



COS 584

Advanced Natural Language Processing

P3: Word Embeddings

Spring 2021

Improving Distributional Similarity with Lessons Learned from Word Embeddings

Omer Levy

Yoav Goldberg

Ido Dagan

Computer Science Department

Bar-Ilan University

Ramat-Gan, Israel

`{omerlevy, yogo, dagan}@cs.biu.ac.il`

Key takeaways

Count-based approaches

- Used since the 90s
- Sparse word-context PPMI matrix
- Decomposed with SVD

Prediction-based approaches (word embeddings)

- Formulated as a machine learning problem
- Word2vec (Mikolov et al., 2013)
- GloVe (Pennington et al., 2014)

Underlying theory: The Distributional Hypothesis (*Firth, '57*)
“Similar words occur in similar contexts”

- Count-based and prediction-based approaches perform comparably, if you tune the **hyper-parameters** extensively...
- The **hyper-parameters** used in word2vec/GloVe can be **transferable** to count-based approaches!
- **Hyper-parameters** have stronger effects than algorithms and *more data*.

Historical context :)

(Mikolov et al., NIPS'2013)

Distributed Representations of Words and Phrases and their Compositionality

Tomas Mikolov
Google Inc.
Mountain View
mikolov@google.com

Ilya Sutskever
Google Inc.
Mountain View
ilyasu@google.com

Kai Chen
Google Inc.
Mountain View
kai@google.com

Greg Corrado
Google Inc.
Mountain View
gcorrado@google.com

Jeffrey Dean
Google Inc.
Mountain View
jeff@google.com

(Baroni et al., ACL'2014)

Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors

Marco Baroni and **Georgiana Dinu** and **Germán Kruszewski**
Center for Mind/Brain Sciences (University of Trento, Italy)
(marco.baroni|georgiana.dinu|german.kruszewski)@unitn.it

(Pennington et al., EMNLP'2014)

GloVe: Global Vectors for Word Representation

Jeffrey Pennington, **Richard Socher**, **Christopher D. Manning**
Computer Science Department, Stanford University, Stanford, CA 94305
jpennin@stanford.edu, richard@socher.org, manning@stanford.edu

Historical context :)

(Levy and Goldberg, NIPS'2014)

Neural Word Embedding as Implicit Matrix Factorization

Omer Levy

Department of Computer Science
Bar-Ilan University
omerlevy@gmail.com

Yoav Goldberg

Department of Computer Science
Bar-Ilan University
yoav.goldberg@gmail.com

SGNS (= skip-gram with negative-sampling)'s corpus-level achieves its optimal value when:

$$\vec{w} \cdot \vec{c} = \text{PMI}(w, c) - \log k$$

k = # of sampled negative examples and the analysis is based on unigram probability for negative sampling

Four types of word representations

- PPMI
- PPMI + SVD
- SGNS
- GloVe

$$\text{PMI}(\text{word}_1, \text{word}_2) = \log_2 \frac{P(\text{word}_1, \text{word}_2)}{P(\text{word}_1)P(\text{word}_2)}$$

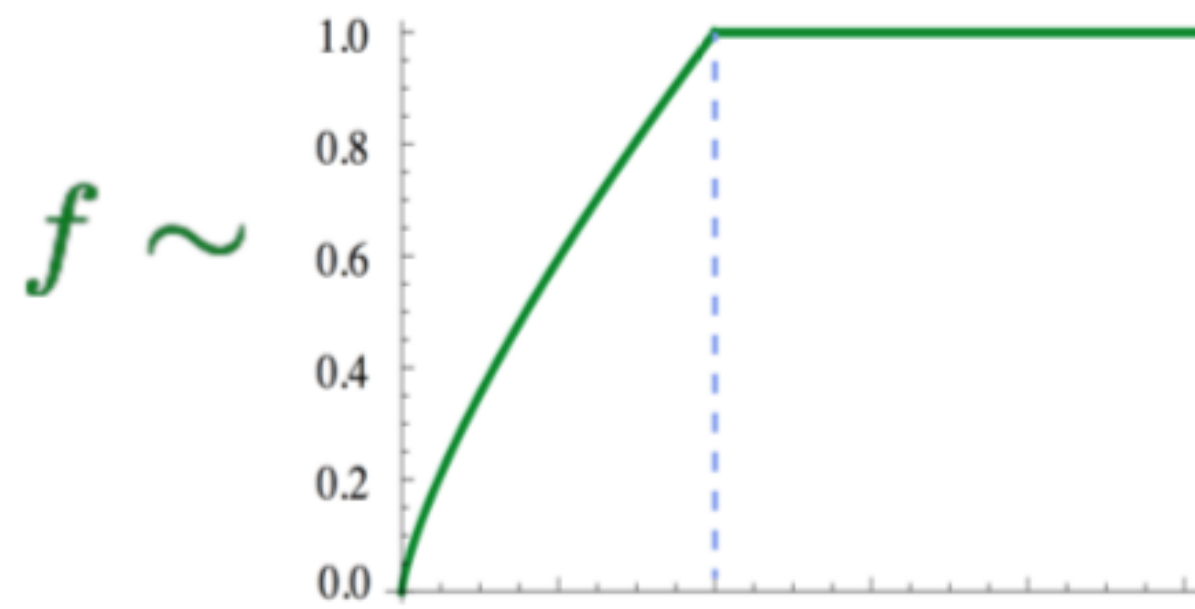
$$\text{PPMI}(\text{word}_1, \text{word}_2) = \max\left(\log_2 \frac{P(\text{word}_1, \text{word}_2)}{P(\text{word}_1)P(\text{word}_2)}, 0\right)$$

$$\begin{bmatrix} X \\ |V| \times |V| \end{bmatrix} = \begin{bmatrix} W \\ |V| \times k \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 \\ 0 & 0 & \sigma_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \sigma_k \end{bmatrix} \begin{bmatrix} C \\ k \times |V| \end{bmatrix}$$

$$y = -\log(\sigma(\mathbf{u}_x \cdot \mathbf{v}_c)) - \sum_{i=1}^K \mathbb{E}_{j \sim P(w)} \log(\sigma(-\mathbf{u}_x \cdot \mathbf{v}_j))$$

GloVe: **G**lobal **V**ectors

- Key idea: let's approximate $\mathbf{u}_i \cdot \mathbf{v}_j$ using their co-occurrence counts $X_{i,j}$ directly.



$$J(\theta) = \sum_{i,j \in V} f(X_{i,j}) \left(\mathbf{u}_i \cdot \mathbf{v}_j + b_i + \tilde{b}_j - \log X_{i,j} \right)^2$$

If we take $b_i = \log(X_i)$, $\tilde{b}_j = \log(X_j)$, GloVe is also similar to factorizing the PMI matrix!

$$\text{PMI}(\text{word}_1, \text{word}_2) = \log_2 \frac{P(\text{word}_1, \text{word}_2)}{P(\text{word}_1)P(\text{word}_2)}$$

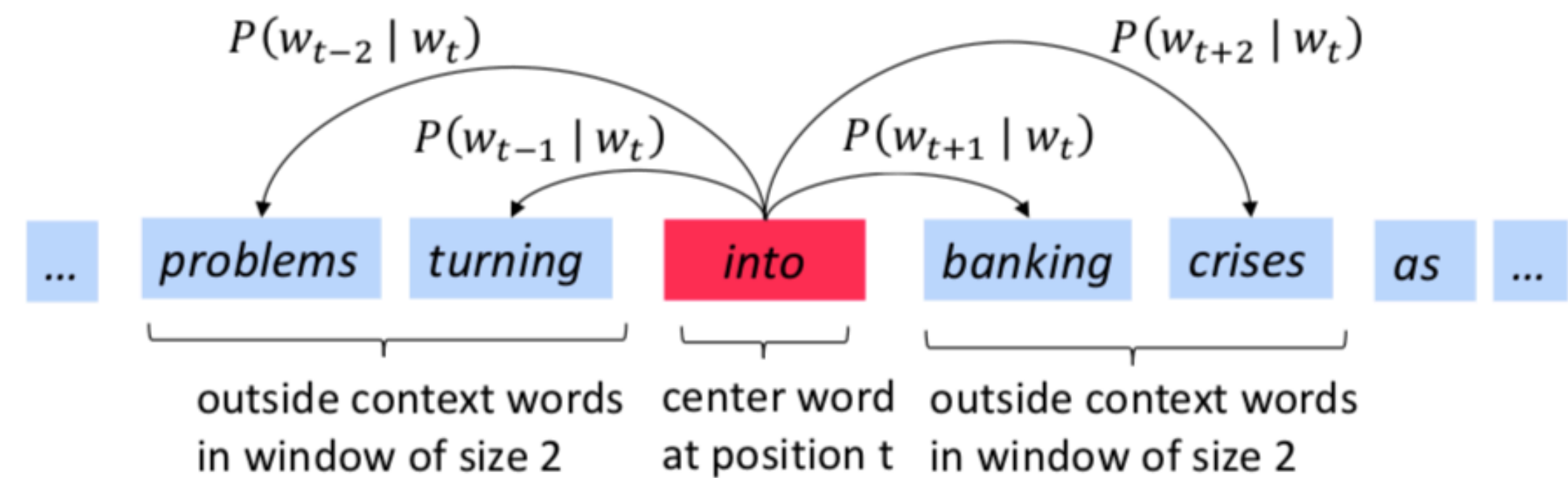


(Pennington et al, 2014): GloVe: Global Vectors for Word Representation

The Big Impact of “Small” Hyperparameters

What are these hyperparameters?

Hyperparameter	Explored Values	Applicable Methods
win	2, 5, 10	All
dyn	none, with	All
sub	none, dirty, clean [†]	All
del	none, with [†]	All
neg	1, 5, 15	PPMI, SVD, SGNS
cds	1, 0.75	PPMI, SVD, SGNS
w+c	only w , $w + c$	SVD, SGNS, GloVe
eig	0, 0.5, 1	SVD
nrm	none [†] , row, col [†] , both [†]	All

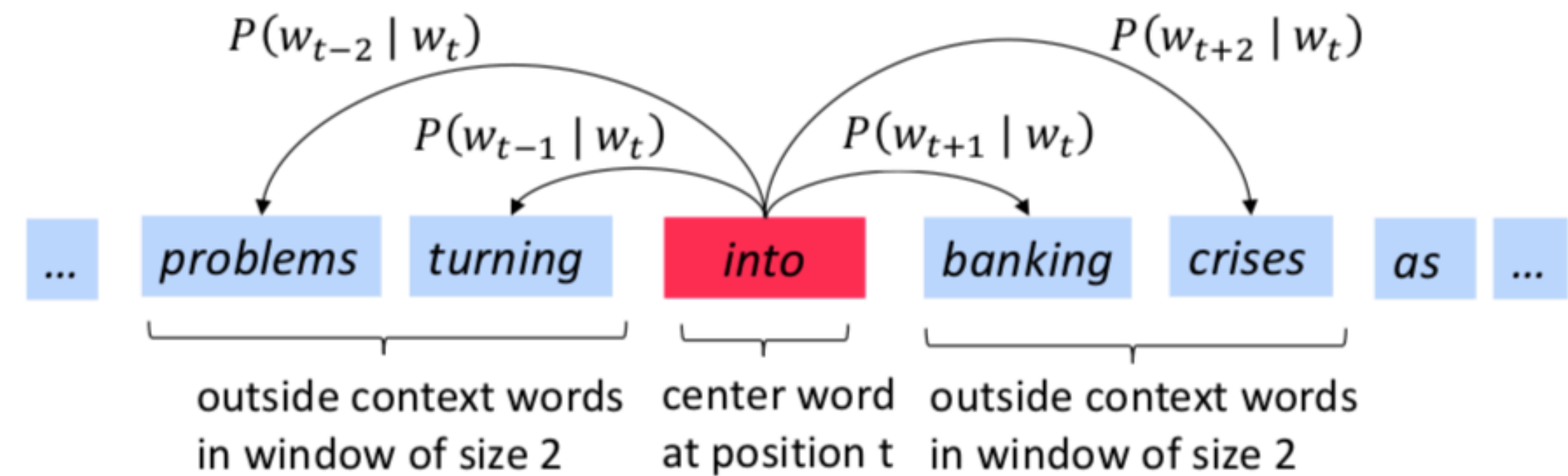


win = window size (# words to the left/right)

The Big Impact of “Small” Hyperparameters

What are these hyperparameters?

Hyperparameter	Explored Values	Applicable Methods
win	2, 5, 10	All
dyn	none, with	All
sub	none, dirty, clean [†]	All
del	none, with [†]	All
neg	1, 5, 15	PPMI, SVD, SGNS
cds	1, 0.75	PPMI, SVD, SGNS
w+c	only w , $w + c$	SVD, SGNS, GloVe
eig	0, 0.5, 1	SVD
nrm	none [†] , row, col [†] , both [†]	All



Dyn = dynamic context window (different weighting for different positions of context words)

1/5 2/5 3/5 4/5 5/5

vs.

1/5 1/5 1/5 1/5 1/5

The Big Impact of “Small” Hyperparameters

What are these hyperparameters?

Hyper-parameter	Explored Values	Applicable Methods
win	2, 5, 10	All
dyn	none, with	All
sub	none, dirty, clean [†]	All
del	none, with [†]	All
neg	1, 5, 15	PPMI, SVD, SGNS
cds	1, 0.75	PPMI, SVD, SGNS
w+c	only w , $w + c$	SVD, SGNS, GloVe
eig	0, 0.5, 1	SVD
nrm	none [†] , row, col [†] , both [†]	All

sub = sub-sampling (removing very frequent words, e.g., a, the)

Each word w_i in the training set is discarded with probability:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

$t = 10^{-5}$, f = unigram prob.

dirty/clean: remove frequent words before and after collecting (word, context) pairs - perform similarly

The Big Impact of “Small” Hyperparameters

What are these hyperparameters?

Hyper-parameter	Explored Values	Applicable Methods
win	2, 5, 10	All
dyn	none, with	All
sub	none, dirty, clean [†]	All
del	none, with [†]	All
neg	1, 5, 15	PPMI, SVD, SGNS
cds	1, 0.75	PPMI, SVD, SGNS
w+c	only w , $w + c$	SVD, SGNS, GloVe
eig	0, 0.5, 1	SVD
nrm	none [†] , row, col [†] , both [†]	All

neg= K in negative sampling

$$y = -\log(\sigma(\mathbf{u}_x \cdot \mathbf{v}_c)) - \sum_{i=1}^K \mathbb{E}_{j \sim P(w)} \log(\sigma(-\mathbf{u}_x \cdot \mathbf{v}_j))$$

$$SPPMI(w, c) = \max(PMI(w, c) - \log k, 0)$$

The Big Impact of “Small” Hyperparameters

What are these hyperparameters?

Hyper-parameter	Explored Values	Applicable Methods
win	2, 5, 10	All
dyn	none, with	All
sub	none, dirty, clean [†]	All
del	none, with [†]	All
neg	1, 5, 15	PPMI, SVD, SGNS
cds	1, 0.75	PPMI, SVD, SGNS
w+c	only $w, w + c$	SVD, SGNS, GloVe
eig	0, 0.5, 1	SVD
nrm	none [†] , row, col [†] , both [†]	All

In word2vec, they sample negative words according to the frequency:

$$P_{\alpha}(w) = \frac{\text{count}(w)^{\alpha}}{\sum_{w'} \text{count}(w')^{\alpha}}$$

$$PMI_{\alpha}(w, c) = \log \frac{\hat{P}(w, c)}{\hat{P}(w) \hat{P}_{\alpha}(c)}$$

$$\hat{P}_{\alpha}(c) = \frac{\#(c)^{\alpha}}{\sum_c \#(c)^{\alpha}}$$

$\alpha = 0.75$

The Big Impact of “Small” Hyperparameters

What are these hyperparameters?

Hyper-parameter	Explored Values	Applicable Methods
win	2, 5, 10	All
dyn	none, with	All
sub	none, dirty, clean [†]	All
del	none, with [†]	All
neg	1, 5, 15	PPMI, SVD, SGNS
cds	1, 0.75	PPMI, SVD, SGNS
w+c	only w , $w + c$	SVD, SGNS, GloVe
eig	0, 0.5, 1	SVD
nrm	none [†] , row, col [†] , both [†]	All

w+c: adding context vectors

$$sim(x, y) = \frac{sim_2(x, y) + sim_1(x, y)}{\sqrt{sim_1(x, x) + 1} \sqrt{sim_1(y, y) + 1}}$$

“They appear in similar contexts” +
“they appear in context of each other”

The Big Impact of “Small” Hyperparameters

What are these hyperparameters?

Hyper-parameter	Explored Values	Applicable Methods
win	2, 5, 10	All
dyn	none, with	All
sub	none, dirty, clean [†]	All
del	none, with [†]	All
neg	1, 5, 15	PPMI, SVD, SGNS
cds	1, 0.75	PPMI, SVD, SGNS
w+c	only w , $w + c$	SVD, SGNS, GloVe
eig	0, 0.5, 1	SVD
nrm	none [†] , row, col [†] , both [†]	All

$$W^{\text{SVD}} = U_d \cdot \Sigma_d$$

$$C^{\text{SVD}} = V_d$$

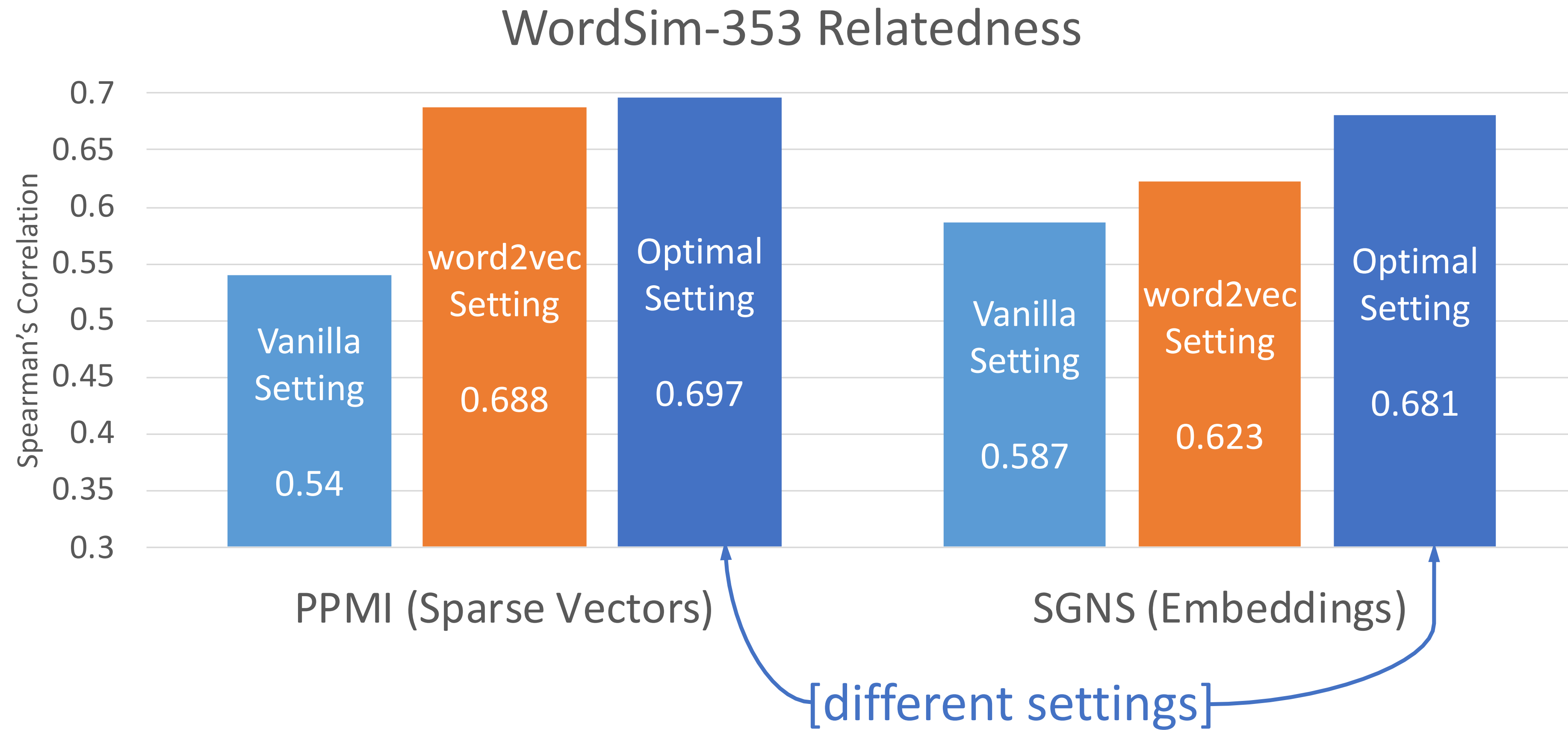
$$W = U_d \cdot \sqrt{\Sigma_d}$$

$$C = V_d \cdot \sqrt{\Sigma_d}$$

$$W = U_d$$

$$C = V_d$$

Main results



Main results

win	Method	WordSim Similarity	WordSim Relatedness	Bruni et al. MEN	Radinsky et al. M. Turk	Luong et al. Rare Words	Hill et al. SimLex	Google Add / Mul	MSR Add / Mul
2	PPMI	.732	.699	.744	.654	.457	.382	.552 / .677	.306 / .535
	SVD	.772	.671	.777	.647	.508	.425	.554 / .591	.408 / .468
	SGNS	.789	.675	.773	.661	.449	.433	.676 / .689	.617 / .644
	GloVe	.720	.605	.728	.606	.389	.388	.649 / .666	.540 / .591
5	PPMI	.732	.706	.738	.668	.442	.360	.518 / .649	.277 / .467
	SVD	.764	.679	.776	.639	.499	.416	.532 / .569	.369 / .424
	SGNS	.772	.690	.772	.663	.454	.403	.692 / .714	.605 / .645
	GloVe	.745	.617	.746	.631	.416	.389	.700 / .712	.541 / .599
10	PPMI	.735	.701	.741	.663	.235	.336	.532 / .605	.249 / .353
	SVD	.766	.681	.770	.628	.312	.419	.526 / .562	.356 / .406
	SGNS	.794	.700	.775	.678	.281	.422	.694 / .710	.520 / .557
	GloVe	.746	.643	.754	.616	.266	.375	.702 / .712	.463 / .519
10	SGNS-LS	.766	.681	.781	.689	.451	.414	.739 / .758	.690 / .729
	GloVe-LS	.678	.624	.752	.639	.361	.371	.732 / .750	.628 / .685



3COSADD vs 3COSMUL

a is to a^* as b is to ? $\arg \max_{b^*} (\cos(b^*, b - a + a^*))$

$$\arg \max_{b^*} \left(\frac{\cos(b^*, b) \cos(b^*, a^*)}{\cos(b^*, a)} \right)$$

<i>queen</i> \cap <i>king</i>	<i>queen</i> \cap <i>woman</i>
uncrowned	Elizabeth
majesty	Katherine
second	impregnate
...	...

Main results

- (Baroni et al, 2014): “word2vec is better than count-based methods” 
- (Pennington et al, 2014): “GloVe is better than word2vec” 
- Two things that made their SVD results much better than previously reported:
 - Use context distribution smoothing
 - Don't use default SVD (eig = 1)
- Use many negative examples for SGNS but shifted PPMI doesn't help.

win	eig	Average Performance
2	0	.612
	0.5	.611
	1	.551
5	0	.616
	0.5	.612
	1	.534
10	0	.584
	0.5	.567
	1	.484

Discussion

- Q2: Which claims in Levy et al., 2015 are most interesting or surprising to you? If you are going to construct a set of word embeddings on a new domain of text, what will be your choices (models and hyper-parameters)?

Conclusions: Methodology

- **Look** for hyperparameters
- **Adapt** hyperparameters across different algorithms
- For good **results**: **tune** hyperparameters
- For good **science**: **tune baselines'** hyperparameters

Thank you :)