Natural Language Processing 1

Lecture 8: Compositional semantics and discourse processing

Katia Shutova

ILLC
University of Amsterdam

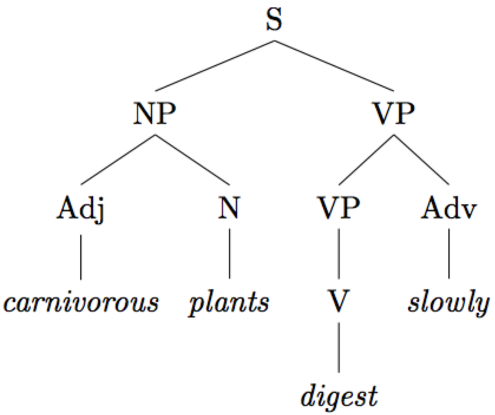
26 November 2018

Outline.

Compositional semantics

- Principle of Compositionality: meaning of each whole phrase derivable from meaning of its parts.
- Sentence structure conveys some meaning
- Deep grammars: model semantics alongside syntax, one semantic composition rule per syntax rule

Compositional semantics alongside syntax



Semantic composition is non-trivial

- Similar syntactic structures may have different meanings:

  it barks
  it rains; it snows – pleonastic pronouns

- Different syntactic structures may have the same meaning:

  Kim seems to sleep.
  It seems that Kim sleeps.

- Not all phrases are interpreted compositionally, e.g. idioms:

  red tape
  kick the bucket

  but they can be interpreted compositionally too, so we can not simply block them.

Semantic composition is non-trivial

- Elliptical constructions where additional meaning arises through composition, e.g. logical metonymy:

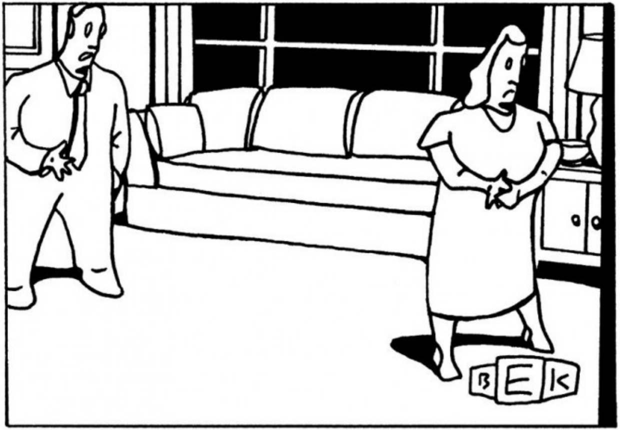
  fast programmer
  fast plane

- Meaning transfer and additional connotations that arise through composition, e.g. metaphor

  *I cant **buy** this story.*
  *This sum will **buy** you a ride on the train.*

- Recursion

Recursion



"Of course I care about how you imagined I thought you perceived I wanted you to feel."

Compositional semantic models

1. Compositional **distributional semantics**
   - model composition in a vector space
   - unsupervised
   - general-purpose representations
2. Compositional semantics in **neural networks**
   - supervised
   - task-specific representations

Outline.

Compositional distributional semantics

Can distributional semantics be extended to account for the meaning of phrases and sentences?

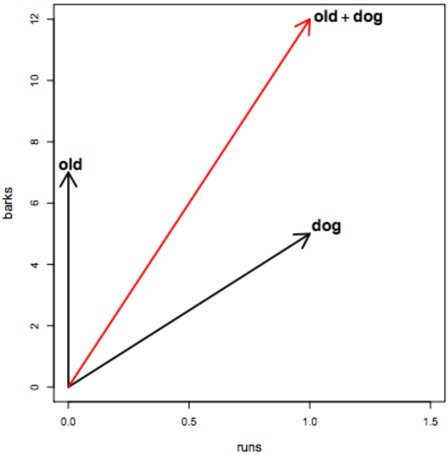
- Language can have an infinite number of sentences, given a limited vocabulary
- So we can not learn vectors for all phrases and sentences
- and need to do composition in a distributional space

1. Vector mixture models

Mitchell and Lapata, 2010.
Composition in Distributional Models of Semantics

Models:

- Additive
- Multiplicative



Additive and multiplicative models

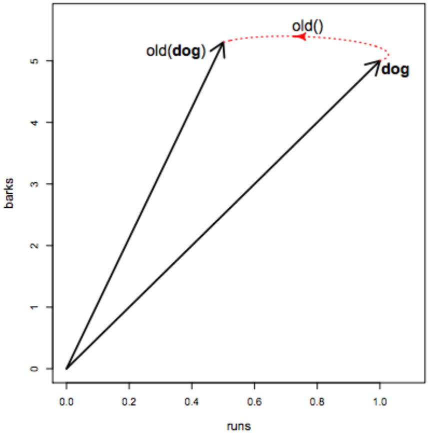
				additive		multiplicative	
	dog	**cat**	**old**	**old** + **dog**	**old** + **cat**	**old** $\odot$ **dog**	**old** $\odot$ **cat**
runs	1	4	0	1	4	0	0
barks	5	0	7	12	7	35	0

- correlate with human similarity judgments about adjective-noun, noun-noun, verb-noun and noun-verb pairs
- but... commutative, hence do not account for word order
  John hit the ball = *The ball hit John*!
- more suitable for modelling content words, would not port well to function words:
  e.g. *some dogs; lice and dogs; lice on dogs*

2. Lexical function models

Distinguish between:

- words whose meaning is directly determined by their distributional behaviour, e.g. nouns
- words that act as functions transforming the distributional profile of other words, e.g., verbs, adjectives and prepositions



Lexical function models

Baroni and Zamparelli, 2010. *Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space*

Adjectives as lexical functions

$$\textit{old dog} = \textit{old}(\textit{dog})$$

- Adjectives are parameter matrices ($\mathbf{A}_{old}$, $\mathbf{A}_{furry}$, etc.).
- Nouns are vectors (**house**, **dog**, etc.).
- Composition is simply **old dog** $= \mathbf{A}_{old} \times$ **dog**.

$$\begin{array}{l|ll} \textbf{OLD} & \text{runs} & \text{barks} \\ \hline \text{runs} & 0.5 & 0 \\ \text{barks} & 0.3 & 1 \end{array} \times \begin{array}{l|l} & \textbf{dog} \\ \hline \text{runs} & 1 \\ \text{barks} & 5 \end{array} = \begin{array}{l|l} & \textbf{OLD(dog)} \\ \hline \text{runs} & (0.5 \times 1) + (0 \times 5) \\ & = 0.5 \\ \text{barks} & (0.3 \times 1) + (5 \times 1) \\ & = 5.3 \end{array}$$

Learning adjective matrices

For each adjective, learn a set of parameters that allow to predict the vectors of adjective-noun phrases

Training set:

house		**old house**
dog		**old dog**
car	$\rightarrow$	**old car**
cat		**old cat**
toy		**old toy**
...		...

Test set:

elephant	$\rightarrow$	**old elephant**
mercedes	$\rightarrow$	**old mercedes**

Learning adjective matrices

1. Obtain a distributional vector $\mathbf{n}_j$ for each noun n_j in the lexicon.
2. Collect adjective noun pairs (a_i, n_j) from the corpus.
3. Obtain a distributional vector $\mathbf{p}_{ij}$ of each pair (a_i, n_j) from the same corpus using a conventional DSM.
4. The set of tuples $\{(\mathbf{n}_j, \mathbf{p}_{ij})\}_j$ represents a dataset $\mathcal{D}(a_i)$ for the adjective a_i.
5. Learn matrix $\mathbf{A}_i$ from $\mathcal{D}(a_i)$ using linear regression.

Minimize the squared error loss:

$$L(\mathbf{A}_i) = \sum_{j \in \mathcal{D}(a_i)} \|\mathbf{p}_{ij} - \mathbf{A}_i \mathbf{n}_j\|^2$$

Verbs as higher-order tensors

Different patterns of subcategorization, i.e. how many (and what kind of) arguments the verb takes

- **Intransitive** verbs: only subject

  Kim slept

  modelled as a matrix (second-order tensor): $N \times M$

- **Transitive** verbs: subject and object

  Kim loves her dog

  modelled as a third-order tensor: $N \times M \times K$

Polysemy in lexical function models

Generally:

- use single representation for all senses
- assume that ambiguity can be handled as long as contextual information is available

Exceptions:

- Kartsaklis and Sadrzadeh (2013): homonymy poses problems and is better handled with prior disambiguation
- Gutierrez et al (2016): literal and metaphorical senses better handled by separate models
- However, this is still an open research question.

Modelling metaphor in lexical function models

Gutierrez et al (2016). *Literal and Metaphorical Senses in Compositional Distributional Semantic Models.*

- trained separate lexical functions for literal and metaphorical senses of adjectives
- mapping from literal to metaphorical sense as a linear transformation
- model can **identify metaphorical expressions**:

  e.g. brilliant person

- and **interpret** them

  brilliant person: clever person
  brilliant person: genius

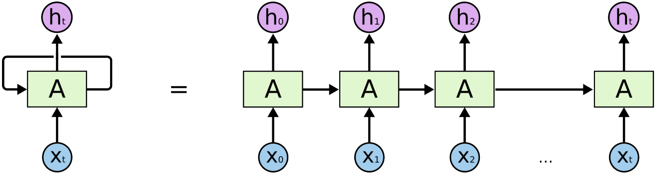
Outline.

Compositional semantics in neural networks

- Supervised learning framework, i.e. train compositional representations for a specific task
- taking word representations as input
- Possible tasks: sentiment analysis; natural language inference; paraphrasing; machine translation etc.

Compositional semantics in neural networks

- **recurrent neural networks** (e.g. LSTM): sequential processing, i.e. no sentence structure



- **recursive neural networks** (e.g. tree LSTM): model compositional semantics alongside syntax

Tree Recursive Neural Networks

Joost Bastings
bastings.github.io

Recap

- Training basics
  - SGD
  - Backpropagation
  - Cross Entropy Loss
- Bag of Words models: BOW, CBOW, Deep CBOW
  - Can encode a sentence of arbitrary length, but loses word order
- Sequence models: RNN and LSTM
  - Sensitive to word order
  - RNN has vanishing gradient problem, LSTM deals with this
  - LSTM has input, forget, and output gates that control information flow

Exploiting tree structure

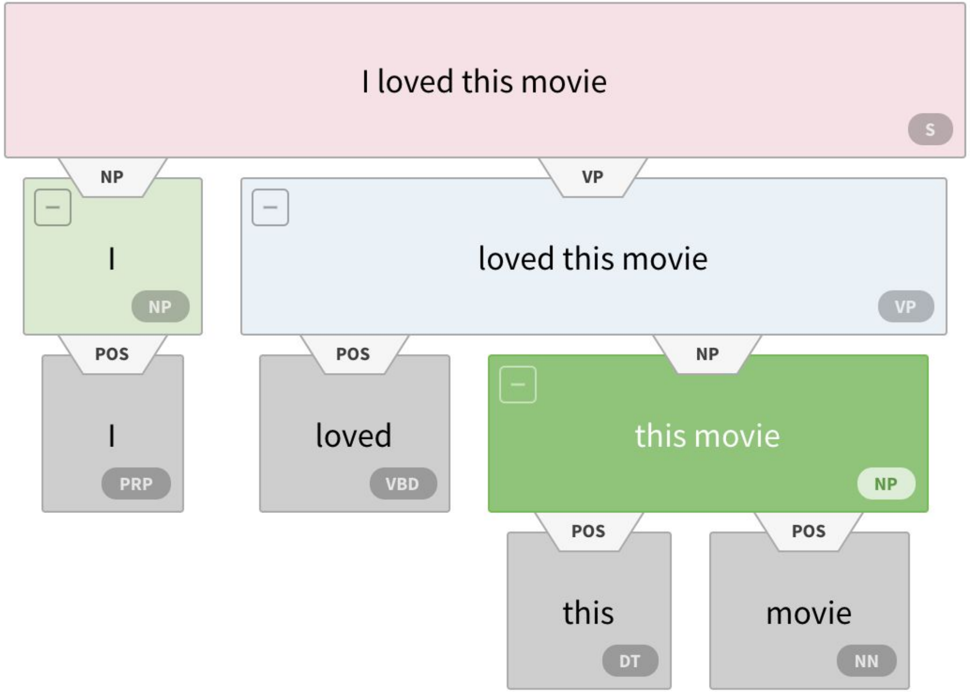
Instead of treating our input as a **sequence**, we can take an alternative approach: assume a **tree structure** and use the principle of **compositionality**.

The meaning (vector) of a sentence is determined by:

1. the meanings of its **words** and
2. the **rules** that combine them

Constituency Parse

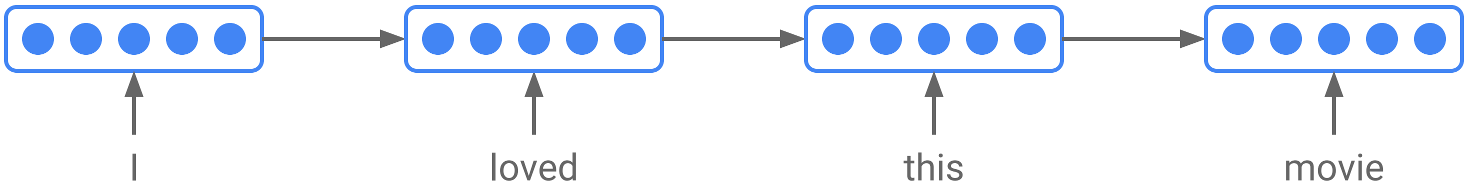
Can we obtain a sentence vector using the tree structure given by a parse?



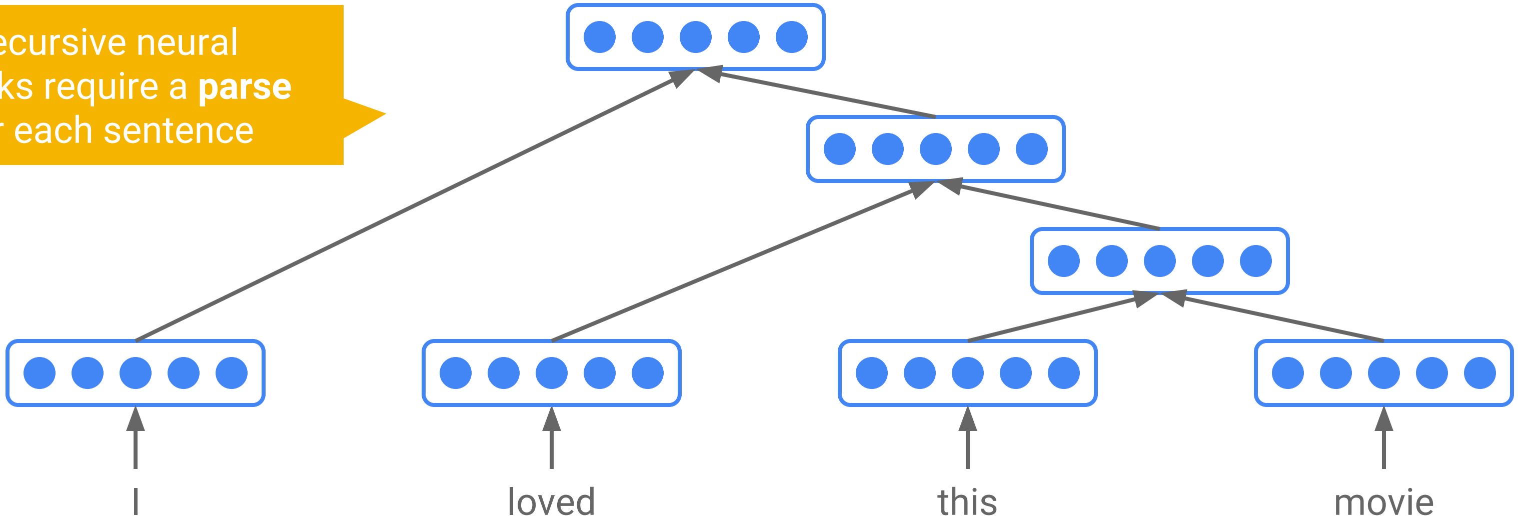
http://demo.allennlp.org/constituency-parsing

Recurrent vs Tree Recursive NN

RNNs cannot capture phrases **without prefix context** and often capture too much of **last words** in final vector



Tree Recursive neural networks require a **parse tree** for each sentence



Practical II data set: Stanford Sentiment Treebank (SST)

sentiment label for **root node**

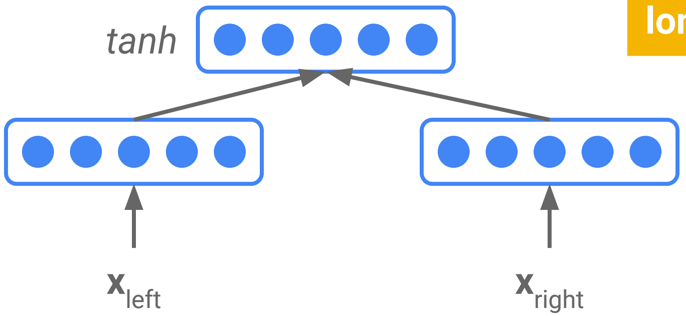
```
             3
 ------------|--------------------
|                                 4
|        -------------------------|----------------------------------------------
|       4                                                                        |
|    ---|--------------                                                          |
|   |                  4                                                         |
|   |         ---------|----------                                               |
|   |        |                    3                                              |
|   |        |               -----|----------------------                        |
|   |        |              |                            4                       |
|   |        |              |            ----------------|-------                |
|   |        |              |           |                        2               |
|   |        |              |           |                 -------|---            |
|   |        3              |           |                |           2           |
|   |    ----|-----         |           |                |        ---|-----      |
|   |   |          4        |           3                |       2         |     |
|   |   |     -----|---     |      -----|-------         |    ---|---      |     |
2   2   2    3         2    2     3             2        2   2       2     2     2
|   |   |    |         |    |     |             |        |   |       |     |     |
It  's  a  lovely     film with lovely     performances  by Buy     and Accorsi  .
```

A naive recursive NN

Combine every two children (left and right) into a parent node **p**:

$$\mathbf{p} = \tanh(W_{left}\mathbf{x}_{left} + W_{right}\mathbf{x}_{right} + \mathbf{b})$$

a bit **simplistic** and does not work well for **longer sentences**



Richard Socher et al. Parsing natural scenes and natural language with recursive neural networks. ICML 2011.

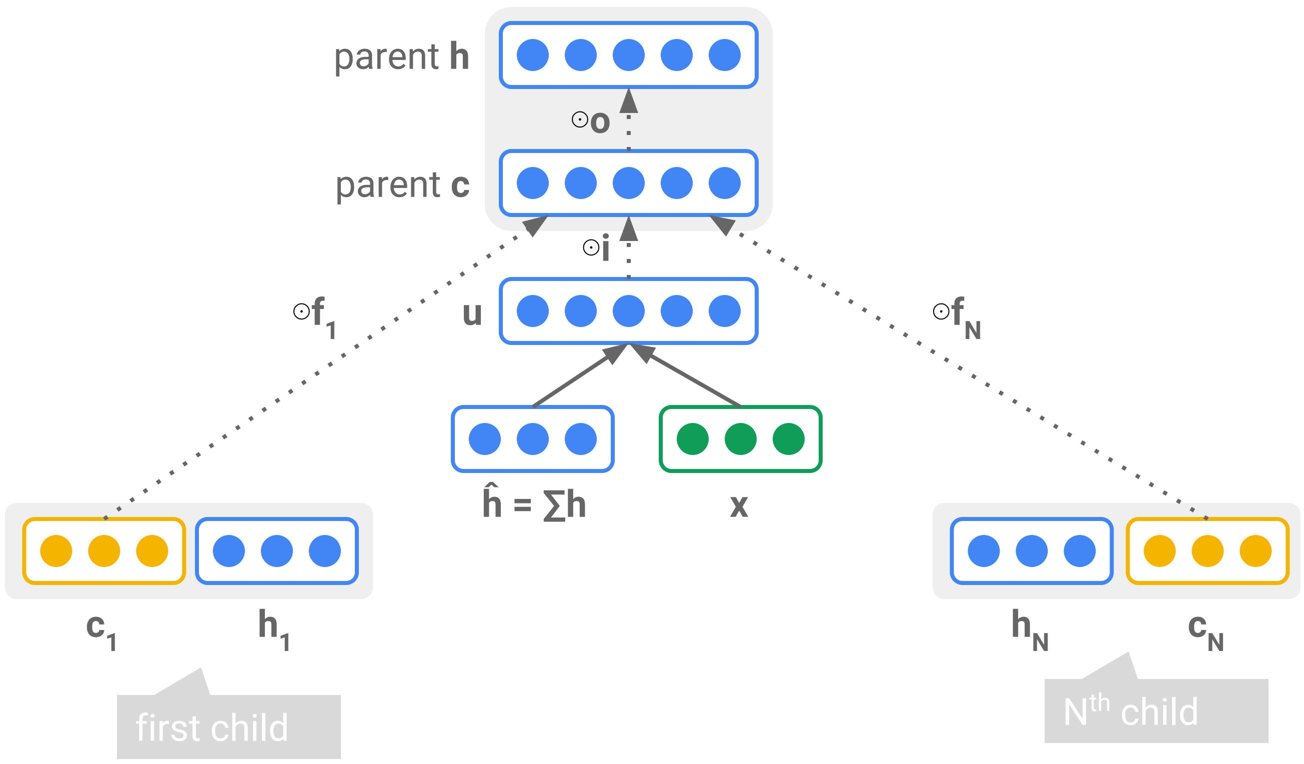
Better idea: generalize LSTM to tree structure

Use the idea of LSTM (gates, memory cell) but allow for multiple inputs (node children)

Proposed by 3 groups in the same summer :-)

- Kai Sheng Tai, Richard Socher, and Christopher D. Manning. *Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks*. ACL 2015.
    - Child-Sum Tree LSTM
    - N-ary Tree LSTM
- Phong Le and Willem Zuidema. *Compositional distributional semantics with long short term memory.* *SEM 2015.
- Xiaodan Zhu, Parinaz Sobihani, and Hongyu Guo. *Long short-term memory over recursive structures*. ICML 2015.

Child-Sum Tree LSTM



Child-Sum Tree LSTM

useful for encoding **dependency** trees

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

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

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

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

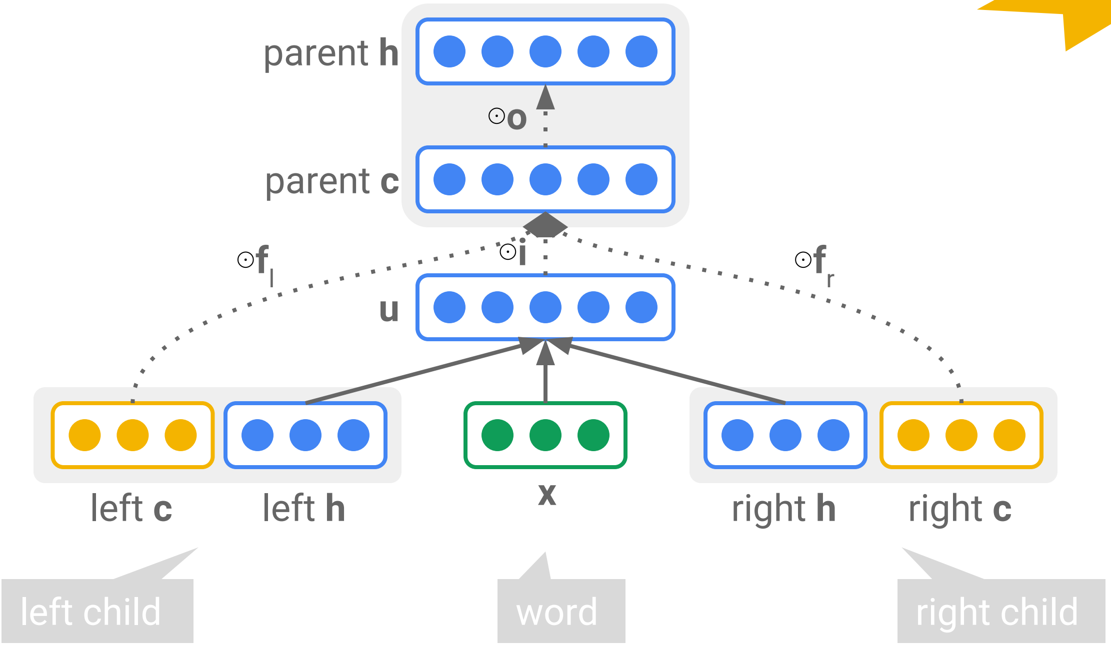
$$u_j = \tanh\left(W^{(u)} x_j + U^{(u)} \tilde{h}_j + b^{(u)}\right)$$

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

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

N-ary Tree LSTM

As seen in Practical II



N-ary Tree LSTM

$$i_j = \sigma\left(W^{(i)}x_j + \sum_{\ell=1}^{N} U_\ell^{(i)} h_{j\ell} + b^{(i)}\right),$$

$$f_{jk} = \sigma\left(W^{(f)}x_j + \sum_{\ell=1}^{N} U_{k\ell}^{(f)} h_{j\ell} + b^{(f)}\right),$$

$$o_j = \sigma\left(W^{(o)}x_j + \sum_{\ell=1}^{N} U_\ell^{(o)} h_{j\ell} + b^{(o)}\right),$$

$$u_j = \tanh\left(W^{(u)}x_j + \sum_{\ell=1}^{N} U_\ell^{(u)} h_{j\ell} + b^{(u)}\right),$$

$$c_j = i_j \odot u_j + \sum_{\ell=1}^{N} f_{j\ell} \odot c_{j\ell},$$

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

useful for encoding **constituency** trees

Transition Sequence Representation

Building a tree with a transition sequence

We can describe a **binary tree** using a *shift-reduce* **transition sequence**

```
(I ( loved ( this movie ) ) )
 S   S       S    S      R R R
```

We start with a buffer (queue) and an empty stack:

```
stack = []
buffer = queue([I, loved, this, movie])
```

Now we follow the transition sequence:

if SHIFT (S): take **first** word (*leftmost*) of the **buffer**, push it to the **stack**

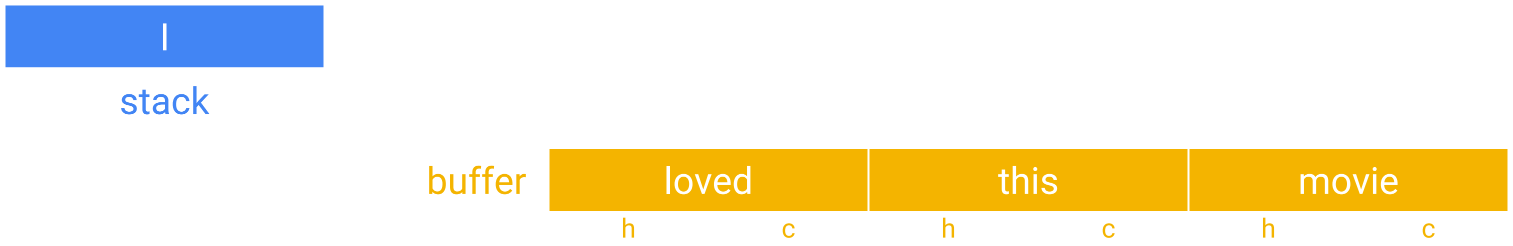
if REDUCE (R): **pop** top 2 words from the **stack** and **reduce** them into one **new node**

Transition sequence example

```
(I ( loved ( this movie ) ) )
 S   S       S    S       R R R
```

Transition sequence example

```
(I ( loved ( this movie ) ) )
 S   S         S    S        R R R
```

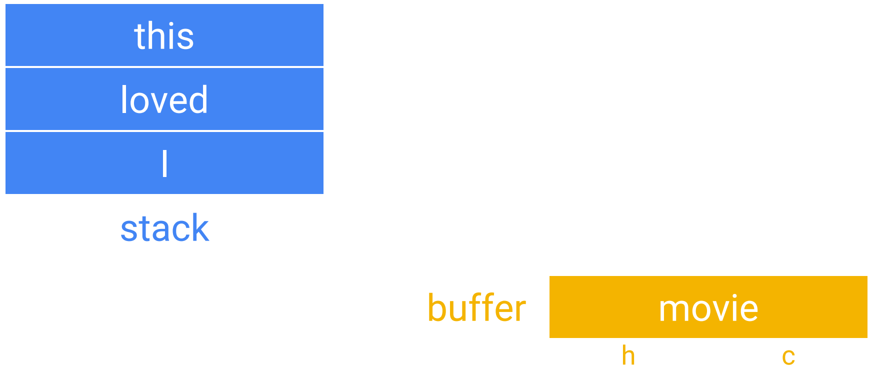


Transition sequence example

```
(I ( loved ( this movie ) ) )
 S   S        S    S     R R R
```

Transition sequence example

```
(I ( loved ( this movie ) ) )
 S   S       S    S     R R R
```



Transition sequence example

```
(I ( loved ( this movie ) ) )
 S   S        S    S      R R R
```

movie

this

loved

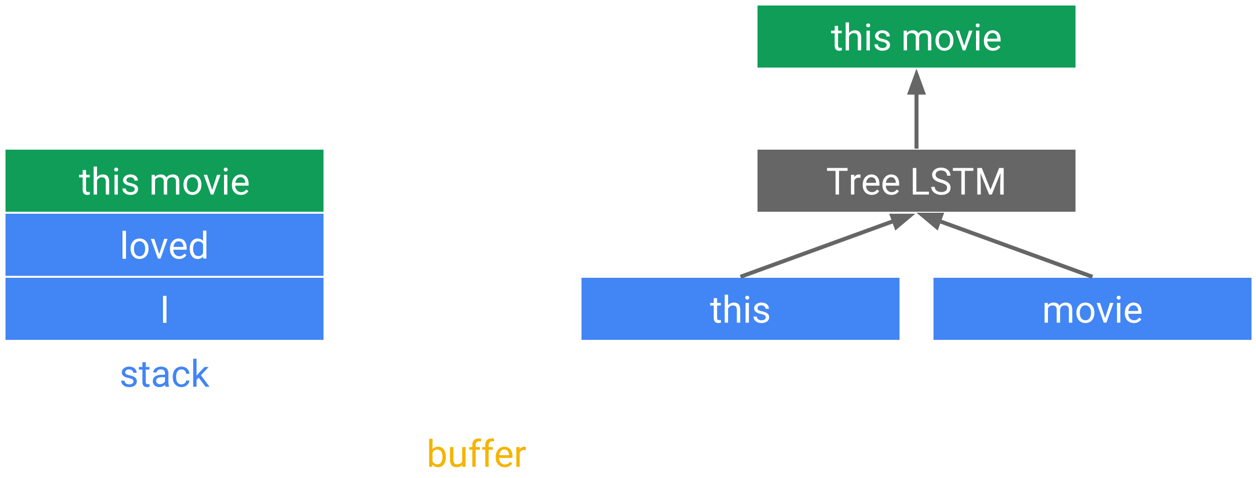
I

stack

buffer

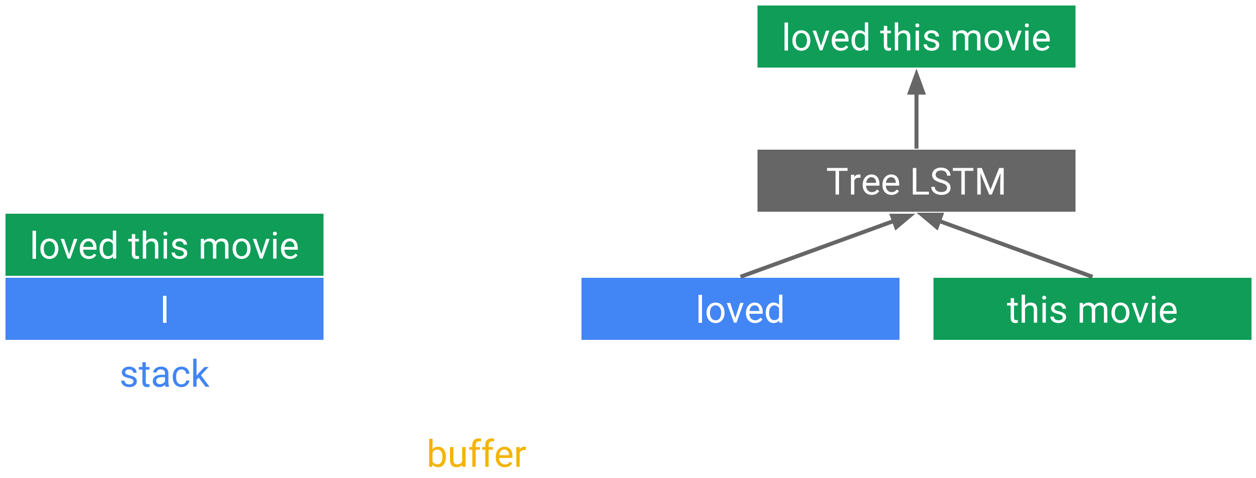
Transition sequence example

```
(I ( loved ( this movie ) ) )
 S   S       S    S      R R R
```



Transition sequence example

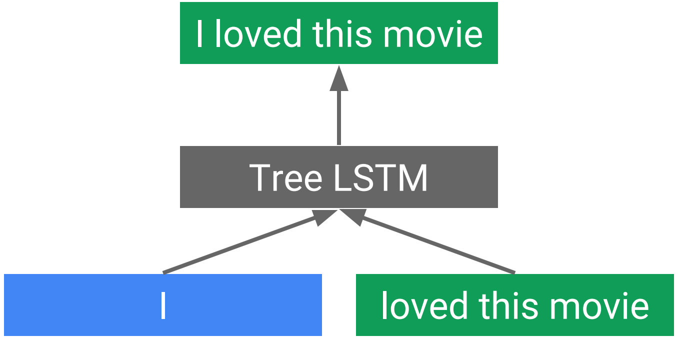
```
(I ( loved ( this movie ) ) )
 S  S        S    S      R R R
```



Transition sequence example

```
(I ( loved ( this movie ) ) )
 S   S       S    S       R R R
```

practical II explains how to obtain this sequence



this is your **root node** for classification

I loved this movie

stack

buffer

Mini-batches

SGD vs GD

Mini-batch SGD strikes a balance between these two

SGD:

```
for epoch in 1..E
  for each training example
    compute loss (forward pass)
    compute gradient of loss (backward)
    update parameters
  end for
end for
```

- **fast**, but **high variance**
- *might* find **better optimum** because of variance

Gradient Descent (GD):

```
for epoch in 1..E
  for each training example
    compute loss (forward pass)
    compute gradient of loss (backward)
    accumulate gradient
  end for
  update parameters
end for
```

- **slow**, but **more stable** (not overly influenced by most recent training example)
- **can get stuck in local optimum**

Source: Neubig.

Transition sequence example (mini-batched)

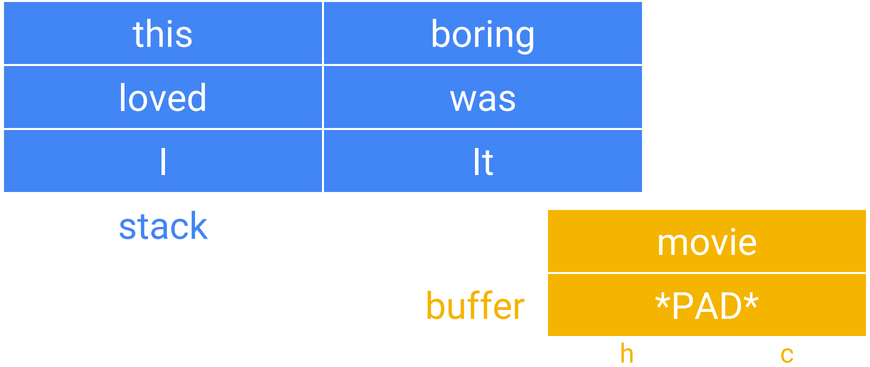
```
(I ( loved ( this movie ) ) )          (It ( was boring ) )
 S   S         S    S      R R R        S     S   S       R R
```

stack

buffer	I	loved	this	movie
	It	was	boring	*PAD*
	h c	h c	h c	h c

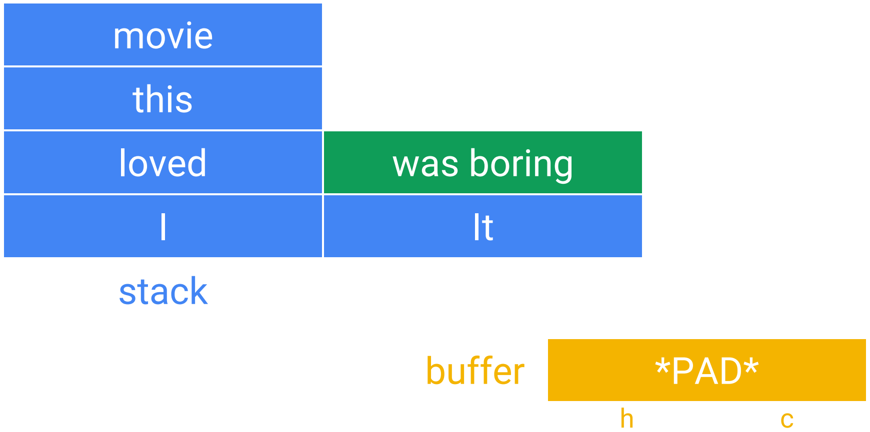
Transition sequence example (mini-batched)

```
(I ( loved ( this movie ) ) )        (It ( was boring ) )
 S   S         S    S      R R R       S   S   S      R R
```



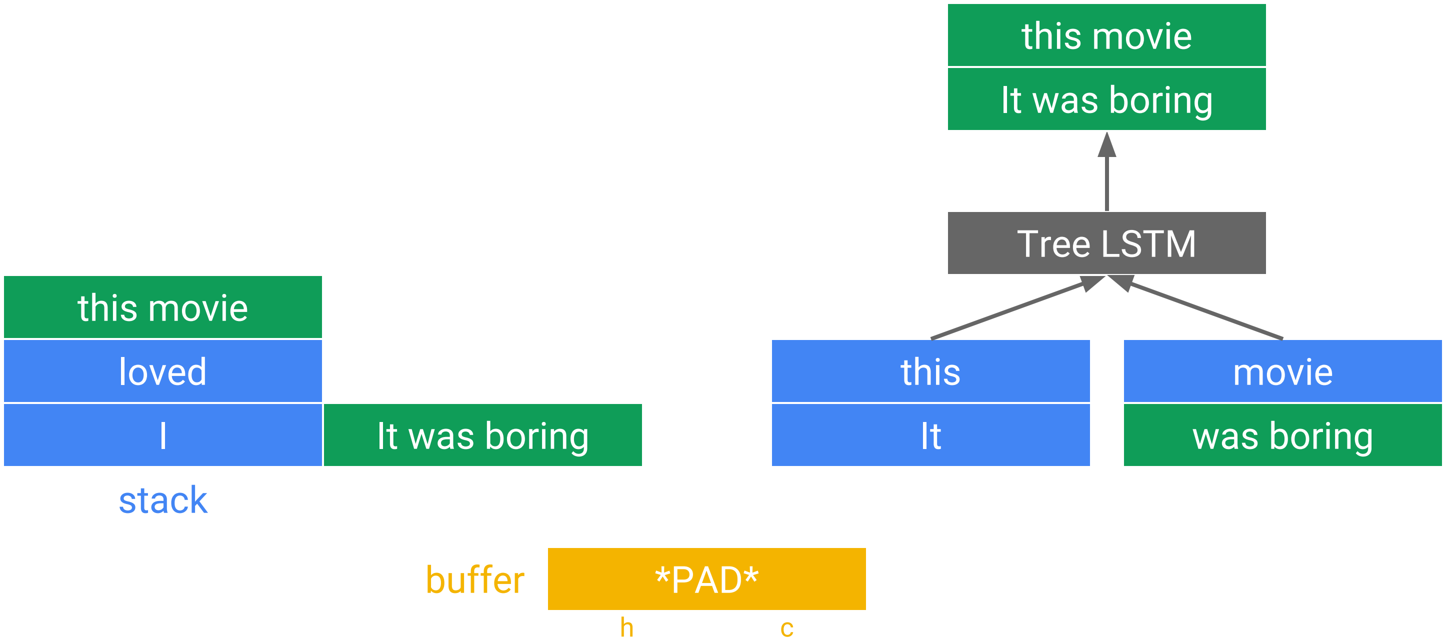
Transition sequence example (mini-batched)

```
(I ( loved ( this movie ) ) )        (It ( was boring ) )
 S   S       S    S      R R R        S     S   S      R R
```



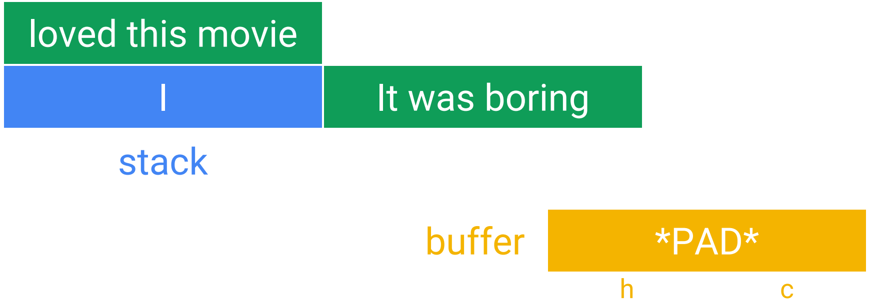
Transition sequence example (mini-batched)

```
(I ( loved ( this movie ) ) )        (It ( was boring ) )
 S   S       S    S     R R R         S    S   S      R R
```



Transition sequence example (mini-batched)

```
(I ( loved ( this movie ) ) )        (It ( was boring ) )
 S   S       S    S     R R R         S    S   S      R R
```



Transition sequence example (mini-batched)

`(I ( loved ( this movie ) ) )`

`S S S S R R R`

`(It ( was boring ) )`

`S S S R R`

Summary

Summary

- Tree-based models: Child-Sum & N-ary Tree LSTM
    - Generalize LSTM to tree structures
    - Exploit compositionality, but require a parse tree
    - Transition sequence
- Mini-batch SGD

Outline.

Document structure and discourse structure

- Most types of document are highly structured, implicitly or explicitly:
  - Scientific papers: conventional structure (differences between disciplines).
  - News stories: first sentence is a summary.
  - Blogs, etc etc
- Topics within documents.
- Relationships between sentences.

Rhetorical relations

Max fell. John pushed him.

can be interpreted as:

1. Max fell because John pushed him.
   EXPLANATION

or

2 Max fell and then John pushed him.
   NARRATION

Implicit relationship: discourse relation or rhetorical relation
because, *and then* are examples of cue phrases

Rhetorical relations

Analysis of text with rhetorical relations generally gives a binary branching structure:

- nucleus (the main phrase) and satellite (the subsidiary phrase: e.g., EXPLANATION, JUSTIFICATION

  Max fell because John pushed him.

- equal weight: e.g., NARRATION

  Max fell and Kim kept running.

Rhetorical relations

Analysis of text with rhetorical relations generally gives a binary branching structure:

- nucleus (the main phrase) and satellite (the subsidiary phrase: e.g., EXPLANATION, JUSTIFICATION

  Max fell because John pushed him.

- equal weight: e.g., NARRATION

  Max fell and Kim kept running.

Coherence

Discourses have to have connectivity to be coherent:

Kim got into her car. Sandy likes apples.

Can be OK in context:

Kim got into her car. Sandy likes apples, so Kim thought she'd go to the farm shop and see if she could get some.

Coherence

Discourses have to have connectivity to be coherent:

Kim got into her car. Sandy likes apples.

Can be OK in context:

Kim got into her car. Sandy likes apples, so Kim thought she'd go to the farm shop and see if she could get some.

Coherence in interpretation

Discourse coherence assumptions can affect interpretation:

John likes Bill. He gave him an expensive Christmas present.

If EXPLANATION - ‘he’ is probably Bill.
If JUSTIFICATION (supplying evidence for another sentence), ‘he’ is John.

Factors influencing discourse interpretation

1. Cue phrases (e.g. *because, and*)
2. Punctuation (also prosody) and text structure.

   Max fell (John pushed him) and Kim laughed.
   Max fell, John pushed him and Kim laughed.
3. Real world content:

   Max fell. John pushed him as he lay on the ground.
4. Tense and aspect.

   Max fell. John had pushed him.
   Max was falling. John pushed him.

Discourse parsing: hard problem, but 'surfacy techniques' (punctuation and cue phrases) work to some extent.

Outline.

Co-reference and referring expressions

Niall Ferguson is prolific, well-paid and a snappy dresser. Stephen Moss hated him — at least until he spent an hour being charmed in the historian's Oxford study.

referent a real world entity that some piece of text (or speech) refers to. the actual Prof. Ferguson

referring expressions bits of language used to perform reference by a speaker. 'Niall Ferguson', 'he', 'him'

antecedent the text initially evoking a referent. 'Niall Ferguson'

anaphora the phenomenon of referring to an antecedent.

cataphora pronouns appear before the referent (rare)

What about *a snappy dresser*?

Pronoun resolution

- Identifying the referents of pronouns
- Anaphora resolution: generally only consider cases which refer to antecedent noun phrases.

Niall Ferguson is prolific, well-paid and a snappy dresser. Stephen Moss hated him — at least until he spent an hour being charmed in the historian's Oxford study.

Pronoun resolution

- Identifying the referents of pronouns
- Anaphora resolution: generally only consider cases which refer to antecedent noun phrases.

Niall Ferguson is prolific, well-paid and a snappy dresser. Stephen Moss hated him — at least until he spent an hour being charmed in the historian's Oxford study.

Outline.

Anaphora resolution as supervised classification

- instances: potential pronoun/antecedent pairings
- class is TRUE/FALSE
- training data labelled with correct pairings
- candidate antecedents are all NPs in current sentence and preceeding 5 sentences (excluding pleonastic pronouns)

Niall Ferguson is prolific, well-paid and a snappy dresser. Stephen Moss hated him — at least until he spent an hour being charmed in the historian's Oxford study.

Hard constraints: Pronoun agreement

- A little girl is at the door — see what she wants, please?
- My dog has hurt his foot — he is in a lot of pain.
- * My dog has hurt his foot — it is in a lot of pain.

Complications:

- I don't know who the new lecturer will be, but I'm sure they'll make changes to the course.
- The team played really well, but now they are all very tired.
- Kim and Sandy are asleep: they are very tired.

Hard constraints: Reflexives

- John$_i$ cut himself$_i$ shaving. (himself = John, subscript notation used to indicate this)
- # John$_i$ cut him$_j$ shaving. ($i \neq j$ — a very odd sentence)

Reflexive pronouns must be coreferential with a preceeding argument of the same verb, non-reflexive pronouns cannot be.

Hard constraints: Pleonastic pronouns

Pleonastic pronouns are semantically empty, and don't refer:

- It is snowing
- It is not easy to think of good examples.
- It is obvious that Kim snores.
- It bothers Sandy that Kim snores.

Soft preferences: Salience

- Recency: More recent antecedents are preferred. They are more accessible.

  Kim has a big car. Sandy has a smaller one. Lee likes to drive it.

- Grammatical role: Subjects $>$ objects $>$ everything else:

  Fred went to the shopping centre with Bill. He bought a CD.

- Repeated mention: Entities that have been mentioned more frequently are preferred.

Soft preferences: Salience

- Parallelism Entities which share the same role as the pronoun in the same sort of sentence are preferred:

  Bill went with Fred to the Grafton Centre. Kim went with him to Lion Yard. Him=Fred

- Coherence effects: The pronoun resolution may depend on the rhetorical / discourse relation that is inferred.

  Bill likes Fred. He has a great sense of humour.

Features

Cataphoric Binary: t if pronoun before antecedent.

Number agreement Binary: t if pronoun compatible with antecedent.

Gender agreement Binary: t if gender agreement.

Same verb Binary: t if the pronoun and the candidate antecedent are arguments of the same verb.

Sentence distance Discrete: { 0, 1, 2 ... }

Grammatical role Discrete: { subject, object, other } The role of the potential antecedent.

Parallel Binary: t if the potential antecedent and the pronoun share the same grammatical role.

Linguistic form Discrete: { proper, definite, indefinite, pronoun }

Feature vectors

Niall Ferguson is prolific, well-paid and a snappy dresser. Stephen Moss hated him — at least until he spent an hour being charmed in the historian's Oxford study.

pron	ante	cat	num	gen	same	dist	role	par	form
him	*Niall F.*	f	t	t	f	1	subj	f	prop
him	*Ste. M.*	f	t	t	t	0	subj	f	prop
him	*he*	t	t	t	f	0	subj	f	pron
he	*Niall F.*	f	t	t	f	1	subj	t	prop
he	*Ste. M.*	f	t	t	f	0	subj	t	prop
he	*him*	f	t	t	f	0	obj	f	pron

Training data, from human annotation

class	cata	num	gen	same	dist	role	par	form
TRUE	f	t	t	f	1	subj	f	prop
FALSE	f	t	t	t	0	subj	f	prop
FALSE	t	t	t	f	0	subj	f	pron
FALSE	f	t	t	f	1	subj	t	prop
TRUE	f	t	t	f	0	subj	t	prop
FALSE	f	t	t	f	0	obj	f	pron

Problems with simple classification model

- Cannot implement ‘repeated mention’ effect.
- Cannot use information from previous links.

Not really pairwise: need a discourse model with real world entities corresponding to clusters of referring expressions.

Evaluation

- link accuracy, i.e. percentage of correct links.

But:

- Identification of non-pleonastic pronouns and antecendent NPs should be part of the evaluation.
- Binary linkages don't allow for chains:

  Sally met Andrew in town and took him to the new restaurant. He was impressed.

Multiple evaluation metrics exist because of such problems.

Acknowledgement

Some slides were adapted from Ann Copestake