4a. Vector-based Semantics
Gerald Penn
Department of Computer Science, University of Toronto

(slides borrowed from Chris Manning)
From symbolic to distributed representations

The vast majority of rule-based and statistical NLP work regarded words as atomic symbols: *hotel*, *conference*, *walk*

In vector space terms, this is a vector with one 1 and a lot of zeroes

\[
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

We call this a “one-hot” representation.

Its problem:

\[
\begin{align*}
\text{motel} & \cdot \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}^T \\
\text{hotel} & \cdot \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} = 0
\end{align*}
\]
Distributional similarity based representations

You can get a lot of value by representing a word by means of its neighbors

“You shall know a word by the company it keeps”

(J. R. Firth 1957: 11)

One of the most successful ideas of modern NLP

government debt problems turning into banking crises as has happened in saying that Europe needs unified banking regulation to replace the hodgepodge

These words will represent banking
With distributed, distributional representations, syntactic and semantic patterning is captured.

\[
\begin{align*}
\text{shown} &= 0.286 \\
&= 0.792 \quad -0.177 \quad -0.107 \quad 0.109 \quad -0.542 \quad 0.349 \quad 0.271
\end{align*}
\]

Synonymy? Hyponymy? Morphology?

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

1. Vector space representations of language

2. Predict! vs. Count!: The GloVe model of word vectors

3. Wanted: meaning composition functions

4. Tree-structured Recursive Neural Networks for Semantics

5. Natural Language Inference with TreeRNNs
LSA vs. word2vec

**LSA: Count!**

- Factorize a (maybe weighted, maybe log scaled) term-document or word-context matrix (Schütze 1992) into $U\Sigma V^T$
- Retain only $k$ singular values, in order to generalize

[Cf. Baroni: Don’t count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. ACL 2014]
LSA vs. word2vec

LSA: **Count!** vs.

word2vec CBOW/SkipGram: **Predict!**

- Train word vectors to try to either:
  - Predict a word given its bag-of-words context (CBOW); or
  - Predict a context word (position-independent) from the center word
- Update word vectors until they can do this prediction well
Word Analogies: word2vec captures dimensions of similarity as linear relations

Test for linear relationships, examined by Mikolov et al. 2013

\[ \text{man:woman :: king:?} \]

\[ a:b :: c:? \]

\[ d = \arg \max_x \frac{(w_b - w_a + w_c)^T w_x}{\|w_b - w_a + w_c\|} \]

\[ \begin{align*}
  \text{king} & \quad \begin{bmatrix} 0.30 & 0.70 \end{bmatrix} \\
  \text{man} & \quad \begin{bmatrix} 0.20 & 0.20 \end{bmatrix} \\
  \text{woman} & \quad \begin{bmatrix} 0.60 & 0.30 \end{bmatrix} \\
  \text{queen} & \quad \begin{bmatrix} 0.70 & 0.80 \end{bmatrix}
\end{align*} \]
COALS model (count-modified LSA) [Rohde, Gonnerman & Plaut, ms., 2005]
<table>
<thead>
<tr>
<th>Analogy</th>
<th>Reported</th>
<th>Index</th>
<th>1st answer</th>
<th>2nd answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miskolov et al. (2013a)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>man king woman</td>
<td>queen</td>
<td>2</td>
<td>king</td>
<td>queen</td>
</tr>
<tr>
<td>Paris France Tokyo</td>
<td>Japan</td>
<td>1</td>
<td>Japan</td>
<td>Tokyo</td>
</tr>
<tr>
<td>brother sister grandson</td>
<td>granddaughter</td>
<td>1</td>
<td>granddaughter</td>
<td>niece</td>
</tr>
<tr>
<td>big bigger cold</td>
<td>colder</td>
<td>2</td>
<td>cold</td>
<td>colder</td>
</tr>
<tr>
<td>Einstein scientist Picasso</td>
<td>painter</td>
<td>1</td>
<td>painter</td>
<td>scientist</td>
</tr>
<tr>
<td>Bolukbasi et al. (2016)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>man computer_programmer woman</td>
<td>homemaker</td>
<td>2</td>
<td>computer_programmer</td>
<td>homemaker</td>
</tr>
<tr>
<td>he doctor he</td>
<td>nurse</td>
<td>2</td>
<td>doctor</td>
<td>nurse</td>
</tr>
<tr>
<td>she interior_designer he</td>
<td>architect</td>
<td>2</td>
<td>interior_designer</td>
<td>architect</td>
</tr>
<tr>
<td>she feminism he</td>
<td>conservatism</td>
<td>4</td>
<td>feminism</td>
<td>liberalism</td>
</tr>
<tr>
<td>she lovely he</td>
<td>brilliant</td>
<td>10</td>
<td>lovely</td>
<td>magnificent</td>
</tr>
<tr>
<td>she sewing he</td>
<td>carpentry</td>
<td>4</td>
<td>sewing</td>
<td>woodworking</td>
</tr>
<tr>
<td>Manzini et al. (2019b)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>black criminal caucasian</td>
<td>lawful</td>
<td>13</td>
<td>legal</td>
<td>statutory</td>
</tr>
<tr>
<td>caucasian lawful black</td>
<td>criminal</td>
<td>2</td>
<td>lawful</td>
<td>criminal</td>
</tr>
<tr>
<td>caucasian hillbilly asian</td>
<td>yuppie</td>
<td>3</td>
<td>hillbilly</td>
<td>hippy</td>
</tr>
<tr>
<td>asian yuppie caucasian</td>
<td>hillbilly</td>
<td>2</td>
<td>yuppie</td>
<td>hillbilly</td>
</tr>
<tr>
<td>asian engineer black</td>
<td>killer</td>
<td>39</td>
<td>operator</td>
<td>jockey</td>
</tr>
<tr>
<td>black killer asian</td>
<td>engineer</td>
<td>7</td>
<td>killer</td>
<td>impostor</td>
</tr>
<tr>
<td>christian conservative jew</td>
<td>liberal</td>
<td>4</td>
<td>centrist</td>
<td>democrat</td>
</tr>
<tr>
<td>jew liberal christian</td>
<td>conservative</td>
<td>2</td>
<td>liberal</td>
<td>conservative</td>
</tr>
<tr>
<td>muslim terrorist jew</td>
<td>journalist</td>
<td>4</td>
<td>hacker</td>
<td>protestor</td>
</tr>
<tr>
<td>jew journalist muslim</td>
<td>terrorist</td>
<td>2</td>
<td>purportedly</td>
<td>terrorist</td>
</tr>
<tr>
<td>christian conservative muslim</td>
<td>regressive</td>
<td>53</td>
<td>moderate</td>
<td>conservative</td>
</tr>
<tr>
<td>muslim regressive christian</td>
<td>conservative</td>
<td>13</td>
<td>regressive</td>
<td>progressive</td>
</tr>
</tbody>
</table>

Count based vs. direct prediction

LSA, HAL (Lund & Burgess), COALS (Rohde et al), Hellinger-PCA (Lebret & Collobert)

- Fast training
- Efficient usage of statistics
- Primarily used to capture word similarity
- Disproportionate importance given to small counts

NNLM, HLBL, RNN, word2vec Skip-gram/CBOW, (Bengio et al; Collobert & Weston; Huang et al; Mnih & Hinton; Mikolov et al; Mnih & Kavukcuoglu)

- Scales with corpus size
- Inefficient usage of statistics
- Generate improved performance on other tasks
- Can capture complex patterns beyond word similarity
**Encoding meaning in vector differences**

[Pennington, Socher, and Manning, EMNLP 2014]

**Crucial insight:** Ratios of co-occurrence probabilities can encode meaning components

\[
\begin{align*}
x &= \text{solid} & x &= \text{gas} & x &= \text{water} & x &= \text{random} \\
\frac{P(x|\text{ice})}{P(x|\text{steam})} &= \begin{array}{cccc}
\text{large} & \text{small} & \text{large} & \text{small} \\
\text{small} & \text{large} & \text{large} & \text{small}
\end{array}
\end{align*}
\]
Encoding meaning in vector differences
[Pennington, Socher, and Manning, EMNLP 2014]

Crucial insight: Ratios of co-occurrence probabilities can encode meaning components

<table>
<thead>
<tr>
<th></th>
<th>$x = \text{solid}$</th>
<th>$x = \text{gas}$</th>
<th>$x = \text{water}$</th>
<th>$x = \text{fashion}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(x</td>
<td>\text{ice})$</td>
<td>$1.9 \times 10^{-4}$</td>
<td>$6.6 \times 10^{-5}$</td>
<td>$3.0 \times 10^{-3}$</td>
</tr>
<tr>
<td>$P(x</td>
<td>\text{steam})$</td>
<td>$2.2 \times 10^{-5}$</td>
<td>$7.8 \times 10^{-4}$</td>
<td>$2.2 \times 10^{-3}$</td>
</tr>
<tr>
<td>$\frac{P(x</td>
<td>\text{ice})}{P(x</td>
<td>\text{steam})}$</td>
<td>$8.9$</td>
<td>$8.5 \times 10^{-2}$</td>
</tr>
</tbody>
</table>
Encoding meaning in vector differences

Q: How can we capture ratios of co-occurrence probabilities as meaning components in a word vector space?

A: Log-bilinear model: \[ w_i \cdot w_j = \log P(i|j) \]
with vector differences \[ w_x \cdot (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)} \]
GloVe: A new model for learning word representations [Pennington et al., EMNLP 2014]

\[ w_i \cdot w_j = \log P(i|j) \]
\[ w_x \cdot (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)} \]

\[ J = \sum_{i,j=1}^{V} f \left( X_{ij} \right) \left( w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij} \right)^2 \]
Word similarities

Nearest words to **frog**:

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

http://nlp.stanford.edu/projects/glove/
Word Analogies

[![Mikolov et al., 2012, 2013](image)]

**Task:** predict the last column

<table>
<thead>
<tr>
<th>Type of relationship</th>
<th>Word Pair 1</th>
<th>Word Pair 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common capital city</td>
<td>Athens</td>
<td>Oslo</td>
</tr>
<tr>
<td>All capital cities</td>
<td>Astana</td>
<td>Harare</td>
</tr>
<tr>
<td>Currency</td>
<td>Angola</td>
<td>Iran</td>
</tr>
<tr>
<td>City-in-state</td>
<td>Chicago</td>
<td>Stockton</td>
</tr>
<tr>
<td>Man-Woman</td>
<td>brother</td>
<td>grandson</td>
</tr>
<tr>
<td>Adjective to adverb</td>
<td>apparently</td>
<td>rapidly</td>
</tr>
<tr>
<td>Opposite</td>
<td>possibly</td>
<td>unethical</td>
</tr>
<tr>
<td>Comparative</td>
<td>great</td>
<td>tougher</td>
</tr>
<tr>
<td>Superlative</td>
<td>easy</td>
<td>luckiest</td>
</tr>
<tr>
<td>Present Participle</td>
<td>think</td>
<td>reading</td>
</tr>
<tr>
<td>Nationality adjective</td>
<td>Switzerland</td>
<td>Cambodia</td>
</tr>
<tr>
<td>Past tense</td>
<td>walking</td>
<td>swam</td>
</tr>
<tr>
<td>Plural nouns</td>
<td>mouse</td>
<td>dollars</td>
</tr>
<tr>
<td>Plural verbs</td>
<td>work</td>
<td>speaks</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Word analogy task [Mikolov, Yih & Zweig 2013a]

<table>
<thead>
<tr>
<th>Model</th>
<th>Dimensions</th>
<th>Corpus size</th>
<th>Performance (Syn + Sem)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBOW (Mikolov et al. 2013b)</td>
<td>300</td>
<td>1.6 billion</td>
<td>36.1</td>
</tr>
</tbody>
</table>
Glove Visualizations

http://nlp.stanford.edu/projects/glove/
Glove Visualizations: Company - CEO
Glove Visualizations: Superlatives
Analogy evaluation and hyperparameters

Figure 4: Overall accuracy on the word analogy task as a function of training time, which is governed by the number of iterations for GloVe and by the number of negative samples for CBOW (a) and skip-gram (b). In all cases, we train 300-dimensional vectors on the same 6B token corpus (Wikipedia 2014 + Gigaword 5) with the same 400,000 word vocabulary, and use a symmetric context window of size 10.

In (a) and (b), we illustrate the impact of model design choices on accuracy. GloVe and skip-gram are two methods for training word representations. GloVe is a global log-bilinear model parameterized by a count matrix that models the co-occurrence statistics of words in a corpus. Skip-gram is a prediction-based model that seeks to maximize the log-likelihood of target words given a vector representation of the context words.

We fix the number of training words and vary the number of iterations for GloVe and the number of negative samples for CBOW and skip-gram. Accuracy is measured on a held-out set of 1,000 analogy pairs. In (a), we observe that GloVe outperforms CBOW across all numbers of iterations. In (b), we see that skip-gram achieves higher accuracy with a larger number of negative samples.

NB! 

Acknowledgments

We thank the anonymous reviewers for their valuable comments. Stanford University gratefully acknowledges the support of the Defense Threat Reduction Agency (DTRA) under Air Force Research Laboratory (AFRL) contract no. FA8650-10-C-7020 and the Defense Advanced Research Projects Agency (DARPA) Deep Exploration and Filtering of Text (DEFT) Program under AFRL contract no. FA8750-13-2-0040. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DTRA, AFRL, DEFT, or the US government.
Analogy evaluation and hyperparameters

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Semantic</th>
<th>Syntactic</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wiki2010 1B tokens</td>
<td>80.5</td>
<td>71.0</td>
<td>75.8</td>
</tr>
<tr>
<td>Wiki2014 1.6B tokens</td>
<td>82.0</td>
<td>72.5</td>
<td>77.2</td>
</tr>
<tr>
<td>Gigaword5 4.3B tokens</td>
<td>83.5</td>
<td>74.0</td>
<td>78.8</td>
</tr>
<tr>
<td>Gigaword5 + Wiki2014 6B tokens</td>
<td>85.0</td>
<td>75.5</td>
<td>79.8</td>
</tr>
<tr>
<td>Common Crawl 42B tokens</td>
<td>86.5</td>
<td>76.0</td>
<td>81.5</td>
</tr>
</tbody>
</table>

Figure 3: Accuracy on the analogy task for 300-dimensional vectors trained on different corpora.

- Wiki2010: 1B tokens
- Wiki2014: 1.6B tokens
- Gigaword5: 4.3B tokens
- Gigaword5 + Wiki2014: 6B tokens
- Common Crawl: 42B tokens

Entries are updated to assimilate new knowledge, whereas Gigaword is a fixed news repository with outdated and possibly incorrect information.

4.6 Model Analysis: Run-time

The total run-time is split between populating $X$ and training the model. The former depends on many factors, including window size, vocabulary size, and corpus size. Though we did not do so, this step could easily be parallelized across multiple machines (see, e.g., Lebret and Collobert (2014) for some benchmarks). Using a single thread of a dual 2.1GHz Intel Xeon E5-2658 machine, populating $X$ with a 10 word symmetric context window, a 400,000 word vocabulary, and a 6 billion token corpus takes about 85 minutes. Given $X$, the time it takes to train the model depends on the vector size and the number of iterations. For 300-dimensional vectors with the above settings (and using all 32 cores of the above machine), a single iteration takes 14 minutes. See Fig. 4 for a plot of the learning curve.

4.7 Model Analysis: Comparison with word2vec

A rigorous quantitative comparison of GloVe with word2vec is complicated by the existence of many parameters that have a strong effect on performance. We control for the main sources of variation that we identified in Sections 4.4 and 4.5 by setting the vector length, context window size, corpus, and vocabulary size to the configuration mentioned in the previous subsection.

The most important remaining variable to control for is training time. For GloVe, the relevant parameter is the number of training iterations. For word2vec, the obvious choice would be the number of training epochs. Unfortunately, the code is currently designed for only a single epoch:
Word Embeddings Conclusion

Developed a model that can translate meaningful relationships between word-word co-occurrence probabilities into linear relations in the word vector space.

GloVe shows the connection between Count! work and Predict! work – appropriate scaling of counts gives the properties and performance of Predict! models.

Can one explain word2vec’s linear structure? See Arora, Li, Liang, Ma, & Risteski. 2015. Random Walks on Context Spaces: Towards an Explanation of the Mysteries of Semantic Word Embeddings. [Develops a generative model.]
Compositionality
Artificial Intelligence requires understanding bigger things from knowing about smaller things.
WE need more! What of larger semantic units?

How can we know when larger units are similar in meaning?

• Two senators received contributions engineered by lobbyist Jack Abramoff in return for political favors.

• Jack Abramoff attempted to bribe two legislators.

People interpret the meaning of larger text units – entities, descriptive terms, facts, arguments, stories – by semantic composition of smaller elements.
Representing Phrases as Vectors

Vector for single words are useful as features but limited!
the country of my birth
the place where I was born

Can we extend the ideas of word vector spaces to phrases?
How should we map phrases into a vector space?

Use the principle of compositionality!

The meaning (vector) of a sentence is determined by
(1) the meanings of its words and
(2) a method that combine them.
\[
\Phi(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}
\]
Basic computational unit: Recursive Neural Network

(Goller & Küchler 1996, Costa et al. 2003, Socher et al. ICML, 2011)
Version 1: Simple concatenation Tree RNN

\[ p = \tanh\left( W \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} + b \right), \]

where \( \tanh:\)

\[ \text{score} = V^T p \]

Only a single weight matrix = composition function!

No really interaction between the input words!

Not adequate for human language composition function
Version 2: PCFG + Syntactically-Untied RNN

- A symbolic Context-Free Grammar (CFG) backbone is adequate for basic syntactic structure
- We use the discrete syntactic categories of the children to choose the composition matrix
- An RNN can do better with a different composition matrix for different syntactic environments
- The result gives us a better semantics
SU-RNN

Learns soft notion of head words

Initialization: \[ W^{(\ast)} = 0.5[I_{n \times n}I_{n \times n}0_{n \times 1}] + \epsilon \]
Version 3: Matrix-vector RNNs
[Socher, Huval, Bhat, Manning, & Ng, 2012]

\[ p = f \left( W \begin{bmatrix} a \\ b \end{bmatrix} \right) \]

\[ p = f \left( W \begin{bmatrix} Ba \\ Ab \end{bmatrix} \right) \]
Version 3: Matrix-vector RNNs
[Socher, Huval, Bhat, Manning, & Ng, 2012]

\[ p = f \left( W \begin{bmatrix} B_a \\ A_b \end{bmatrix} \right) \]

\[ P = g(A, B) = W_M \begin{bmatrix} A \\ B \end{bmatrix} \]

\( W_M \in \mathbb{R}^{n \times 2n} \)
Classification of Semantic Relationships

- Can an MV-RNN learn how a large syntactic context conveys a semantic relationship?

  - My \textit{[apartment]}_{e1} has a pretty large \textit{[kitchen]}_{e2} \rightarrow \text{component-whole relationship (e2,e1)}

- Build a single compositional semantics for the minimal constituent including both terms

\[ \ldots \]

\text{Classifier: Message-Topic}

\[ \ldots \]

\text{the [movie] showed [wars] \ldots} \]
# Classification of Semantic Relationships

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Features</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>POS, stemming, syntactic patterns</td>
<td>60.1</td>
</tr>
<tr>
<td>MaxEnt</td>
<td>POS, WordNet, morphological features, noun compound system, thesauri, Google n-grams</td>
<td>77.6</td>
</tr>
<tr>
<td>SVM</td>
<td>POS, WordNet, prefixes, morphological features, dependency parse features, Levin classes, PropBank, FrameNet, NomLex-Plus, Google n-grams, paraphrases, TextRunner</td>
<td>82.2</td>
</tr>
<tr>
<td>RNN</td>
<td>–</td>
<td>74.8</td>
</tr>
<tr>
<td>MV-RNN</td>
<td>–</td>
<td>79.1</td>
</tr>
<tr>
<td>MV-RNN</td>
<td>POS, WordNet, NER</td>
<td>82.4</td>
</tr>
</tbody>
</table>
Version 4: Recursive Neural Tensor Network

- Less parameters than MV-RNN
- Allows the two word or phrase vectors to interact multiplicatively
Version 4: Recursive Neural Tensor Network

- Idea: Allow both additive and mediated multiplicative interactions of vectors
Recursive Neural Tensor Network

\[
\begin{pmatrix}
\mathbf{b}^T \\
\mathbf{c}
\end{pmatrix} \mathbf{V}^{[1:2]} \begin{pmatrix}
\mathbf{b} \\
\mathbf{c}
\end{pmatrix}
\]
Recursive Neural Tensor Network

\[
\begin{align*}
\mathbf{b}^T V^{[1:2]} \mathbf{b} + W \mathbf{b}
\end{align*}
\]
Recursive Neural Tensor Network

- Use resulting vectors in tree as input to a classifier like logistic regression
- Train all weights jointly with gradient descent
Version 5: Improving Deep Learning Semantic Representations using a TreeLSTM

[Tai et al., ACL 2015]

Goals:

- Still trying to represent the meaning of a sentence as a location in a (high-dimensional, continuous) vector space
- In a way that accurately handles semantic composition and sentence meaning
- Beat Paragraph Vector!
Tree-Structured Long Short-Term Memory Networks

Use Long Short-Term Memories (Hochreiter and Schmidhuber 1997)

Use syntactic structure
• An LSTM creates a sentence representation via *left-to-right composition*
• Natural language has syntactic structure
• We can use this *additional structure* over inputs to guide how representations should be composed
Tree-Structured Long Short-Term Memory Networks

[Tai et al., ACL 2015]
## Results: Semantic Relatedness

**SICK 2014** (Sentences Involving Compositional Knowledge)

<table>
<thead>
<tr>
<th>Method</th>
<th>Pearson correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meaning Factory (Bjerva et al. 2014)</td>
<td>0.827</td>
</tr>
<tr>
<td>ECNU (Zhao et al. 2014)</td>
<td>0.841</td>
</tr>
<tr>
<td>LSTM (sequence model)</td>
<td>0.853</td>
</tr>
<tr>
<td>Tree LSTM</td>
<td><strong>0.868</strong></td>
</tr>
</tbody>
</table>
Natural Language Inference

Can we tell if one piece of text follows from another?

• *Two senators received contributions engineered by lobbyist Jack Abramoff in return for political favors.*

• *Jack Abramoff attempted to bribe two legislators.*

Natural Language Inference = Recognizing Textual Entailment [Dagan 2005, MacCartney & Manning, 2009]
The task: Natural language inference

James Byron Dean refused to move without blue jeans

{entails, contradicts, neither}

James Dean didn’t dance without pants
MacCartney’s natural logic

An implementable logic for natural language inference without logical forms. (MacCartney and Manning ‘09)

- Sound logical interpretation (Icard and Moss ‘13)

<table>
<thead>
<tr>
<th>p</th>
<th>James Dean refused to move without blue jeans</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>James Byron did n't dance without pants</td>
</tr>
<tr>
<td>edit index</td>
<td>1 2 3 4 5 6 7 8</td>
</tr>
<tr>
<td>edit type</td>
<td>SUB DEL INS INS SUB MAT DEL SUB</td>
</tr>
<tr>
<td>lex feats</td>
<td>smsim 0.67 implic: -/o cat:aux cat:neg hypo hyper</td>
</tr>
<tr>
<td>lex entrel</td>
<td>=</td>
</tr>
<tr>
<td>projectivity</td>
<td>↑</td>
</tr>
<tr>
<td>atomic entrel</td>
<td>=</td>
</tr>
</tbody>
</table>

inversion
The task: Natural language inference

Claim: Simple task to define, but engages the full complexity of compositional semantics:

- Lexical entailment
- Quantification
- Coreference
- Lexical/scope ambiguity
- Commonsense knowledge
- Propositional attitudes
- Modality
- Factivity and implicativity
...
## Natural logic: relations

Seven possible relations between phrases/sentences:

<table>
<thead>
<tr>
<th>Relation</th>
<th>Symbol</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equivalence</td>
<td>$x \equiv y$</td>
<td>$couch \equiv sofa$</td>
</tr>
<tr>
<td>Forward entailment (strict)</td>
<td>$x \sqsubseteq y$</td>
<td>$crow \sqsubseteq bird$</td>
</tr>
<tr>
<td>Reverse entailment (strict)</td>
<td>$x \sqsupseteq y$</td>
<td>$European \sqsupseteq French$</td>
</tr>
<tr>
<td>Negation (exhaustive exclusion)</td>
<td>$x \neg y$</td>
<td>$human \neg nonhuman$</td>
</tr>
<tr>
<td>Alternation (non-exhaustive exclusion)</td>
<td>$x \mid y$</td>
<td>$cat \mid dog$</td>
</tr>
<tr>
<td>Cover (exhaustive non-exclusion)</td>
<td>$x \curvearrowright y$</td>
<td>$animal \curvearrowright nonhuman$</td>
</tr>
<tr>
<td>Independence</td>
<td>$x # y$</td>
<td>$hungry # hippo$</td>
</tr>
</tbody>
</table>
Natural logic: relation joins

Can our NNs learn to make these inferences over pairs of embedding vectors?
A minimal NN for lexical relations
[Bowman 2014]

- Words are learned embedding vectors.
- One plain TreeRNN or TreeRNTN layer
- Softmax emits relation labels
- Learn everything with SGD.

\[ P(\text{entailment}) = 0.9 \]

Comparison N(T)N layer

Softmax classifier

\text{jeans vs. pants}

\text{jeans} \quad \text{pants}

Learned word vectors
Lexical relations: results

<table>
<thead>
<tr>
<th></th>
<th>Train</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td># only</td>
<td>53.8 (10.5)</td>
<td>53.8 (10.5)</td>
</tr>
<tr>
<td>15d NN</td>
<td>99.8 (99.0)</td>
<td>94.0 (87.0)</td>
</tr>
<tr>
<td>15d NTN</td>
<td><strong>100 (100)</strong></td>
<td><strong>99.6 (95.5)</strong></td>
</tr>
</tbody>
</table>

- Both models tuned, then trained to convergence on five randomly generated datasets
- Reported figures: % correct (macroaveraged F1)
- Both NNs used 15d embeddings, 75d comparison layer
Quantifiers

Experimental paradigm: Train on relational statements generated from some formal system, test on other such relational statements.

The model needs to:

● Learn the relations between individual words. (lexical relations)
● Learn how lexical relations impact phrasal relations. (projectivity)
● Quantifiers present some of the harder cases of both of these.
Quantifiers

- Small vocabulary
  - Three basic types:
    - Quantifiers: *some*, *all*, *no*, *most*, *two*, *three*, *not-all*, *not-most*,
      *less-than-two*, *less-than-three*
    - Predicates: *dog*, *cat*, *mammal*, *animal* ...
    - Negation: *not*

- 60k examples generated using a generative implementation of the relevant portion of MacCartney and Manning’s logic.

- All sentences of the form $QPP$, with optional negation on each predicate.

$$(\text{most warthogs}) \text{ walk} \quad \wedge \quad (\text{not}-\text{most warthogs}) \text{ walk}$$

$$(\text{most mammals}) \text{ move} \quad \# \quad (\text{not-most (not turtles)}) \text{ move}$$

$$(\text{most (not pets)}) \text{ (not swim)} \quad \Box \quad (\text{not-most (not pets)}) \text{ move}$$
Quantifier results

<table>
<thead>
<tr>
<th>Model</th>
<th>Train</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Most freq. class (# only)</td>
<td>35.4%</td>
<td>35.4%</td>
</tr>
<tr>
<td>25d SumNN (sum of words)</td>
<td>96.9%</td>
<td>93.9%</td>
</tr>
<tr>
<td>25d TreeRNN</td>
<td>99.6%</td>
<td>99.2%</td>
</tr>
<tr>
<td>25d TreeRNTN</td>
<td>100%</td>
<td>99.7%</td>
</tr>
</tbody>
</table>
Natural language inference data
[Bowman, Manning & Potts 2015]

● To do NLI on real English, we need to teach an NN model English almost from scratch.
● What data do we have to work with:
  ○ GloVe/word2vec (useful w/ any data source)
  ○ SICK: Thousands of examples created by editing and pairing hundreds of sentences.
  ○ RTE: Hundreds of examples created by hand.
  ○ DenotationGraph: Millions of extremely noisy examples (~73% correct?) constructed fully automatically.
## Results on SICK (+DG, +tricks) so far

<table>
<thead>
<tr>
<th></th>
<th>SICK Train</th>
<th>DG Train</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Most freq. class</strong></td>
<td>56.7%</td>
<td>50.0%</td>
<td>56.7%</td>
</tr>
<tr>
<td><strong>30 dim TreeRNN</strong></td>
<td>95.4%</td>
<td>67.0%</td>
<td>74.9%</td>
</tr>
<tr>
<td><strong>50 dim TreeRNTN</strong></td>
<td>97.8%</td>
<td>74.0%</td>
<td>76.9%</td>
</tr>
</tbody>
</table>
Is it competitive? Sort of...

Best result (Ullinois) ≈ interannotator agreement!
Median submission (out of 18):
  TreeRNTN: 76.9%

TreeRNTN is a purely-learned system
None of the ones in the competition were
Natural language inference data

- To do NLI on real English, we need to teach an NN model English almost from scratch.
- What data do we have to work with:
  - GloVe/word2vec (useful w/ any data source)
  - SICK: Thousands of examples created by editing and pairing hundreds of sentences.
  - RTE: Hundreds of examples created by hand.
  - DenotationGraph: Millions of extremely noisy examples (~73% correct?) constructed fully automatically.
  - Stanford NLI corpus: ~600k examples, written by Turkers.
The Stanford NLI corpus

Instructions

The Stanford University NLP Group is collecting data for use in research on computer understanding of English. We appreciate your help!

We will show you the caption for a photo. We will not show you the photo. Using only the caption and what you know about the world:

- Write one alternate caption that is definitely a true description of the photo.
- Write one alternate caption that might be a true description of the photo.
- Write one alternate caption that is definitely an false description of the photo.

Photo caption A little boy in an apron helps his mother cook.

Definitely correct Example: For the caption "Two dogs are running through a field." you could write "There are animals outdoors."

Write a sentence that follows from the given caption.

Maybe correct Example: For the caption "Two dogs are running through a field." you could write "Some puppies are running to catch a stick."

Write a sentence which may be true given the caption, and may not be.

Definitely incorrect Example: For the caption "Two dogs are running through a field." you could write "The pets are sitting on a couch."

Write a sentence which contradicts the caption.

Problems (optional) If something is wrong with the caption that makes it difficult to understand, do your best above and let us know here.
Envoi

There are very good reasons to want to represent meaning with distributed representations

So far, distributional learning has been most effective for this

But cf. [Young, Lai, Hodosh & Hockenmaier 2014] on denotational representations, using visual scenes

However, we want not just word meanings! We want:

Meanings of larger units, calculated compositionally

The ability to do natural language inference