Outline

- Properties of language
- Distributional semantics
- Frame semantics
- Model-theoretic semantics
Properties of language

- Analyses: syntax, semantics, pragmatics

**Syntax**: what is grammatical?

**Semantics**: what does it mean?

**Pragmatics**: what does it do?

For coders:

**Syntax**: no compiler errors

**Semantics**: no implementation bugs

**Pragmatics**: implemented the right algorithm
Properties of language

- Lexical semantics: synonymy, hyponymy/meronymy

Hyponymy (is-a):

a cat is a mammal

Meronomy (has-a):

a cat has a tail
Properties of language

- Challenges: polysemy, vagueness, ambiguity, uncertainty

Vagueness: does not specify full information

I had a late lunch.

Ambiguity: more than one possible (precise) interpretations

One morning I shot an elephant in my pajamas.

How he got in my pajamas, I don’t know. —— Groucho Marx

Uncertainty: due to an imperfect statistical model

The witness was being contumacious.
Outline

- Properties of language
- Distributional semantics
- Frame semantics
- Model-theoretic semantics
Distributional semantics

**Premise:** semantics = context of word/phrase

**Recipe:** form word-context matrix + dimensionality reduction

![Diagram of word-context matrix]

**Models:** Latent semantic analysis, Word2vec (Recall last talk)
Outline

- Properties of language
- Distributional semantics
- Frame semantics
- Model-theoretic semantics
Frame semantics

Distributional semantics: all the contexts in which *sold* occurs

..was sold by... ...sold me that piece of

Can find similar words/contexts and generalize (dimensionality reduction), but no internal structure on word vectors

Frames: meaning given by a frame, a stereotypical situation

- **Commercial transaction**
  - SELLER : ?
  - BUYER : ?
  - GOODS : ?
  - PRICE : ?
Frame semantics

Semantic role labeling (FrameNet, PropBank):

Task:

Input:  
Output:  

Subtasks:

1. Frame identification (PREDICATE)
   [Hermann/Das/Weston/Ganchev, 2014]

2. Argument identification (SELLER, GOODS, etc.)
   [Punyakanok/Roth/Yih, 2008; Tackstrom/Ganchev/Das, 2015]
Frame semantics

Abstract meaning representation (AMR)
[Banarescu et al., 2013]
[Flanigan/Thomson/Carbonell/Dyer/Smith, 2014]

Motivation of AMR: unify all semantic annotation
Semantic role labeling
Named-entity recognition
Coreference resolution
Frame semantics-AMR parsing task

**Input**: sentence

*The boy wants to go to New York City.*

**Output**: graph
Frame semantics

- Both distributional semantics (DS) and frame semantics (FS) involve compression/abstraction

- Frame semantics exposes more structure, more tied to an external world, but requires more supervision
Outline

- Properties of language
- Distributional semantics
- Frame semantics
- Model-theoretic semantics
Model-theoretic semantics

**Every** non-blue block is next to **some** blue block.

Distributional semantics: **block** is like **brick**, **some** is like **every**

Frame semantics: **is next to** has two arguments, **block** and **block**

Model-theoretic semantics: tell the difference between
Model-theoretic semantics

Framework: map natural language into logical forms

Factorization: understanding and knowing

*What is the largest city in California?*

\[
\text{argmax}(\lambda x. \text{city}(x) \land \text{loc}(x, \text{CA}), \lambda x. \text{population}(x))
\]

Applications: question answering, natural language interfaces to robots, programming by natural language
Sequence-to-Sequence Learning and Attention Model

Slides are from Kyunghyun Cho, Dzmitry Bahdanau
MACHINE TRANSLATION

Topics: Statistical Machine Translation

\[ \log p(f|e) = \log p(e|f) + \log p(f) \]

- Language Model
  - \( \log p(f) \)
- Translation Model
  - \( \log p(e|f) \)
- Decoding Algorithm
  - given a language model, a translation model and a new sentence \( e \), find translation \( f \) maximizing

\[ \log p(f|e) = \log p(e|f) + \log p(f) \]

The whole task is conditional language modelling.
NEURAL MACHINE TRANSLATION

(Forcada & Ñeco, 1997;
Castaño & Casacuberta, 1997;
Kalchbrenner & Blunsom, 2013;
Sutskever et al., 2014;
Cho et al., 2014)
Sequence-to-Sequence Learning — Encoder

- Encoder
  - 1-of-k
  - Continuous-space representation
    - \( s_{t'} = W^T x_{t'} \), where \( W \in \mathbb{R}^{|V| \times d} \)
  - Recursively read words
    - \( h_t = f(h_{t-1}, s_t) \), for \( t = 1, \ldots, T \)
Sequence-to-Sequence Learning — Encoder

- Encoder

\[ e = (\text{Economic, growth, has, slowed, down, in, recent, years, .}) \]
Sequence-to-Sequence Learning — Decoder

- Decoder
  - Recursively update the memory
    - $z_{t'} = f(z_{t' - 1}, u_{t' - 1}, b_1)$
  - Compute the next word prob
    - $p(u_{t'} | u_{< t'}) \propto \exp(R_{u_{t'}, z_{t'} + b_{u_{t'}}})$
  - Sample a next word
    - Beam search is a good idea
Sequence-to-Sequence Learning — Decoder

\[ f = (La, croissance, \text{économique, s'est, ralentie, ces, dernières, années, .}) \]

\[ u_i \]

Word Sample

\[ p_i \]

Word Probability

\[ z_i \]

Recurrent State

\[ (1) \]

\[ (2) \]

\[ (3) \]

\[ e = (\text{Economic, growth, has, slowed, down, in, recent, years, .}) \]
RNN Encoder-Decoder: Issues

- has to remember the whole sentence
- fixed size representation can be the bottleneck
- humans do it differently
Key Idea of Attention (D Bahdanau et.al, ICLR 2015)

Tell Decoder what is now translated:

The agreement on European Economic Area was signed in August 1992.

L'accord sur ????

L'accord sur l'Espace économique européen a été signé en ???

Have such hints computed by the net itself!
New Encoder

Bidirectional RNN: $h_j$ contains $x_j$ together with its context ($..., x_{j-1}, x_{j+1}, ...$).

$(h_1, ..., h_L)$ is the new *variable-length* representation instead of *fixed-length* $c$. 
New Decoder

Step i:

- Compute alignment
- Compute context
- Generate new output
- Compute new decoder state
Alignment Model

\[ e_{ij} = v^T \tanh(Ws_{i-1} + Vh_j) \]
\[ \alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{L} \exp(e_{ik})} \]

(1) \hspace{1cm} \hspace{1cm} (2)

nonlinearity (tanh) is crucial!

simplest model possible
Experiment: English to French

Model:

- RNN Search, 1000 units

Baseline:

- RNN Encoder-Decoder, 1000 units
- Moses, a SMT system (Koehn et al. 2007)

Data:

- English to French translation, 348 million words,
- 30000 words + UNK token for the networks, all words for Moses

Training:

- Minimize mean log P(y|x,θ) w.r. θ
- log P(y|x,θ) is differentiable w.r. θ => usual methods
Quantitative Results

- no performance drop on long sentences
- much better than RNN Encoder-Decoder

<table>
<thead>
<tr>
<th>Model</th>
<th>All</th>
<th>No UNK</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNNencdec-30</td>
<td>13.93</td>
<td>24.19</td>
</tr>
<tr>
<td>RNNsearch-30</td>
<td>21.50</td>
<td>31.44</td>
</tr>
<tr>
<td>RNNencdec-50</td>
<td>17.82</td>
<td>26.71</td>
</tr>
<tr>
<td>RNNsearch-50</td>
<td>26.75</td>
<td>34.16</td>
</tr>
<tr>
<td>RNNsearch-50*</td>
<td>28.45</td>
<td>36.15</td>
</tr>
<tr>
<td>Moses</td>
<td>33.30</td>
<td>35.63</td>
</tr>
</tbody>
</table>

without unknown words comparable with the SMT system
Qualitative Results: Alignment

The agreement on the European Economic Area was signed in August 1992.


It is known, that the verb often occupies the last position in German sentences

Es ist bekannt, dass das Verb oft die letzte Position in deutschen Strafen einnimmt.
Still Some Issue...

- Very large target vocabulary (Jean et al., 2015)
- Subword-level Machine Translation (Sennrich et al., 2015)
- Incorporating Target Language Model (Gulcehre&Firat et al., 2015)

  - Recall: $\log p(f|e) = \log p(e|f) + \log p(f)$

- ...
Even Beyond Natural Languages

Image Caption Generation

\[
p(\text{Two, dolphins, are, diving}) = ?
\]

- **Encoder**: convolutional network
  - Pretrained as a classifier or autoencoder
- **Decoder**: recurrent neural network
  - RNN Language model
  - With attention mechanism
    - (Xu et al., 2015)
Image Caption Generation (Examples)
Memory Network

Slides are from Jiasen Lu and Jason Weston


Memory Networks

• Class of models that combine large memory with learning component that can read and write to it.

• Most ML has limited memory which is more-or-less all that’s needed for “low level” tasks e.g. object detection.

• **Motivation**: long-term memory is required to read a story (or watch a movie) and then e.g. answer questions about it.

• We study this by building a simple simulation to generate `stories`. We also try on some real QA data
James the Turtle was always getting in trouble. Sometimes he'd reach into the freezer and empty out all the food. Other times he'd sled on the deck and get a splinter. His aunt Jane tried as hard as she could to keep him out of trouble, but he was sneaky and got into lots of trouble behind her back.

One day, James thought he would go into town and see what kind of trouble he could get into. He went to the grocery store and pulled all the pudding off the shelves and ate two jars. Then he walked to the fast food restaurant and ordered 15 bags of fries. He didn't pay, and instead headed home.

His aunt was waiting for him in his room. She told James that she loved him, but he would have to start acting like a well-behaved turtle.

After about a month, and after getting into lots of trouble, James finally made up his mind to be a better turtle.

Q: What did James pull off of the shelves in the grocery store?

A) pudding  B) fries  C) food  D) splinters
James the Turtle was always getting in trouble. Sometimes he'd reach into the freezer and empty out all the food. Other times he'd sled on the deck and get a splinter. His aunt Jane tried as hard as she could to keep him out of trouble, but he was sneaky and got into lots of trouble behind her back.

One day, James thought he would go into town and see what kind of trouble he could get into. He went to the grocery store and pulled all the pudding off the shelves and ate two jars. Then he walked to the fast food restaurant and ordered 15 bags of fries. He didn't pay, and instead headed home.

His aunt was waiting for him in his room. She told James that she loved him, but he would have to start acting like a well-behaved turtle.

After about a month, and after getting into lots of trouble, James finally made up his mind to be a better turtle.

Q: What did James pull off of the shelves in the grocery store?
A) pudding B) fries C) food D) splinters

Q: Where did James go after he went to the grocery store?

...
Example

Dataset in simulation command format. Dataset after adding a simple grammar.

- antoine go kitchen
- antoine get milk
- antoine go office
- antoine drop milk
- antoine go bathroom
- where is milk? (A: office)
- where is antoine? (A: bathroom)

- Antoine went to the kitchen.
- Antoine picked up the milk.
- Antoine travelled to the office.
- Antoine left the milk there.
- Antoine went to the bathroom.
- Where is the milk now? (A: office)
- Where is Antoine? (A: bathroom)
Simulation Data Generation

**Aim:** built a simple simulation which behaves much like a classic text adventure game. The idea is that generating text within this simulation allows us to ground the language used.

**Actions:**

`go <location>`, `get <object>`, `get <object1> from <object2>`, `put <object1> in/on <object2>`, `give <object> to <actor>`, `drop <object>`, `look`, `inventory`, `examine <object>`.

**Constraints on actions:**

- an actor cannot get something that they or someone else already has
- they cannot go to a place they are already at
- cannot drop something they do not already have
- ...
(1) Factoid QA with Single Supporting Fact

John is in the playground.
Bob is in the office.
Where is John? A: playground

(2) Factoid QA with Two Supporting Facts

John is in the playground.
Bob is in the office.
John picked up the football.
Bob went to the kitchen.
Where is the football? A: playground
Where was Bob before the kitchen? A: office

... (total 20 Tasks)
Memory Networks

MemNNs have four component networks (which may or may not have shared parameters):

**I**: (input feature map) this converts incoming data to the internal feature representation.

**G**: (generalization) this updates memories given new input.

**O**: this produces new output (in feature representation space) given the memories.

**R**: (response) converts output O into a response seen by the outside world.

This process is applied both train and test time, only difference is model parameter I, G, O and R are not update during test time.
**Basic Model (Weston et.al. "Memory networks.")**

**I:** (input feature map) no conversion, keep original text $x$.

**G:** (generalization) stores $I(x)$ in next available slot $m_N$

**O:** Loops over all memories $k=1$ or 2 times:

- 1st loop max: finds best match $m_i$ with $x$.
- 2nd loop max: finds best match $m_j$ with $(x, m_i)$.
- The output $o$ is represented with $(x, m_i, m_j)$.

**R:** (response) ranks all words in the dictionary given $o$ and returns best single word. (OR: use a full RNN here)

RNN: $[x, o1, o2, \ldots, r]$ feed into RNN, Test time: $[x, o1, o2, \ldots]$
Matching function

- For a given Q, we want a good match to the relevant memory slot(s) containing the answer, e.g.:

  Match (Where is the football ?, John picked up the football)

- We use a $q^T U^T U_d$ embedding model with word embedding features.
  - LHS features: $Q$:Where $Q$:is $Q$:the $Q$:football $Q$:
  - RHS features: $D$:John $D$:picked $D$:up $D$:the $D$:football
  - QDMatch:the QDMatch:football

(QDMatch:football is a feature to say there’s a Q&A word match, which can help.)

The parameters U are trained with a margin ranking loss: supporting facts should score higher than non-supporting facts.
Matching function: 2nd hop

• On the 2nd hop we match question & 1st hop to new fact:

Match( [Where is the football ?, John picked up the football],
      John is in the playground)

• We use the same $q^T U^T U_d$ embedding model:
  • LHS features: 
    Q:Where Q:is Q:the Q:football Q:? Q2: John Q2:picked Q2:up Q2:the Q2:football
  • RHS features: 
    D:John D:is D:in D:the D:playground QDMatch:the QDMatch:is ..Q2DMatch:John

• We also need time information for bAbI simulation.

We tried adding absolute time differences (between two memories) as a feature: tricky to get to work.
Some Extensions

Some options and extensions:

- **Efficient Memory Via Hashing**
  - Hashing word
    - Memory will be considered if only share at least one word
  - **Clustering word embedding**
    - Run K-means to cluster word vectors $U$, given $K$ buckets
    - Hash a given sentence into all the buckets that it’s individual words falls into.

- **Modeling Previous Unseen words**
  - Store bag of words it has co-occurred with.
  - Increase the feature representation $D$ from $3|W|$ to $5|W|$.
  - Using kind of the dropout technique.
Results: QA on Reverb data from (Fader et al.)

- 14M statements stored in the memNN memory.
- k=1 loops MemNN, 128-dim embedding.
- R response simply outputs top scoring statement.
- Time features are not necessary, hence not used.
- We also tried adding bag of words (BoW) features.

<table>
<thead>
<tr>
<th>Method</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Fader et al., 2013)</td>
<td>0.54</td>
</tr>
<tr>
<td>(Bordes et al., 2014)</td>
<td>0.73</td>
</tr>
<tr>
<td>MemNN</td>
<td>0.72</td>
</tr>
<tr>
<td>MemNN (with BoW features)</td>
<td>0.82</td>
</tr>
</tbody>
</table>
Results: QA on Reverb data from (Fader et al.)

Scoring all 14M candidates in the memory is slow.

We consider speedups using hashing in S and O as mentioned earlier:

- Hashing via words (essentially: inverted index)
- Hashing via k-means in embedding space (k=1000)

<table>
<thead>
<tr>
<th>Method</th>
<th>Embedding</th>
<th>Embed+BoW</th>
<th>candidates</th>
</tr>
</thead>
<tbody>
<tr>
<td>MemNN (no hashing)</td>
<td>0.72</td>
<td>0.82</td>
<td>14M</td>
</tr>
<tr>
<td>MemNN (word hash)</td>
<td>0.63</td>
<td>0.68</td>
<td>13k (1000x)</td>
</tr>
<tr>
<td>MemNN (clust hash)</td>
<td>0.71</td>
<td>0.80</td>
<td>177k (80x)</td>
</tr>
</tbody>
</table>
bAbI Experiment

10k sentences. (Actor: only ask questions about actors.)
- Difficulty: how many sentences in the past when entity mentioned.
- Fully supervised (supporting sentences are labeled).
- Compare RNN (no supervision) and MemNN hops $k = 1$ or $2$, & with/without time features.

Table 3: Test accuracy on the simulation QA task.

<table>
<thead>
<tr>
<th>Method</th>
<th>Difficulty 1</th>
<th></th>
<th>Difficulty 5</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>actor w/o before</td>
<td>actor</td>
<td>actor+object</td>
<td>actor</td>
</tr>
<tr>
<td>RNN</td>
<td>100%</td>
<td>60.9%</td>
<td>27.9%</td>
<td>23.8%</td>
</tr>
<tr>
<td>LSTM</td>
<td>100%</td>
<td>64.8%</td>
<td>49.1%</td>
<td>35.2%</td>
</tr>
<tr>
<td>MemNN $k = 1$</td>
<td>97.8%</td>
<td>31.0%</td>
<td>24.0%</td>
<td>21.9%</td>
</tr>
<tr>
<td>MemNN $k = 1$ (+time)</td>
<td>99.9%</td>
<td>60.2%</td>
<td>42.5%</td>
<td>60.8%</td>
</tr>
<tr>
<td>MemNN $k = 2$ (+time)</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>
End-to-end Memory networks
"End-to-end memory networks." Sukhbaatar et.al.

Problem of Memory Network:
• not easy to train via back propagation
• required supervision at each layer of the network.

Continuous form of memory network
• Attention paper:
  • RNN search:
  • Show attention and tell:
Model-single layer

Embedding Parameter: \( A, B, C, W \)

Input memory representation

\[ p_i = \text{Softmax}(u^T m_i) \]

Output memory representation

\[ o = \sum_i p_i c_i. \]

Generating the final prediction

\[ \hat{a} = \text{Softmax}(W(o + u)) \]

Memory size 50

Slide credit: Reference paper
Model - multiple layers

Weight type

1: Adjacent:
- \( A^{k+1} = C^k \)
- \( W^T = C^K \) (final output embedding)
- \( B = A^1 \)

2: Layer-Wise
- \( A^1 = A^2 = \ldots \)
- \( u^{k+1} = Hu^k + o^k \)
Some extensions

1. **Sentence Representation (PE)**
   \[ c_i = \sum_j A x_{ij} \quad \Rightarrow \quad m_i = \sum_j l_j \cdot A x_{ij} \quad l_{kj} = (1 - j / J) - (k / d)(1 - 2 j / J) \]

2. **Temporal Encoding**
   \[ m_i = \sum_j A x_{ij} + T_A(i) \quad T_A(i) : \text{Special matrix encode temporal info} \]

3. **Inject Random noise (RN)**
   random add 10% of the empty memories

4. **Linear Start (LS)**
   initial train with remove all the non-linear without the final softmax
### Experiment – Synthetic QA

<table>
<thead>
<tr>
<th>Task</th>
<th>Baseline</th>
<th>MemN2N</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Strongly Supervised</td>
<td>LSTM</td>
</tr>
<tr>
<td>1: 1 supporting fact</td>
<td>0.0</td>
<td>50.0</td>
</tr>
<tr>
<td>2: 2 supporting facts</td>
<td>0.0</td>
<td>80.0</td>
</tr>
<tr>
<td>3: 3 supporting facts</td>
<td>0.0</td>
<td>80.0</td>
</tr>
<tr>
<td>4: 2 argument relations</td>
<td>0.0</td>
<td>39.0</td>
</tr>
<tr>
<td>5: 3 argument relations</td>
<td>2.0</td>
<td>30.0</td>
</tr>
<tr>
<td>6: yes/no questions</td>
<td>0.0</td>
<td>52.0</td>
</tr>
<tr>
<td>7: counting</td>
<td>15.0</td>
<td>51.0</td>
</tr>
<tr>
<td>8: lists/sets</td>
<td>9.0</td>
<td>55.0</td>
</tr>
<tr>
<td>10: indefinite knowledge</td>
<td>0.0</td>
<td>36.0</td>
</tr>
<tr>
<td>11: basic coreference</td>
<td>0.0</td>
<td>38.0</td>
</tr>
<tr>
<td>12: conjunction</td>
<td>0.0</td>
<td>26.0</td>
</tr>
<tr>
<td>13: compound coreference</td>
<td>0.0</td>
<td>6.0</td>
</tr>
<tr>
<td>14: time reasoning</td>
<td>1.0</td>
<td>73.0</td>
</tr>
<tr>
<td>15: basic deduction</td>
<td>0.0</td>
<td>79.0</td>
</tr>
<tr>
<td>16: basic induction</td>
<td>0.0</td>
<td>77.0</td>
</tr>
<tr>
<td>17: positional reasoning</td>
<td>35.0</td>
<td>49.0</td>
</tr>
<tr>
<td>18: size reasoning</td>
<td>5.0</td>
<td>48.0</td>
</tr>
<tr>
<td>19: path finding</td>
<td>64.0</td>
<td>92.0</td>
</tr>
<tr>
<td>20: agent’s motivation</td>
<td>0.0</td>
<td>9.0</td>
</tr>
</tbody>
</table>

| Mean error (%)                    | 6.7      | 51.3   | 40.2  | 25.1| 20.3 | 16.3| 13.9| 25.8| 15.6  | 13.3   | 12.4   | 15.2| 15.2| 15.2| 15.2|
| Failed tasks (err. > 5%)          | 4        | 20     | 18    | 15  | 13    | 12 | 11 | 17 | 11    | 11     | 11     | 10 | 10 | 10 | 10 |

| On 10k training data              |          |        |       |     |      |    |    |    |      |       |       |    |    |    |    |
| Mean error (%)                    | 3.2      | 36.4   | 39.2  | 15.4| 9.4   | 7.2 | 6.6 | 24.5| 10.9  | 7.9    | 7.5    | 11.0| 11.0| 11.0| 11.0|
| Failed tasks (err. > 5%)          | 2        | 16     | 17    | 9   | 6     | 4  | 4  | 16 | 7     | 6      | 6      | 6   | 6  | 6  | 6  |

Close to MemNN and beat weakly supervised baseline (MemNN WSH)

Slide credit: Reference paper
Figure 2: Example predictions on the QA tasks of [21]. We show the labeled supporting facts (support) from the dataset which MemN2N does not use during training, and the probabilities $p$ of each hop used by the model during inference. MemN2N successfully learns to focus on the correct supporting sentences.
## Experiment – Language Modeling

<table>
<thead>
<tr>
<th>Model</th>
<th># of hidden</th>
<th># of hops</th>
<th>memory size</th>
<th>Valid. perp.</th>
<th>Test perp.</th>
<th># of hidden</th>
<th># of hops</th>
<th>memory size</th>
<th>Valid. perp.</th>
<th>Test perp.</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN [15]</td>
<td>300</td>
<td>-</td>
<td>-</td>
<td>133</td>
<td>129</td>
<td>500</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>184</td>
</tr>
<tr>
<td>LSTM [15]</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>120</td>
<td>115</td>
<td>500</td>
<td>-</td>
<td>-</td>
<td>122</td>
<td>154</td>
</tr>
<tr>
<td>SCRN [15]</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>120</td>
<td>115</td>
<td>500</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>161</td>
</tr>
<tr>
<td>MemN2N</td>
<td>150</td>
<td>2</td>
<td>100</td>
<td>128</td>
<td>121</td>
<td>500</td>
<td>2</td>
<td>100</td>
<td>152</td>
<td>187</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>3</td>
<td>100</td>
<td>129</td>
<td>122</td>
<td>500</td>
<td>3</td>
<td>100</td>
<td>142</td>
<td>178</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>4</td>
<td>100</td>
<td>127</td>
<td>120</td>
<td>500</td>
<td>4</td>
<td>100</td>
<td>129</td>
<td>162</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>5</td>
<td>100</td>
<td>127</td>
<td>118</td>
<td>500</td>
<td>5</td>
<td>100</td>
<td>123</td>
<td>154</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>6</td>
<td>100</td>
<td>122</td>
<td>115</td>
<td>500</td>
<td>6</td>
<td>100</td>
<td>124</td>
<td>155</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>7</td>
<td>100</td>
<td>120</td>
<td>114</td>
<td>500</td>
<td>7</td>
<td>100</td>
<td>118</td>
<td>147</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>6</td>
<td>25</td>
<td>125</td>
<td>118</td>
<td>500</td>
<td>6</td>
<td>25</td>
<td>131</td>
<td>163</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>6</td>
<td>50</td>
<td>121</td>
<td>114</td>
<td>500</td>
<td>6</td>
<td>50</td>
<td>132</td>
<td>166</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>6</td>
<td>75</td>
<td>122</td>
<td>114</td>
<td>500</td>
<td>6</td>
<td>75</td>
<td>126</td>
<td>158</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>6</td>
<td>100</td>
<td>122</td>
<td>115</td>
<td>500</td>
<td>6</td>
<td>100</td>
<td>124</td>
<td>155</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>6</td>
<td>125</td>
<td>120</td>
<td>112</td>
<td>500</td>
<td>6</td>
<td>125</td>
<td>125</td>
<td>157</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>6</td>
<td>150</td>
<td>121</td>
<td>114</td>
<td>500</td>
<td>6</td>
<td>150</td>
<td>123</td>
<td>154</td>
</tr>
</tbody>
</table>

Table 2: The perplexity on the test sets of Penn Treebank and Text8 corpora. Note that increasing the number of memory hops improves performance.
Figure 3: Average activation weight of memory positions during 6 memory hops. White color indicates where the model is attending during the $k^{th}$ hop. For clarity, each row is normalized to have maximum value of 1. A model is trained on (left) Penn Treebank and (right) Text8 dataset.
Reflections

1. Three types of semantics
   a. Distributional semantics:
      i. Pro: Most broadly applicable, ML-friendly
      ii. Con: Monolithic representations
   b. Frame semantics:
      i. Pro: More structured representations
      ii. Con: Not full representation of world
   c. Model-theoretic semantics:
      i. Full world representation, rich semantics, end-to-end
      ii. Narrower in scope
Reflections

2. Neural MT and Attention Mechanisms
   a. Novel approach to neural machine translation
   b. Applicable to many other structured input/output problems

3. Memory Network
   a. learn to do reasoning tasks end-to-end from scratch
   b. How to get real data and how much do we need to make it work?
   c. Can the model incorporate some structure without getting too complex?
Thanks!