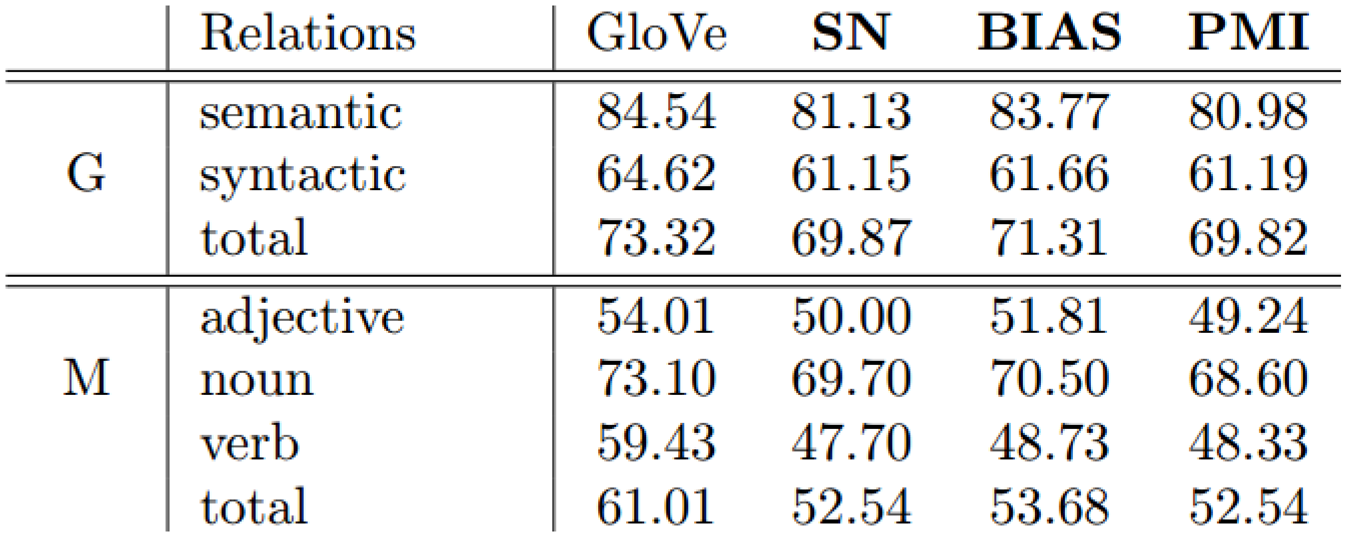
"Solving for" $v_a - v_b$, we get $v_a - v_b = V^{\dagger}(d\,\tilde{\mu}_R + d(\tilde{n}_{a,b,R}(\chi) - \epsilon_{a,b}(\chi)))$

Hence, denoting $\mu_R := V^{\dagger} d\,\tilde{\mu}_R$ and $n_{a,b,R} := \tilde{n}_{a,b,R} - \epsilon_{a,b}$, we get

$v_a - v_b = \mu_R + n_{a,b,R}$, with $\|n_{a,b,R}\| \ll \|\tilde{n}_{a,b,R}\|$ as we wanted.

Experimental results

- Main thm suggest extremely simple fitting algorithm:
  minimize $\sum_{ij}\left(\langle v_i, v_j\rangle - PMI(i,j)\right)^2$
- Despite simplicity, competitive w/ CBOW/SkipGram/GloVe

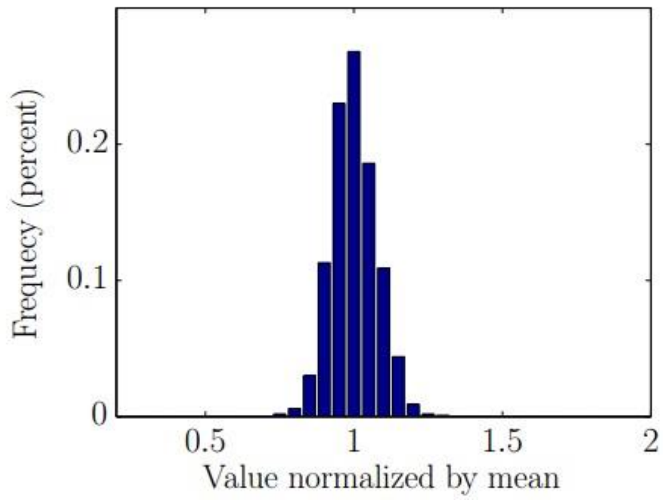


	Relations	GloVe	**SN**	**BIAS**	**PMI**
G	semantic	84.54	81.13	83.77	80.98
	syntactic	64.62	61.15	61.66	61.19
	total	73.32	69.87	71.31	69.82
M	adjective	54.01	50.00	51.81	49.24
	noun	73.10	69.70	70.50	68.60
	verb	59.43	47.70	48.73	48.33
	total	61.01	52.54	53.68	52.54

Verification of generative assumptions

"Partition function is roughly constant": we compute for a 1000 random unit vectors c, $Z_c = \sum_w \exp\langle v_w c\rangle$

Vector isotropy: $||V||_F/\sqrt{d}$ is 34.3 while the minimum non-zero singular value is 11.



Partition function concentration

Polysemy

- **Linear algebraic** decomposition of "meaning"?
  Performing sparse coding on the word vectors results in a "dictionary" of ~2000 "atoms" of discourse. (Arora, Li, Liang, Ma, Risteski '18)

- Each word has a sparse representation in the basis of these atoms.

- More precise and useful than distance-based clustering.

DISK

D1	memory, disk, disks, floppy, storage
D2	install, booting, linux, reinstall, windows
D3	sphere, spherical, geometry, dimensional, euclidean, hyperbolic
D4	kg, lbs, kilograms, weighing, weighs
D5	remastered, dvd, cd, vinyl, disc, reissue