COMP 786 (Fall 2020) Natural Language Processing

Lecture 2 (Aug 19): LMs, Embeddings, Classification

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Mohit Bansal

(various slides adapted/borrowed from courses by Dan Klein, Richard Socher, Chris Manning, others)

Language Modeling

- A language model is a distribution over sequences of words (sentences)

$$P(\mathbf{w}) = P(w_{1} \dots w_{n})$$

- Purpose is to usually assign high weights to plausible sentences, e.g., in speech recognition or machine translation

- Also used for language generation now (predict next word given previous words), esp. w/ new RNN models

Traditional N-gram LMs

- Use chain rule to generate words left-to-right

$$P(w_1 \dots w_n) = \prod_i P(w_i | w_1 \dots w_{i-1})$$

- Can’t condition on the entire left context

P(??? | Turn to page 134 and look at the picture of the)

- N-gram models make a Markov assumption

$$P(w_1 \dots w_n) = \prod_i P(w_i | w_{i-k} \dots w_{i-1})$$

$$P(\text{please close the door}) =$$

$$P(\text{please}|\text{START})P(\text{close}|\text{please}) \dots P(\text{STOP}|door)$$

Traditional N-gram LMs

- How do we know P(w | history)?
  - Use statistics from data (examples using Google N-Grams)
  - E.g. what is P(door | the)?

Training Counts

```
198015222 the first
194623024 the same
168504105 the following
158562063 the world

14112454 the door
-----------------
23135851162 the *
```

$$\hat{P}(\text{door}|\text{the}) = \frac{14112454}{23135851162}$$

$$= 0.0006$$

- This is the *maximum likelihood* estimate

https://netspeak.org/#q=*+the+door&corpus=web-en

Sparsity Issue & Parameter Estimation

- New words all the time (*antidisestablishmentarianism, kakorrhaphiophobia,, www.xyzabc156.com)*….worse for new bigrams and trigrams!
  - Maximum likelihood estimates won't get us very far

$$\hat{P}(w|w_{-1}) = \frac{c(w_{-1}, w)}{\sum_{w'} c(w_{-1}, w')}$$

```
3516 wipe off the excess
1034 wipe off the dust
547 wipe off the sweat
518 wipe off the mouthpiece
…
120 wipe off the grease
0 wipe off the sauce
0 wipe off the mice
-----------------
28048 wipe off the *
```

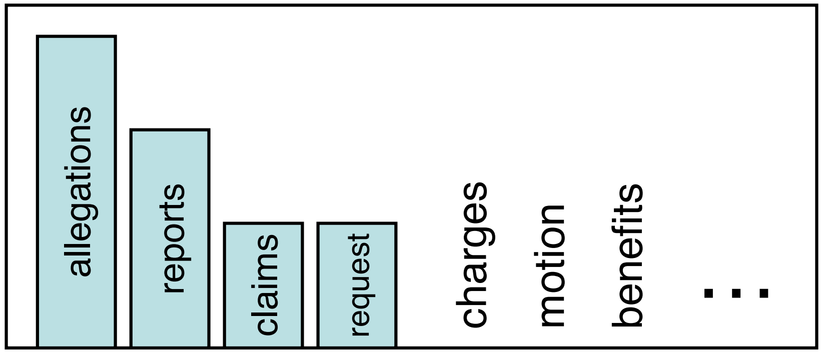
  - Need to *smooth* these estimates
  - General method (procedurally)
    - Take your empirical counts
    - Modify them in various ways to improve estimates
  - General method (mathematically)
    - Often can give estimators a formal statistical interpretation
    - … but not always
    - Approaches that are mathematically obvious aren't always what works

Smoothing Techniques

- We often want to make estimates from sparse statistics:

P(w | denied the)
3 allegations
2 reports
1 claims
1 request

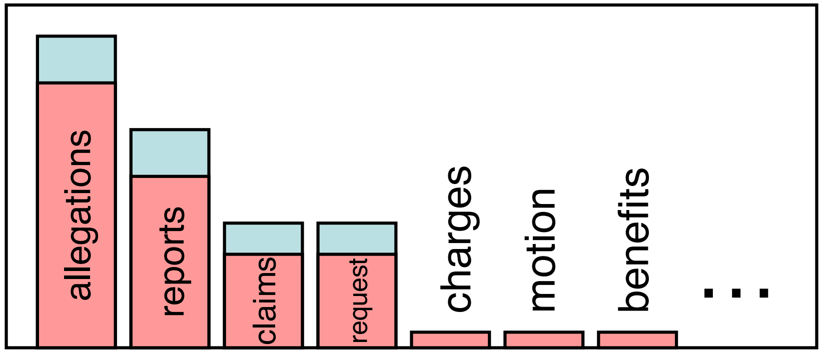
7 total



- Smoothing flattens spiky distributions so they generalize better

P(w | denied the)
2.5 allegations
1.5 reports
0.5 claims
0.5 request
2 other

7 total



- Very important all over NLP, but easy to do badly!

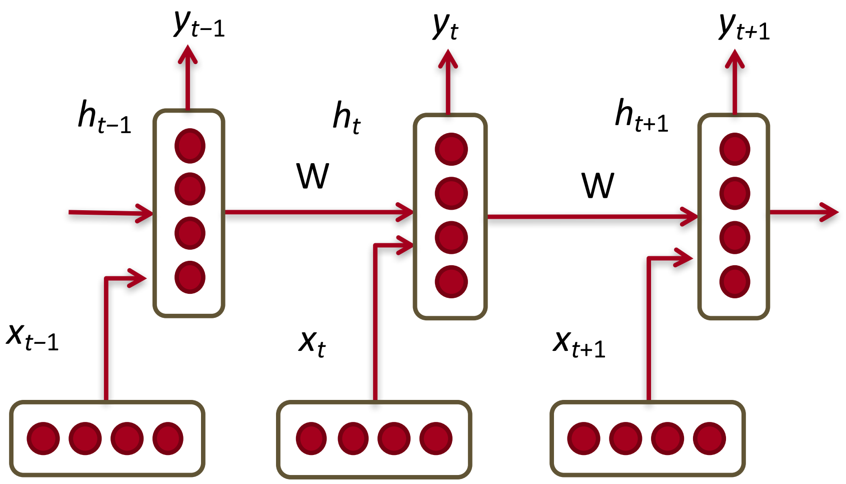
Smoothing Techniques

- Classic Solution: add-one or add small priors to numer/denom
- Backing off to smaller n-grams
- Held-out Reweighting: Important to optimize/estimate how models generalize! So use held-out data to estimate the map of old count to new count
- Kneser-Ney Discounting: two successful ideas:
  - Idea 1: observed n-grams occur more in training than they will later
  - Idea 2: Type-based fertility (based on how common the word type is)
- Read Chen and Goodman, 1996 for various details and graphs!

http://wordnetweb.princeton.edu/perl/webwn?s=bank

RNN Language Models

- Avoid huge number of n-grams; Memory requirement only scales with #words
- Can condition on all previous history (with forget gates)
- Loss function on identity of predicted word at each time step
- But harder/slower to train and reach optimum (and less interpretable)?



Distributional Semantics

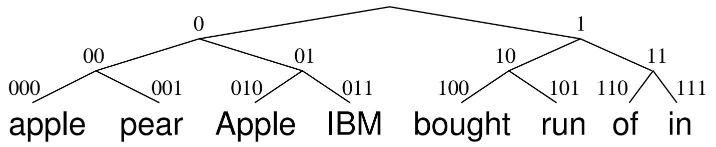
- Words occurring in similar context have similar linguistic behavior (meaning) [Harris, 1954; Firth, 1957]

$$\textit{food} \Rightarrow \begin{bmatrix} 0.6 \\ -0.2 \\ 0.9 \\ 0.3 \\ -0.4 \\ 0.5 \end{bmatrix}$$

- Traditional approach: context-counting vectors
  - Count left and right context in window
  - Reweight with PMI or LLR
  - Reduce dimensionality with SVD or NNMF

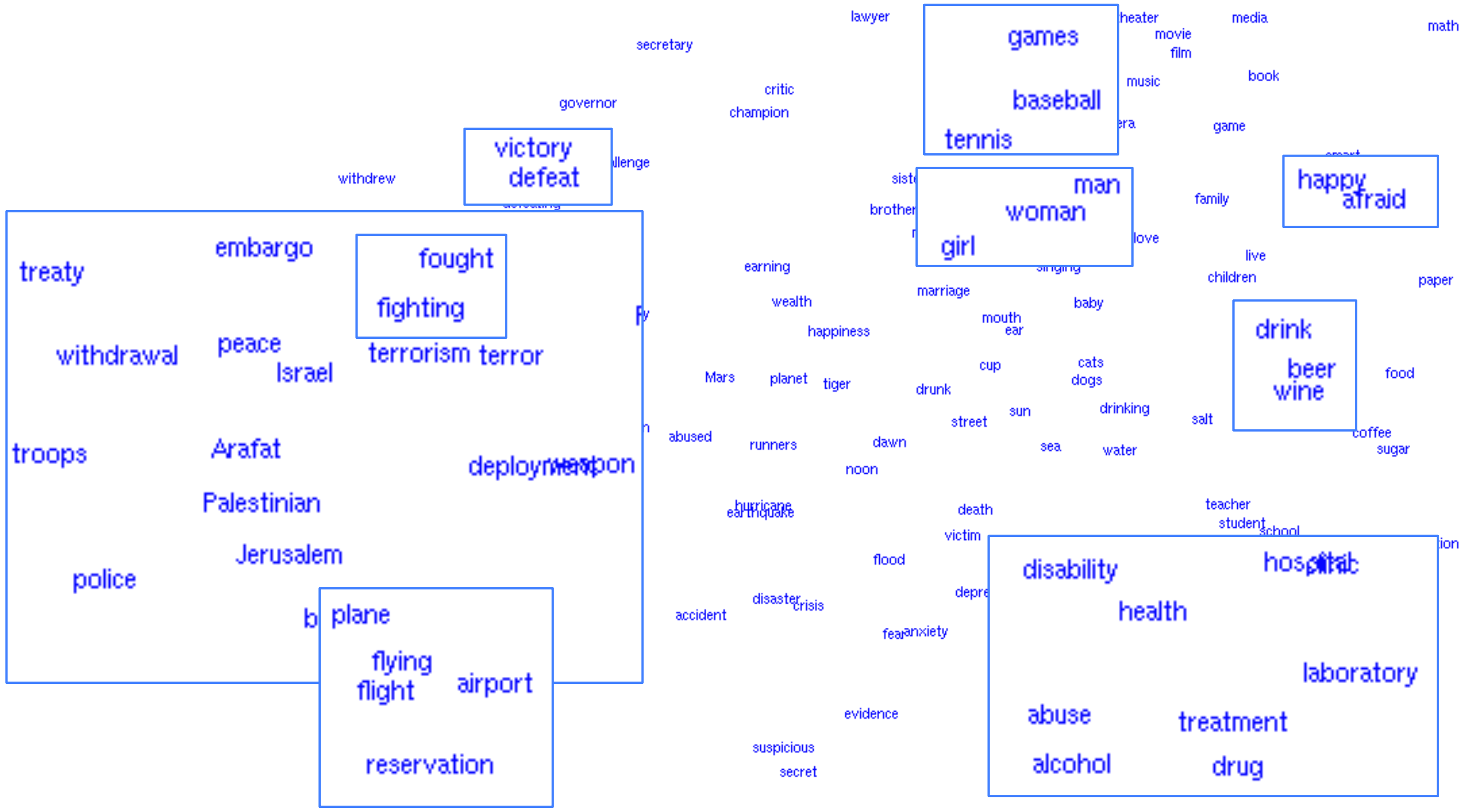
  [Pereira et al., 1993; Lund & Burgess, 1996; Lin, 1998; Lin and Pantel, 2001; Sahlgren, 2006; Pado & Lapata, 2007; Turney and Pantel, 2010; Baroni and Lenci, 2010]

- More word representations: hierarchical clustering based on bigram LM LL
  [Brown et al., 1992]



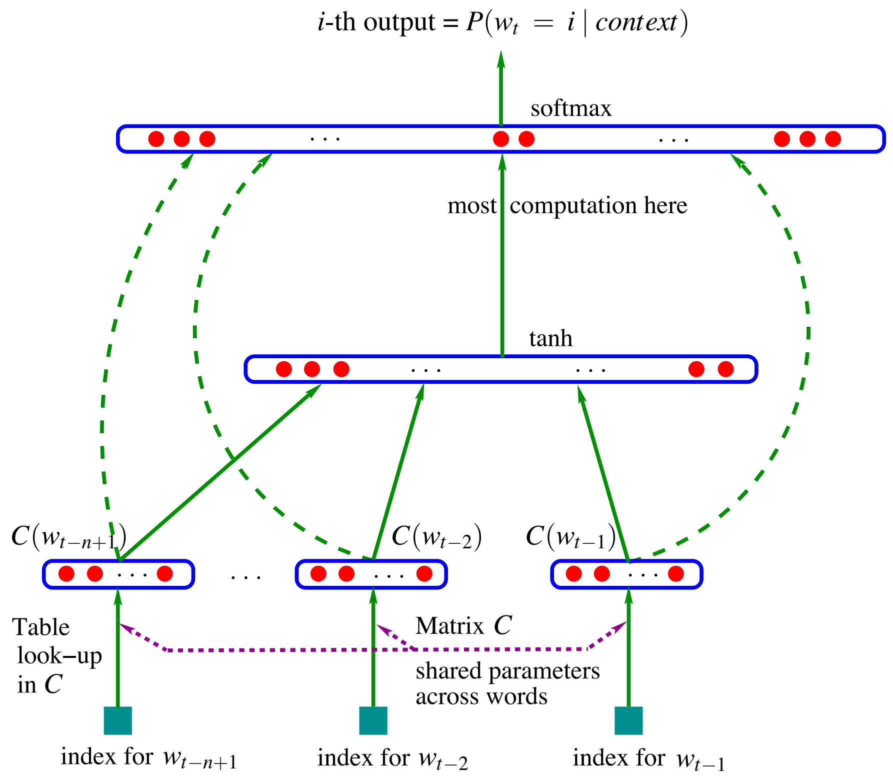
Unsupervised Embeddings

- Vector space representations learned on unlabeled linear context (i.e., left/right words): distributional semantics (Harris, 1954; Firth, 1957)



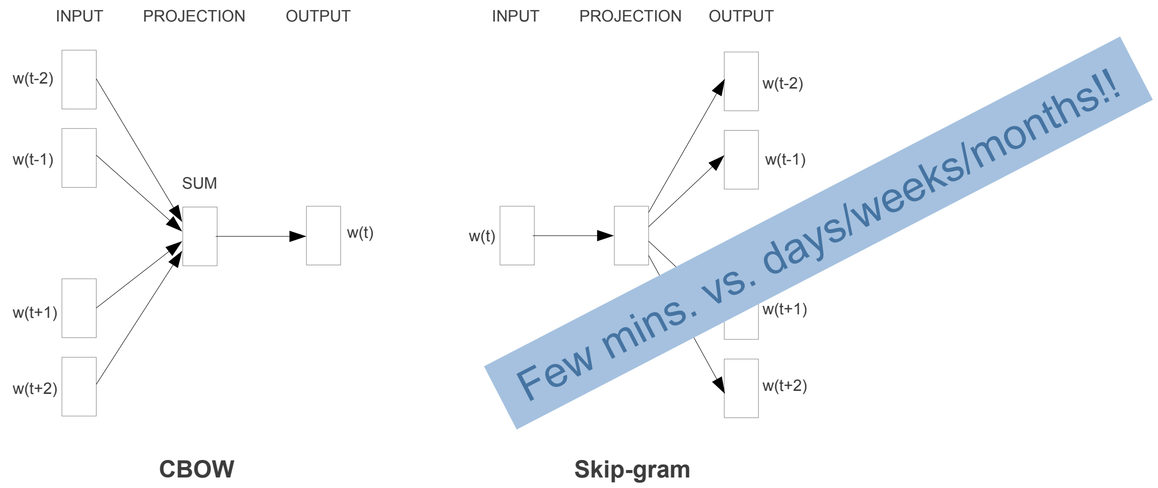
Distributional Semantics -- NNs

- Newer approach: context-predicting vectors (NNs)
  - SENNA [Collobert and Weston, 2008; Collobert et al., 2011]: Multi-layer DNN w/ ranking-loss objective; BoW and sentence-level feature layers, followed by std. NN layers. Similar to [Bengio et al., 2003].



Distributional Semantics -- NNs

- CBOW, SKIP, word2vec [Mikolov et al., 2013]: Simple, super-fast NN w/ no hidden layer. Continuous BoW model predicts word given context, skip-gram model predicts surrounding context words given current word

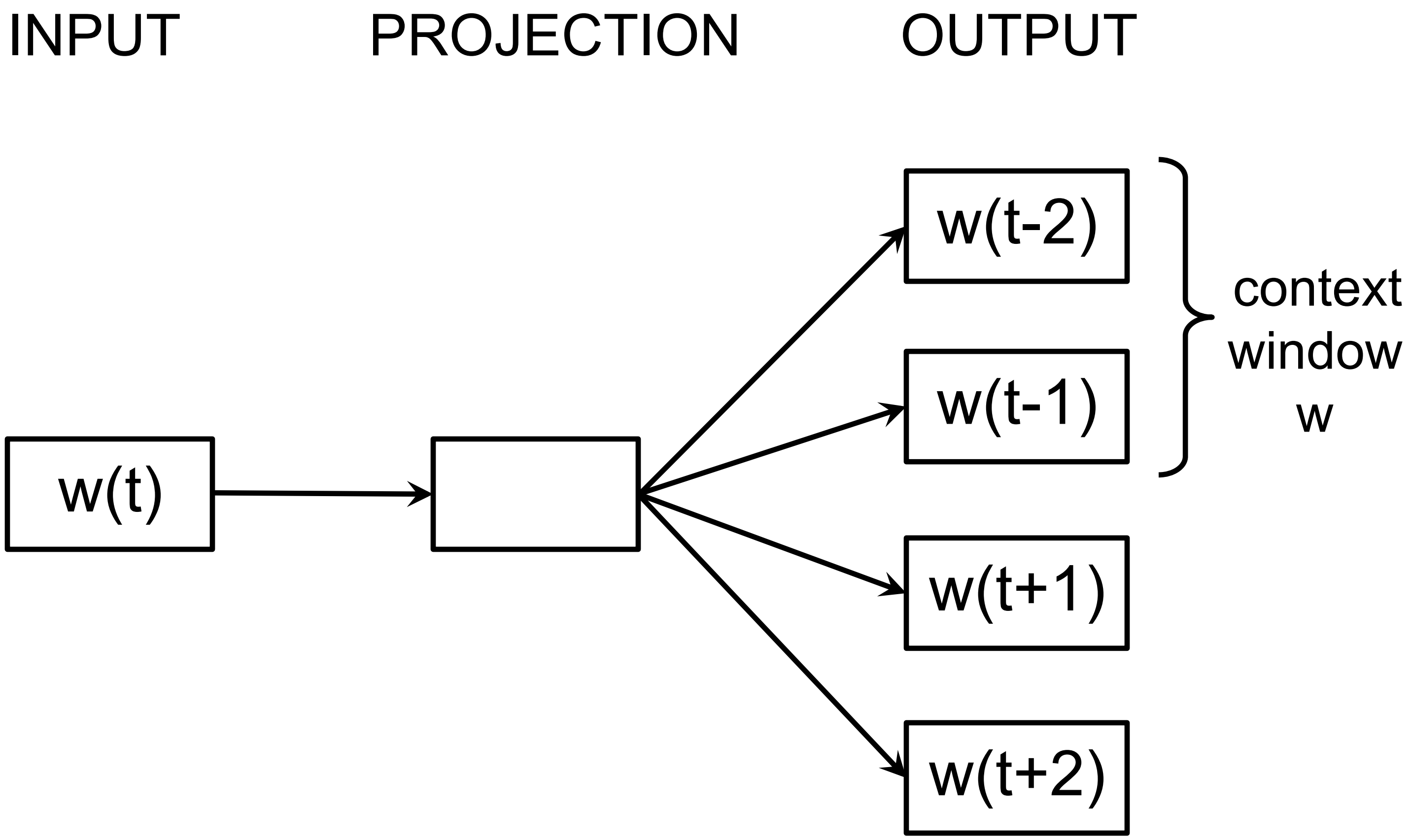


  - Other: [Mnih and Hinton, 2007; Turian et al., 2010]

- Demos: https://code.google.com/p/word2vec, http://metaoptimize.com/projects/wordreprs/, http://ml.nec-labs.com/senna/

Skipgram word2vec

[Mikolov et al., 2013]



Few mins. vs. days/weeks/months!!

Skip-gram word2vec Objective Function

[Mikolov et al., 2013]

- Objective of Skip-gram model is to max. the avg. log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

- The above conditional probability is defined via the softmax function:

$$p(w_O|w_I) = \frac{\exp\left({v'_{w_O}}^\top v_{w_I}\right)}{\sum_{w=1}^{W}\exp\left({v'_w}^\top v_{w_I}\right)}$$

where v and v′ are the “input” and “output” vector representations of w, and W is the number of words in the vocabulary

Efficient Skip-gram word2vec:

[Mikolov et al., 2013]

- Negative Sampling:

$$\log \sigma({v'_{w_O}}^\top v_{w_I}) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)} \left[\log \sigma(-{v'_{w_i}}^\top v_{w_I}) \right]$$

- I.e., to distinguish the target word w_o from draws from the noise distribution $P_n(w)$ using logistic regression, where there are k negative samples for each data sample.

Efficient Skip-gram word2vec:

[Mikolov et al., 2013]

- Hierarchical Softmax:

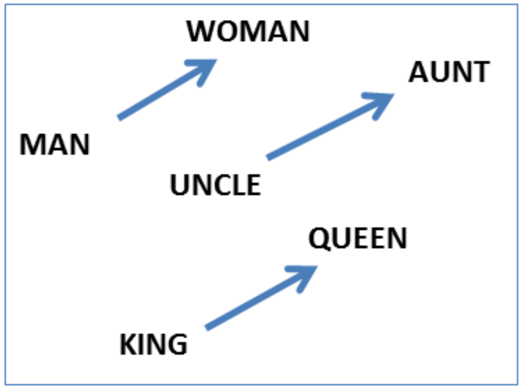
$$p(w|w_I) = \prod_{j=1}^{L(w)-1} \sigma\left([\![n(w,j+1) = \mathrm{ch}(n(w,j))]\!] \cdot {v'_{n(w,j)}}^\top v_{w_I} \right)$$

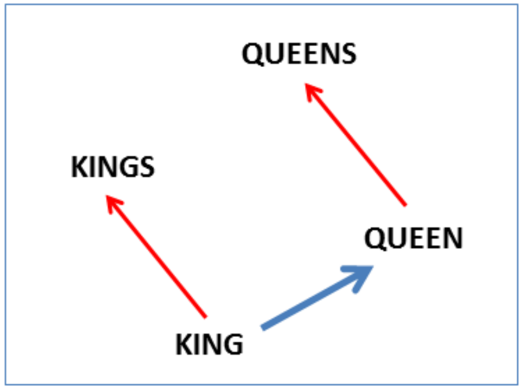
- Instead of evaluating W output nodes in the neural network to obtain the probability distribution, it is needed to evaluate only about $\log_2$(W) nodes.

- Uses a binary tree representation of the output layer with the W words as its leaves and, for each node, explicitly represents the relative probabilities of its child nodes. These define a random walk that assigns probabilities to words.

Analogy Properties Learned

[Mikolov et al., 2013]





Analogy Properties Learned

[Mikolov et al., 2013]

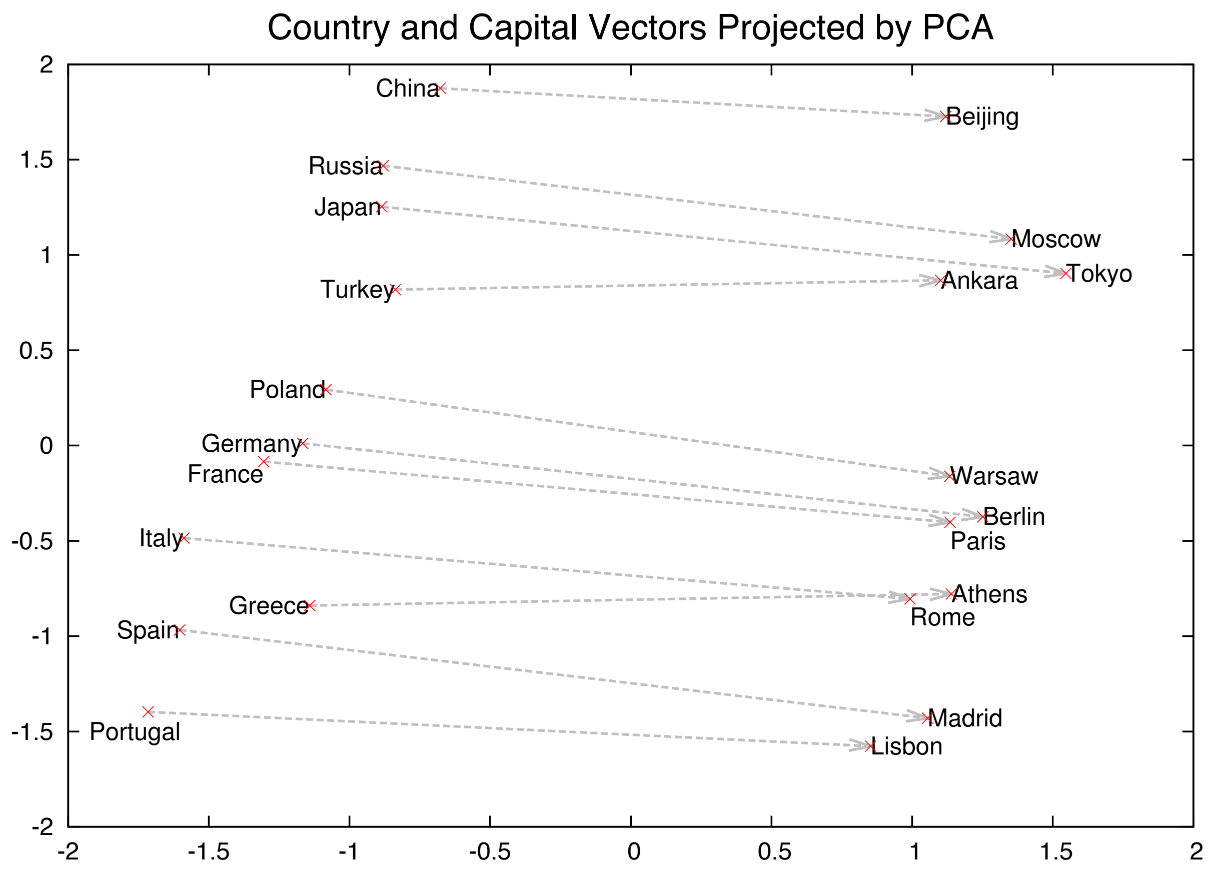


Figure 2: Two-dimensional PCA projection of the 1000-dimensional Skip-gram vectors of countries and their capital cities. The figure illustrates ability of the model to automatically organize concepts and learn implicitly the relationships between them, as during the training we did not provide any supervised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

Analogy Properties Learned

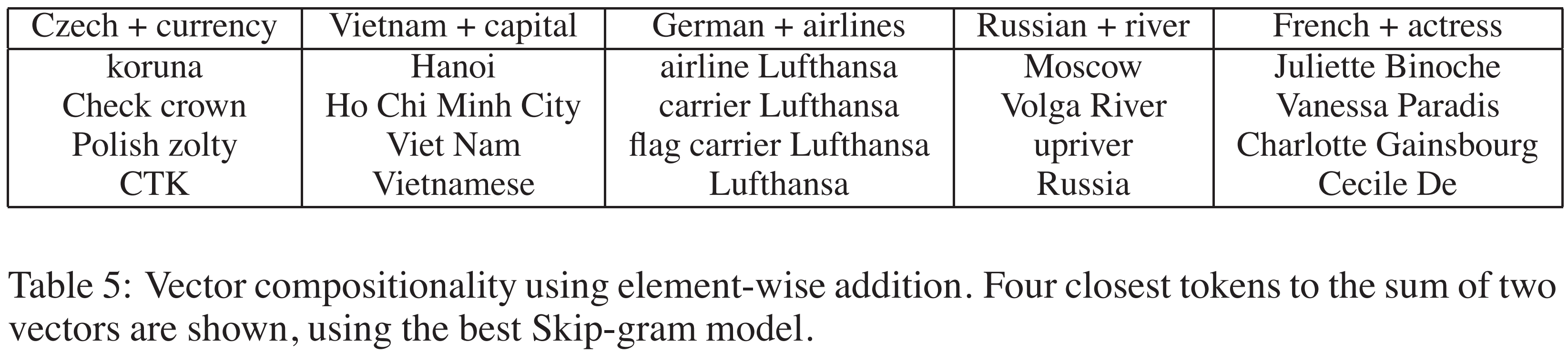
[Mikolov et al., 2013]

Newspapers			
New York	New York Times	Baltimore	Baltimore Sun
San Jose	San Jose Mercury News	Cincinnati	Cincinnati Enquirer
NHL Teams			
Boston	Boston Bruins	Montreal	Montreal Canadiens
Phoenix	Phoenix Coyotes	Nashville	Nashville Predators
NBA Teams			
Detroit	Detroit Pistons	Toronto	Toronto Raptors
Oakland	Golden State Warriors	Memphis	Memphis Grizzlies
Airlines			
Austria	Austrian Airlines	Spain	Spainair
Belgium	Brussels Airlines	Greece	Aegean Airlines
Company executives			
Steve Ballmer	Microsoft	Larry Page	Google
Samuel J. Palmisano	IBM	Werner Vogels	Amazon

Table 2: Examples of the analogical reasoning task for phrases (the full test set has 3218 examples). The goal is to compute the fourth phrase using the first three. Our best model achieved an accuracy of 72% on this dataset.

Analogy Properties Learned

[Mikolov et al., 2013]



Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

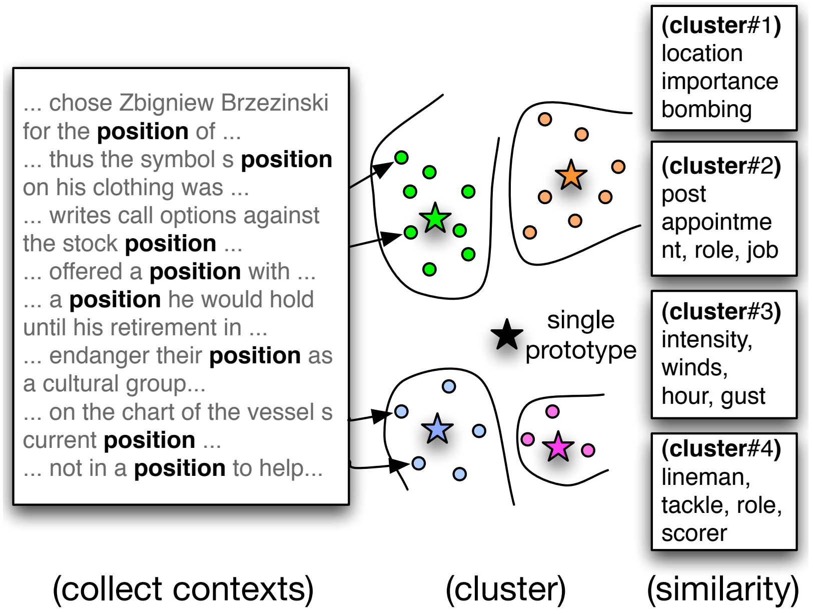
Table 5: Vector compositionality using element-wise addition. Four closest tokens to the sum of two vectors are shown, using the best Skip-gram model.

Distributional Semantics

- Other approaches: spectral methods, e.g., CCA
  - Word-context correlation [Dhillon et al., 2011, 2012]
  - Multilingual correlation [Faruqui and Dyer, 2014; Lu et al., 2015]
- Multi-sense embeddings [Reisinger and Mooney, 2010; Neelakantan et al., 2014]
- Some later ideas: Train task-tailored embeddings to capture specific types of similarity/semantics, e.g.,
  - Dependency context [Bansal et al., 2014, Levy and Goldberg, 2014]
  - Predicate-argument structures [Hashimoto et al., 2014; Madhyastha et al., 2014]
  - Lexicon evidence (PPDB, WordNet, FrameNet) [Xu et al., 2014; Yu and Dredze, 2014; Faruqui et al., 2014; Wieting et al., 2015]
  - Combining advantages of global matrix factorization and local context window methods [GloVe; Pennington et al., 2014]

Multi-sense Embeddings

- Different vectors for each sense of a word



... chose Zbigniew Brzezinski for the **position** of ...
... thus the symbol s **position** on his clothing was ...
... writes call options against the stock **position** ...
... offered a **position** with ...
... a **position** he would hold until his retirement in ...
... endanger their **position** as a cultural group...
... on the chart of the vessel s current **position** ...
... not in a **position** to help...
single prototype
(cluster#1) location importance bombing
(cluster#2) post appointment, role, job
(cluster#3) intensity, winds, hour, gust
(cluster#4) lineman, tackle, role, scorer
(collect contexts)
(cluster)
(similarity)

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, 2010]

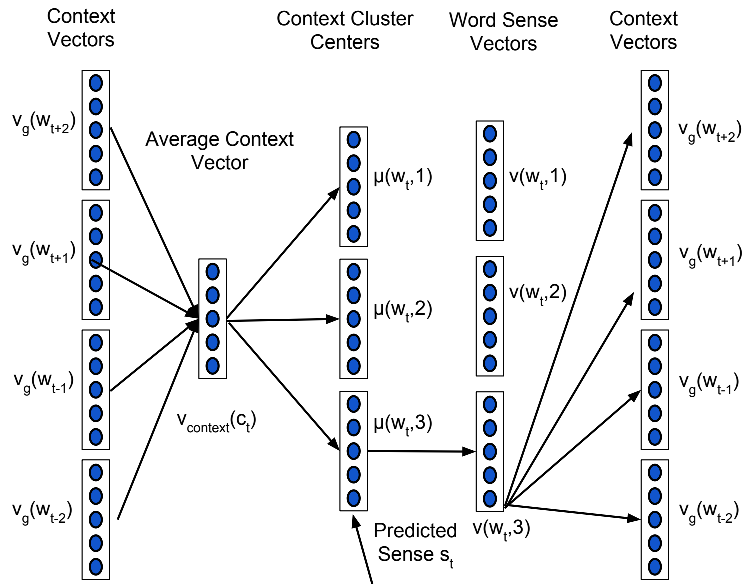


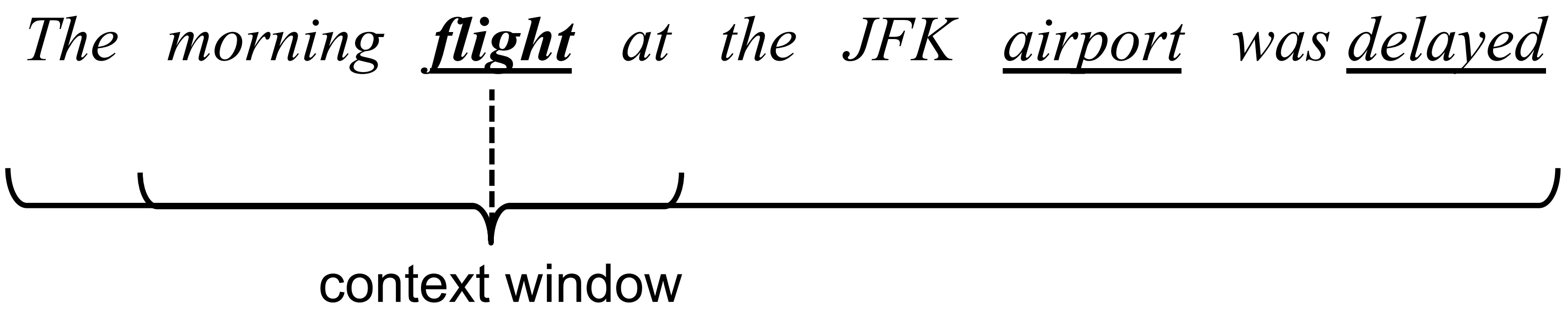
Figure 2: Architecture of Multi-Sense Skip-gram (MSSG) model with window size $R_t = 2$ and $K = 3$. Context c_t of word w_t consists of $w_{t-1}, w_{t-2}, w_{t+1}, w_{t+2}$. The sense is predicted by finding the cluster center of the context that is closest to the average of the context vectors.

[Neelakantan et al., 2014]

Syntactically Tailored Embeddings

[Bansal et al., 2014]

- Context window size (SKIP)
  - Smaller window → syntactic/functional similarity
  - Larger window → topical similarity



- Similar effect in distributional representations (Lin and Wu, 2009)

Cluster Examples

[Bansal et al., 2014]

- SKIP, w = 10:

  [*attendant, takeoff, airport, carry-on, airplane, flown, landings, flew, fly, cabins, …*]

  [*maternity, childbirth, clinic, physician, doctor, medical, health-care, day-care, …*]

  [*transactions, equity, investors, capital, financing, stock, fund, purchases, …*]

Cluster Examples

[Bansal et al., 2014]

- SKIP, w = 1

  [*Mr., Mrs., Ms., Prof., III, Jr., Dr.*]

  [*Jeffrey, William, Dan, Robert, Stephen, Peter, John, Richard, ...*]

  [*Portugal, Iran, Cuba, Ecuador, Greece, Thailand, Indonesia, ...*]

  [*his, your, her, its, their, my, our*]

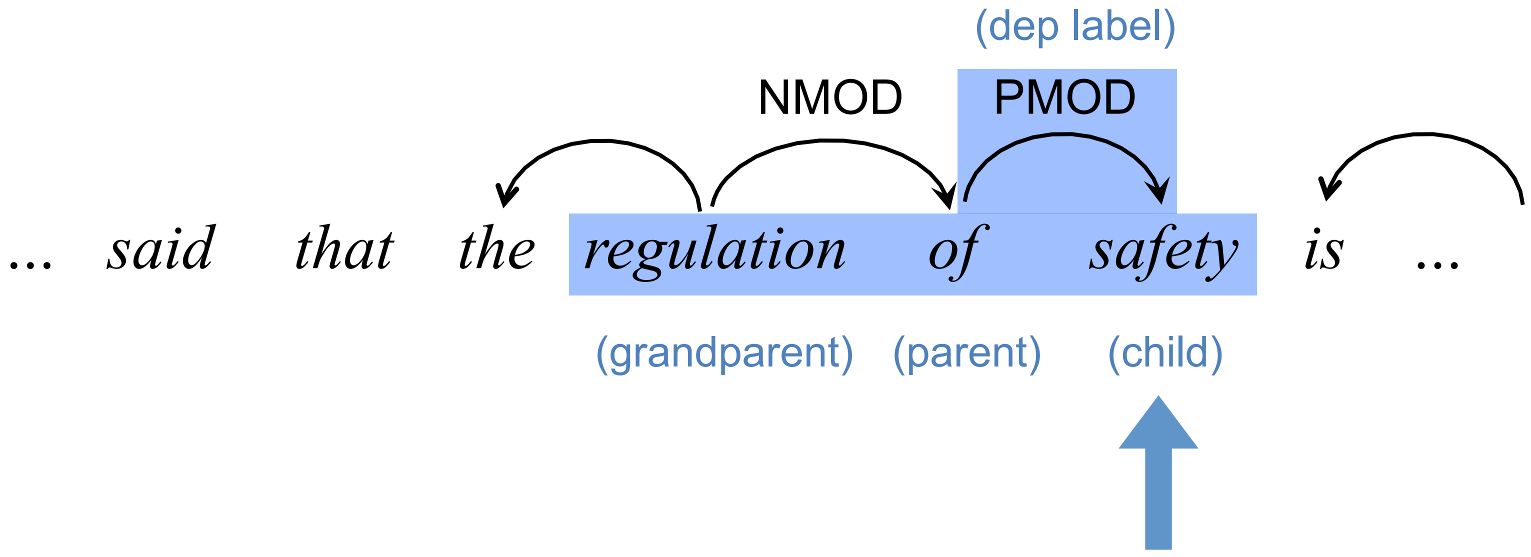
  [*Your, Our, Its, My, His, Their, Her*]

  [*truly, wildly, politically, financially, completely, potentially, ...*]

Syntactically Tailored Embeddings

[Bansal et al., 2014]

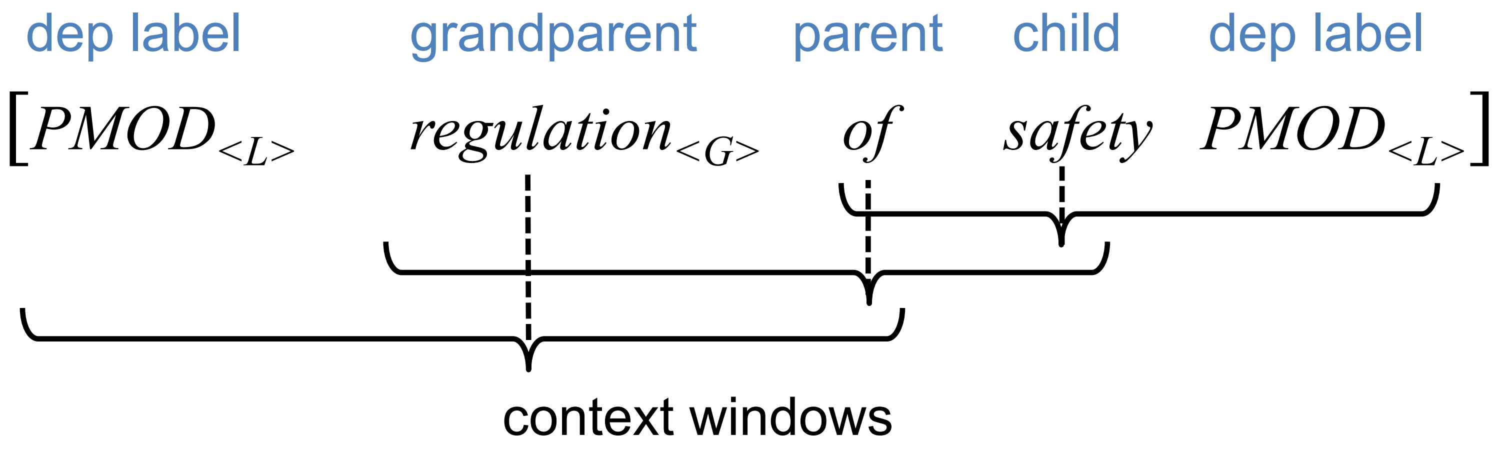
- Syntactic context (SKIP$_{\text{DEP}}$)
  - Condition on dependency context instead of linear
  - First parse a large corpus with baseline parser:



Syntactically Tailored Embeddings

[Bansal et al., 2014]

- Syntactic context (SKIP$_{\text{DEP}}$)
  - Condition on dependency context instead of linear
  - Then convert each dependency to a tuple:



- Syntactic information in clustering, topic, semantic space models

(Sagae and Gordon, 2009; Haffari et al., 2011; Grave et al., 2013; Boyd-Graber and Blei, 2008; Pado and Lapata, 2007)

Intrinsic Evaluation

[Bansal et al., 2014]

(Finkelstein et al., 2002)

Representation	SIM	TAG
BROWN	–	**89.3**
SENNA	49.8	85.2
HUANG	**62.6**	78.1
SKIP, $w = 10$	44.6	71.5
SKIP, $w = 5$	44.4	81.1
SKIP, $w = 1$	37.8	86.6
SKIP$_{\text{DEP}}$	34.6	88.3

Topical

Syntactic/ Functional

Parsing Experiments

[Bansal et al., 2014]

- Main WSJ results:

System	Test	
Baseline	91.9	
BROWN	**92.7**	
SENNA	92.3	
TURIAN	92.3	
HUANG	92.4	
SKIP	92.3	
SKIP$_{\text{DEP}}$	**92.7**	(faster)
Ensemble Results		
ALL – BROWN	92.9	(complementary)
ALL	93.0	

Task-Trained Embeddings

[Chen and Manning, 2014; CS224n]

- Can also directly train word embeddings on the task, via back-prop from the task supervision (XE errors), e.g., dependency parsing:

Softmax probabilities

Output layer y

$y = \text{softmax}(Uh + b_2)$

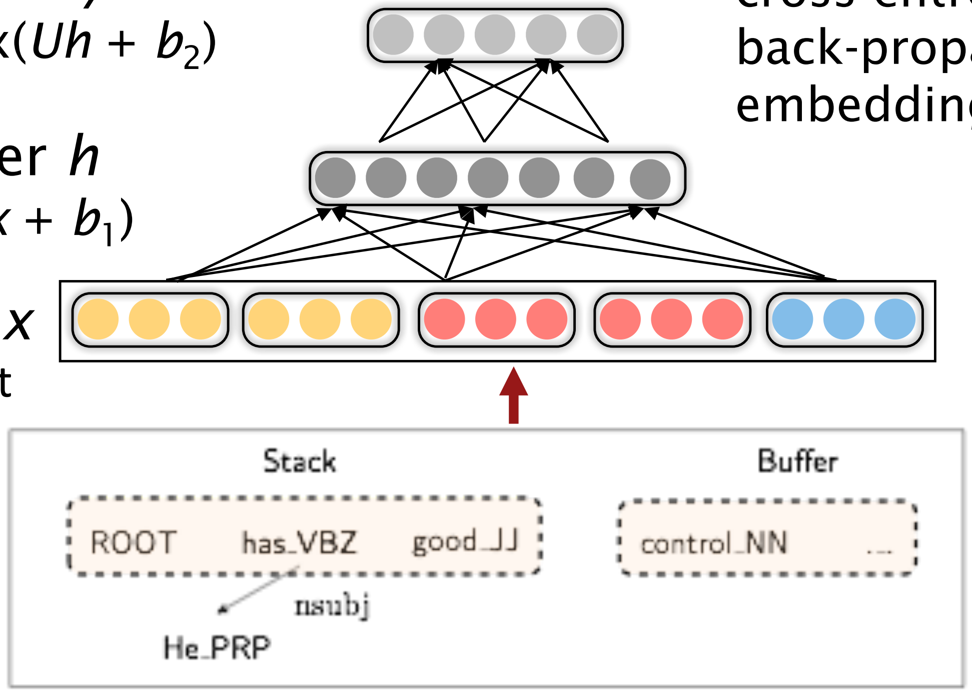
cross-entropy error will be back-propagated to the embeddings.

Hidden layer h

$h = \text{ReLU}(Wx + b_1)$

Input layer x

lookup + concat



Multilingual Embeddings via CCA

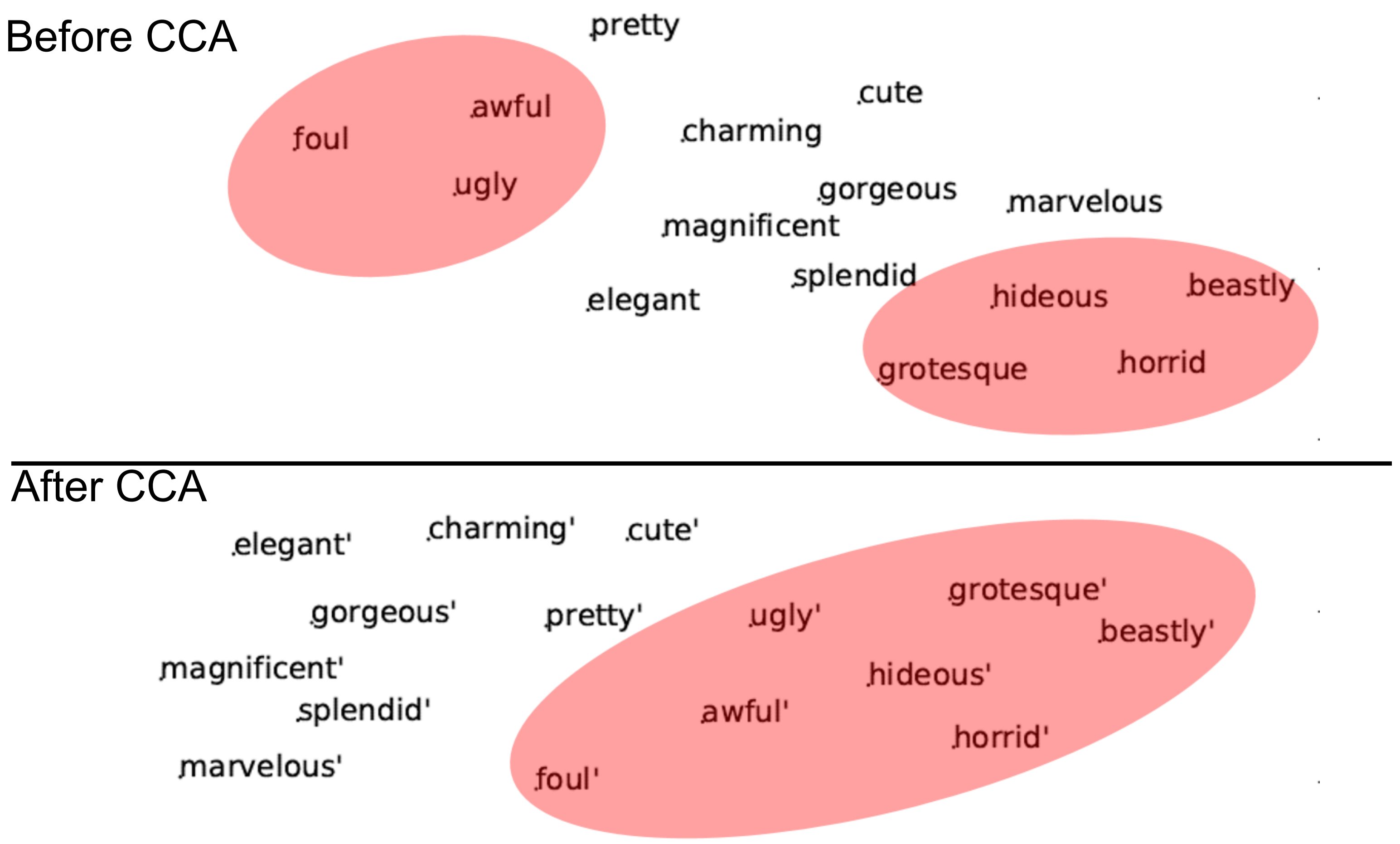
- Translational context (say, English ←→ German) can help learn stronger embeddings, e.g., separate antonyms vs. synonyms

- CCA on translation pairs to map them to shared space

$$\max_{\mathbf{u}\in\mathbb{R}^{D_x},\mathbf{v}\in\mathbb{R}^{D_y}} \frac{\mathbb{E}\left[(\mathbf{u}^\top\mathbf{x})(\mathbf{v}^\top\mathbf{y})\right]}{\sqrt{\mathbb{E}\left[(\mathbf{u}^\top\mathbf{x})^2\right]}\sqrt{\mathbb{E}\left[(\mathbf{v}^\top\mathbf{y})^2\right]}} = \frac{\mathbf{u}^\top\mathbf{\Sigma}_{xy}\mathbf{v}}{\sqrt{\mathbf{u}^\top\mathbf{\Sigma}_{xx}\mathbf{u}}\sqrt{\mathbf{v}^\top\mathbf{\Sigma}_{yy}\mathbf{v}}}$$

[Faruqui and Dyer, 2014]

Multi-view Embeddings via CCA



[Faruqui and Dyer, 2014]

Linear vs Deep CCA

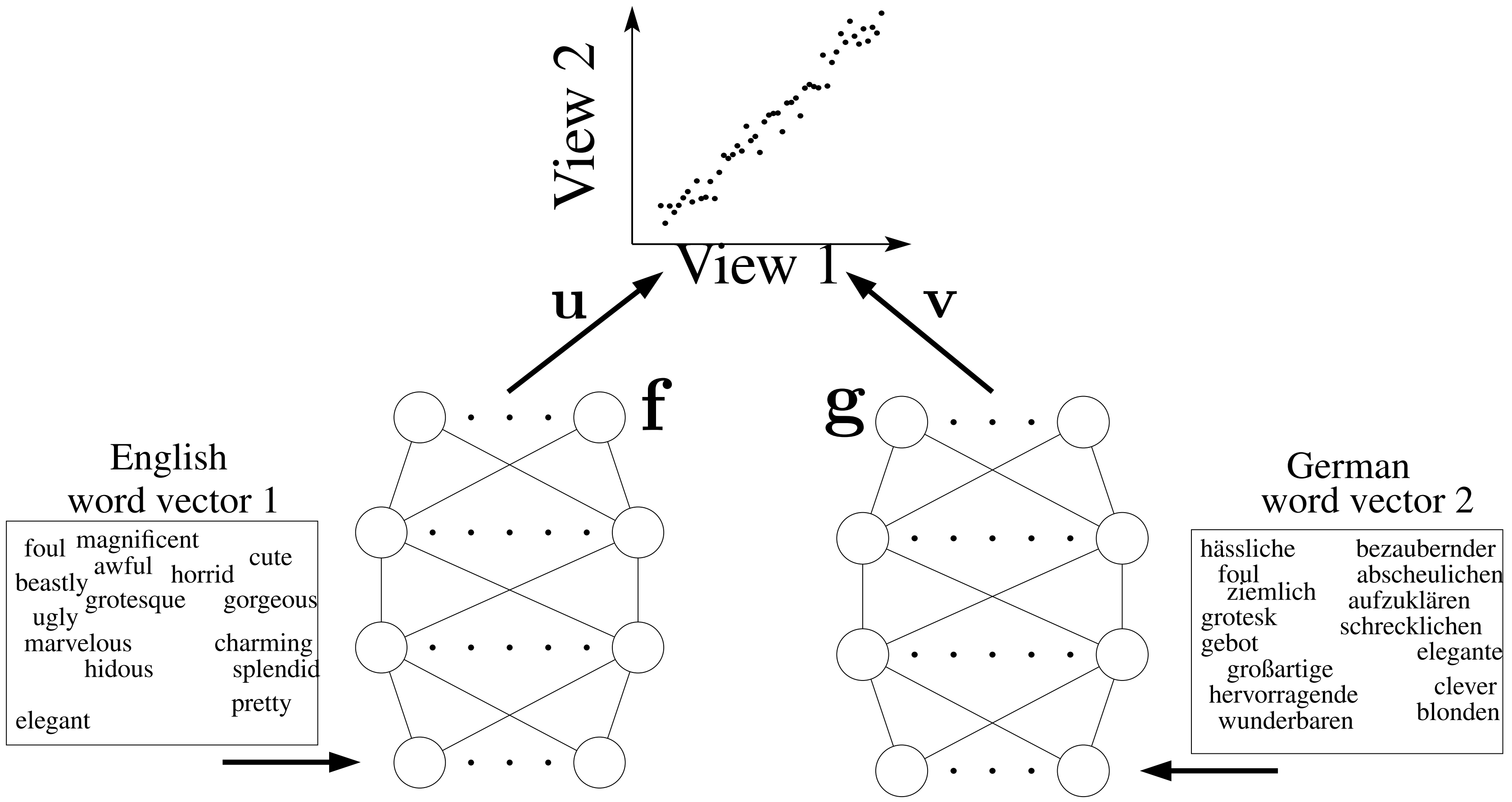
- Linear CCA results:

Embeddings	WS-353	WS-SIM	WS-REL	SL-999
Original	46.7	56.3	36.6	26.5
CCA-1	67.2	73.0	63.4	40.7
CCA-Ens	67.5	73.1	63.7	40.4

- Linear feature mapping not sufficiently powerful to capture hidden, non-linear relationships within data

- Use deep NNs to learn non-linear transformations of orig. embeddings to space where linear correlation maximized

Deep-CCA



[Lu, Wang, Bansal, Gimpel, Livescu, 2015]

Deep-CCA

- 2 DNNs **f**, **g** extract features from the 2 input views **x** and **y**
- DNNs are trained to maximize output linear correlation of 2 views
- DNN weights and linear projections optimized **together**:

$$\max_{\mathbf{W_f},\mathbf{W_g},\mathbf{u},\mathbf{v}} \quad \frac{\mathbf{u}^\top \boldsymbol{\Sigma}_{fg}\mathbf{v}}{\sqrt{\mathbf{u}^\top \boldsymbol{\Sigma}_{ff}\mathbf{u}}\sqrt{\mathbf{v}^\top \boldsymbol{\Sigma}_{gg}\mathbf{v}}}$$

- Covariance matrices computed for $\{\mathbf{f}(\mathbf{x}_i),\mathbf{g}(\mathbf{y}_i)\}_{i=1}^N$, as in CCA
- Mini-batch SGD: Feed-forward a sample to estimate (**u**, **v**) and gradient and then update NN weights via back-propagation

[Andrew et al., 2013]

Results

- Word-similarity improvements

Embeddings	WS-353	WS-SIM	WS-REL	SL-999
Original	46.7	56.3	36.6	26.5
CCA-1	67.2	73.0	63.4	40.7
CCA-Ens	67.5	73.1	63.7	40.4
DCCA-1 (BestAvg)	69.6	73.9	65.6	38.9
DCCA-Ens (BestAvg)	**70.8**	**75.2**	**67.3**	41.7
DCCA-1 (MostBeat)	68.6	73.5	65.7	**42.3**
DCCA-Ens (MostBeat)	69.9	74.4	66.7	**42.3**

- Also gets improvements on bigram similarity datasets

[Lu, Wang, Bansal, Gimpel, Livescu, 2015]

Analysis

- High-similarity word pairs that change most with DCCA

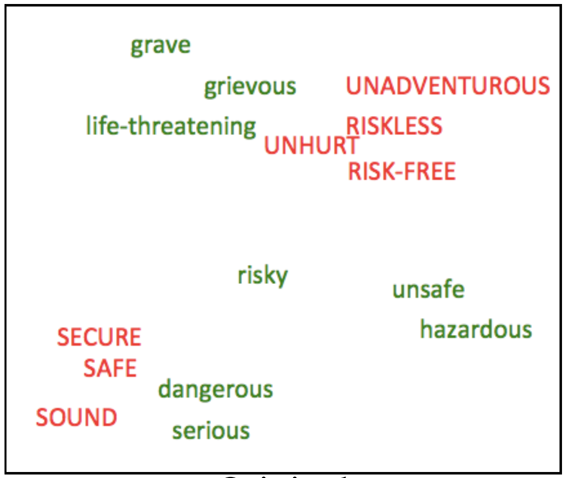
better with DCCA		worse with DCCA	
arrive	come	author	creator
locate	find	leader	manager
way	manner	buddy	companion
recent	new	crowd	bunch
take	obtain	achieve	succeed
boundary	border	attention	interest
win	accomplish	join	add
contemplate	think	mood	emotion

- DCCA discards hypernymy, separates senses

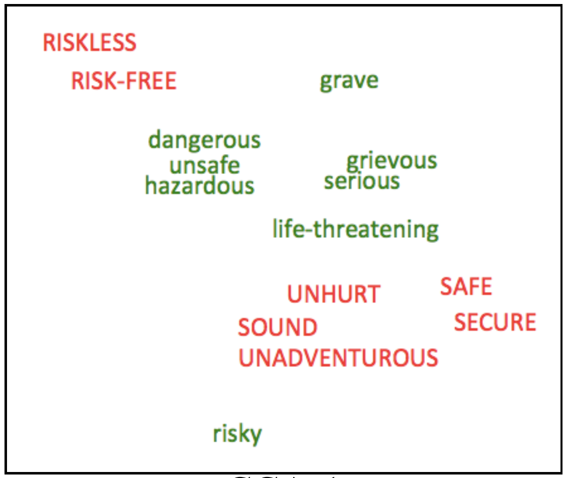
[Lu, Wang, Bansal, Gimpel, Livescu, 2015]

Analysis

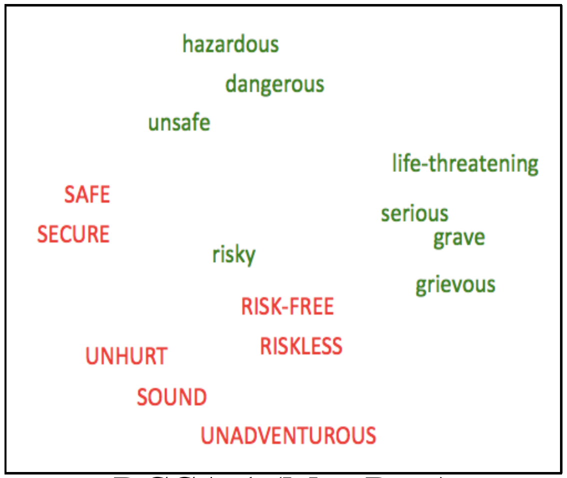
- DCCA more cleanly separates synonym-antonym lists



Original



CCA-1



DCCA-1 (MostBeat)

[Lu, Wang, Bansal, Gimpel, Livescu, 2015]

Retrofitting Word Embeddings to Lexicons

- We want the inferred word vector to be close to the observed value q̂ and close to its neighbors q_j, ∀j such that (i, j) ∈ E, where E is the set of relations in a dictionary/lexicon (e.g., WordNet, PPDB, etc.)

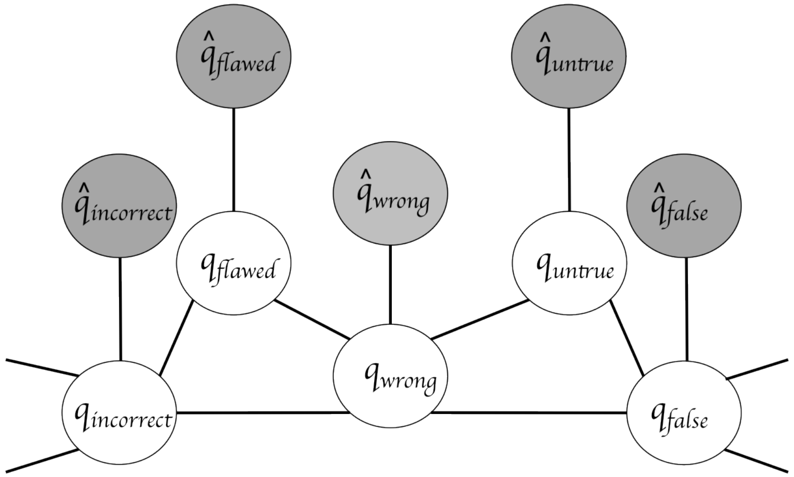
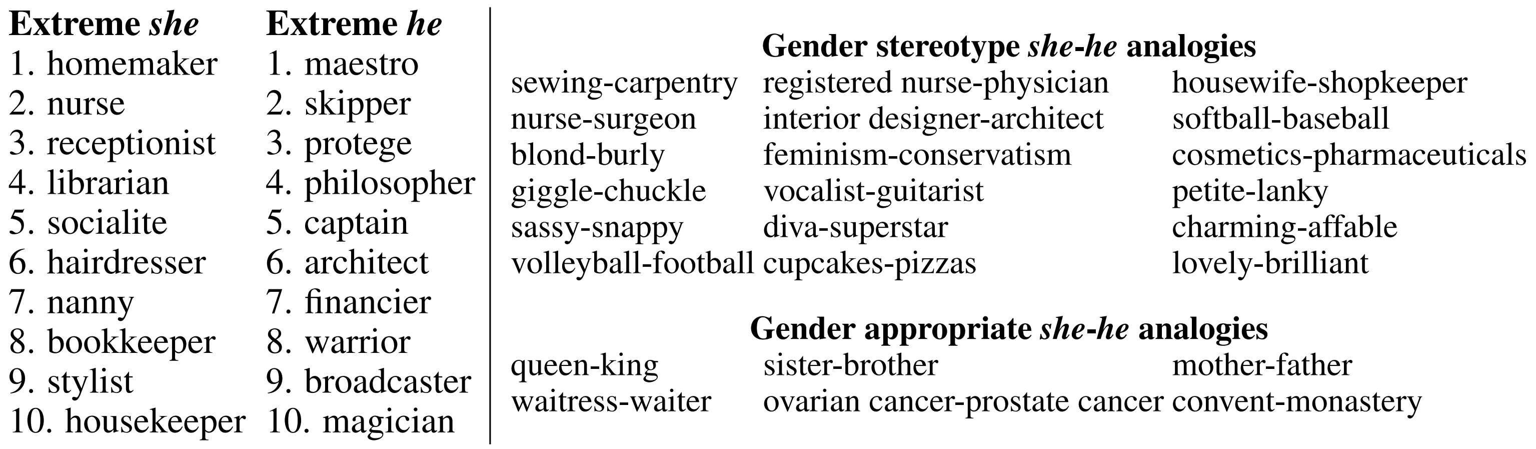


Figure 1: Word graph with edges between related words showing the observed (grey) and the inferred (white) word vector representations.

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

[Faruqui et al., 2015]

Bias in Word Embeddings



Extreme *she*	**Extreme *he***
1. homemaker	1. maestro
2. nurse	2. skipper
3. receptionist	3. protege
4. librarian	4. philosopher
5. socialite	5. captain
6. hairdresser	6. architect
7. nanny	7. financier
8. bookkeeper	8. warrior
9. stylist	9. broadcaster
10. housekeeper	10. magician

Gender stereotype *she*-*he* analogies

sewing-carpentry	registered nurse-physician	housewife-shopkeeper
nurse-surgeon	interior designer-architect	softball-baseball
blond-burly	feminism-conservatism	cosmetics-pharmaceuticals
giggle-chuckle	vocalist-guitarist	petite-lanky
sassy-snappy	diva-superstar	charming-affable
volleyball-football	cupcakes-pizzas	lovely-brilliant

Gender appropriate *she*-*he* analogies

queen-king	sister-brother	mother-father
waitress-waiter	ovarian cancer-prostate cancer	convent-monastery

Figure 1: **Left** The most extreme occupations as projected on to the *she*−*he* gender direction on w2vNEWS. Occupations such as *businesswoman*, where gender is suggested by the orthography, were excluded. **Right** Automatically generated analogies for the pair *she*-*he* using the procedure described in text. Each automatically generated analogy is evaluated by 10 crowd-workers to whether or not it reflects gender stereotype.

- Debiasing word embeddings via identifying pairs (sets) of words to correct/neutralize, identify bias direction (subspace), and then debias via neutralize+equalize or soften algorithms.

[Bolukbasi et al., 2016]

Compositional Semantics with NNs

- Composing, combining word vectors to representations for longer units: phrases, sentences, paragraphs, …
- Initial approaches: point-wise sum, multiplication [Mitchell and Lapata, 2010; Blacoe and Lapata, 2012]
- Vector-matrix compositionality [Baroni and Zamparelli, 2010; Zanzotto et al., 2010; Grefenstette and Sadrzadeh, 2011; Socher et al., 2011; Yessenalina and Cardie, 2011]
- Linguistic information added via say parses in RvNNs [Socher et al., 2011b, 2012, 2013a, 2013b, 2014; Hermann and Blunsom, 2013]
- Sequential RNNs (with GRU/LSTM gates)

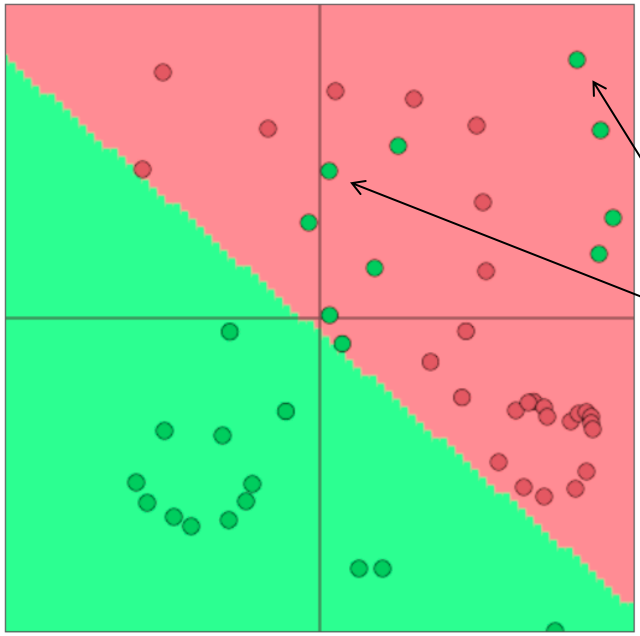
(Simple vector averaging w/ updating sometimes competitive)

Compositional Semantics with NNs

- Feed-forward NNs with back-propagation

Softmax (= logistic regression) is not very powerful

- Softmax only linear decision boundaries



→ Lame when problem is complex

Wouldn't it be cool to get these correct?

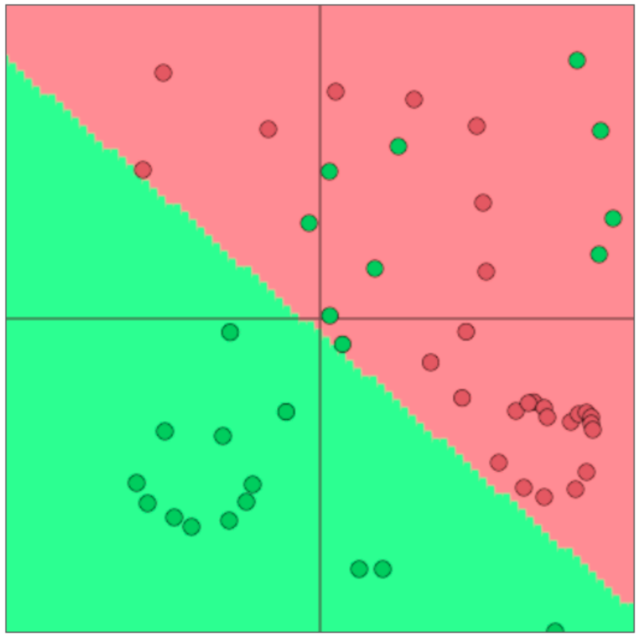
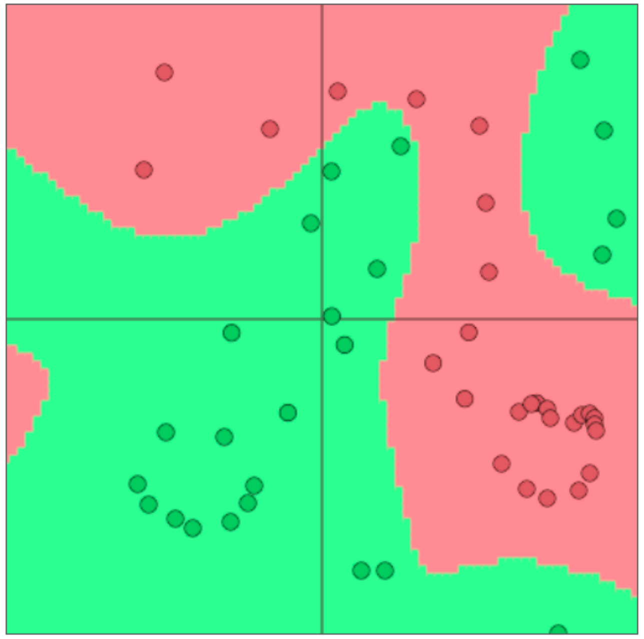
NN and backprop slides from CS224d – Richard Socher

Compositional Semantics with NNs

- Feed-forward NNs with back-propagation

Neural Nets for the Win!

- Neural networks can learn much more complex functions and nonlinear decision boundaries!

Compositional Semantics with NNs

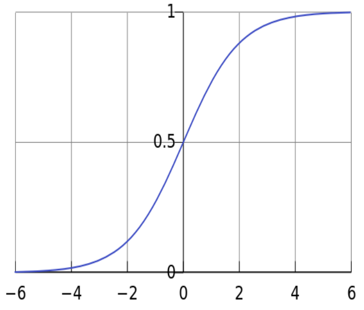
- Feed-forward NNs with back-propagation

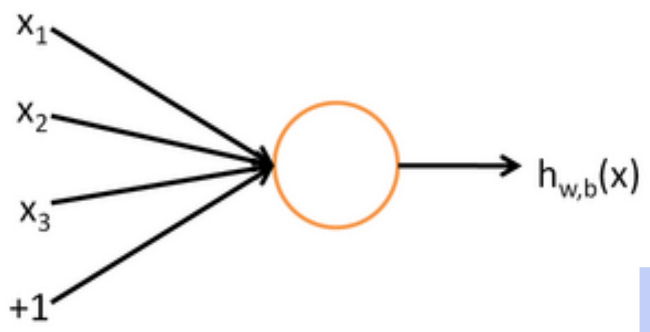
A neuron is essentially a binary logistic regression unit

$$h_{w,b}(x) = f(w^{\top}x + b)$$

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

b: We can have an "always on" feature, which gives a class prior, or separate it out, as a bias term





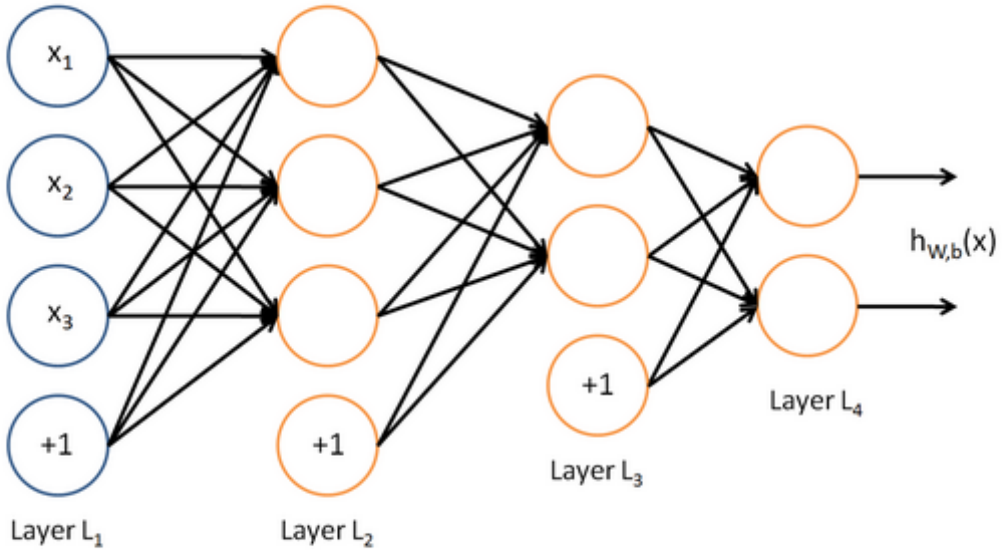
w, *b* are the parameters of this neuron i.e., this logistic regression model

Compositional Semantics with NNs

- Feed-forward NNs with back-propagation

A neural network
= running several logistic regressions at the same time

Before we know it, we have a multilayer neural network….



Compositional Semantics with NNs

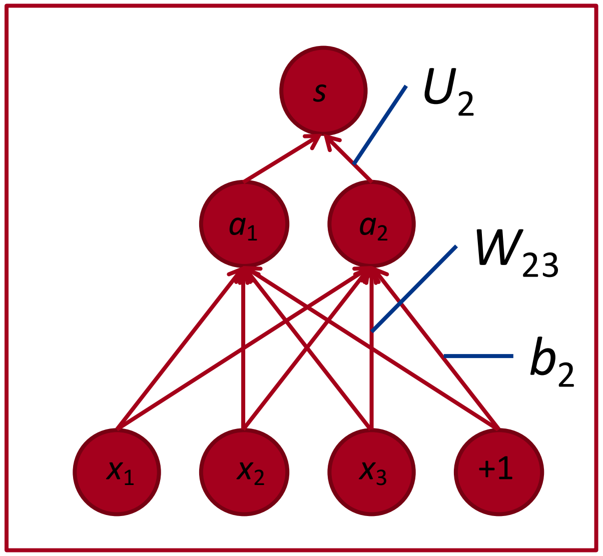
- Feed-forward NNs with back-propagation

Training with Backpropagation

- Let's consider the derivative of a single weight W_{ij}

$$\frac{\partial s}{\partial W} = \frac{\partial}{\partial W} U^T a = \frac{\partial}{\partial W} U^T f(z) = \frac{\partial}{\partial W} U^T f(Wx + b)$$

- This only appears inside a_i
- For example: W_{23} is only used to compute a_2



Compositional Semantics with NNs

- Feed-forward NNs with back-propagation

Training with Backpropagation

$$\frac{\partial s}{\partial W} = \frac{\partial}{\partial W} U^T a = \frac{\partial}{\partial W} U^T f(z) = \frac{\partial}{\partial W} U^T f(Wx+b)$$

Derivative of weight W_{ij}:

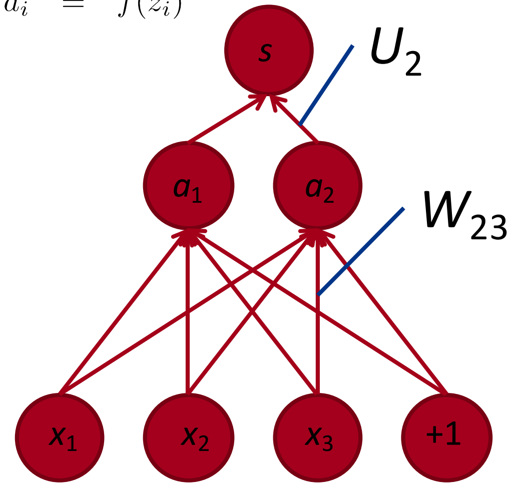
$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial u}\frac{\partial u}{\partial x}$$

$$\frac{\partial}{\partial W_{ij}} U^T a \rightarrow \frac{\partial}{\partial W_{ij}} U_i a_i$$

$$z_i = W_{i\cdot}x + b_i = \sum_{j=1}^{3} W_{ij}x_j + b_i$$

$$a_i = f(z_i)$$

$$\begin{aligned} U_i \frac{\partial}{\partial W_{ij}} a_i &= U_i \frac{\partial a_i}{\partial z_i} \frac{\partial z_i}{\partial W_{ij}} \\ &= U_i \frac{\partial f(z_i)}{\partial z_i} \frac{\partial z_i}{\partial W_{ij}} \\ &= U_i f'(z_i) \frac{\partial z_i}{\partial W_{ij}} \\ &= U_i f'(z_i) \frac{\partial W_{i\cdot}x + b_i}{\partial W_{ij}} \end{aligned}$$



Compositional Semantics with NNs

- Feed-forward NNs with back-propagation

Training with Backpropagation

Derivative of single weight W_{ij} :

$$z_i = W_{i\cdot}x + b_i = \sum_{j=1}^{3} W_{ij}x_j + b_i$$

$$a_i = f(z_i)$$

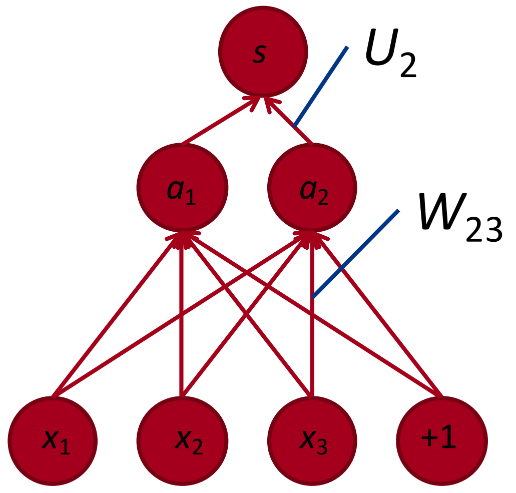
$$U_i \frac{\partial}{\partial W_{ij}} a_i = U_i f'(z_i) \frac{\partial W_{i\cdot}x + b_i}{\partial W_{ij}}$$

$$= U_i f'(z_i) \frac{\partial}{\partial W_{ij}} \sum_k W_{ik}x_k$$

$$= \underbrace{U_i f'(z_i)}\, x_j$$

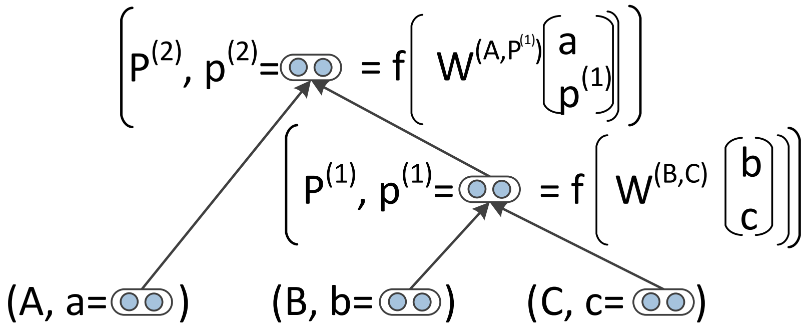
$$= \underbrace{\delta_i}_{\text{Local error signal}} \quad \underbrace{x_j}_{\text{Local input signal}}$$

where $f'(z) = f(z)(1 - f(z))$ for logistic f

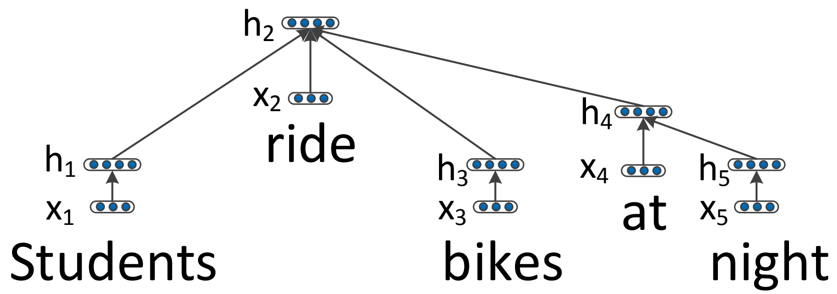


Syntactically Recursive NNs

- Socher et al., 2013a, 2014: RvNNs on constituent and dependency parse trees



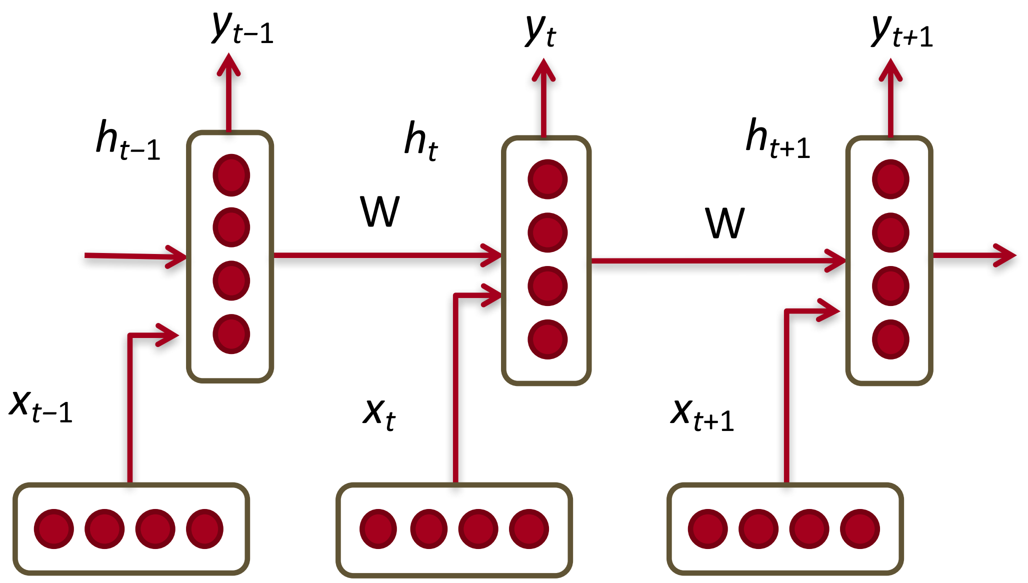
$P^{(2)}, p^{(2)} = f\left(W^{(A,P^{(1)})}\begin{bmatrix} a \\ p^{(1)} \end{bmatrix}\right)$
$P^{(1)}, p^{(1)} = f\left(W^{(B,C)}\begin{bmatrix} b \\ c \end{bmatrix}\right)$
(A, a=)
(B, b=)
(C, c=)



https://stanfordnlp.github.io/CoreNLP/demo.html
https://parser.kitaev.io/

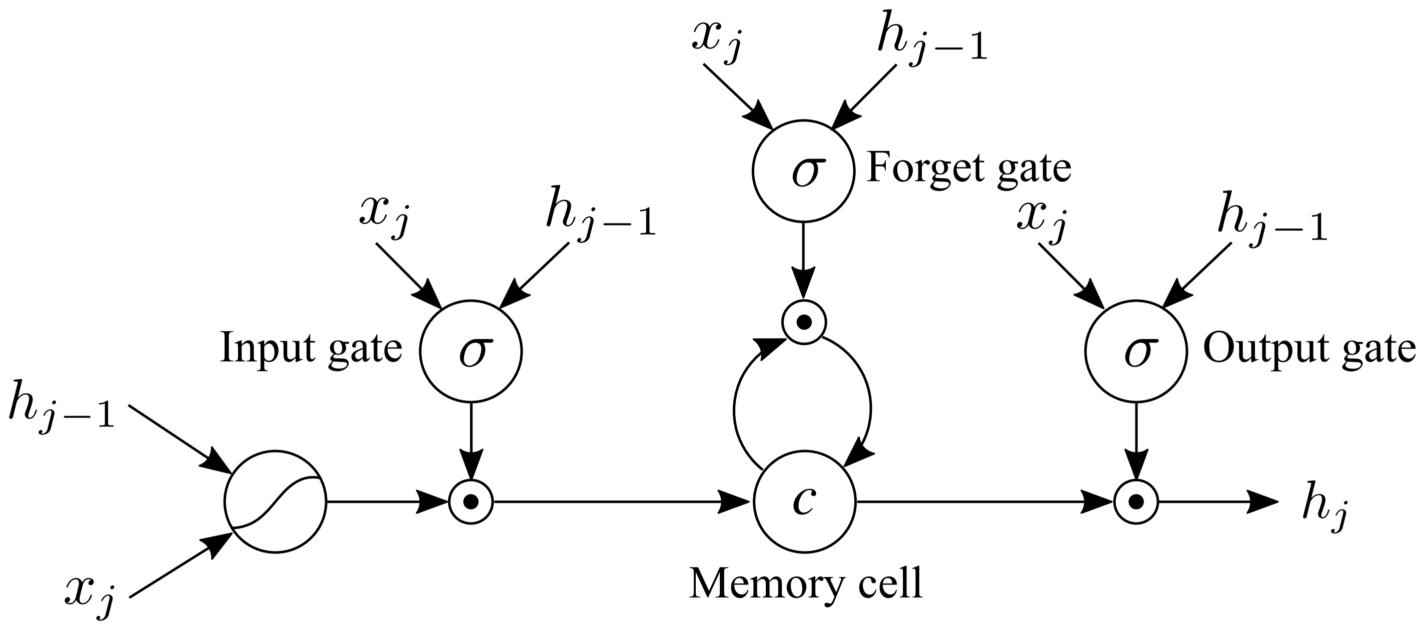
Recurrent NNs

- Recurrent NNs (RNNs) are non-tree, sequential versions of recursive RvNNs
- Weights tied together for each time step
- Loss function on identity of predicted word at each time step



LSTM RNNs

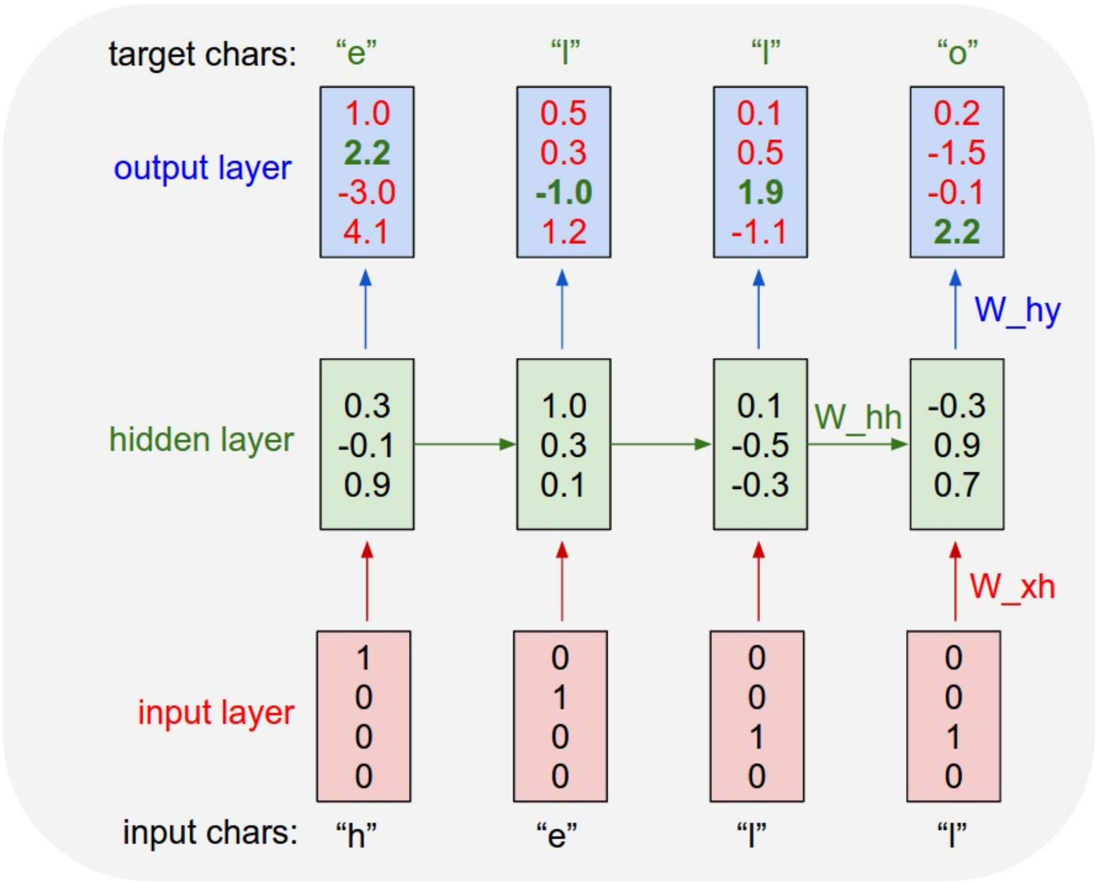
- LSTM (Long short term memory) RNNs have gates for forgetting, allowing learning of longer-term connections by avoiding vanishing/ exploding gradients



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Character RNNs

- Can directly process each character as a unit!
- Helps learn prefixes, stems, suffixes (form vs. function, rare/ unseen words, etc.)



http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Supervised Sentence Embedding Models

- Just like word embeddings were supervised using lexicons, dictionaries, taxonomies (WordNet) etc., sentence embeddings also benefit greatly from supervision!
- 2 examples: supervision based on bidirectional sentence similarity (paraphrases) or directed similarity (entailment vs contradiction vs neutral)

Paraphrase-based Sentence Embeddings

- Phrases that mean the same, are replaceable in context

main reason why	|||	*principal reason for*
informed about the outcome	|||	*notified of the results*
with particular emphasis	|||	*with specific focus*
we 'll have a good time	|||	*we 're gonna have fun*
50 years ago	|||	*five decades ago*
that , according to	|||	*which , in accordance with*
program is aimed at	|||	*programme aims to*
are under the obligation	|||	*have a duty*
a critical component	|||	*an essential element*

Paraphrase-based Sentence Embeddings

- PPDB: Massive, useful resource (220M) automatically extracted from parallel bilingual corpora [Ganitkevitch et al., 2013]

- Idea summary: carefully extract a few (< 0.05%) +ve and -ve pairs from unannotated PPDB as weak supervision

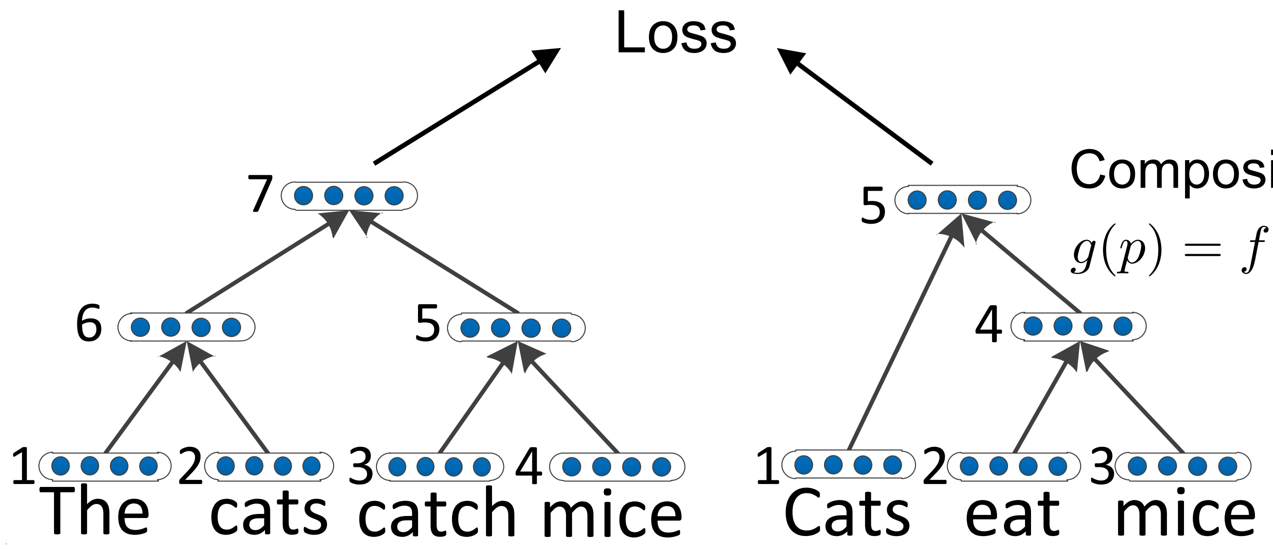
- Train a parametric paraphrase model (2-view RNN with hinge loss) on these pairs, to be able to represent arbitrary phrases as embeddings

- This learns strong word/phrase embeddings that better predict paraphrases on new annotated PPDB subset and gets SoA on word/bigram similarity datasets

[Wieting, Bansal, Gimpel, Livescu, Roth, 2015]

Paraphrase Model

- 2 parse-based RvNNs with a hinge-based loss function



Composition =

$g(p) = f(W[g(c_1); g(c_2)] + b)$

[Socher et al., 2011]

Paraphrase Model

- Loss: +ve pairs closer than -ve pairs with margin δ

Positive training pairs

Negative training pairs

$$\min_{W,b,W_w} \frac{1}{|X|}\left(\boxed{\sum_{\langle x_1,x_2\rangle\in X}} \max(0, \delta - \boxed{g(x_1)\cdot g(x_2)} + \boxed{g(x_1)\cdot g(t_1)}) \right.$$

$$\left. + \max(0, \delta - \boxed{g(x_1)\cdot g(x_2)} + \boxed{g(x_2)\cdot g(t_2)})\right)$$

$$\underbrace{\hphantom{g(x_1)\cdot g(x_2)}}\quad \underbrace{\hphantom{g(x_1)\cdot g(t_1)}}$$

Regularization terms

[Wieting, Bansal, Gimpel, Livescu, Roth, 2015]

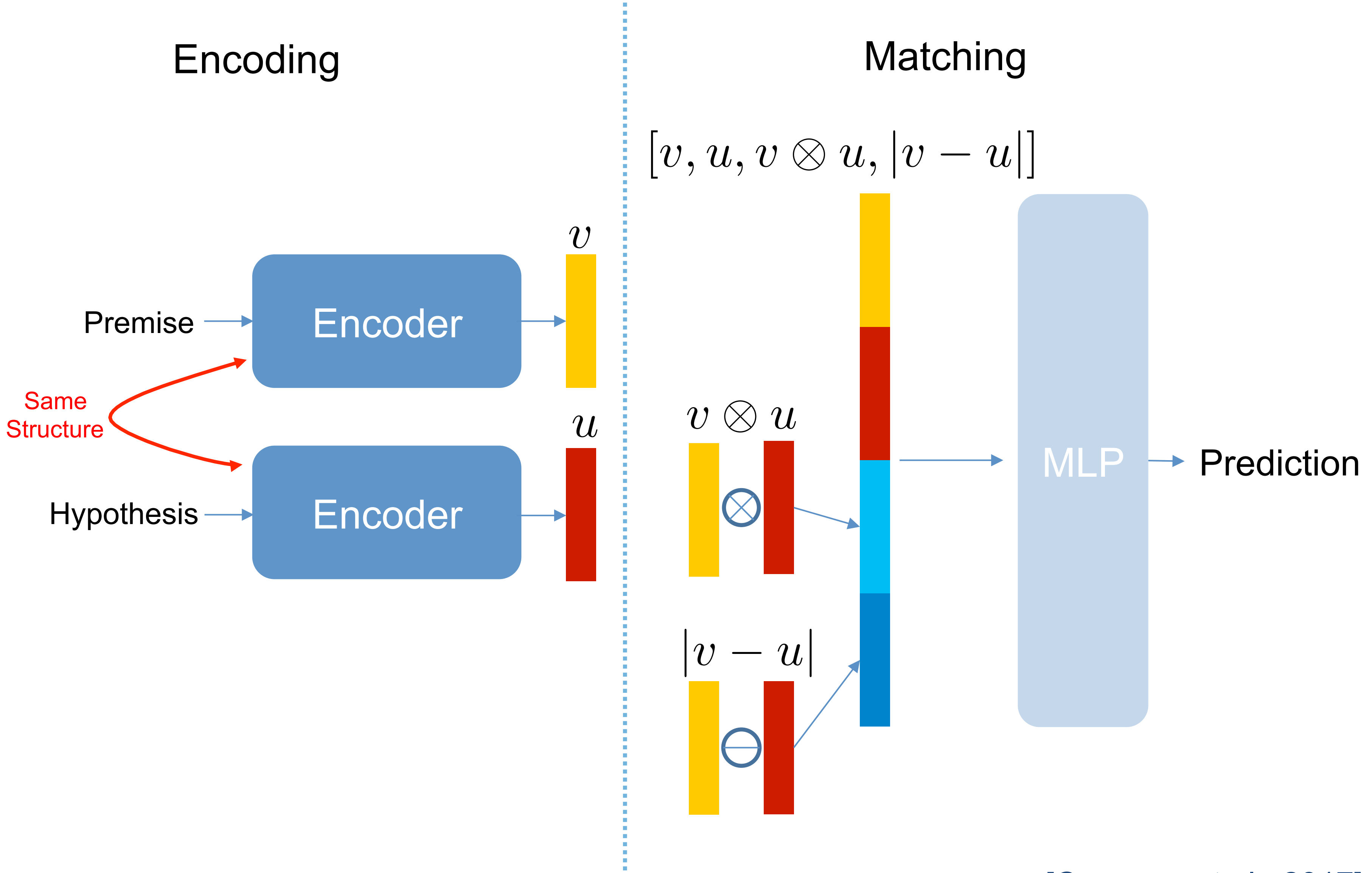
Entailment-based Embeddings

- SNLI and Multi-NLI corpora with sentence pairs of 3 relationships: entailment, contradiction, neutral/unrelated

Premise	Label	Hypothesis	Genre
The Old One always comforted Ca'daan, except today.	*neutral*	Ca'daan knew the Old One very well.	*Fiction*
Your gift is appreciated by each and every student who will benefit from your generosity.	*neutral*	Hundreds of students will benefit from your generosity.	*Letters*
yes now you know if if everybody like in August when everybody's on vacation or something we can dress a little more casual or	*contradiction*	August is a black out month for vacations in the company.	*Telephone Speech*
At the other end of Pennsylvania Avenue, people began to line up for a White House tour.	*entailment*	People formed a line at the end of Pennsylvania Avenue.	*9/11 Report*
A black race car starts up in front of a crowd of people.	*contradiction*	A man is driving down a lonely road.	*SNLI*

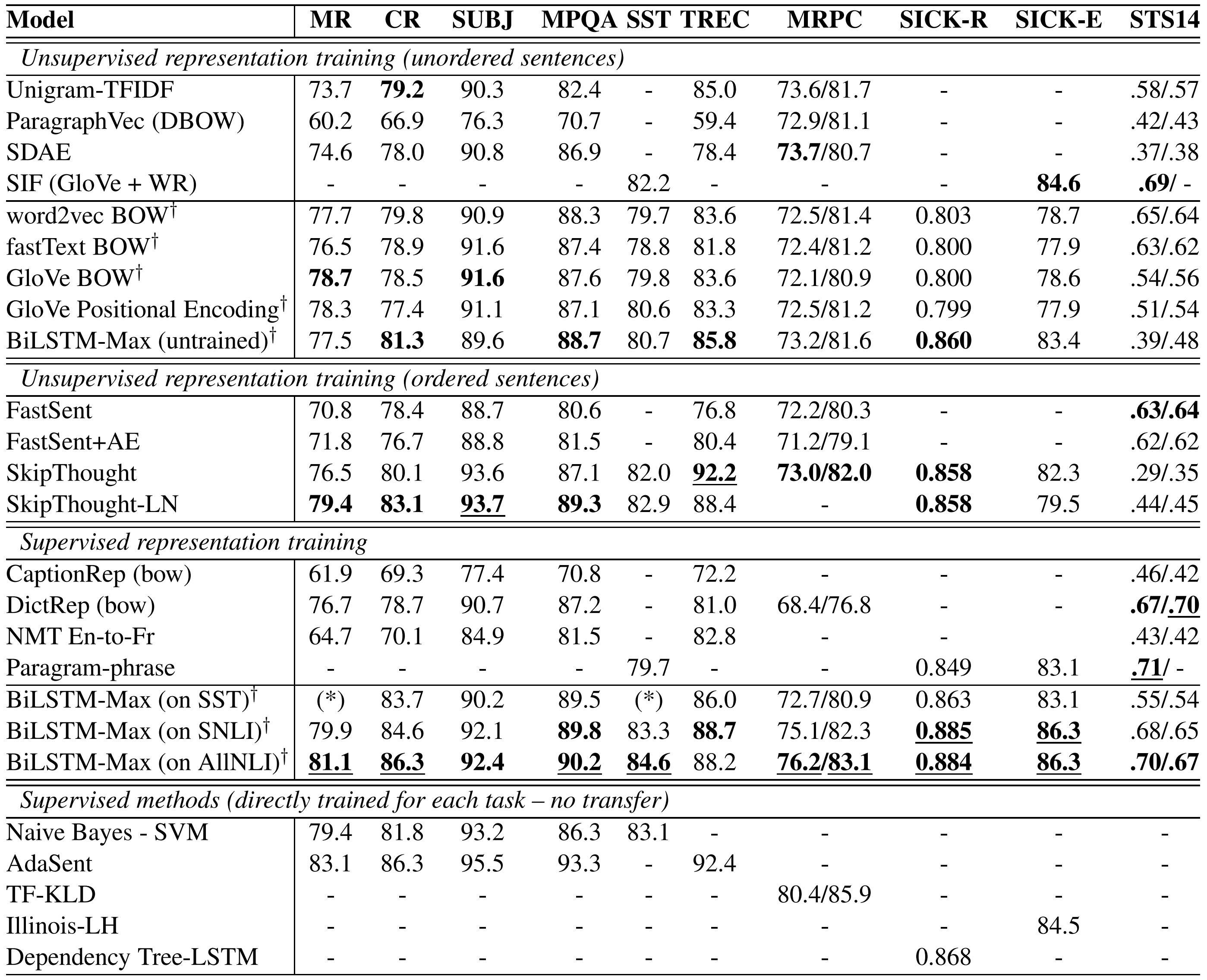
[Bowman et al., 2015; Williams et al., 2017]

Entailment-based Embeddings



[Conneau et al., 2017]

Entailment-based Embeddings

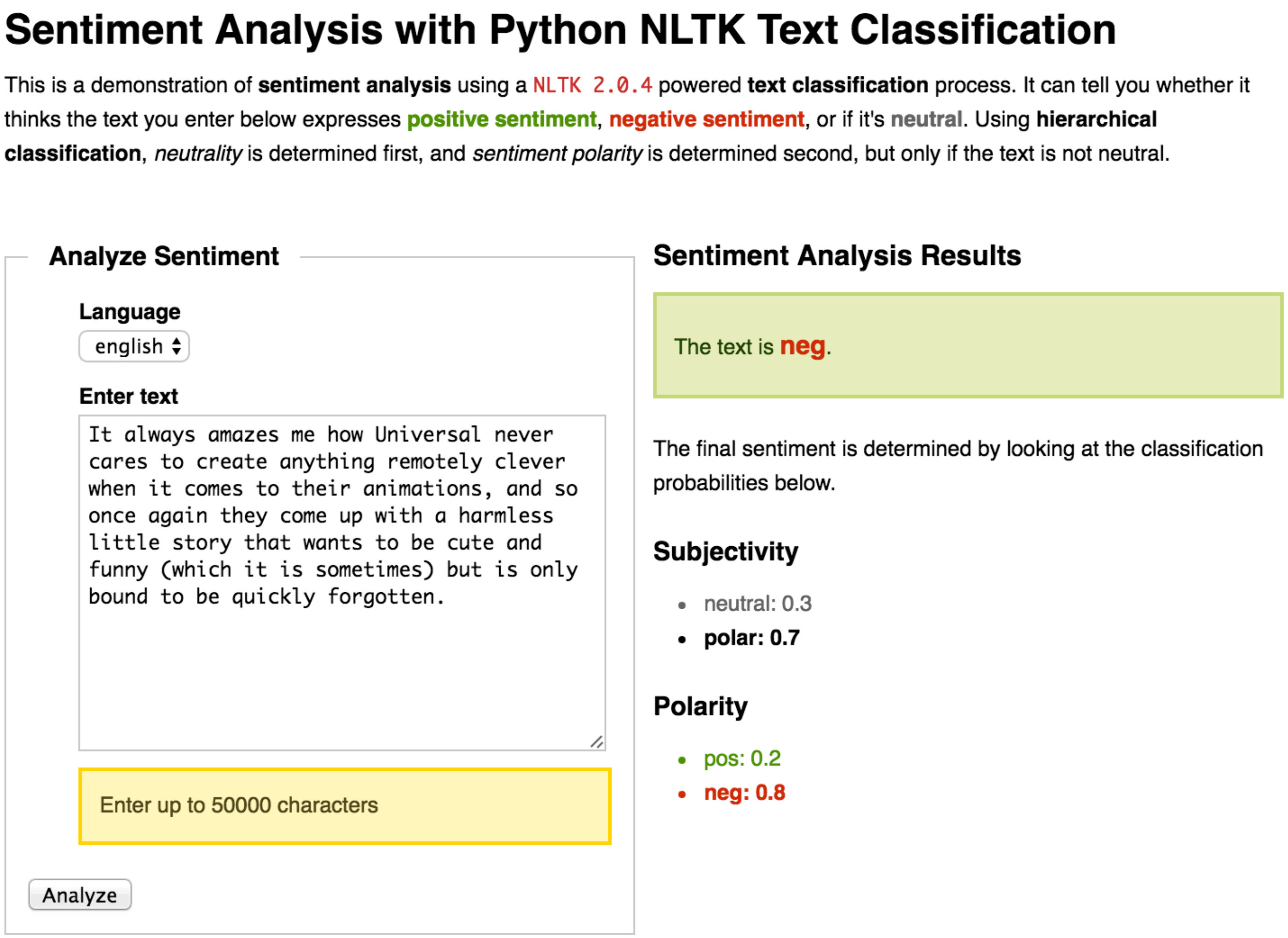


Model	MR	CR	SUBJ	MPQA	SST	TREC	MRPC	SICK-R	SICK-E	STS14
Unsupervised representation training (unordered sentences)										
Unigram-TFIDF	73.7	**79.2**	90.3	82.4	-	85.0	73.6/81.7	-	-	.58/.57
ParagraphVec (DBOW)	60.2	66.9	76.3	70.7	-	59.4	72.9/81.1	-	-	.42/.43
SDAE	74.6	78.0	90.8	86.9	-	78.4	**73.7**/80.7	-	-	.37/.38
SIF (GloVe + WR)	-	-	-	-	82.2	-	-	-	**84.6**	**.69**/ -
word2vec BOW†	77.7	79.8	90.9	88.3	79.7	83.6	72.5/81.4	0.803	78.7	.65/.64
fastText BOW†	76.5	78.9	91.6	87.4	78.8	81.8	72.4/81.2	0.800	77.9	.63/.62
GloVe BOW†	**78.7**	78.5	**91.6**	87.6	79.8	83.6	72.1/80.9	0.800	78.6	.54/.56
GloVe Positional Encoding†	78.3	77.4	91.1	87.1	80.6	83.3	72.5/81.2	0.799	77.9	.51/.54
BiLSTM-Max (untrained)†	77.5	**81.3**	89.6	**88.7**	80.7	**85.8**	73.2/81.6	**0.860**	83.4	.39/.48
Unsupervised representation training (ordered sentences)										
FastSent	70.8	78.4	88.7	80.6	-	76.8	72.2/80.3	-	-	**.63/.64**
FastSent+AE	71.8	76.7	88.8	81.5	-	80.4	71.2/79.1	-	-	.62/.62
SkipThought	76.5	80.1	93.6	87.1	82.0	**92.2**	**73.0/82.0**	**0.858**	82.3	.29/.35
SkipThought-LN	**79.4**	**83.1**	**93.7**	**89.3**	82.9	88.4	-	**0.858**	79.5	.44/.45
Supervised representation training										
CaptionRep (bow)	61.9	69.3	77.4	70.8	-	72.2	-	-	-	.46/.42
DictRep (bow)	76.7	78.7	90.7	87.2	-	81.0	68.4/76.8	-	-	**.67/.70**
NMT En-to-Fr	64.7	70.1	84.9	81.5	-	82.8	-	-		.43/.42
Paragram-phrase	-	-	-	-	79.7	-	-	0.849	83.1	**.71**/ -
BiLSTM-Max (on SST)†	(*)	83.7	90.2	89.5	(*)	86.0	72.7/80.9	0.863	83.1	.55/.54
BiLSTM-Max (on SNLI)†	79.9	84.6	92.1	**89.8**	83.3	**88.7**	75.1/82.3	**0.885**	**86.3**	.68/.65
BiLSTM-Max (on AllNLI)†	**81.1**	**86.3**	**92.4**	**90.2**	**84.6**	88.2	**76.2/83.1**	**0.884**	**86.3**	**.70/.67**
Supervised methods (directly trained for each task – no transfer)										
Naive Bayes - SVM	79.4	81.8	93.2	86.3	83.1	-	-	-	-	-
AdaSent	83.1	86.3	95.5	93.3	-	92.4	-	-	-	-
TF-KLD	-	-	-	-	-	-	80.4/85.9	-	-	-
Illinois-LH	-	-	-	-	-	-	-	-	84.5	-
Dependency Tree-LSTM	-	-	-	-	-	-	-	0.868	-	-

[Conneau et al., 2017]

Classification Tasks: Sentiment Analysis

FROM THE HUMANS BEHIND DESPICABLE ME
Ever wonder what your pets do when you're not home?
ILLUMINATION PRESENTS
THE SECRET LIFE OF PETS
JULY 8
IN THEATERS AND REAL D 3D



Sentiment Analysis with Python NLTK Text Classification

This is a demonstration of **sentiment analysis** using a `NLTK 2.0.4` powered **text classification** process. It can tell you whether it thinks the text you enter below expresses **positive sentiment**, **negative sentiment**, or if it's **neutral**. Using **hierarchical classification**, *neutrality* is determined first, and *sentiment polarity* is determined second, but only if the text is not neutral.

Analyze Sentiment

Language

english

Enter text

```
It always amazes me how Universal never
cares to create anything remotely clever
when it comes to their animations, and so
once again they come up with a harmless
little story that wants to be cute and
funny (which it is sometimes) but is only
bound to be quickly forgotten.
```

Enter up to 50000 characters

Sentiment Analysis Results

The text is **neg**.

The final sentiment is determined by looking at the classification probabilities below.

Subjectivity

- neutral: 0.3
- **polar: 0.7**

Polarity

- pos: 0.2
- **neg: 0.8**

Sentiment Analysis

- Earlier methods used bag of words, e.g., lexicons of positive and negative words and phrases
- Cannot distinguish tricky cases like:

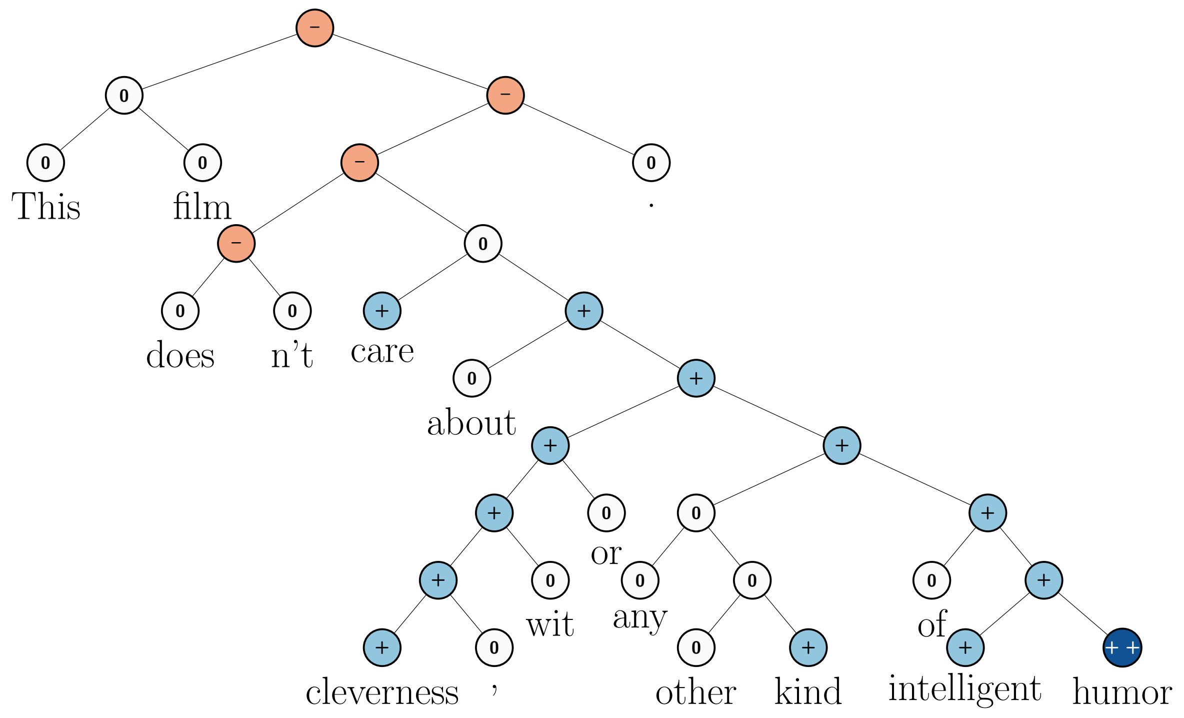
+	*white*	*blood*	*cells*	*destroying*	*an*	*infection*
–	*an*	*infection*	*destroying*	*white*	*blood*	*cells*

+ *There are slow and repetitive parts but it has just enough spice to keep it interesting.*

– *Stealing Harvard doesn't care about cleverness, wit or any other kind of intelligent humor.*

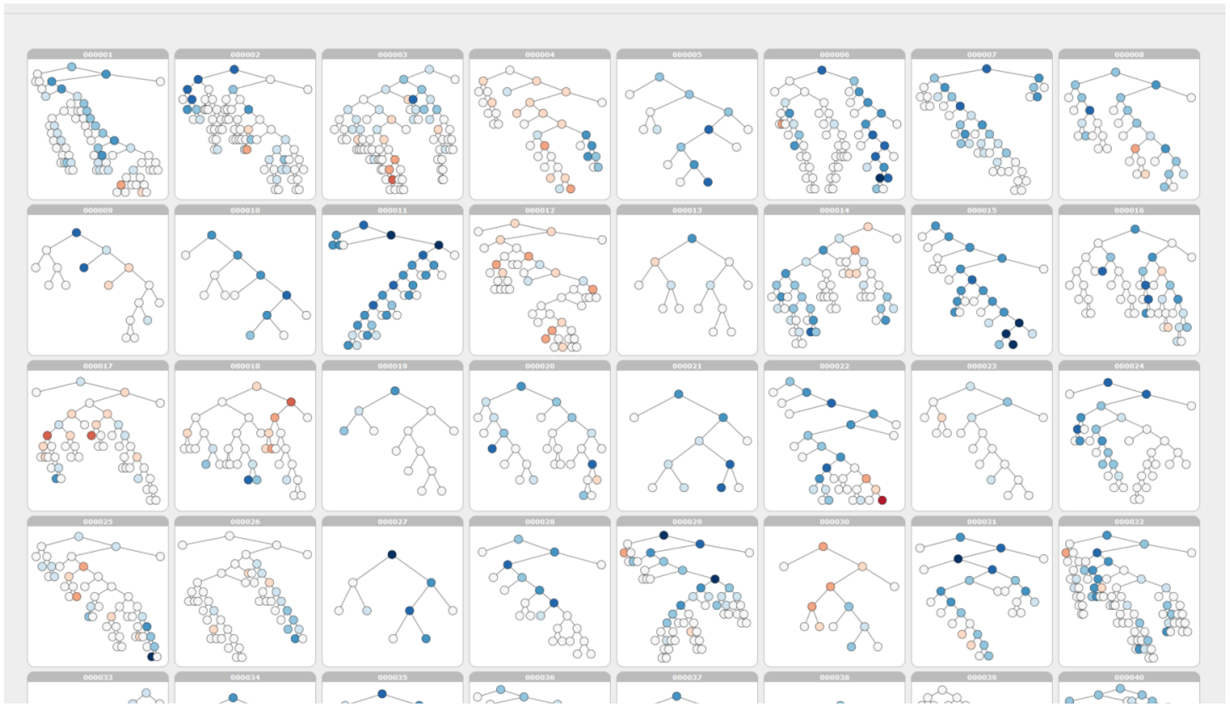
Sentiment Analysis

- Even simpler issues like negation hard to understand
- Socher et al., 2013b present new compositional training data and new composition model



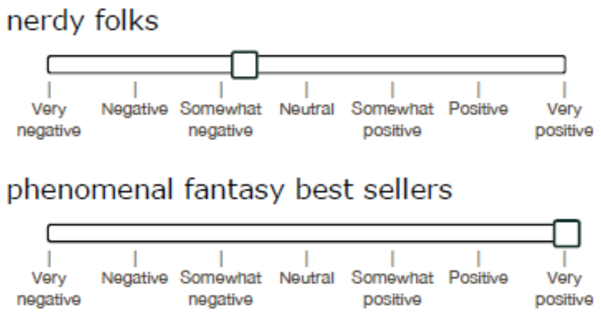
Sentiment Analysis

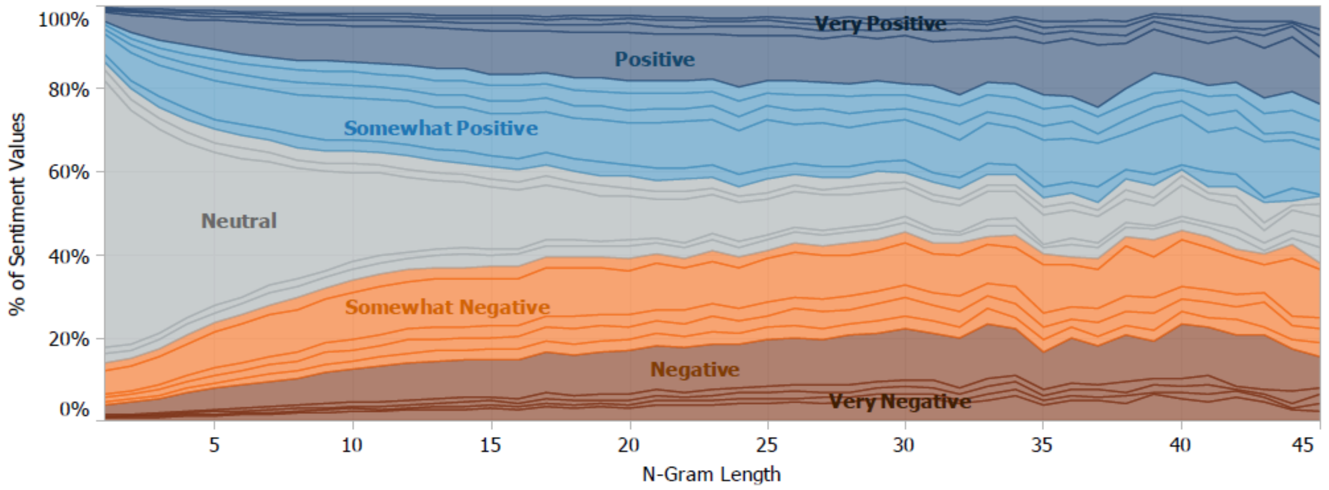
- Even simpler issues like negation hard to understand
- Socher et al., 2013b present new compositional training data and new composition model



Sentiment Analysis

- Sentiment Compositionality:
  - Parse trees of 11,855 sentences
  - 215,154 phrases with labels
  - Allows training and evaluating with compositional information



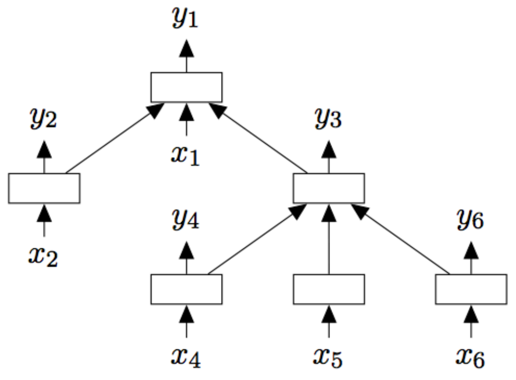


Sentiment Analysis

Better Models: Tree-based LSTM-RNNs

Tree LSTMs

- We can use those ideas in grammatical tree structures!
- Paper: Tai et al. 2015: Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks



- Idea: Sum the child vectors in a tree structure
- Each child has its own forget gate
- Same softmax on h

$$\boxed{\tilde{h}_j = \sum_{k \in C(j)} h_k,}$$

$$i_j = \sigma\left(W^{(i)} x_j + U^{(i)} \boxed{\tilde{h}_j} + b^{(i)}\right),$$

$$\boxed{f_{jk}} = \sigma\left(W^{(f)} x_j + U^{(f)} h_k + b^{(f)}\right),$$

$$o_j = \sigma\left(W^{(o)} x_j + U^{(o)} \boxed{\tilde{h}_j} + b^{(o)}\right),$$

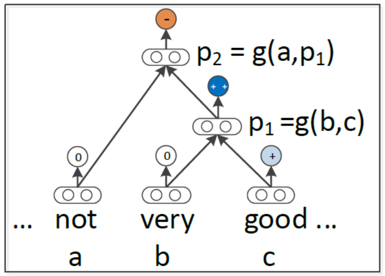
$$u_j = \tanh\left(W^{(u)} x_j + U^{(u)} \tilde{h}_j + b^{(u)}\right),$$

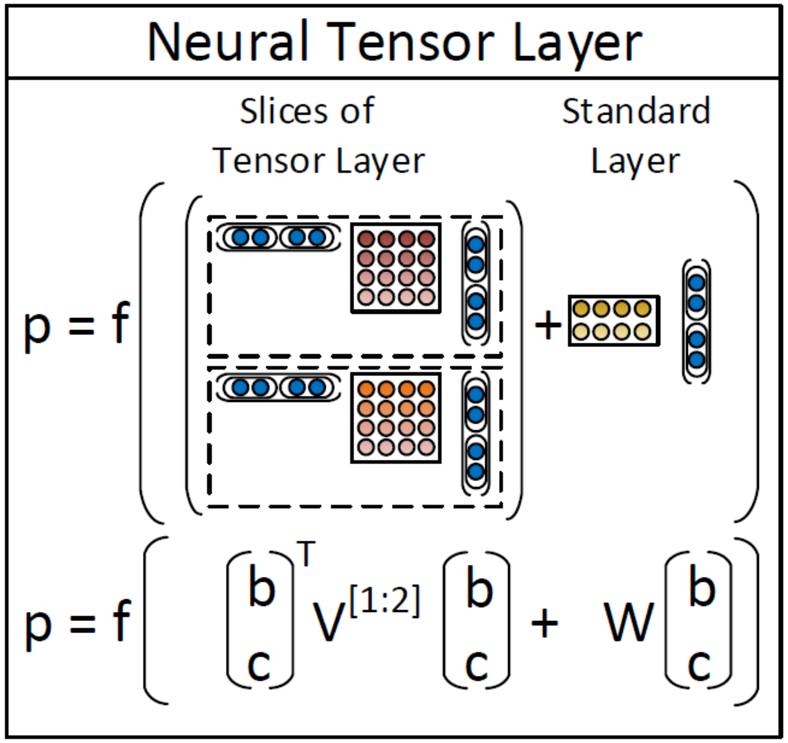
$$c_j = i_j \odot u_j + \boxed{\sum_{k \in C(j)} f_{jk} \odot c_k,}$$

$$h_j = o_j \odot \tanh(c_j),$$

Sentiment Analysis

- Better Models: Recursive Neural Tensor Network (RNTN)

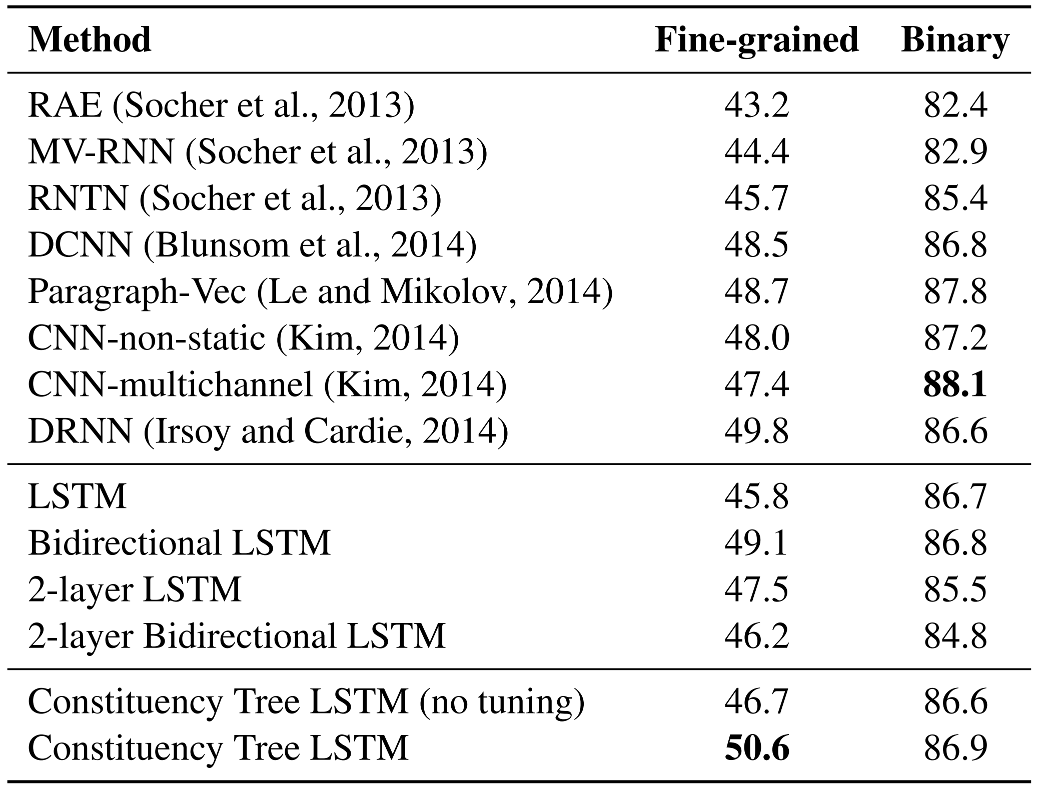




$$p = f\left(\begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:2]} \begin{bmatrix} b \\ c \end{bmatrix} + W \begin{bmatrix} b \\ c \end{bmatrix} \right)$$

Sentiment Compositionality

Results on Stanford Sentiment Treebank



Method	Fine-grained	Binary
RAE (Socher et al., 2013)	43.2	82.4
MV-RNN (Socher et al., 2013)	44.4	82.9
RNTN (Socher et al., 2013)	45.7	85.4
DCNN (Blunsom et al., 2014)	48.5	86.8
Paragraph-Vec (Le and Mikolov, 2014)	48.7	87.8
CNN-non-static (Kim, 2014)	48.0	87.2
CNN-multichannel (Kim, 2014)	47.4	**88.1**
DRNN (Irsoy and Cardie, 2014)	49.8	86.6
LSTM	45.8	86.7
Bidirectional LSTM	49.1	86.8
2-layer LSTM	47.5	85.5
2-layer Bidirectional LSTM	46.2	84.8
Constituency Tree LSTM (no tuning)	46.7	86.6
Constituency Tree LSTM	**50.6**	86.9

of word vectors

- Demos: http://nlp.stanford.edu:8080/sentiment/rntnDemo.html

[Yessenalina and Cardie, 2011; Socher et al., 2013b]

Other Classification Tasks

- Sentence similarity
- Entailment classification
- Spam detection
- Document topic classification
- Others: abusive language, hate speech, humor, irony, rumor, sarcasm detection, etc.

SemEval has great new tasks every year with novel datasets in many cases! Some recent years:

http://alt.qcri.org/semeval2019/index.php?id=tasks
http://alt.qcri.org/semeval2018/index.php?id=tasks
http://alt.qcri.org/semeval2017/index.php?id=tasks
http://alt.qcri.org/semeval2016/index.php?id=tasks
http://alt.qcri.org/semeval2015/index.php?id=tasks