Distributed word representations

Christopher Potts

CS 244U: Natural language understanding
April 9

Related materials

- For people starting to implement these models:
  - Socher et al. 2012a; Socher and Manning 2013
  - Unsupervised Feature Learning and Deep Learning
  - Deng and Yu (2014)
  - `http://www.stanford.edu/class/cs224u/code/shallow_neuralnet_with_backprop.py`
- For people looking for new application domains:
  - Baroni et al. (2012)
  - Huang et al. (2012)
  - Unsupervised Feature Learning and Deep Learning: Recommended readings

Goals of semantics (from class meeting 2)

How are distributional vector models doing on our core goals?

1. Word meanings $\approx$
2. Connotations $\checkmark$
3. Compositionality
4. Syntactic ambiguities
5. Semantic ambiguities ?
6. Entailment and monotonicity ?
7. Question answering

(Items in red seem like reasonable goals for lexical models.)

Thought experiment: vectors as classifier features

Class	Word
0	awful
0	terrible
0	lame
0	worst
0	disappointing
1	nice
1	amazing
1	wonderful
1	good
1	awesome

(a) Training set.

$\Pr(\text{Class} = 1)$	Word
?	w_1
?	w_2
?	w_3
?	w_4

(b) Test/prediction set.

Figure: A hopeless supervised set-up.

Thought experiment: vectors as classifier features

Class	Word	excellent	terrible
0	awful	−0.69	1.13
0	terrible	−0.13	3.09
0	lame	−1.00	0.69
0	worst	−0.94	1.04
0	disappointing	0.19	0.09
1	nice	0.08	−0.07
1	amazing	0.71	−0.06
1	wonderful	0.66	−0.76
1	good	0.21	0.11
1	awesome	0.67	0.26

(a) Training set.

Pr(Class=1)	Word	excellent	terrible
≈ 0	w_1	−0.47	0.82
≈ 0	w_2	−0.55	0.84
≈ 1	w_3	0.49	−0.13
≈ 1	w_4	0.41	−0.11

(b) Test/prediction set.

Figure: Values derived from a PMI weighted word $\times$ word matrix and used as features in a logistic regression fit on the training set. The test examples are, from top to bottom, *bad*, *horrible*, *great*, and *best*.

Distributed and distributional

All the representations we discuss are vectors, matrices, and perhaps higher-order tensors. They are all 'distributed' in a sense.

1. 'Distributional' suggests a basis in counts gathered from co-occurrence statistics (perhaps with reweighting, etc.).
2. 'Distributed' connotes deep learning and suggests that the dimensions (or subsets thereof) capture meaningful aspects of natural language objects. See also 'word embedding'.
3. The line will be blurred if we begin with distributional vectors and derive hidden representations from them.
4. For discussion, see Turian et al. 2010:§3, 4.
5. We can reserve 'neural' for representations trained with neural networks. These are always 'distributed' and might or might not have distributional aspects in the sense of 1 above.
6. (But be careful who you say 'neural' to.)

Applications of distributed representations to date

- Sentiment analysis (Socher et al. 2011b, 2012b, 2013b)
- Morphology (Luong et al. 2013)
- Parsing (Socher et al. 2013a)
- Semantic parsing (Lewis and Steedman 2013)
- Paraphrase (Socher et al. 2011a)
- Analogies (Mikolov et al. 2013)
- Language modeling (Collobert et al. 2011)
- Named entity recognition (Collobert et al. 2011)
- Part of speech tagging (Collobert et al. 2011)
- . . .

(With apologies to everyone in speech, cogsci, vision, . . .)

Plan and goals for today

Plan

1. Discuss how to capture entailment
2. (Shallow) neural networks as extensions of discriminative classifier models
3. Unsupervised training of distributed word representations
4. Modeling lexical ambiguity with distributed representations

Goals

- Help you navigate the literature
- Relate this material to things you already know about
- Address the foundational issues of entailment and ambiguity

Entailment in vector space

Last time, we focused exclusively on the relation VSMs capture best: similarity (fuzzy synonymy).

What about entailment? Its asymmetric nature poses challenges.

1. *poodle* $\Rightarrow$ *dog* $\Rightarrow$ *mammal*
2. *run* $\Rightarrow$ *move*
3. *will* $\Rightarrow$ *might*
4. *superb* $\Rightarrow$ *good*
5. *awful* $\Rightarrow$ *bad*
6. *every* $\Rightarrow$ *most* $\Rightarrow$ *some*
7. *probably* $\Rightarrow$ *possibly*

My review is based on Kotlerman et al. 2010.

Lexical relations in WordNet: many entailment concepts

method	adjective	noun	adverb	verb
hypernyms	0	74389	0	13208
instance_hypernyms	0	7730	0	0
hyponyms	0	16693	0	3315
instance_hyponyms	0	945	0	0
member_holonyms	0	12201	0	0
substance_holonyms	0	551	0	0
part_holonyms	0	7859	0	0
member_meronyms	0	5553	0	0
substance_meronyms	0	666	0	0
part_meronyms	0	3699	0	0
attributes	620	320	0	0
entailments	0	0	0	390
causes	0	0	0	218
also_sees	1333	0	0	1
verb_groups	0	0	0	1498
similar_tos	13205		0	0
total	18156	82115	3621	13767

Table: Synset-level relations.

Lexical relations in WordNet: many entailment concepts

method	adjective	noun	adverb	verb
antonyms	3872	2120	707	1069
derivationally_related_forms	10531	26758	1	13102
also_sees	0	0	0	324
verb_groups	0	0	0	2
pertainyms	46650	0	3220	0
topic_domains	6	3	0	1
region_domains	1	14	0	0
usage_domains	1	365	0	2
total	61061	29260	3928	14500

Table: Lemma-level relations.

Conceptualizing the problem

Which row vectors entail which others?

	d_1	d_2	d_3
w_1	1	0	0
w_2	0	0	10
w_3	0	0	20
w_4	0	10	10
w_5	20	20	20

Possible criteria:

- Subset relationship on environments
- Score sizes
- Similarity of score vectors
- ...

Measures: preliminaries

Definition (Feature functions)

Let u be a vector of dimension n. Then F_u is the partial function from $[1, n]$ such that $F_u(i)$ is defined iff $1 \leqslant i \leqslant n$ and $u_i > 0$. Where defined, $F_u(i) = u_i$.

Definition (Feature function membership)

$i \in F_u$ iff i is defined for F_u

Definition (Feature function intersection)

$F_u \cap F_v = \{i : i \in F_u \text{ and } i \in F_v\}$

Definition (Feature function cardinality)

$|F_u| = \Big|\{i : i \in F_u\}\Big|$

Measure: *WeedsPrec*

Definition (Weeds and Weir 2003)

$$WeedsPrec(u,v) \stackrel{def}{=} \frac{\sum_{i \in F_u \cap F_v} F_u(i)}{\sum_{i \in F_u} F_u(i)}$$

	d_1	d_2	d_3
w_1	1	0	0
w_2	0	0	10
w_3	0	0	20
w_4	0	10	10
w_5	20	20	20

(a) Original matrix

	w_1	w_2	w_3	w_4	w_5
w_1	1.0	0.0	0.0	0.0	1.0
w_2	0.0	1.0	1.0	1.0	1.0
w_3	0.0	1.0	1.0	1.0	1.0
w_4	0.0	0.5	0.5	1.0	1.0
w_5	0.3	0.3	0.3	0.7	1.0

(b) Predictions. Max values highlighted. Entailment testing from row to column.

Table: *WeedsPrec*

Measure: *ClarkeDE*

Definition (Clarke 2009)

$$ClarkeDE(u,v) \stackrel{def}{=} \frac{\sum_{i \in F_u \cap F_v} \min\big(F_u(i), F_v(i)\big)}{\sum_{i \in F_u} F_u(i)}$$

	d_1	d_2	d_3
w_1	1	0	0
w_2	0	0	10
w_3	0	0	20
w_4	0	10	10
w_5	20	20	20

(a) Original matrix

	w_1	w_2	w_3	w_4	w_5
w_1	1.0	0.0	0.0	0.0	1.0
w_2	0.0	1.0	1.0	1.0	1.0
w_3	0.0	0.5	1.0	0.5	1.0
w_4	0.0	0.5	0.5	1.0	1.0
w_5	0.0	0.2	0.3	0.3	1.0

(b) Predictions. Max values highlighted. Entailment testing from row to column.

Table: *ClarkeDE*

Measure: *APinc*

Definition (Kotlerman et al. 2010)

$$APinc(u,v) \stackrel{def}{=} \frac{\sum_{i \in F_u} P(i) \cdot rel(F_r)}{|F_v|}$$

1. $rank(i, F_u) =$ the rank of $F_u(i)$ according to the value of $F_u(i)$
2. $P(i) = \frac{\left|\{j \in F_v : rank(j, F_u) \leqslant rank(i, F_u)\}\right|}{rank(i, F_u)}$
3. $rel(i) = \begin{cases} 1 - \frac{rank(i, F_v)}{|F_v|+1} & \text{if } i \in F_v \\ 0 & \text{if } i \notin F_v \end{cases}$

	d_1	d_2	d_3
w_1	1	0	0
w_2	0	0	10
w_3	0	0	20
w_4	0	10	10
w_5	20	20	20

(a) Original matrix

	w_1	w_2	w_3	w_4	w_5
w_1	0.5	0.0	0.0	0.0	0.2
w_2	0.0	0.5	0.5	0.2	0.1
w_3	0.0	0.5	0.5	0.2	0.1
w_4	0.0	0.2	0.2	0.5	0.2
w_5	0.5	0.2	0.2	0.3	0.5

(b) Predictions. Max values highlighted. Entailment testing from row to column.

Balancing

Definition (Lin 1998)

$$LIN(u,v) \stackrel{def}{=} \frac{\sum_{i \in F_u \cap F_v} F_u(i) + F_v(i)}{\sum_{i \in F_u} F_u(i) + \sum_{i \in F_v} F_v(i)}$$

Definition (Kotlerman et al. 2010)

If $E \in \{WeedsPrec, ClarkeDE, APinc\}$, then

$$balE(u,v) \stackrel{def}{=} \sqrt{LIN(u,v) \cdot E(u,v)}$$

Comparisons

	d_1	d_2	d_3
w_1	1	0	0
w_2	0	0	10
w_3	0	0	20
w_4	0	10	10
w_5	20	20	20

	w_1	w_2	w_3	w_4	w_5
w_1	1.0	0.0	0.0	0.0	1.0
w_2	0.0	1.0	1.0	1.0	1.0
w_3	0.0	1.0	1.0	1.0	1.0
w_4	0.0	0.5	0.5	1.0	1.0
w_5	0.3	0.3	0.3	0.7	1.0

(a) *WeedsPrec*

	w_1	w_2	w_3	w_4	w_5
w_1	1.0	0.0	0.0	0.0	0.6
w_2	0.0	1.0	1.0	0.8	0.7
w_3	0.0	1.0	1.0	0.9	0.7
w_4	0.0	0.6	0.6	1.0	0.9
w_5	0.3	0.4	0.4	0.7	1.0

(b) *balWeedsPrec*

Table: *WeedsPrec* with and without balancing.

Comparisons

	d_1	d_2	d_3
w_1	1	0	0
w_2	0	0	10
w_3	0	0	20
w_4	0	10	10
w_5	20	20	20

	w_1	w_2	w_3	w_4	w_5
w_1	1.0	0.0	0.0	0.0	1.0
w_2	0.0	1.0	1.0	1.0	1.0
w_3	0.0	0.5	1.0	0.5	1.0
w_4	0.0	0.5	0.5	1.0	1.0
w_5	0.0	0.2	0.3	0.3	1.0

(a) *ClarkeDE*

	w_1	w_2	w_3	w_4	w_5
w_1	1.0	0.0	0.0	0.0	0.6
w_2	0.0	1.0	1.0	0.8	0.7
w_3	0.0	0.7	1.0	0.6	0.7
w_4	0.0	0.6	0.6	1.0	0.9
w_5	0.1	0.3	0.4	0.5	1.0

(b) *balClarkeDE*

Table: *ClarkeDE* with and without balancing.

Comparisons

	d_1	d_2	d_3
w_1	1	0	0
w_2	0	0	10
w_3	0	0	20
w_4	0	10	10
w_5	20	20	20

	w_1	w_2	w_3	w_4	w_5
w_1	0.5	0.0	0.0	0.0	0.2
w_2	0.0	0.5	0.5	0.2	0.1
w_3	0.0	0.5	0.5	0.2	0.1
w_4	0.0	0.2	0.2	0.5	0.2
w_5	0.5	0.2	0.2	0.3	0.5

(a) *APinc*

	w_1	w_2	w_3	w_4	w_5
w_1	0.7	0.0	0.0	0.0	0.3
w_2	0.0	0.7	0.7	0.3	0.2
w_3	0.0	0.7	0.7	0.4	0.2
w_4	0.0	0.4	0.4	0.7	0.4
w_5	0.4	0.3	0.3	0.5	0.7

(b) *balAPinc*

Table: *APinc* with and without balancing.

Entailment between nouns (Baroni et al. 2012)

	Relationship	Size
Positive class	$A\ N \Rightarrow N$	1246 pairs
Negative class	$A\ N_2 \not\Rightarrow N_1$	1246 pairs

Table: Training data. All the data were manually checked after generation, and all the phrase types have at least 100 tokens in their data.

Positive

- *tall student* $\Rightarrow$ *student*
- *wooden desk* $\Rightarrow$ *desk*
- *skillful linguist* $\Rightarrow$ *linguist*

Negative

- *tall student* $\Rightarrow$ *desk*
- *wooden desk* $\Rightarrow$ *linguist*
- *skillful linguist* $\Rightarrow$ *criminal*

- *alleged criminal* $\not\Rightarrow$ *criminal*
- *fake gun* $\not\Rightarrow$ *gun*

Entailment between nouns (Baroni et al. 2012)

	Relationship	Size
Positive class	$A\ N \Rightarrow N$	1246 pairs
Negative class	$A\ N_2 \not\Rightarrow N_1$	1246 pairs

Table: Training data. All the data were manually checked after generation, and all the phrase types have at least 100 tokens in their data.

	Relationship	Size
Positive class	$N_1 \Rightarrow N_2$	1385 pairs, from WordNet hypernym chains
Negative class	$N_1 \not\Rightarrow N_2$	1385 pairs, by inverting and shuffling the positive pairs

Table: Test data.

Unsupervised method (Baroni et al. 2012)

The authors use *balAPinc* as defined above and find that it beats their frequency- and similarity-based baselines on the nouns task but that it performs poorly on their quantifier task. (See page 30 for details on the performance and the thresholds used to define entailment categorically.)

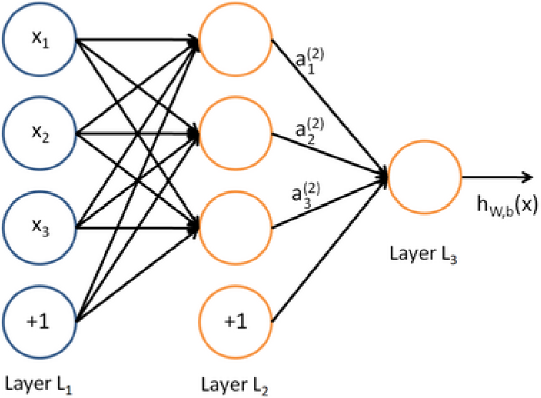
Supervised method (Baroni et al. 2012)

- In the supervised approach, the authors train Support Vector Machines (SVMs) on concatenation of vector representations, reduced to 300 each dimensions with SVD/LSA.
- Their SVMs have polynomial kernels that captures feature interactions (p. 29).
- This method is successful for both the nouns task and the quantifiers task (Tables 3, 4).
- In the ‘quantifier-out’ set-up, performance ranges from 34% accuracy (*either*) to 98% (*each*).
- In addition, they tried working with just quantifier vectors (no N complements) and judged the model unsuccessful (p. 30).

Summary, lessons, and prospects

- Defining entailment a priori in terms of vectors is challenging conceptually and empirically.
- Training supervised classifiers to learn entailment between vectors is more promising.
- We'll now move to more powerful models that might do even better at this and other semantic tasks.
- (Once we figure out entailment, we should worry about contradiction.)

Shallow neural nets



L_1 = representation of the data

L_2 to $L_3 \approx$ classifier using a hidden representation L_2

L_3 = Output signal/prediction

Linear models and discriminative training

1. Feature representations: $\phi(x, y) \in \mathbf{R}^d$
2. Scoring: $\text{Score}_{\mathbf{w}}(x, y) = \mathbf{w} \cdot \phi(x, y) = \sum_{j=1}^{d} w_j \phi(x, y)_j$
3. Objective function:

$$\min_{\mathbf{w} \in \mathbf{R}^d} \sum_{(x,y) \in \mathcal{D}} \max_{y' \in \mathcal{Y}} [\text{Score}_{\mathbf{w}}(x, y') + c(y, y')] - \text{Score}_{\mathbf{w}}(x, y)$$

   where $\mathcal{D}$ is a set of (x, y) training examples and $c(y, y')$ is the cost for predicting y' when the correct output is y.

4. Optimization:

   $\textsc{StochasticGradientDescent}(\mathcal{D}, T, \eta)$

   Initialize $\mathbf{w} \leftarrow \mathbf{0}$
   Repeat T times
       for each $(x, y) \in \mathcal{D}$ (in random order)
           $\tilde{y} \leftarrow \arg\max_{y' \in \mathcal{Y}} \text{Score}_{\mathbf{w}}(x, y') + c(y, y')$
           $\mathbf{w} \leftarrow \mathbf{w} + \eta(\phi(x, y) - \phi(x, \tilde{y}))$
   Return $\mathbf{w}$

Simple supervised learning example

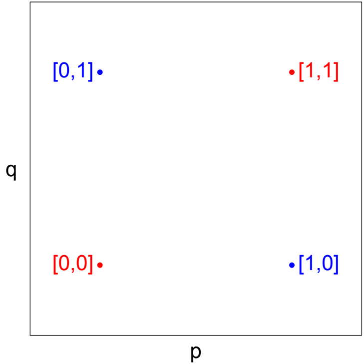
		Feature representations $\phi(x, y)$		
	(x, y)	'empty string'	'last word'	'all words'
Train	(twenty five, 0)	ϵ	five	[twenty, five]
	(thirty one, 0)	ϵ	eight	[thirty, one]
	(forty nine, 0)	ϵ	nine	[forty, nine]
	(fifty two, E)	ϵ	two	[fifty, two]
	(eighty two, E)	ϵ	two	[eighty, two]
	(eighty four, E)	ϵ	four	[eighty, four]
	(eighty six, E)	ϵ	six	[eighty, six]
Test	(eighty five, 0)	$\epsilon \rightarrow \mathtt{E}$	five $\rightarrow$ 0	[eighty, five] $\rightarrow \mathtt{E}$

Table: Tradeoffs in machine learning.

XOR and related examples (Rumelhart et al. 1986a,b)

p	q	$(p \mathbin{\bar{\vee}} q)$
1	1	0
1	0	1
0	1	1
0	0	0

Table: Exclusive ‘or’ (XOR)

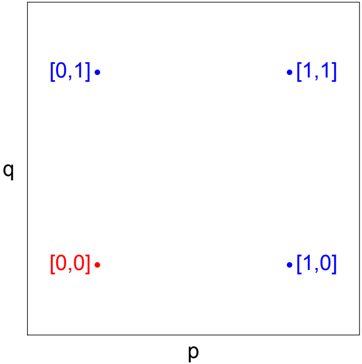


No linear separation into the two desired classes.

XOR and related examples (Rumelhart et al. 1986a,b)

p	q	$(p \vee q)$
1	1	1
1	0	1
0	1	1
0	0	0

Table: Inclusive 'or'

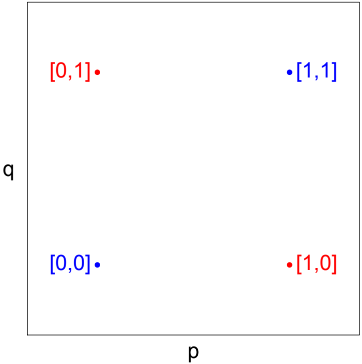


Easy linear separation into the two desired classes.

XOR and related examples (Rumelhart et al. 1986a,b)

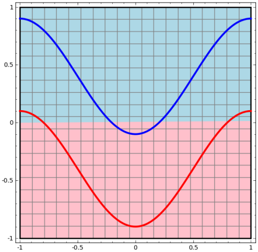
p	q	$(p \leftrightarrow q)$
1	1	1
1	0	0
0	1	0
0	0	1

Table: Biconditional (IFF)

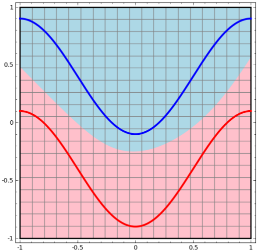


No linear separation into the two desired classes.

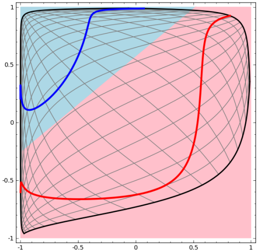
A glimpse of hidden representations



Linear classifier



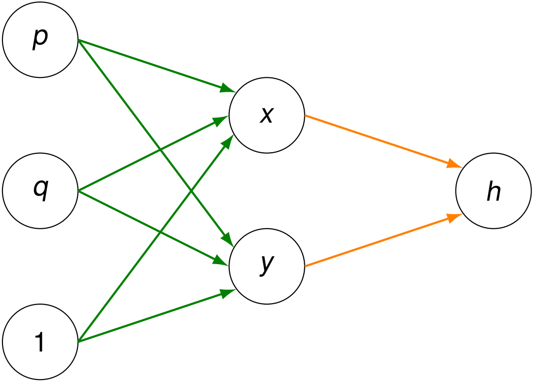
Shallow network



Hidden reps

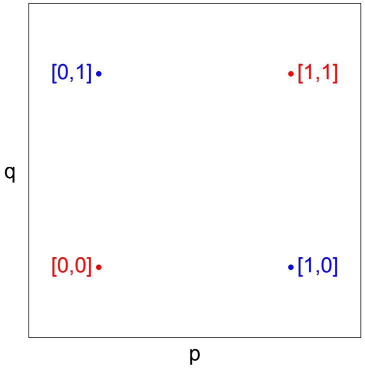
From http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/

A shallow XOR network with forward propagation



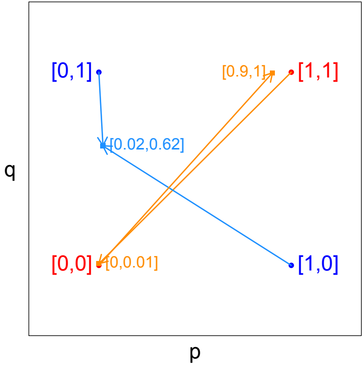
$$f\left([p, q, 1]\begin{bmatrix} p_1 & p_2 \\ q_1 & q_2 \\ b_1 & b_2 \end{bmatrix}\right) = [x, y] \qquad f\left([x, y]\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}\right) = h \qquad \boxed{f(x) = \frac{1}{1+e^{-x}}}$$

Hidden XOR representations



Hidden XOR representations

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



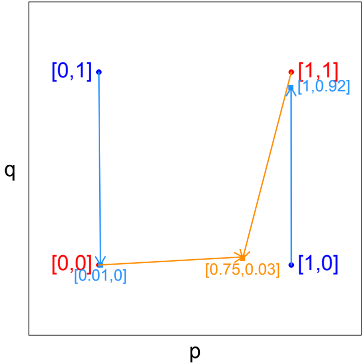
$$f\left([p, q, 1]\begin{bmatrix} -6.09 & -5.22 \\ -6.05 & -5.22 \\ 2.22 & 5.71 \end{bmatrix}\right)$$

Example:

$$f\left([0, 1, 1]\begin{bmatrix} -6.09 & -5.22 \\ -6.05 & -5.22 \\ 2.22 & 5.71 \end{bmatrix}\right) = [0.02, 0.62]$$

Hidden XOR representations

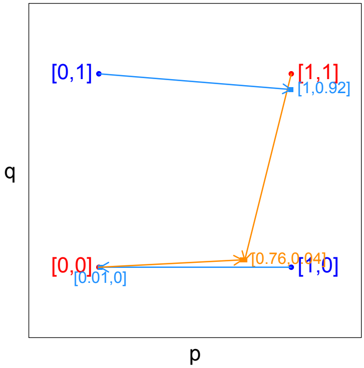
$$f(x) = \frac{1}{1 + \mathrm{e}^{-x}}$$



$$f\left([p, q, 1]\begin{bmatrix} 5.90 & 5.57 \\ -5.90 & -5.81 \\ 1.09 & -3.13 \end{bmatrix}\right)$$

Hidden XOR representations

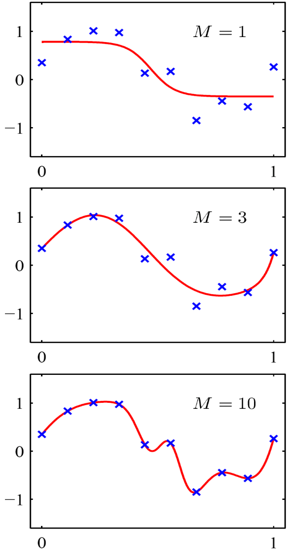
$$f(x) = \frac{1}{1 + \mathrm{e}^{-x}}$$



$$f\left([p, q, 1]\begin{bmatrix} -5.97 & -5.69 \\ 6.04 & 5.65 \\ 1.07 & -3.23 \end{bmatrix}\right)$$

The role of the non-linear activation function

- The activation function bends the representation dimensions around to help satisfy the objective function.
- The more dimensions in the representation, the more complex the functions we can approximate.
- Networks without non-linear activation functions are coherent, but they just perform lots of linear transformations between dimensions and so can be reduced to a single layer model.



Socher and Manning 2013:31

Learning with backpropagation

Same framework for feature representation and scoring as in the classifier model presented earlier ▸ link to the slide. The only changes concern propagating the error signal through the hidden layer:

$\textsc{BackwardPropagationViaStochasticDescent}(\mathcal{D}, T, \eta)$

Initialize input weights $W^{i \times h}$ with small, normally distributed values
Initialize output weights $H^{h \times 1}$ with small, normally distributed values
Repeat T times
  for each $(x, y) \in \mathcal{D}$ (in random order)
    $a \leftarrow f(x \cdot W)$ `# forward prop input to hidden`
    $z \leftarrow f(a \cdot H)$ `# forward prop hidden to output`
    $\delta_2 \leftarrow (y - z) \cdot f'(z)$ `# output errors`
    $\delta_1 \leftarrow \delta_2 \cdot H^T \cdot f'(a)$ `# hidden errors`
    $H \leftarrow \eta \cdot a^T \cdot \delta_2$ `# hidden weights update`
    $W \leftarrow \eta \cdot x^T \cdot \delta_1$ `# input weights update`
Return W, H

Application to sentiment

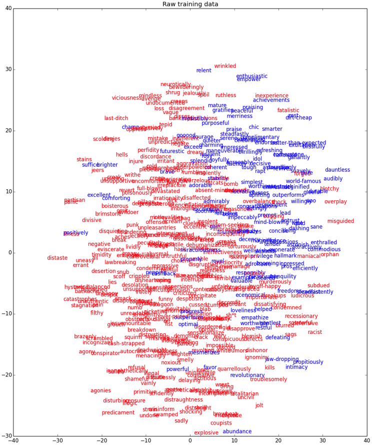
Word	Class
good	$+1$
excellent	$+1$
superior	$+1$
correct	$+1$
bad	-1
poor	-1
unfortunate	-1
wrong	-1

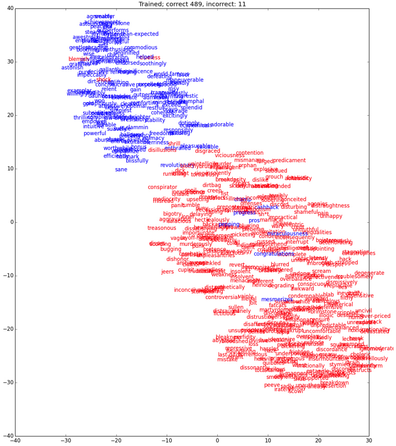
Word	against	age	agent	ages	ago	agree
good	-0.19	-0.07	-0.12	-0.07	0.03	0.08
excellent	-0.14	0.01	-0.10	0.41	0.17	-0.01
superior	0.32	-0.39	-0.18	0.24	-0.41	0.14
correct	-0.09	-0.21	0.16	0.58	0.70	0.08
bad	-0.26	-0.54	-0.03	-0.48	-0.02	-0.01
poor	-0.02	-0.31	0.02	-0.06	-0.26	0.01
unfortunate	0.39	-0.06	0.04	-0.96	-0.09	0.26
wrong	-0.11	-0.20	-0.01	-0.18	-0.05	0.16

Code for these experiments: `http://www.stanford.edu/class/cs224u/code/shallow_neuralnet_with_backprop.py` and the Python t-SNE implementation `http://homepage.tudelft.nl/19j49/t-SNE.html`

Application to sentiment

Input (left): 200*d* PMI reps. Output (right): 100*d* hidden reps.

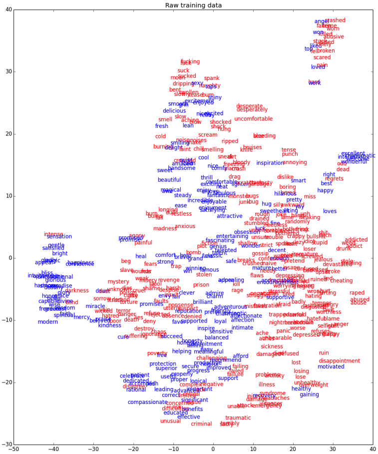


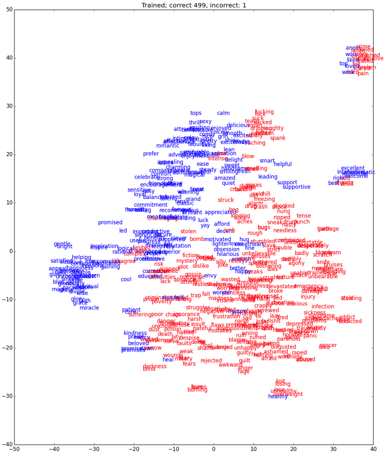


All visualizations with t-SNE (van der Maaten and Geoffrey 2008)

Application to sentiment

Input (left): 100*d* PMI+LSA reps. Output (right): 100*d* hidden reps.

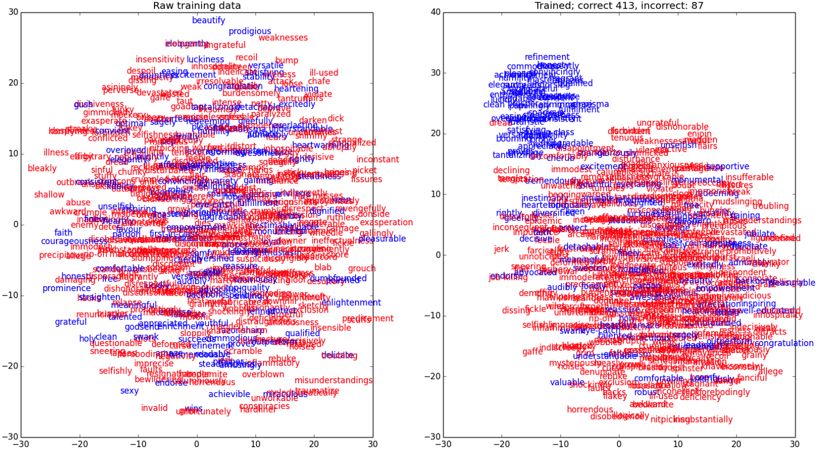




All visualizations with t-SNE (van der Maaten and Geoffrey 2008)

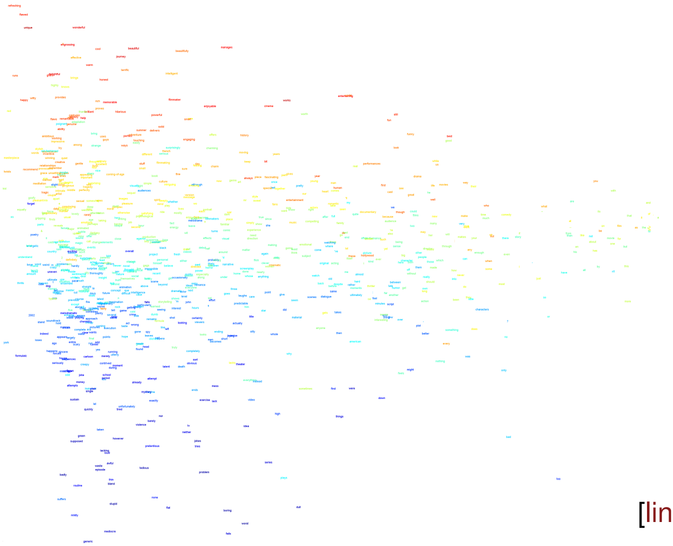
Application to sentiment

Input (left): random 100d reps. Output (right): 100d hidden reps.



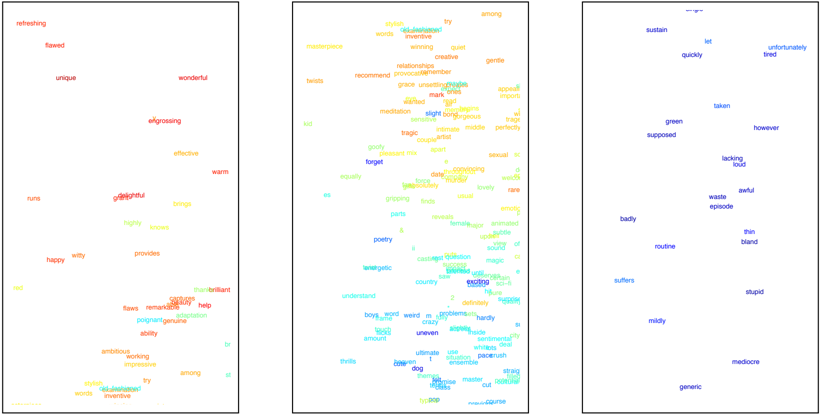
All visualizations with t-SNE (van der Maaten and Geoffrey 2008)

Semi-supervised auto-encoders (Socher et al. 2011b)



[link]

Semi-supervised auto-encoders (Socher et al. 2011b)



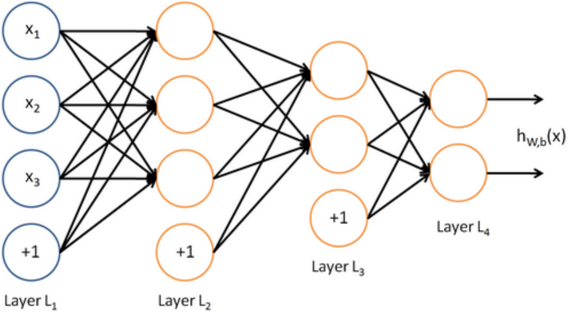
negative middle positive

Lexical entailment (Bowman 2014)

1. Learns not only entailment pairs like *puppy* $\Rightarrow$ *animal* but also contradiction pairs like *dog* | *bird*.
2. (The set of relations is even richer; MacCartney 2009.)
3. Recursive neural tensor network (Socher et al. 2013b).
4. Hold-one-out evaluation: train on the entire lexical network except for a pair of words (x, y), and then predict the relation between x and y.
5. "The results are modestly promising. Of a sample of 69 test examples [. . .] 61 (88.4%) were labeled correctly"
6. Optimization with AdaGrad (Duchi et al. 2011)
7. Rectified linear activation function (Maas et al. 2013): $f(x) = \max(x, 0) + 0.01 \min(x, 0)$
8. Full code release: link
9. More on this model later in the term!

Some extensions and modifications

Deeper and higher dimensional networks:



http://deeplearning.stanford.edu/wiki/index.php/Neural_Networks

Some extensions and modifications

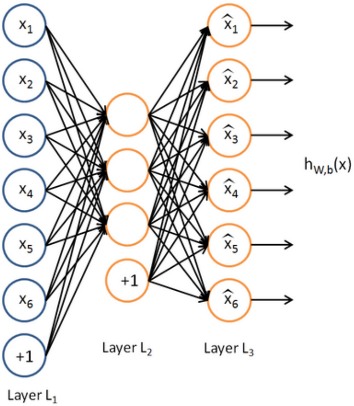
Different activation functions; some examples:

Name	Function	Derivative
sigmoid	$f(x) = \frac{1}{1+\mathrm{e}^{-x}}$	$f(x) \cdot (1 - f(x))$
softmax	$\frac{\mathrm{e}^{x_j}}{\sum_{k=1}^{n} \mathrm{e}^{x_k}}$	$f(x_j) \cdot (1 - f(x_j))$
tanh	$f(x) = \frac{\mathrm{e}^{x}-\mathrm{e}^{-x}}{\mathrm{e}^{x}+\mathrm{e}^{-x}}$	$1 - f(x)^2$
softplus	$f(x) = \log(1 + e^{x})$	$\frac{1}{1+e^{-x}}$

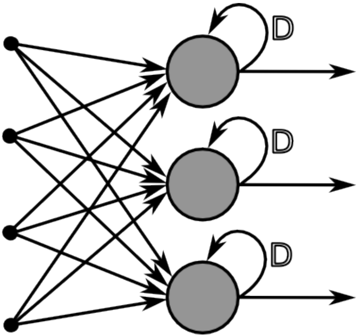
The choice of activation function affects the freedom one has for the output variables and the nature of the error function.

Some extensions and modifications

Radically different network structures:



Autoencoder [link]



Recurrent [link]

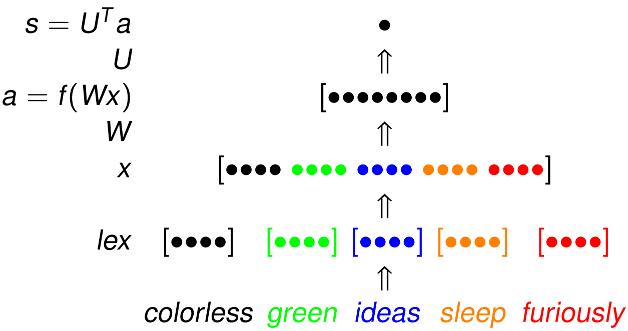
Lexical ambiguity

Ambiguity is *everywhere* in language and is the source of most linguistic humor (e.g., the funniest joke in the world):

1. *crane* and *crane*
2. *pitch* and *pitch*
3. *try* and *try*
4. *sanction* (permit) and *sanction* (penalize)
5. *flat (tire)*, *flat (note)*, *flat (beer)*, *flat (note)*
6. *throw (a party)*, *throw (a stone)*, *throw (a fight)*
7. *into (the tunnel)* and *into (jazz)*
8. *still*
9. *mean*
10. . . .

VSMs might seem constitutionally unable to model ambiguity because of the way they are constructed.

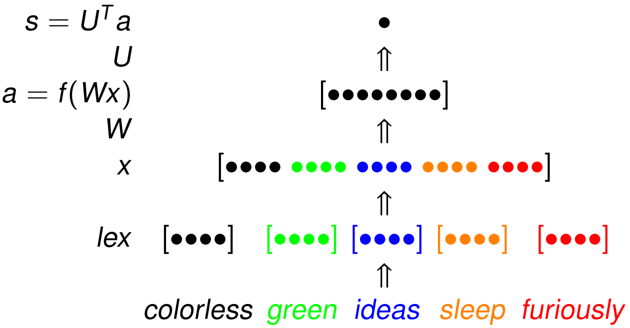
Scores without supervision



(Collobert and Weston 2008; Turian et al. 2010)

Scores without supervision

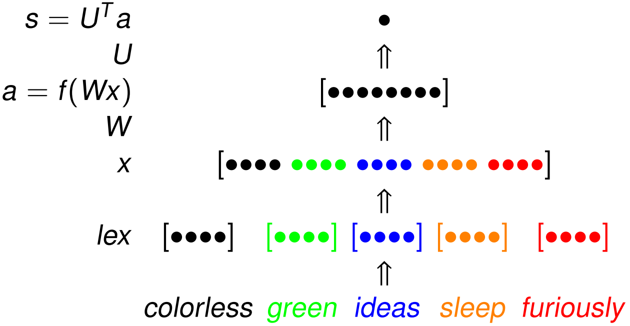
1. $s = score(colorless\ green\ ideas\ sleep\ \underline{furiously})$



(Collobert and Weston 2008; Turian et al. 2010)

Scores without supervision

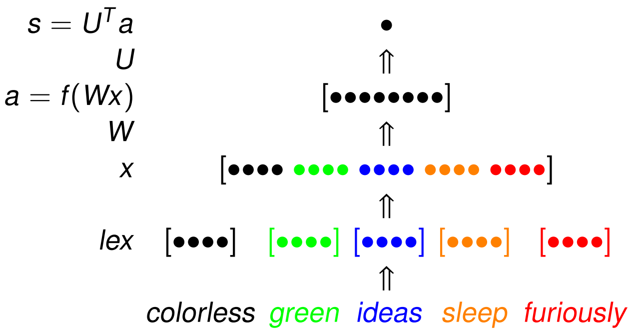
1. $s = score(colorless\ green\ ideas\ sleep\ \underline{furiously})$
2. $s_c = score(colorless\ green\ ideas\ sleep\ \boxed{might})$



(Collobert and Weston 2008; Turian et al. 2010)

Scores without supervision

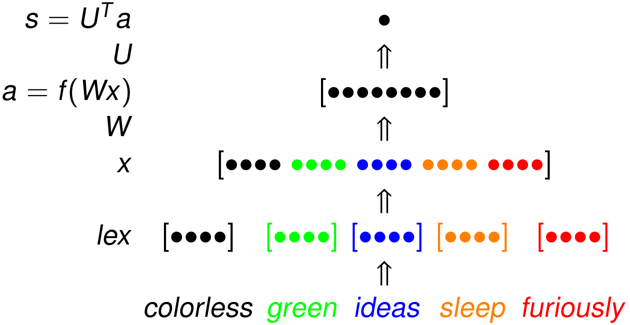
1. $s = score(colorless\ green\ ideas\ sleep\ \underline{furiously})$
2. $s_c = score(colorless\ green\ ideas\ sleep\ \boxed{might})$
3. Objective: minimize $\sum_{w \in \mathcal{D}} \frac{1}{|\mathcal{D}|} \max(0, 1 - s_w + s_c)$
   (seek to make s_w at least $+1$ of s_c)



(Collobert and Weston 2008; Turian et al. 2010)

Scores without supervision

1. $s = score(colorless\ green\ ideas\ sleep\ \underline{furiously})$
2. $s_c = score(colorless\ green\ ideas\ sleep\ \boxed{might})$
3. Objective: minimize $\sum_{w \in \mathcal{D}} \frac{1}{|\mathcal{D}|} \max(0, 1 - s_w + s_c)$
   (seek to make s_w at least $+1$ of s_c)
4. Backpropagation down to the lexical vectors *lex*



(Collobert and Weston 2008; Turian et al. 2010)

Huang et al. (2012)

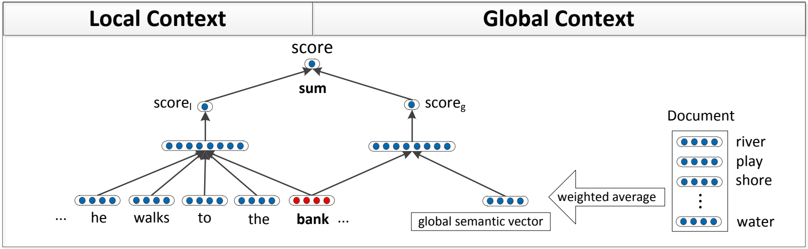


Figure 1: An overview of our neural language model. The model makes use of both local and global context to compute a score that should be large for the actual next word (*bank* in the example), compared to the score for other words. When word meaning is still ambiguous given local context, information in global context can help disambiguation.

Sense disambiguation via clustering

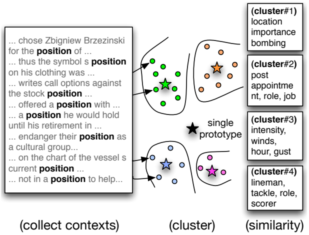


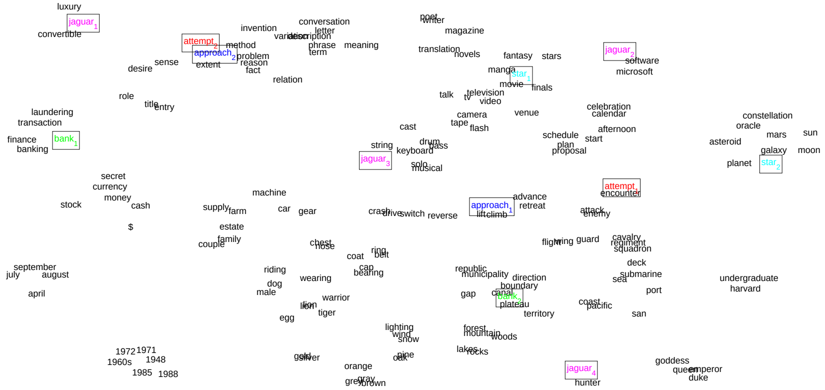
Figure 1: Overview of the multi-prototype approach to near-synonym discovery for a single target word independent of context. Occurrences are clustered and cluster centroids are used as prototype vectors. Note the "hurricane" sense of *position* (cluster 3) is not typically considered appropriate in WSD.

Reisinger and Mooney 2010b

- Cluster the contexts for each word using a standard centroid algorithm.
- Label each token with its cluster's index.
- Construct word representations for this new vocabulary.

See also Schütze 1998; Pantel 2003; Reisinger and Mooney 2010a

Huang et al. (2012) word embeddings



From the paper's website

Word meanings in context

Word 1	Word 2
Located downtown along the east **bank** of the Des Moines River ...	This is the basis of all **money** laundering , a track record of depositing clean money before slipping through dirty money ...
Inside the ruins , there are **bats** and a bowl with Pokeys that fills with sand over the course of the race , and the music changes somewhat while inside ...	An aggressive lower order batsman who usually **bats** at No. 11 , Muralitharan is known for his tendency to back away to leg and slog ...
An example of legacy **left** in the Mideast from these nobles is the Krak des Chevaliers ' enlargement by the Counts of Tripoli and Toulouse ...	... one should not adhere to a particular explanation , only in such measure as to be ready to **abandon** it if it be proved with certainty to be false ...
... and Andy 's getting ready to **pack** his bags and head up to Los Angeles tomorrow to get ready to fly back home on Thursday	... she encounters Ben (Duane Jones) , who arrives in a pickup truck and defends the house against another **pack** of zombies ...
In **practice** , there is an unknown phase delay between the transmitter and receiver that must be compensated by " synchronization " of the receivers local oscillator	... but Gilbert did not believe that she was dedicated enough , and when she missed a **rehearsal** , she was dismissed ...

Table 4: Example pairs from our new dataset. Note that words in a pair can be the same word and have different parts of speech.

(Huang et al. 2012; the data set)

Code and tools

- PyBrain: `http://pybrain.org`
- Google vectors package word2vec: `https://code.google.com/p/word2vec/`
- word2vec reimplemented in Python/Gensim: `http://radimrehurek.com/2013/09/deep-learning-with-word2vec-and-gensim/`
- Richard Socher has released code with almost all his recent papers: `http://www.socher.org`
- Deeply Moving: Deep Learning for Sentiment Analysis `http://nlp.stanford.edu/sentiment/`
- A beautiful t-SNE visualization of Collobert and Weston's (2008) representations: `https://www.cs.toronto.edu/~hinton/turian.png`

Looking ahead

How are distributional vector models doing on our core goals?

1. Word meanings $\approx$
2. Connotations $\checkmark$
3. Compositionality (May 14)
4. Syntactic ambiguities
5. Semantic ambiguities (progress!)
6. Entailment and monotonicity (progress!)
7. Question answering

References I

Baroni, Marco; Raffaella Bernardi; Ngoc-Quynh Do; and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 23–32. Avignon, France: ACL.

Bowman, Samuel R. 2014. Can recursive neural tensor networks learn logical reasoning? In *Proceedings of the International Conference on Learning Representations*.

Clarke, Daoud. 2009. Context-theoretic semantics for natural language: An overview. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, 112–119. Athens, Greece: ACL.

Collobert, Ronan and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine learning*, ICML '08, 160–167. New York: ACM. doi:\bibinfo{doi}{http://doi.acm.org/10.1145/1390156.1390177}. URL http://doi.acm.org/10.1145/1390156.1390177.

Collobert, Ronan; Jason Weston; Léon Bottou; Michael Karlen; Koray Kavukcuoglu; and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12:2493–2537.

Deng, Li and Dong Yu. 2014. *Deep Learning: Methods and Applications*. Now Publishers.

Duchi, John; Elad Hazan; and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 2121–2159.

Huang, Eric; Richard Socher; Christopher D. Manning; and Andrew Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 873–882. Jeju Island, Korea: ACL. URL http://www.aclweb.org/anthology/P12-1092.

Kotlerman, Lili; Ido Dagan; Idan Szpektor; and Maayan Zhitomirsky-geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering* 16(4):359–389. doi:\bibinfo{doi}{10.1017/S1351324910000124}.

Lewis, Mike and Mark Steedman. 2013. Combined distributional and logical semantics. *Transactions of the Association for Computational Linguistics* 1:179–192.

Lin, Dekang. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL*, 768–774. Montreal: ACl.

Luong, Minh-Thang; Richard Socher; and Christopher D. Manning. 2013. Better word representations with recursive neural networks for morphology. In *CoNNL*.

References II

Maas, Andrew L.; Awni Y. Hannun; and Andrew Y. Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the 30th International Conference on Machine Learning*.

van der Maaten, Laurens and Hinton Geoffrey. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9:2579–2605.

MacCartney, Bill. 2009. *Natural Language Inference*. Ph.D. thesis, Stanford University.

Mikolov, Tomas; Wen-tau Yih; and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 746–751. Stroudsburg, PA: ACL. URL http://www.aclweb.org/anthology/N13-1090.

Pantel, Patrick. 2003. *Clustering by Committee*. Ph.D. thesis, University of Edmonton, Edmonton, Alberta.

Reisinger, Joseph and Raymond Mooney. 2010a. A mixture model with sharing for lexical semantics. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 1173–1182. Cambridge, MA: Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D10-1114.

Reisinger, Joseph and Raymond J. Mooney. 2010b. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 109–117. Los Angeles, California: Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N10-1013.

Rumelhart, David E.; Geoffrey E. Hinton; and Ronald J. Williams. 1986a. Learning internal representations by error propagation. In David E. Rumelhart and James L. McClelland, eds., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1: Foundations, 318–362. Cambridge, MA: MIT Press.

Rumelhart, David E.; Geoffrey E. Hinton; and Ronald J. Williams. 1986b. Learning representations by back-propagating errors. *Nature* 323(6088):533–536. doi:\bibinfo{doi}{doi:10.1038/323533a0}.

Schütze, Hinrich. 1998. Automatic word sense discrimination. *Computational Linguistics* 24(1):97–123.

Socher, Richard; John Bauer; Christopher D. Manning; and Ng Andrew Y. 2013a. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, volume 1: Long Papers, 455–465. Stroudsburg, PA: ACL.

Socher, Richard; Yoshua Bengio; and Christopher D. Manning. 2012a. Deep learning for NLP (without magic). Tutorial at ACL 2012, Jeju Island, Korea., URL http://www.socher.org/index.php/DeepLearningTutorial/DeepLearningTutorial.

References III

Socher, Richard; Eric H. Huang; Jeffrey Pennin; Christopher D Manning; and Andrew Y. Ng. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In John Shawe-Taylor; Richard S. Zemel; Peter L. Bartlett; Fernando Pereira; and Kilian Q. Weinberger, eds., *Advances in Neural Information Processing Systems 24*, 801–809.

Socher, Richard; Brody Huval; Christopher D. Manning; and Andrew Y. Ng. 2012b. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing*, 1201–1211. Stroudsburg, PA.

Socher, Richard and Christopher D. Manning. 2013. Deep learning for NLP (without magic). In *NAACL HLT 2013 Tutorial Abstracts*, 1–3. Atlanta, GA: Association for Computational Linguistics. Tutorial at NAACL 2013, Atlanta, Georgia, URL http://nlp.stanford.edu/courses/NAACL2013/.

Socher, Richard; Jeffrey Pennington; Eric H. Huang; Andrew Y. Ng; and Christopher D. Manning. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 151–161. Edinburgh, Scotland, UK.: ACL.

Socher, Richard; Alex Perelygin; Jean Wu; Jason Chuang; Christopher D. Manning; Andrew Y. Ng; and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1631–1642. Stroudsburg, PA: Association for Computational Linguistics.

Turian, Joseph; Lev-Arie Ratinov; and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 384–394. Uppsala, Sweden: ACL.

Weeds, Julie and David Weir. 2003. A general framework for distributional similarity. In Michael Collins and Mark Steedman, eds., *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, 81–88.