Language Technologies Institute

Carnegie Mellon University

Multimodal Machine Learning

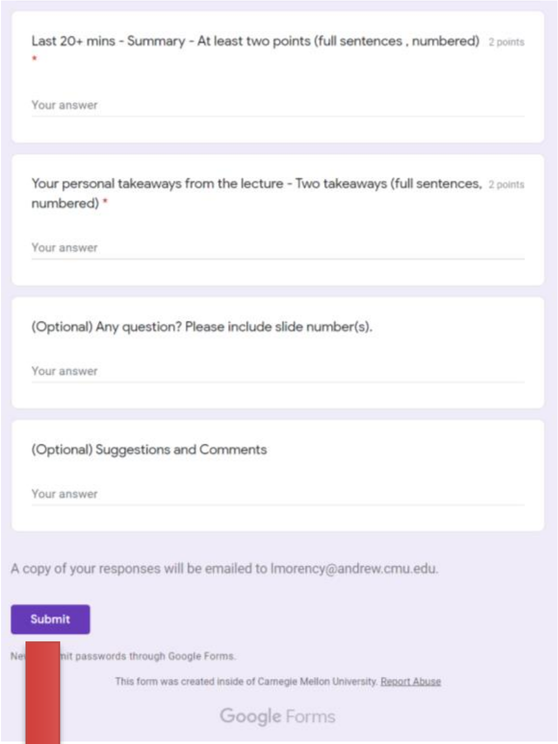
Lecture 3.2: Language Representations and RNNs

Louis-Philippe Morency

** Original course co-developed with Tadas Baltrusaitis.*
Spring 2021 edition taught by Yonatan Bisk

Administrative Stuff

Lecture Highlights - Reminder



Last 20+ mins - Summary - At least two points (full sentences , numbered) * 2 points

Your answer

Your personal takeaways from the lecture - Two takeaways (full sentences, numbered) * 2 points

Your answer

(Optional) Any question? Please include slide number(s).

Your answer

(Optional) Suggestions and Comments

Your answer

A copy of your responses will be emailed to lmorency@andrew.cmu.edu.

Submit

Ne it passwords through Google Forms.

This form was created inside of Carnegie Mellon University. Report Abuse

Google Forms

IMPORTANT: Be sure you received an email after your submission (or revisit the form and your answers should be there).

Reading Assignments – Reminder

Week 3 reading assignment was posted

1. **Friday 8pm:** Post your summary
2. **Monday 8pm:** End of the reading assignment

Be sure to post your discussion comments before Monday 8pm!

➡ Start the discussion early ☺

➡ Late submissions will be penalized

GPUs

- 50$ coupons available for each student
- Pre-registration is required first

More details soon on Piazza …

Language Technologies Institute

Carnegie Mellon University

Multimodal Machine Learning

Lecture 3.2: Language Representations and RNNs

Louis-Philippe Morency

** Original course co-developed with Tadas Baltrusaitis.*
Spring 2021 edition taught by Yonatan Bisk

Lecture Objectives

- Word representations
  - Distributional hypothesis
  - Learning neural representations
- Sentence representations and sequence modeling
  - Recurrent neural networks
  - Gated recurrent neural networks
  - Backpropagation through time
- Syntax and language structure
  - Phrase-structure and dependency grammars
  - Recursive neural network
    - Tree-based RNN, Stack LSTM

Word Representations

What is the meaning of "bardiwac"?

- He handed her her glass of bardiwac.
- Beef dishes are made to complement the bardiwacs.
- Nigel staggered to his feet, face flushed from too much bardiwac.
- Malbec, one of the lesser-known bardiwac grapes, responds well to Australia's sunshine.
- I dined off bread and cheese and this excellent bardiwac.
- The drinks were delicious: blood-red bardiwac as well as light, sweet Rhenish.

⇒ bardiwac is a heavy red alcoholic beverage made from grapes

How to learn (word) features/representations?

- **Distribution hypothesis:** Approximate the word meaning by its surrounding words

- Words used in a similar context will lie close together

- **Instead of capturing co-occurrence counts directly, predict surrounding words of every word**

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

Geometric interpretation

- row vector $\mathbf{x}_{dog}$ describes usage of word *dog* in the corpus

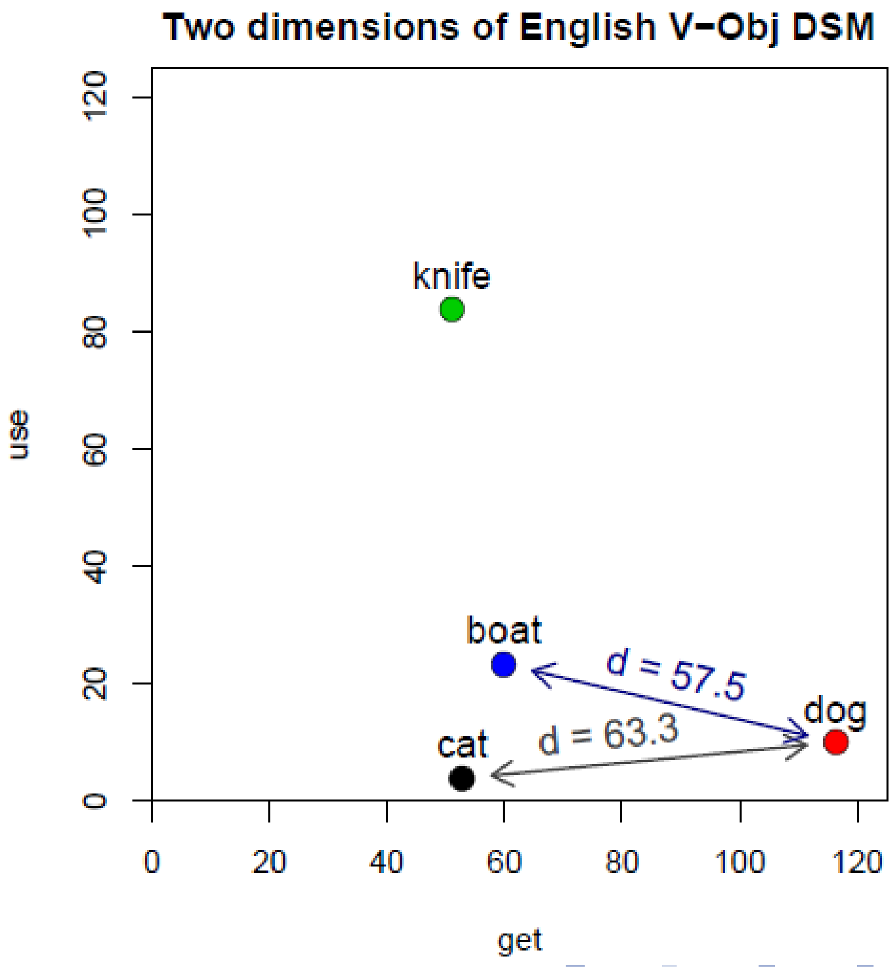
- can be seen as coordinates of point in *n*-dimensional Euclidean space R^n

	get	see	use	hear	eat	kill
knife	51	20	84	0	3	0
cat	52	58	4	4	6	26
dog	115	83	10	42	33	17
boat	59	39	23	4	0	0
cup	98	14	6	2	1	0
pig	12	17	3	2	9	27
banana	11	2	2	0	18	0

co-occurrence matrix M

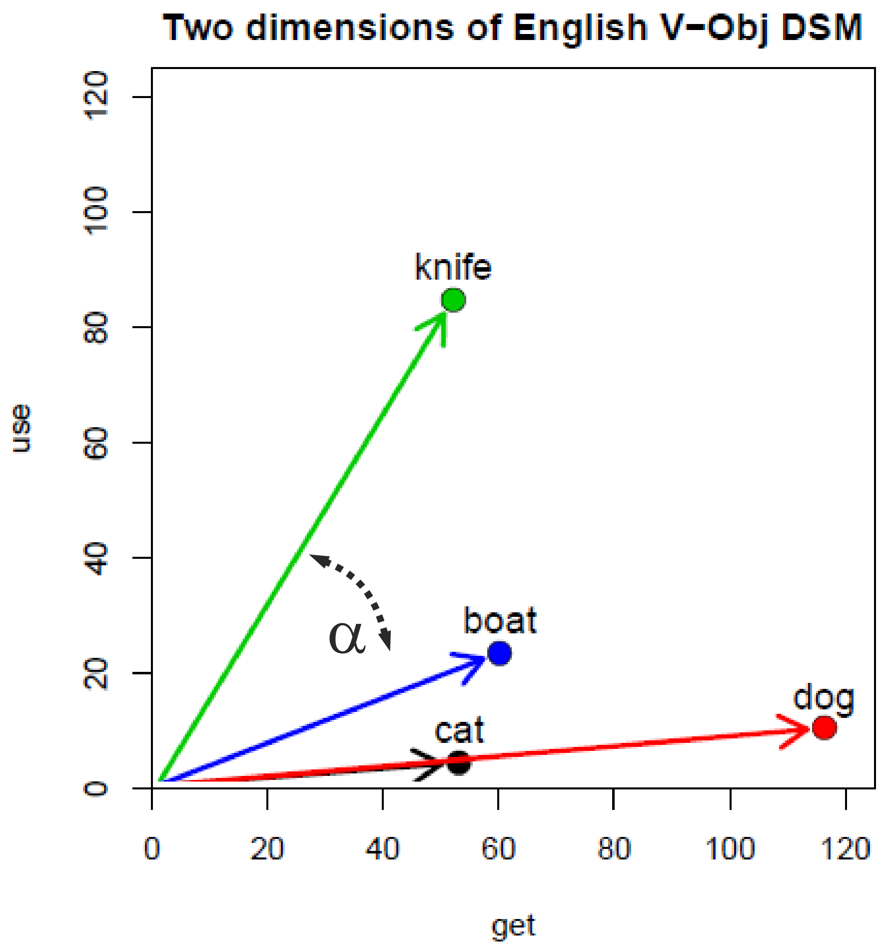
Distance and similarity

- illustrated for two dimensions: *get* and *use*: $\mathbf{x}_{dog} = (115, 10)$
- similarity = spatial proximity (Euclidean distance)
- location depends on frequency of noun ($f_{dog} \approx 2.7 \cdot f_{cat}$)

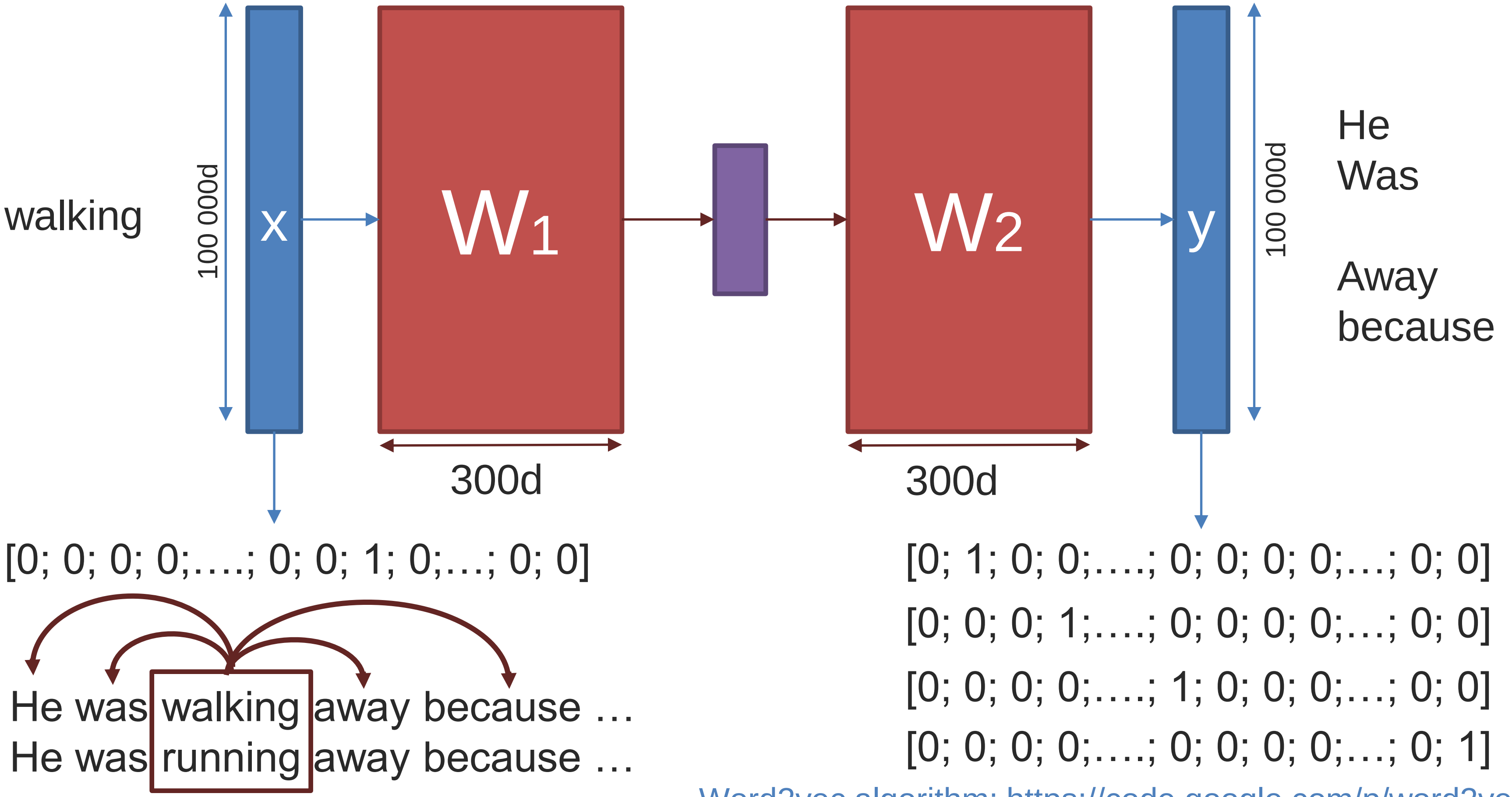


Angle and similarity

- direction more important than location
- normalise "length" $||\mathbf{x}_{dog}||$ of vector
- or use angle α as distance measure



How to learn (word) features/representations?



Word2vec algorithm: https://code.google.com/p/word2vec/

How to use these word representations

If we would have a vocabulary of 100 000 words:

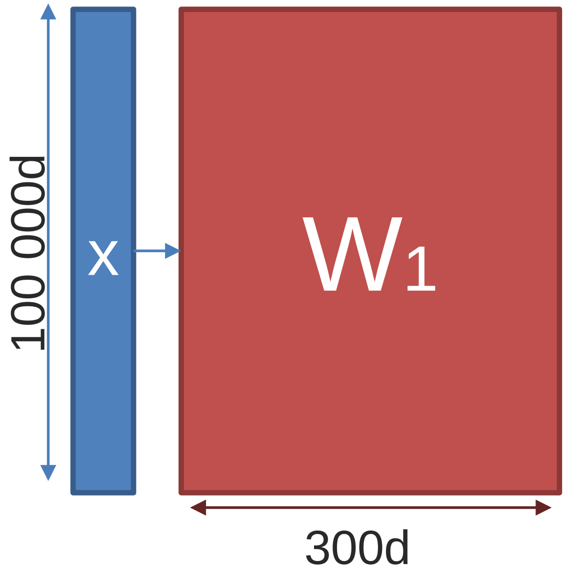
Classic NLP: 100 000 dimensional vector

Walking: [0; 0; 0; 0;….; 0; 0; 1; 0;…; 0; 0]

Running: [0; 0; 0; 0;….; 0; 0; 0; 0;…; 1; 0]

Similarity = 0.0

Transform: x'=x*W



Goal: 300 dimensional vector

Walking: [0,1; 0,0003; 0;….; 0,02; 0.08; 0,05]

Running: [0,1; 0,0004; 0;….; 0,01; 0.09; 0,05]

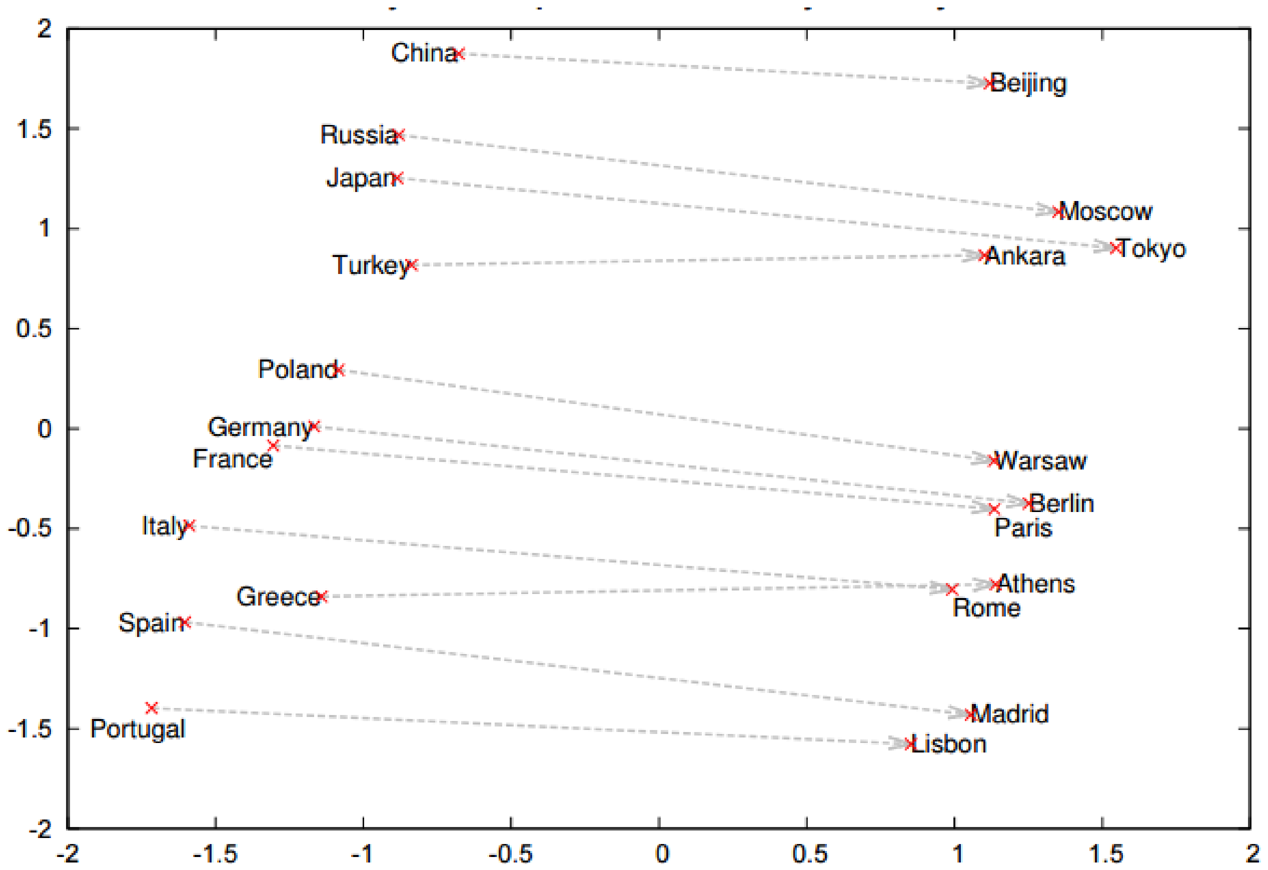
Similarity = 0.9

Vector space models of words

- While learning these word representations, we are actually building a vector space in which all words reside with certain relationships between them
- Encodes both syntactic and semantic relationships
- This vector space allows for algebraic operations:

Vec(king) – vec(man) + vec(woman) ≈ vec(queen)

Vector space models of words: semantic relationships



Trained on the Google news corpus with over 300 billion words

Word Representation Resources

Word-level representations:

Word2Vec (Google, 2013)

https://code.google.com/archive/p/word2vec/

Glove (Stanford, 2014)

https://nlp.stanford.edu/projects/glove/

FastText (Facebook, 2017)

https://fasttext.cc/

Sentence-level representations:

ELMO (Allen Institute for AI, 2018)

https://allennlp.org/elmo

BERT (Google, 2018)

https://github.com/google-research/bert

RoBERTa (Facebook, 2019)

https://github.com/pytorch/fairseq

Word representations are contextualized using all the words in the sentence.

More details later in this lecture and during Week 5

Lexicon-based Word Representation

LIWC: Language Inquiry & Word Count

Manually created dictionaries for different topics and categories:

- Function words: *pronouns, preposition, negation…*
- Affect words: *positive, negative emotions*
- Social words: *family, friends, referents*
- Cognitive processes: *Insight, cause, …*
- Perceptual processes: *Seeing, hearing, feeling*
- Biological processes: *Body, health/illness,…*
- Drives and needs*: Affiliation, achievement, …*
- Time orientation: *past, present, future*
- Relativity: *motion, space, time*
- Personal concerns: *work, leisure, money, religion …*
- Informal speech: *swear words, fillers, assent,…*

LIWC can encode individual words or full sentences.

https://liwc.wpengine.com/

➡ Commercial software. Contact TAs in advance if you would like to use it.

Other Lexicon Resources

Lexicons

- General Inquirer (Stone et al., 1966)
- OpinionFinder lexicon (Wiebe & Riloff, 2005)
- SentiWordNet (Esuli & Sebastiani, 2006)
- LIWC (Pennebaker)

Other Tools

- LightSIDE
- Stanford NLP toolbox
- IBM Watson Tone Analyzer
- Google Cloud Natural Language
- Microsoft Azure Text Analytics

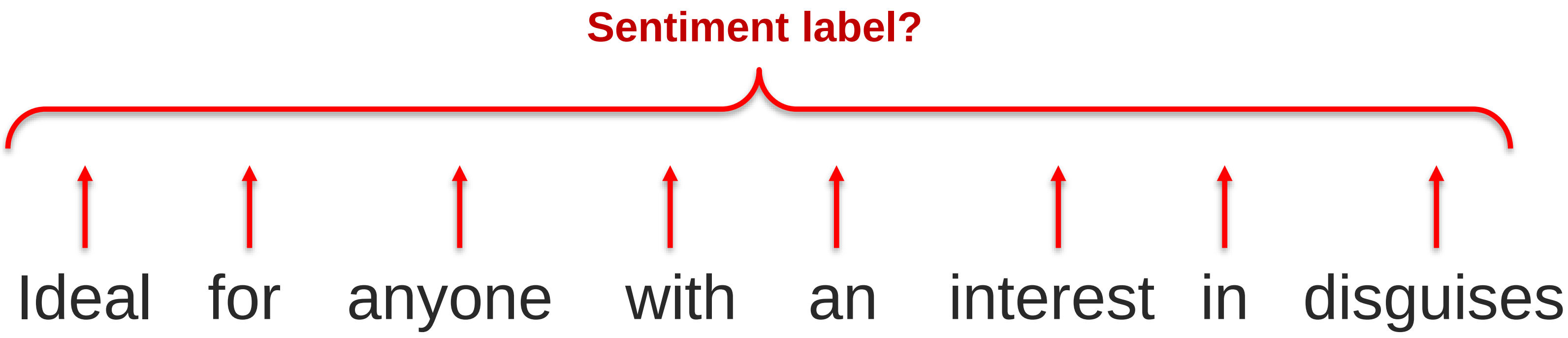
Sentence Modeling

Sentence Modeling: Sequence Label Prediction


Masterful!
By Antony Witheyman - January 12, 2006
Ideal for anyone with an interest in disguises who likes to see the subject tackled in a humourous manner.
0 of 4 people found this review helpful

Prediction →

Sentiment ?
(positive or negative)



Sentence Modeling: Sequence Prediction


Masterful!
By Antony Witheyman - January 12, 2006
Ideal for anyone with an interest in disguises who likes to see the subject tackled in a humourous manner.
0 of 4 people found this review helpful

Prediction →

Part-of-speech ?
(noun, verb,…)

POS?	POS?	POS?	POS?	POS?	POS?	POS?	POS?
↑	↑	↑	↑	↑	↑	↑	↑
Ideal	for	anyone	with	an	interest	in	disguises

Sentence Modeling: Sequence Representation

Masterful!

By Antony Witheyman - January 12, 2006

Ideal for anyone with an interest in disguises who likes to see the subject tackled in a humourous manner.

0 of 4 people found this review helpful

Learning → Sequence representation

[0,1; 0,0004; 0;….; 0,01; 0.09; 0,05]

Ideal for anyone with an interest in disguises

Sentence Modeling: Language Model


Masterful!

By Antony Witheyman - January 12, 2006

Ideal for anyone with an interest in disguises who likes to see the subject tackled in a humourous manner.

0 of 4 people found this review helpful

Prediction → Language Model

Next word?

Ideal for anyone with an interest in disguises

Language Model Application: Language Generation

Embedding

[0,1;
0,0004;
….;
0.09;
0,05]

Generation →

Ideal for anyone with an interest in disguises who likes to see the subject tackled in a humourous manner.

Example: Image captioning

→

[0,1;
0,0004;
….;
0.09;
0,05]

→

The man at bat readies to swing at the pitch while the umpire looks on.

Language Model Application: Speech Recognition

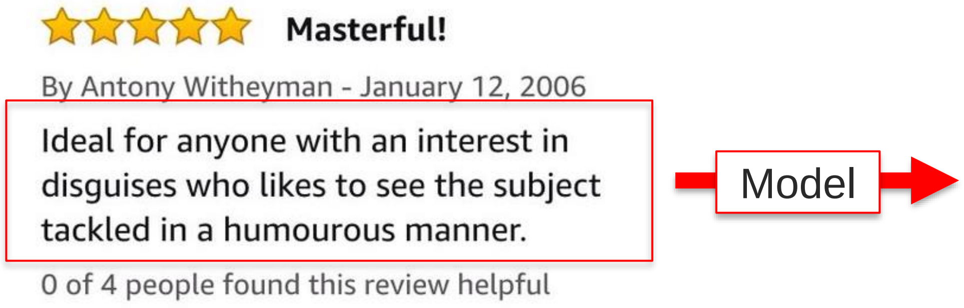
$$\arg\max_{wordsequence} P(wordsequence \mid acoustics) =$$

$$\arg\max_{wordsequence} \frac{P(acoustics \mid wordsequence) \times P(wordsequence)}{P(acoustics)}$$

$$\arg\max_{wordsequence} P(acoustics \mid wordsequence) \times P(wordsequence)$$

↑ **Language model**

Challenges in Sequence Modeling



- Part-of-speech ?
  (noun, verb,…)
- Sentiment ?
  (positive or negative)
- Language Model
- Sequence representation

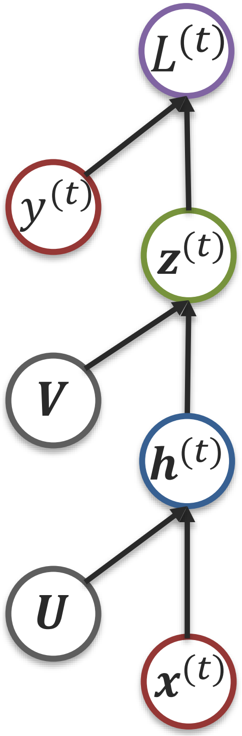
Main Challenges:

- Sequences of variable lengths (e.g., sentences)
- Keep the number of parameters at a minimum
- Take advantage of possible redundancy

Recurrent Neural Networks

Recurrent Neural Network

Feedforward Neural Network



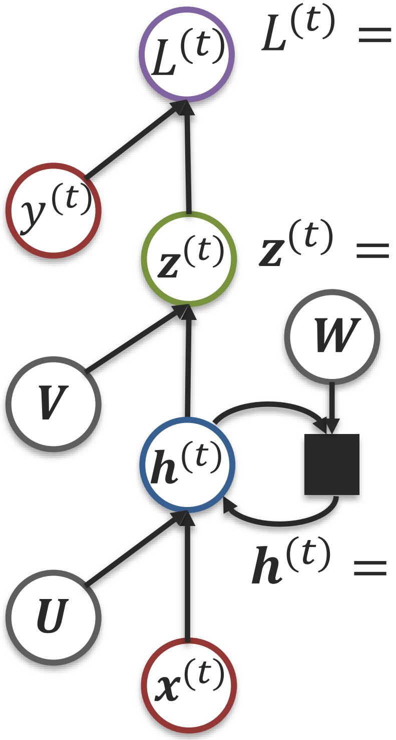
$$L^{(t)} = -logP(Y = y^{(t)} | \boldsymbol{z}^{(t)})$$

$$\boldsymbol{z}^{(t)} = matmult(\boldsymbol{h}^{(t)}, \boldsymbol{V})$$

$$\boldsymbol{h}^{(t)} = tanh(\boldsymbol{U}\boldsymbol{x}^{(t)})$$

Recurrent Neural Networks

$$L = \sum_t L^{(t)}$$



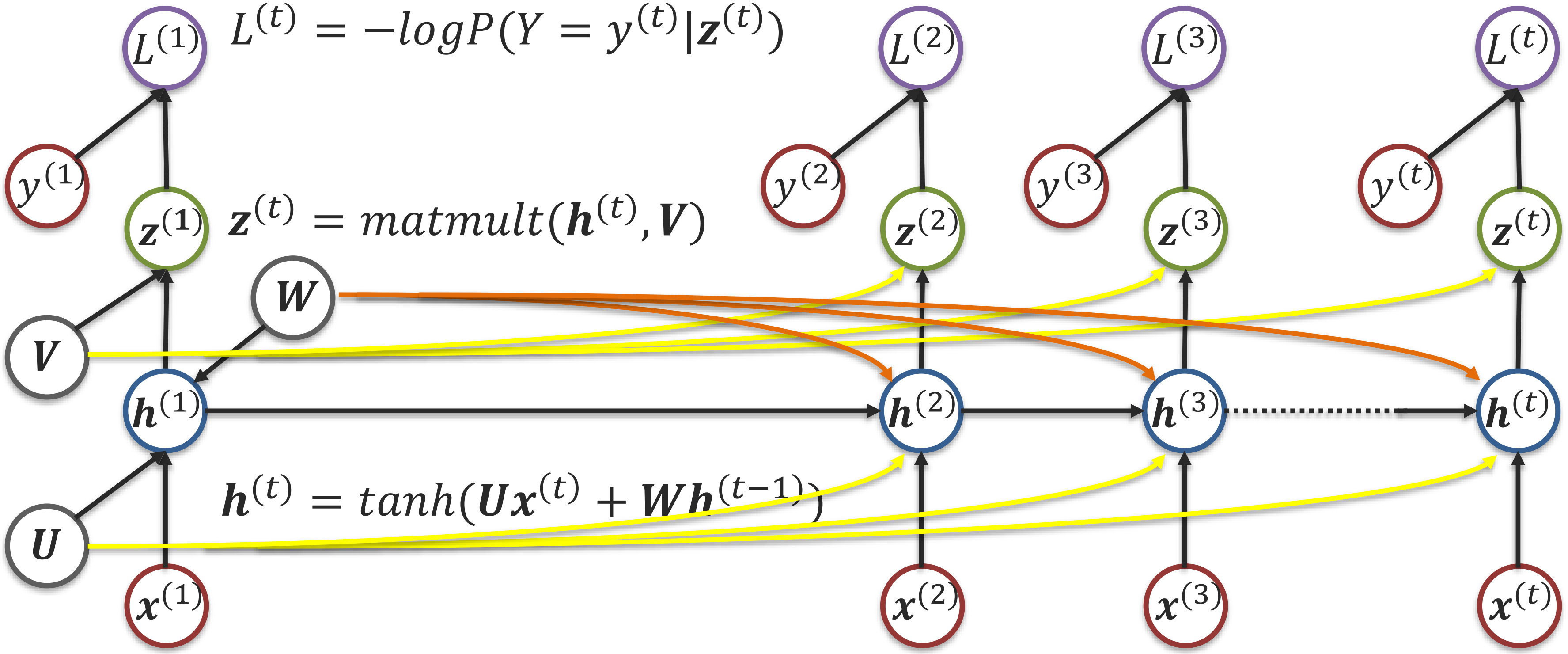
$$L^{(t)} = -logP(Y = y^{(t)}|\boldsymbol{z}^{(t)})$$

$$\boldsymbol{z}^{(t)} = matmult(\boldsymbol{h}^{(t)}, \boldsymbol{V})$$

$$\boldsymbol{h}^{(t)} = tanh(\boldsymbol{U}\boldsymbol{x}^{(t)} + \boldsymbol{W}\boldsymbol{h}^{(t-1)})$$

Recurrent Neural Networks - Unrolling

$$L = \sum_t L^{(t)}$$



$$L^{(t)} = -logP(Y = y^{(t)} | \boldsymbol{z}^{(t)})$$

$$\boldsymbol{z}^{(t)} = matmult(\boldsymbol{h}^{(t)}, \boldsymbol{V})$$

$$\boldsymbol{h}^{(t)} = tanh(\boldsymbol{U}\boldsymbol{x}^{(t)} + \boldsymbol{W}\boldsymbol{h}^{(t-1)})$$

Same model parameters are used for all time parts.

Backpropagation Through Time

$$L = \sum_t L^{(t)} = -\sum_t logP(Y = y^{(t)}|\boldsymbol{z}^{(t)})$$

Gradient = "backprop" gradient

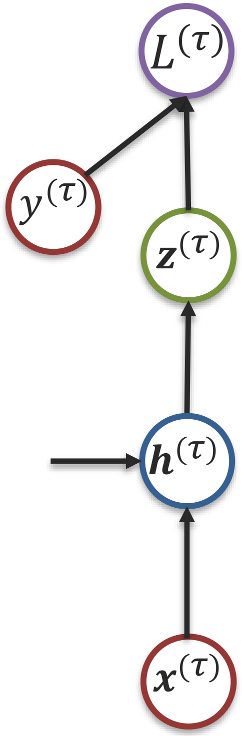
x "local" Jacobian

$L^{(\tau)}$ or $L^{(t)}$ $\quad \frac{\partial L}{\partial L^{(t)}} = 1$

$\boldsymbol{z}^{(\tau)}$ or $\boldsymbol{z}^{(t)}$ $\quad \left(\nabla_{\boldsymbol{z}^{(t)}} L\right)_i = \frac{\partial L}{\partial z_i^{(t)}} = \frac{\partial L}{\partial L^{(t)}} \frac{\partial L^{(t)}}{\partial z_i^{(t)}} = sigmoid\left(z_i^t\right) - \mathbf{1}_{i,y^{(t)}}$

$\boldsymbol{h}^{(\tau)}$ $\quad \nabla_{\boldsymbol{h}^{(\tau)}} L = \nabla_{\boldsymbol{z}^{(\tau)}} L \frac{\partial z^{(\tau)}}{\partial \boldsymbol{h}^{(\tau)}} = \nabla_{\boldsymbol{z}^{(\tau)}} L\boldsymbol{V}$

$\boldsymbol{h}^{(t)} \rightarrow \boldsymbol{h}^{(t+1)}$ $\quad \nabla_{\boldsymbol{h}^{(t)}} L = \nabla_{\boldsymbol{z}^{(t)}} L \frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{h}^{(t)}} + \nabla_{\boldsymbol{z}^{(t+1)}} L \frac{\partial \boldsymbol{h}^{(t+1)}}{\partial \boldsymbol{h}^{(t)}}$

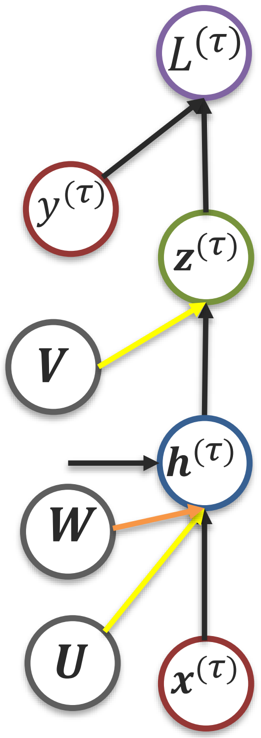


Backpropagation Through Time

$$L = \sum_t L^{(t)} = -\sum_t logP(Y = y^{(t)} | \boldsymbol{z}^{(t)})$$

Gradient = "backprop" gradient

x "local" Jacobian

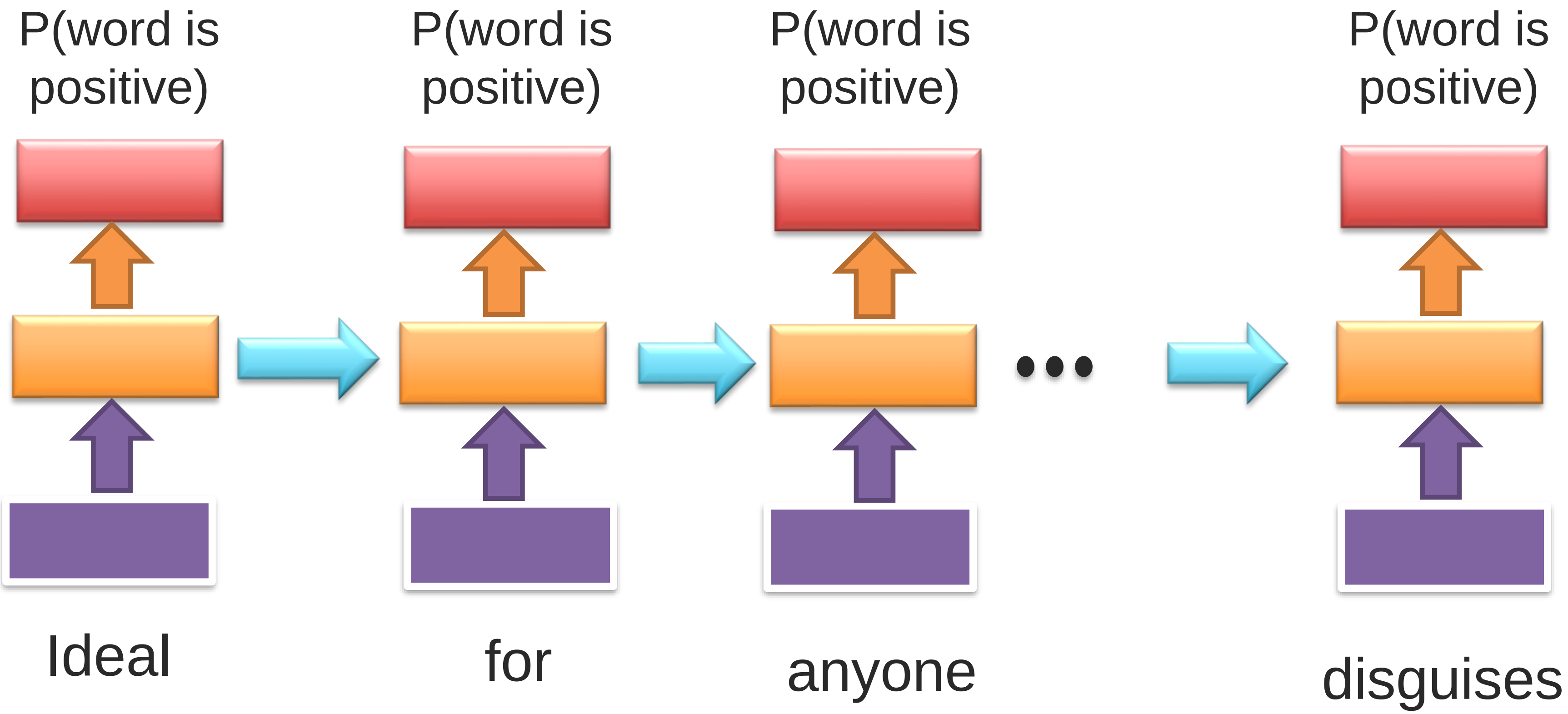


$\boldsymbol{V}$ $$\nabla_{\boldsymbol{V}} L = \sum_t (\nabla_{\boldsymbol{z}^{(t)}} L) \frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{V}}$$

$\boldsymbol{W}$ $$\nabla_{\boldsymbol{W}} L = \sum_t (\nabla_{\boldsymbol{h}^{(t)}} L) \frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{W}}$$

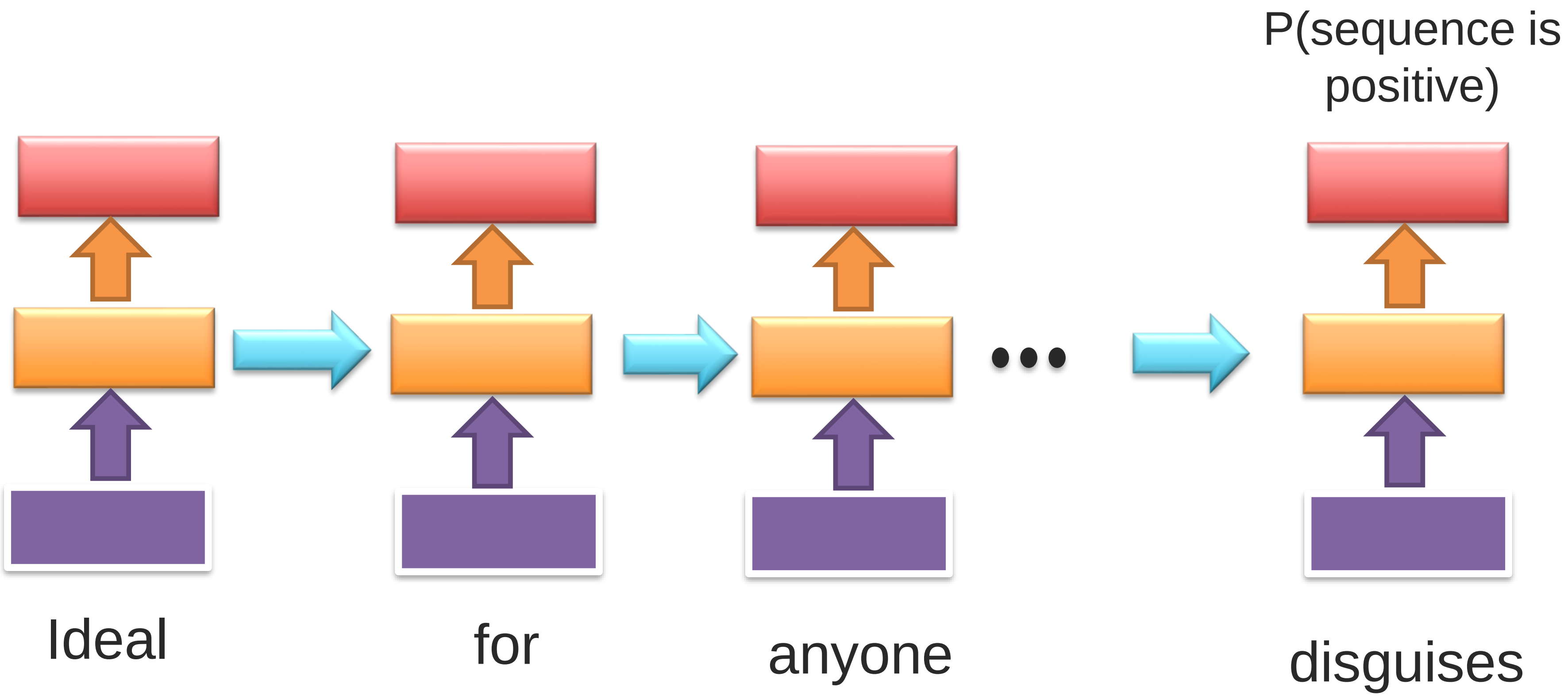
$\boldsymbol{U}$ $$\nabla_{\boldsymbol{U}} L = \sum_t (\nabla_{\boldsymbol{h}^{(t)}} L) \frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{U}}$$

RNN for Sequence Prediction



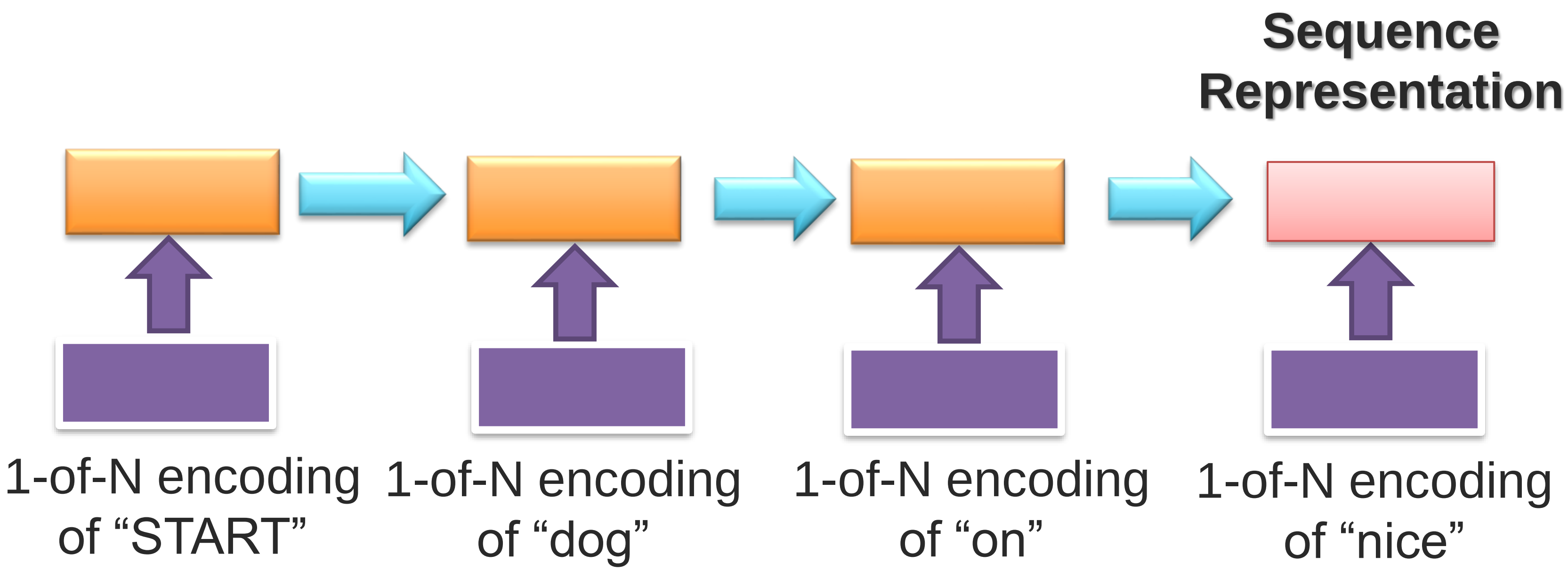
What is the loss? $L = \frac{1}{N}\sum_t L^{(t)} = \frac{1}{N}\sum_t -logP(Y = y^{(t)}|z^{(t)})$

RNN for Sequence Prediction



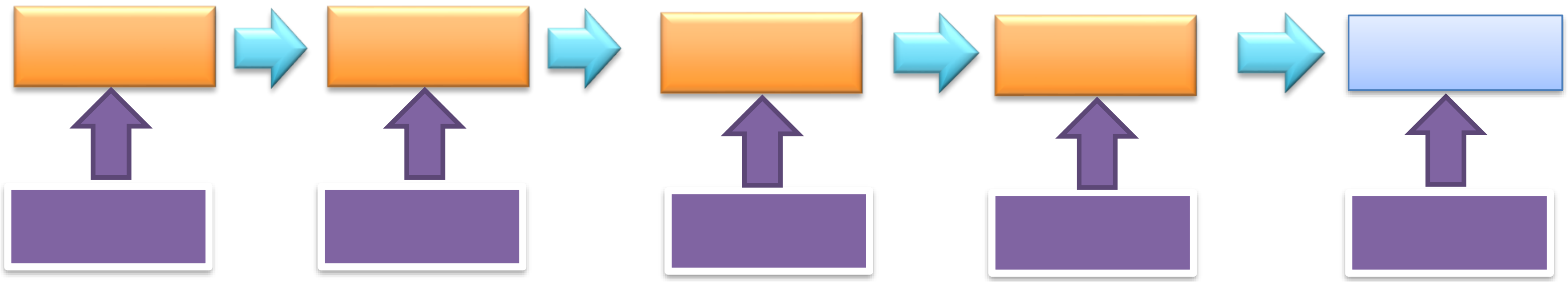
What is the loss? $L = L^{(N)} = -logP(Y = y^{(N)}|z^{(N)})$

RNN for Sequence Representation (Encoder)



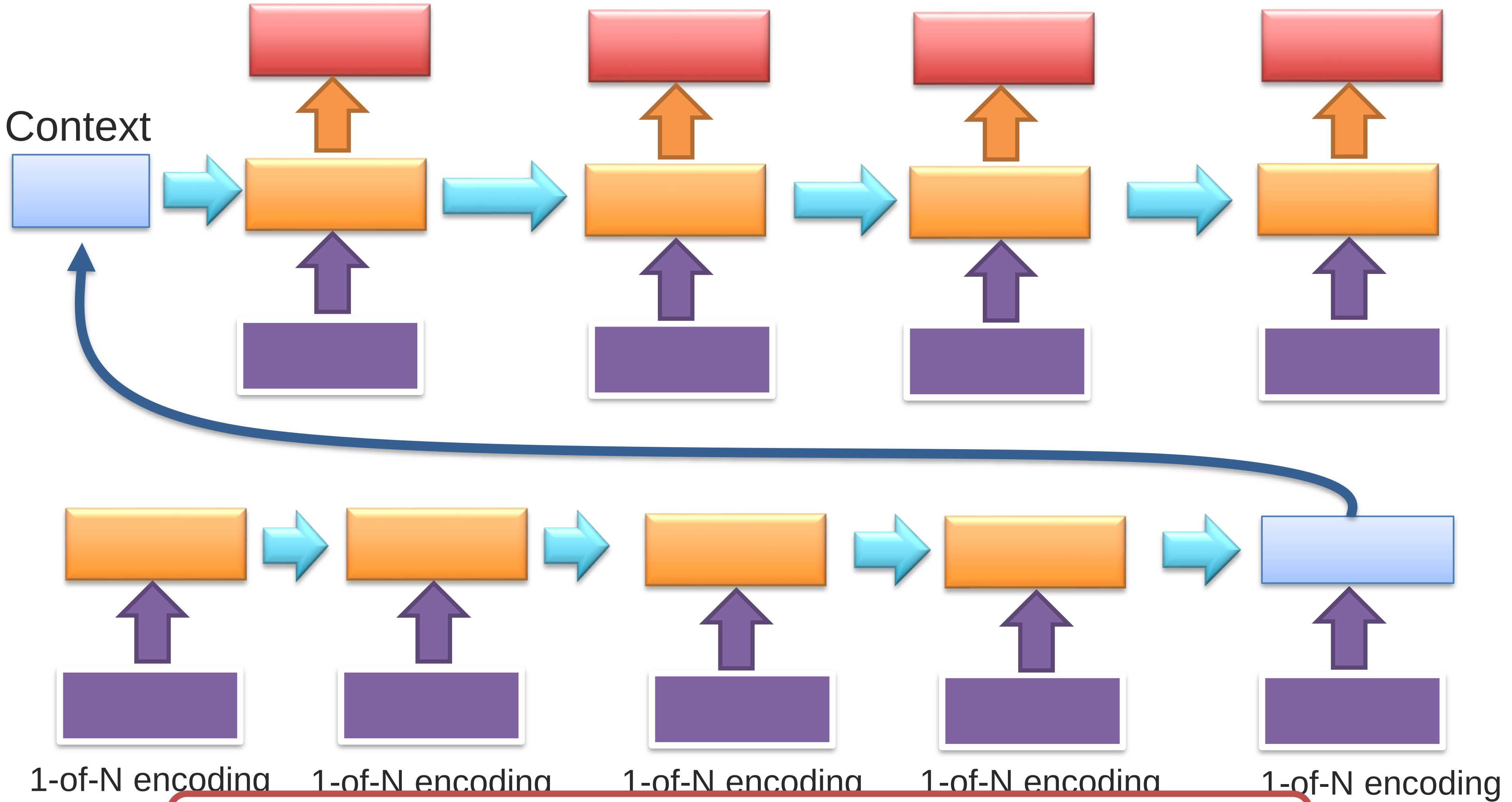
RNN-based for Machine Translation

Le chien sur la plage → The dog on the beach



1-of-N encoding of “le” | 1-of-N encoding of “chien” | 1-of-N encoding of “sur” | 1-of-N encoding of “la” | 1-of-N encoding of “plage”

Encoder-Decoder Architecture



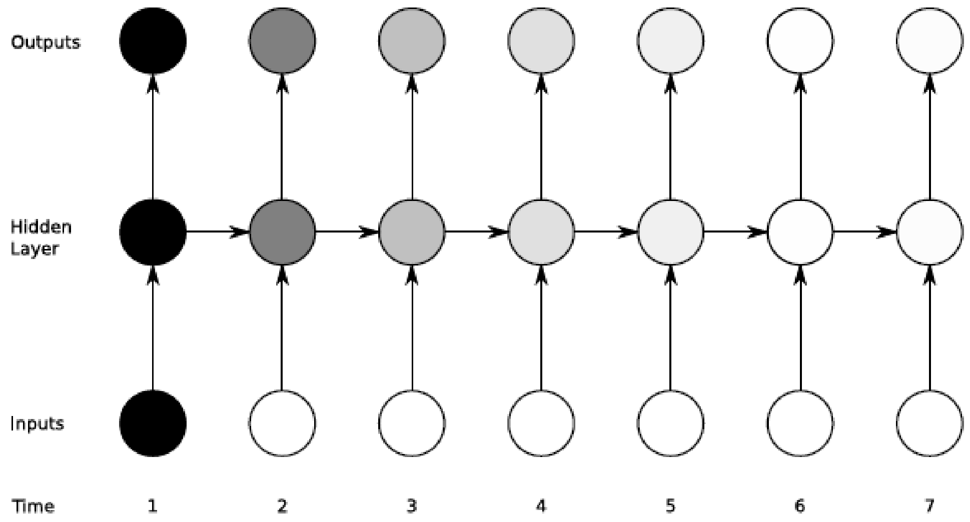
What is the loss function?

Gated Recurrent Neural Networks

Long-term Dependencies

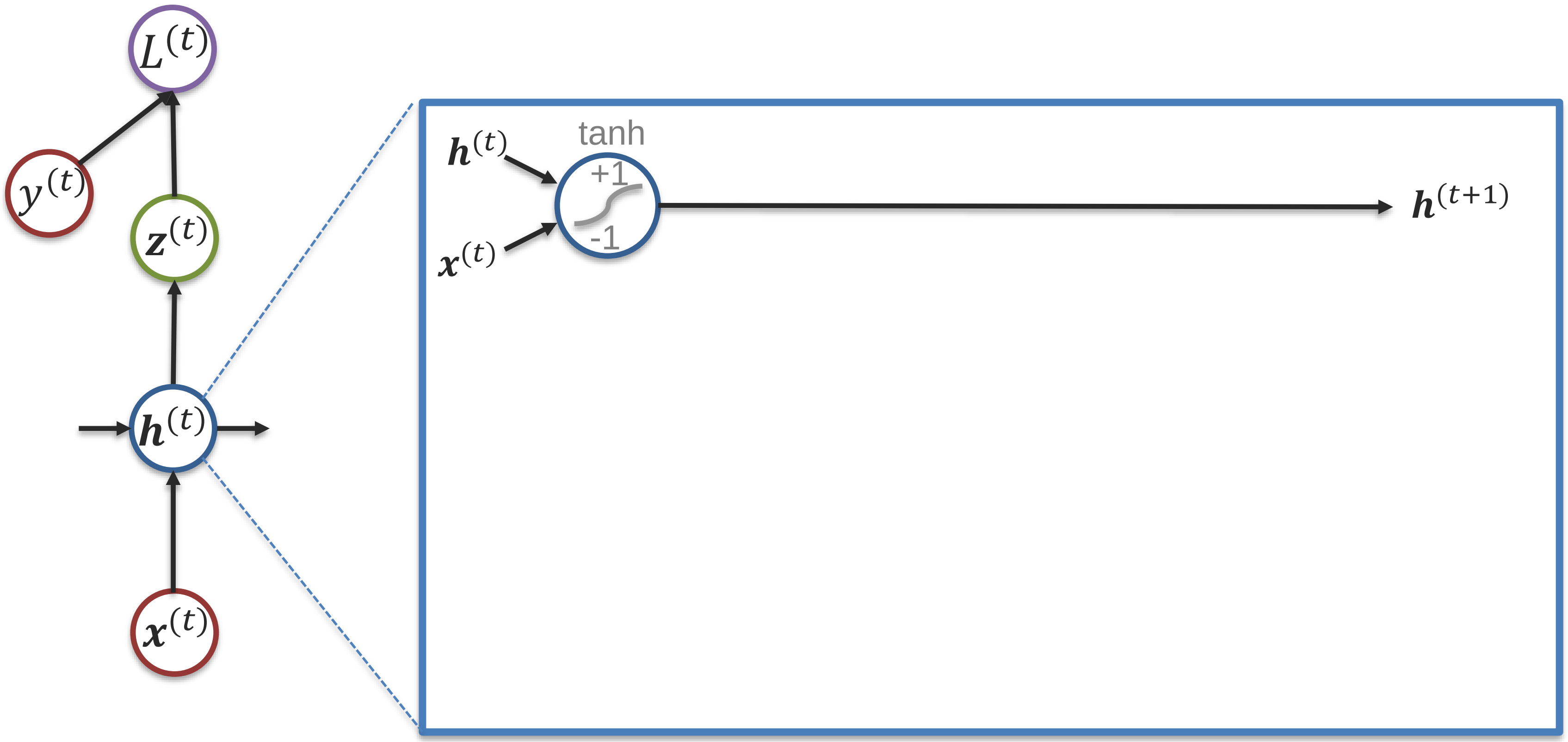
Vanishing gradient problem for RNNs:

$$\boldsymbol{h}^{(t)} \sim tanh(\boldsymbol{W}\boldsymbol{h}^{(t-1)})$$



- The influence of a given input on the hidden layer, and therefore on the network output, either decays or blows up exponentially as it cycles around the network's recurrent connections.

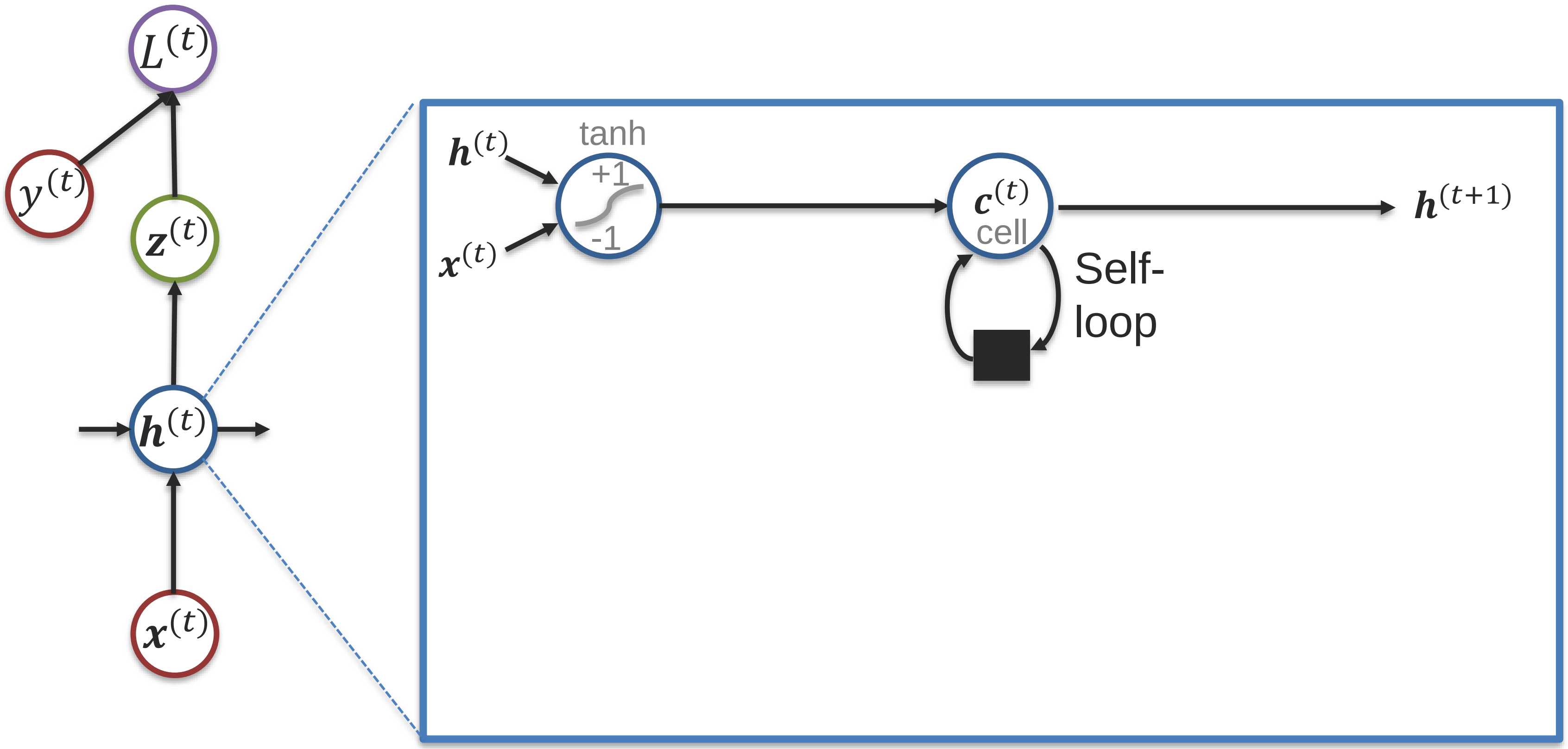
Recurrent Neural Networks



LSTM ideas: (1) “Memory” Cell and Self Loop

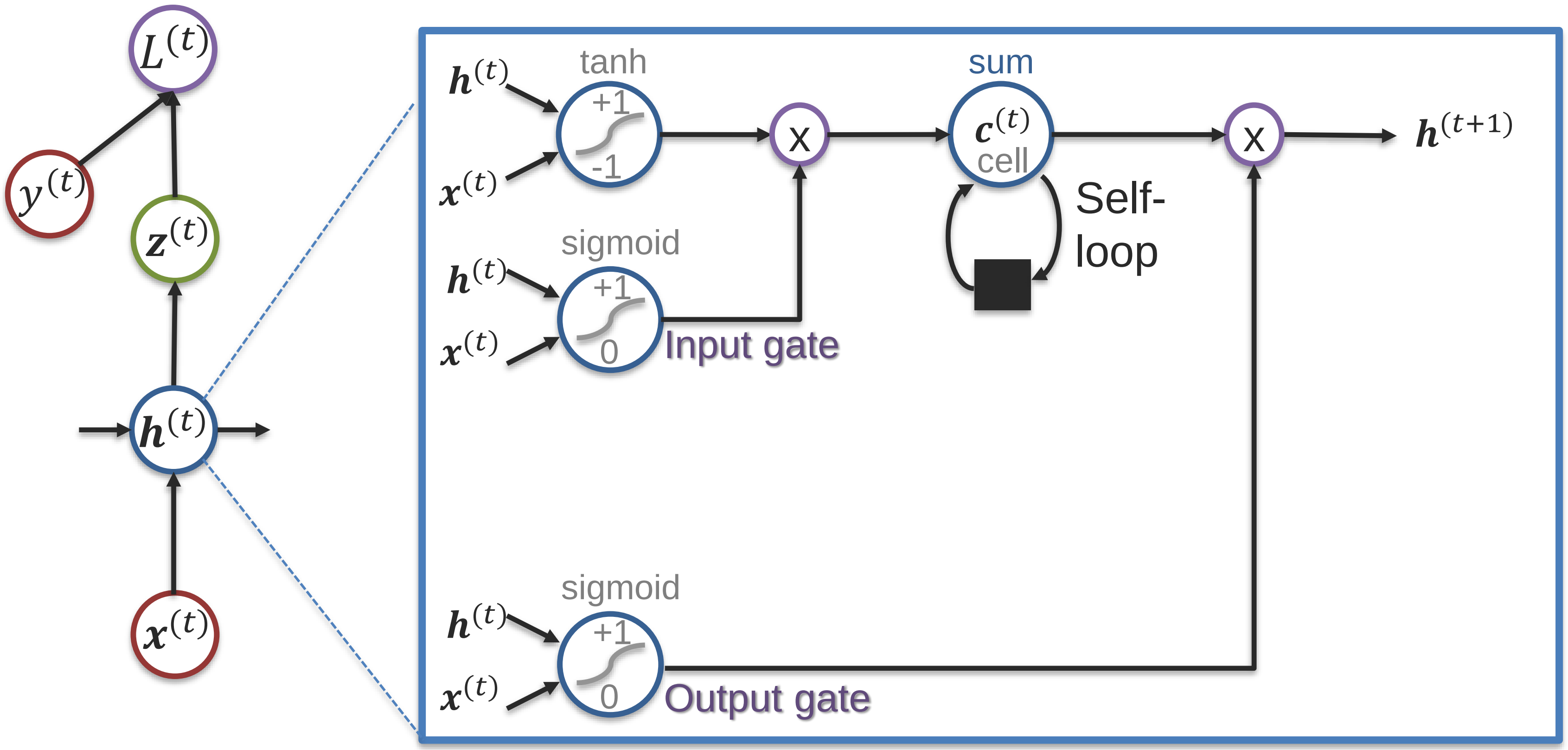
[Hochreiter and Schmidhuber, 1997]

Long Short-Term Memory (LSTM)



LSTM Ideas: (2) Input and Output Gates

[Hochreiter and Schmidhuber, 1997]



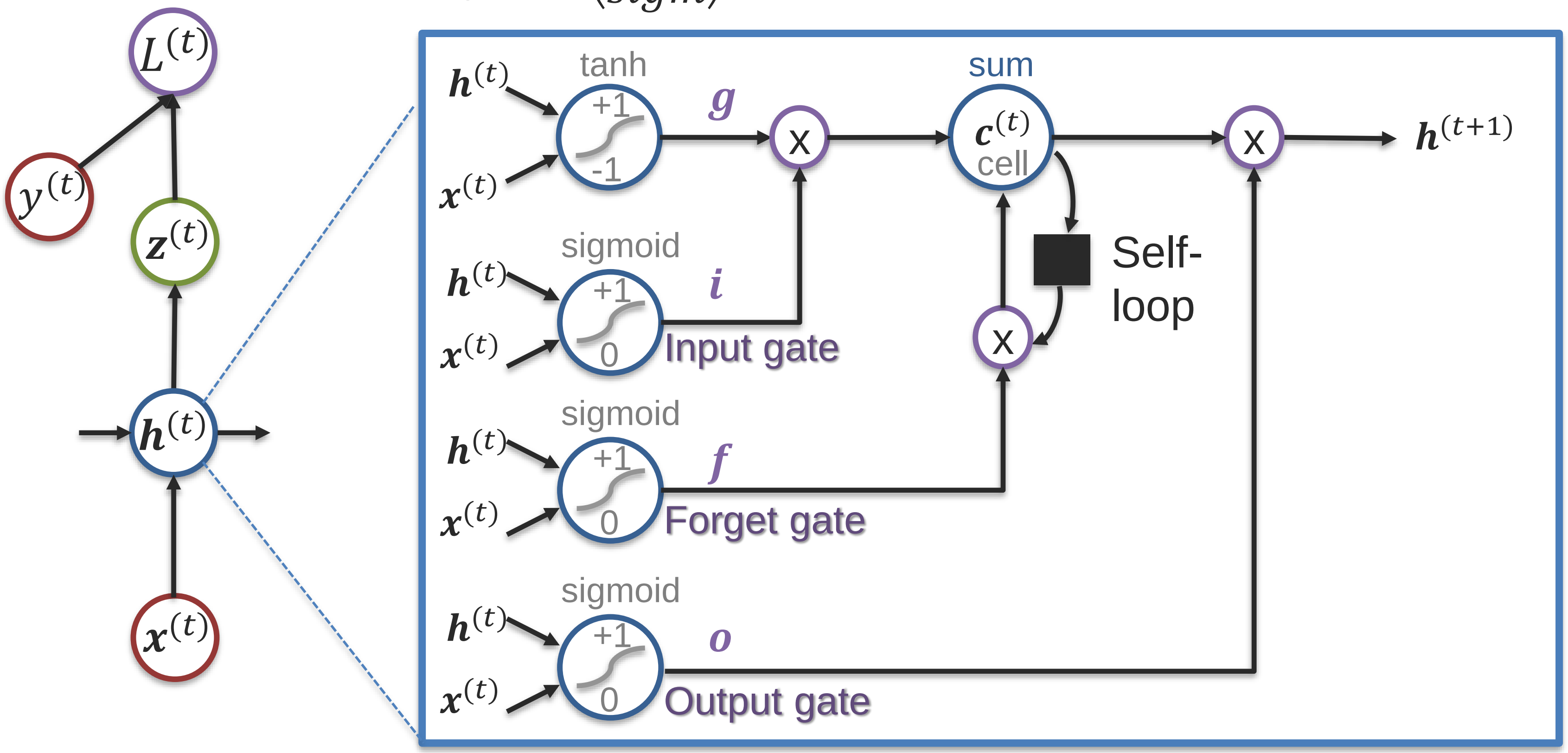
LSTM Ideas: (3) Forget Gate

[Gers et al., 2000]

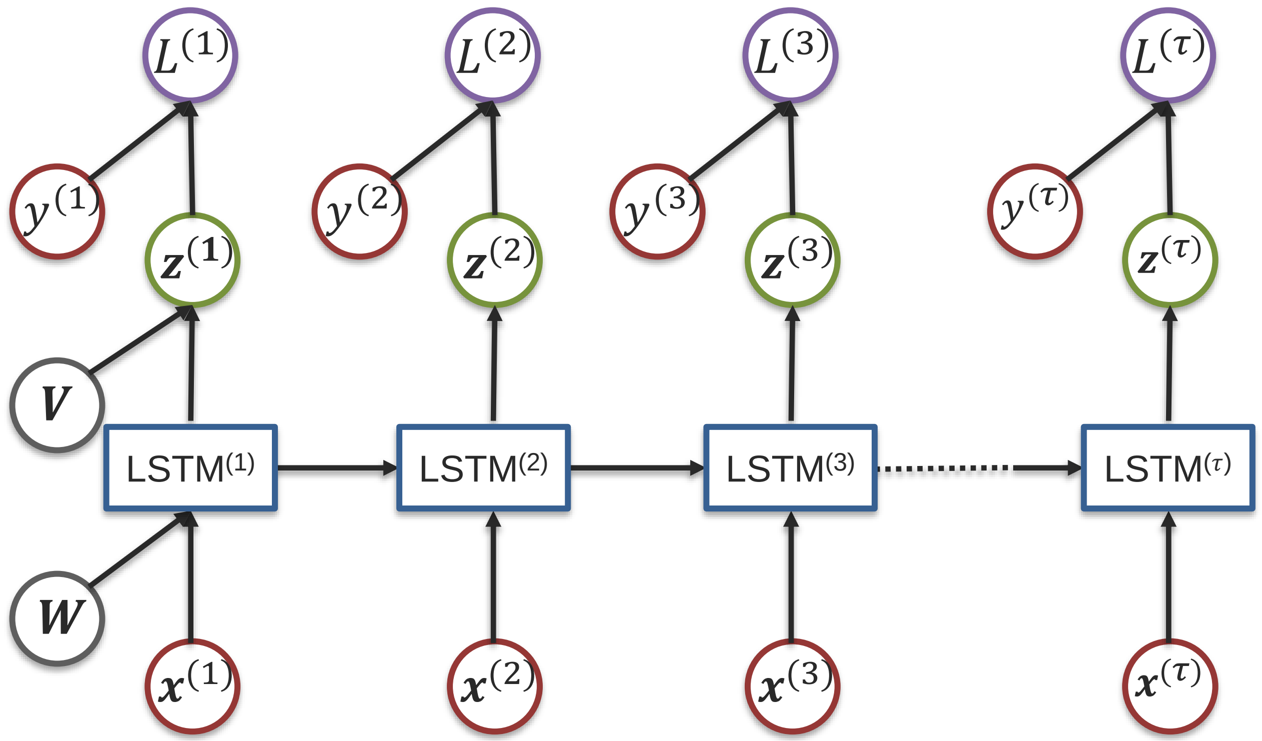
$$\begin{pmatrix} g \\ i \\ f \\ o \end{pmatrix} = \begin{pmatrix} tanh \\ sigm \\ sigm \\ sigm \end{pmatrix} W \begin{pmatrix} h^{(t)} \\ x^{(t)} \end{pmatrix}$$

$$c^{(t)} = f \odot c^{(t-1)} + i \odot g$$

$$h^{(t)} = o \odot \tanh(c^{(t)})$$

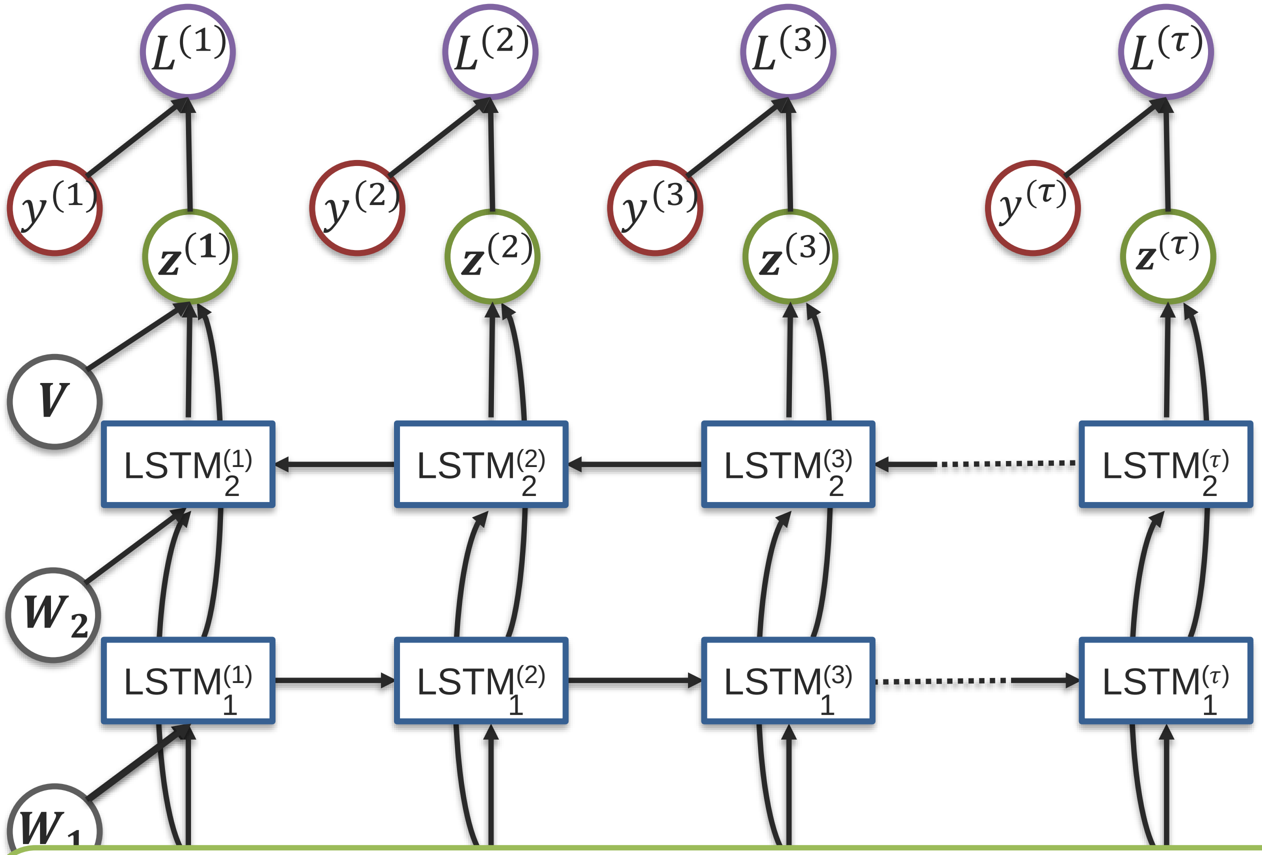


Recurrent Neural Network using LSTM Units



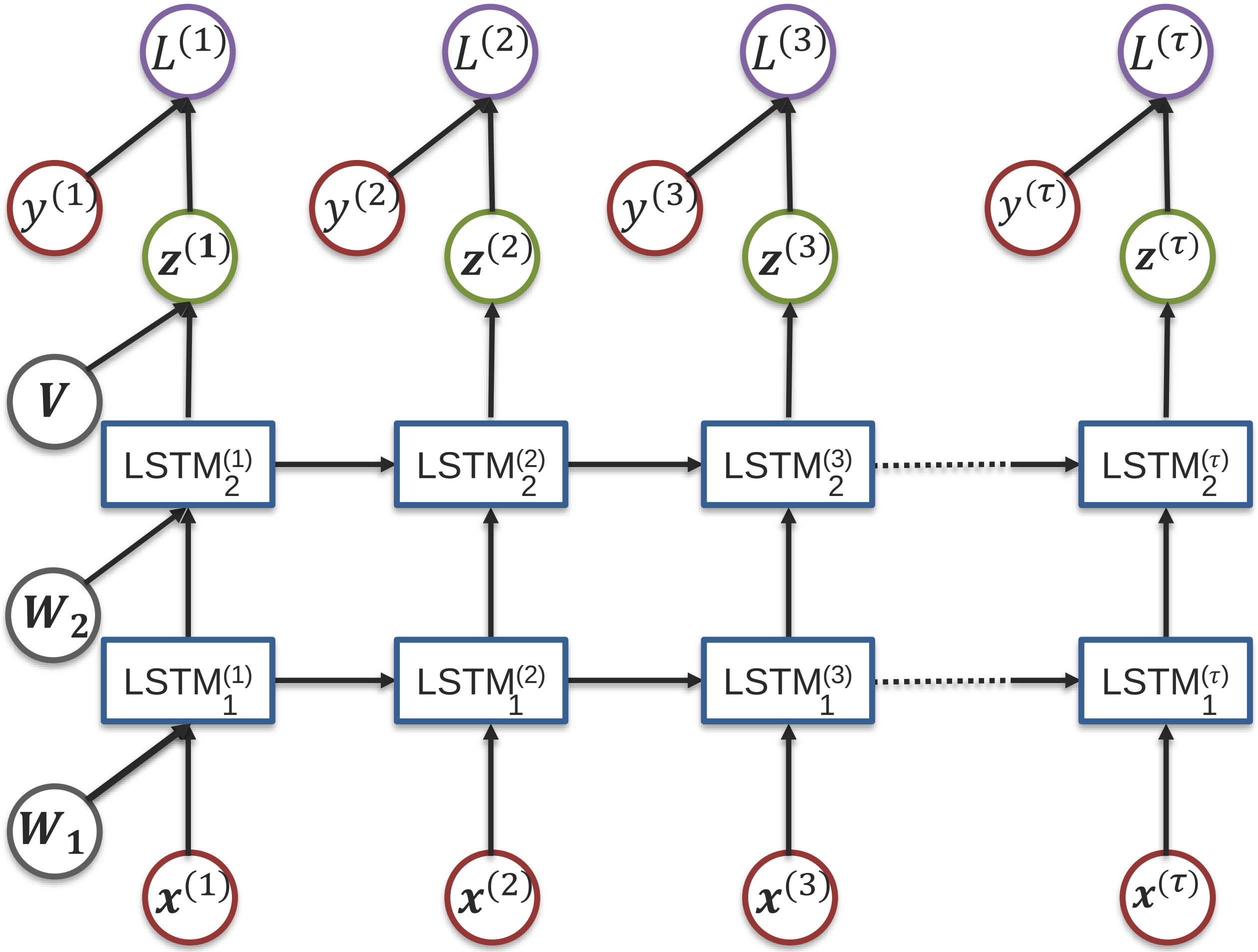
Gradient can still be computer using backpropagation!

Bi-directional LSTM Network



ELMO: Two bi-directional LSTMs are used to contextualize the word embeddings
https://allennlp.org/elmo

Deep LSTM Network

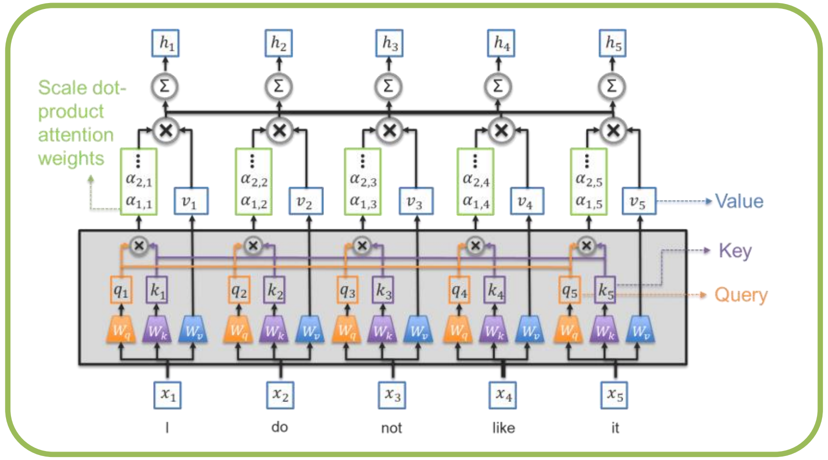


And There Are More Ways To Model Sequences…

… in Week 5!

Self-attention Models

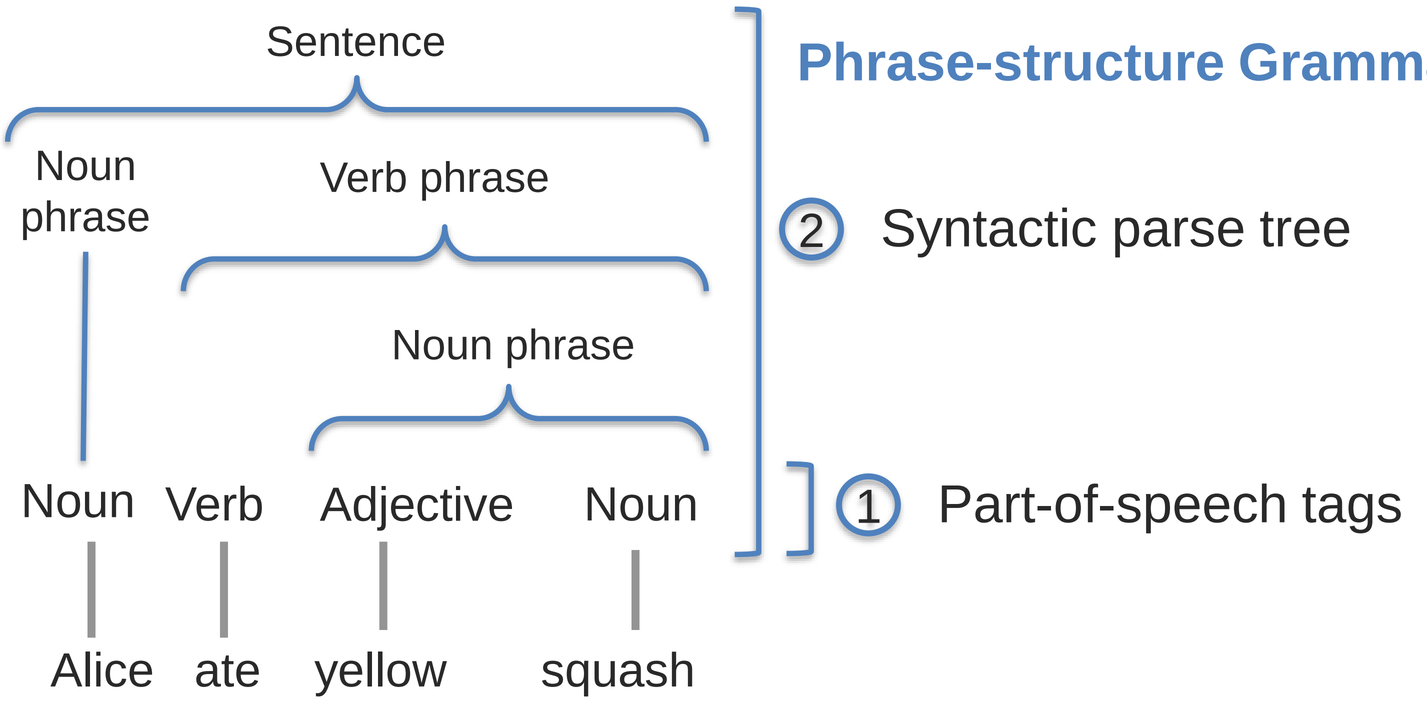
(e.g., BERT, RoBERTa)



Syntax and Language Structure

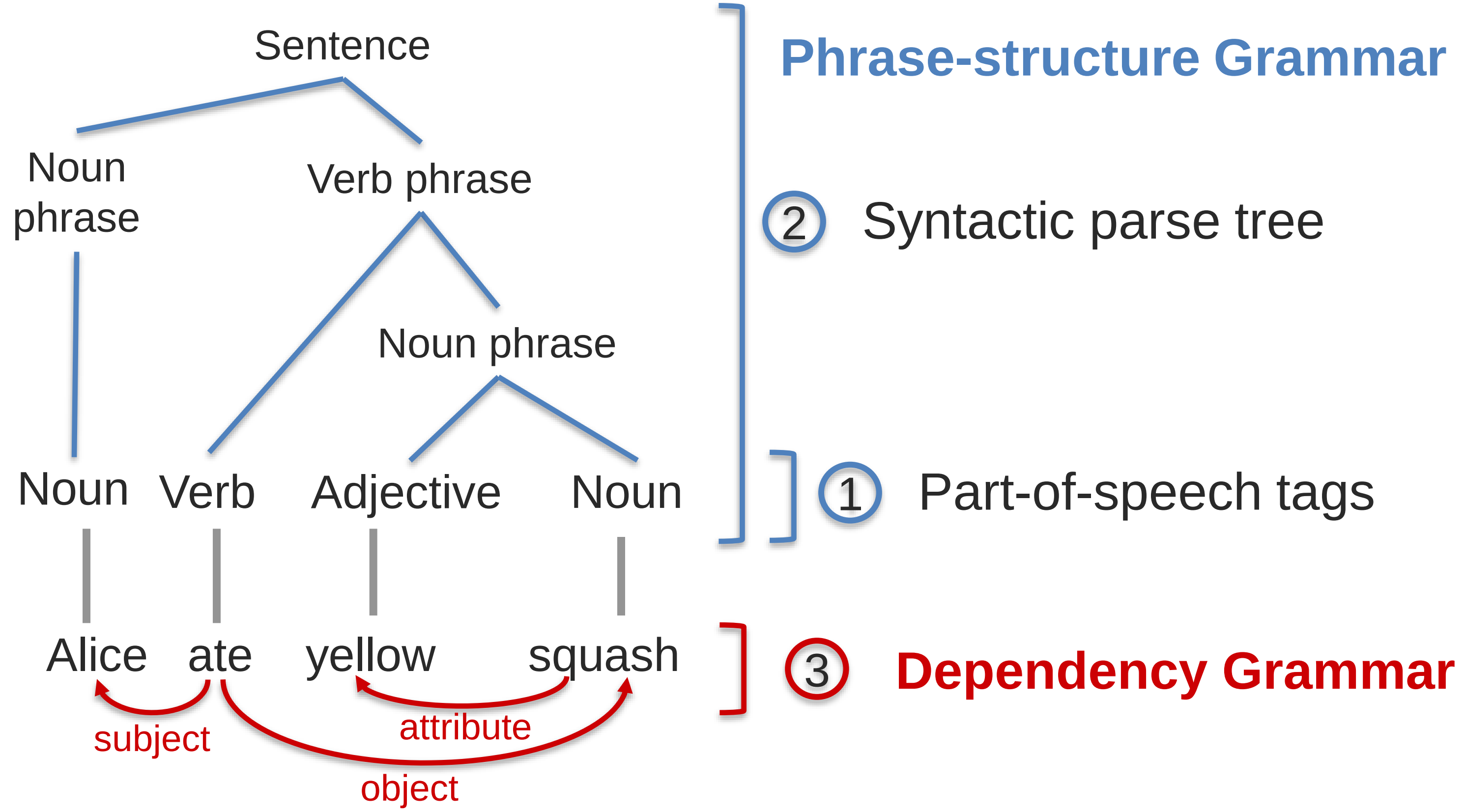
Syntax and Language Structure

What can you tell about this sentence?



Syntax and Language Structure

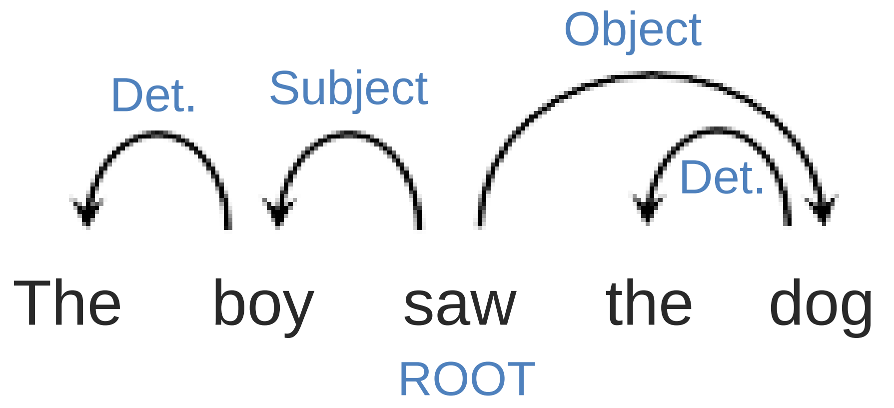
What can you tell about this sentence?



Dependency Grammar

Main idea: Syntactic structure consists of *lexical items*, linked by binary asymmetric relations called *dependencies*

- Easier to convert to predicate-argument structure
- You can try to convert one representation into another
  - But, in general, these formalisms are not equivalent



Ambiguity in Syntactic Parsing

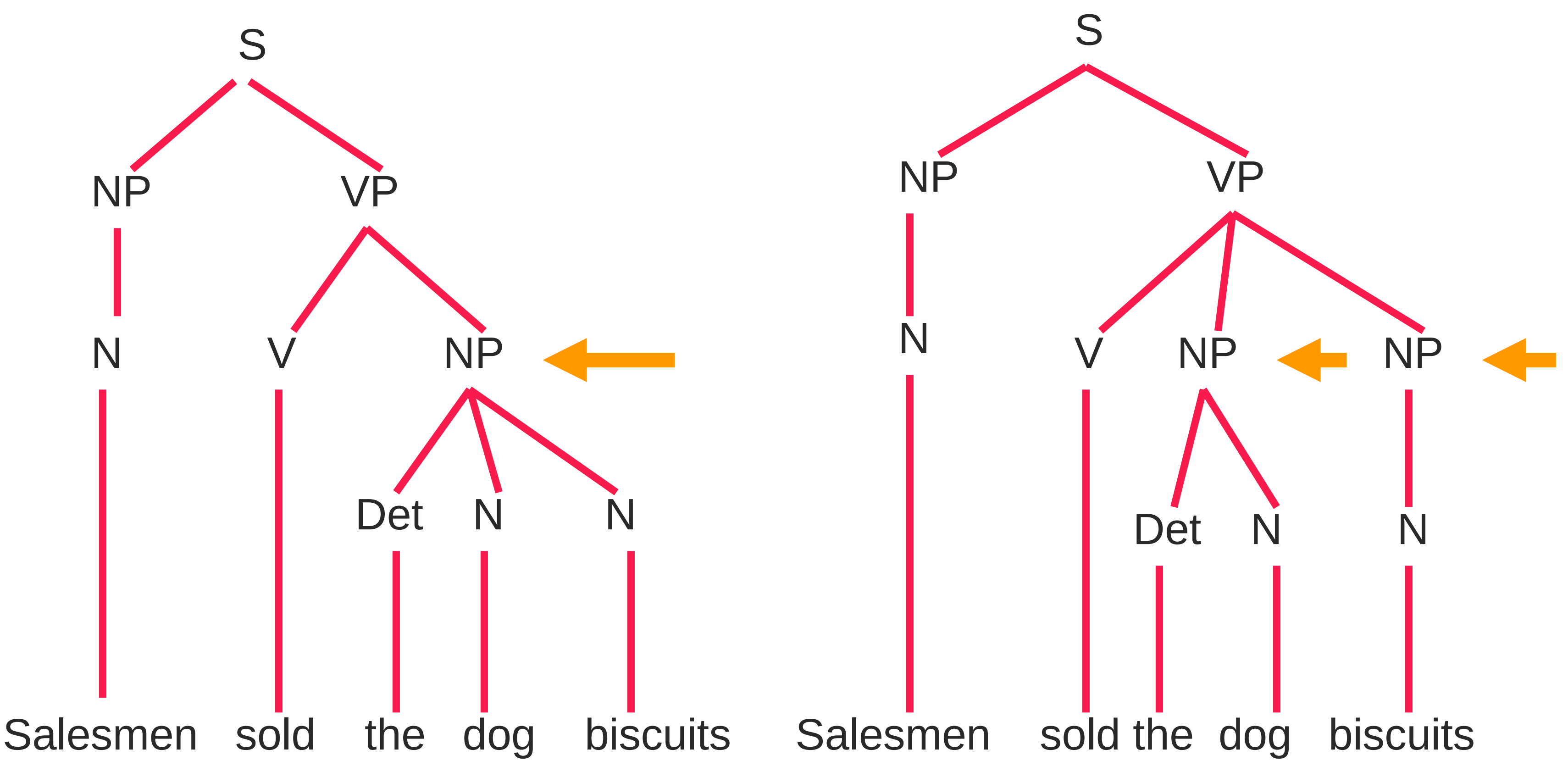
“Like” can be a verb or a preposition

- I like/VBP candy.
- Time flies like/IN an arrow.

“Around” can be a preposition, particle, or adverb

- I bought it at the shop around/IN the corner.
- I never got around/RP to getting a car.
- A new Prius costs around/RB $25K.

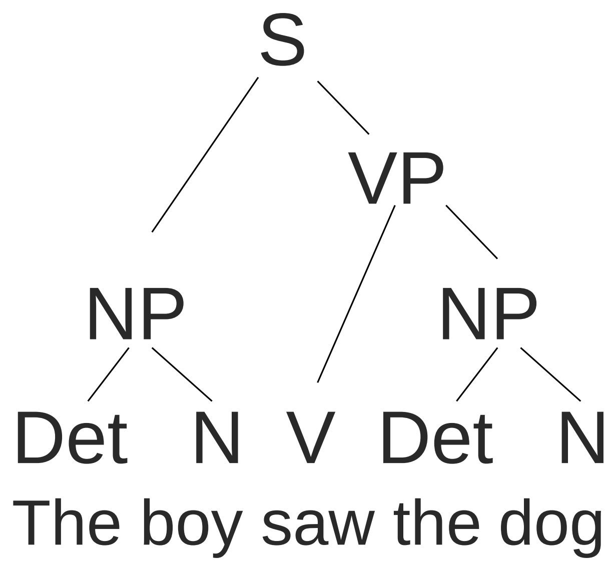
Language Ambiguity



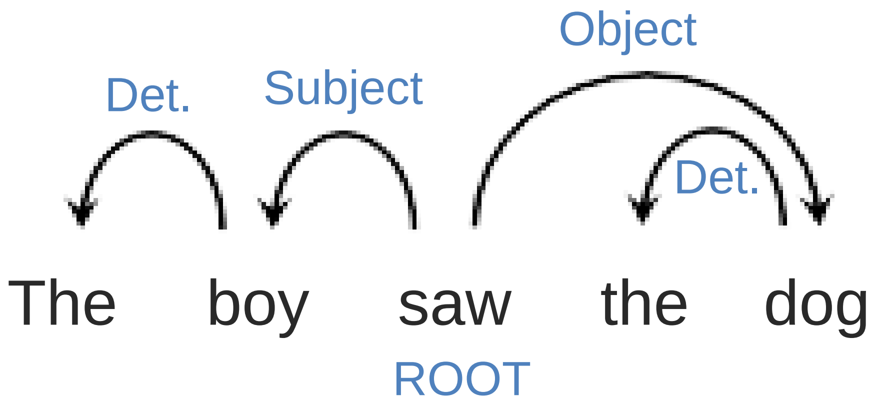
Language Syntax – Examples

Det Noun Verb Det Noun Prep Det Noun
The boy saw the dog in the park

Part of Speech tagging



Constituency Parsing

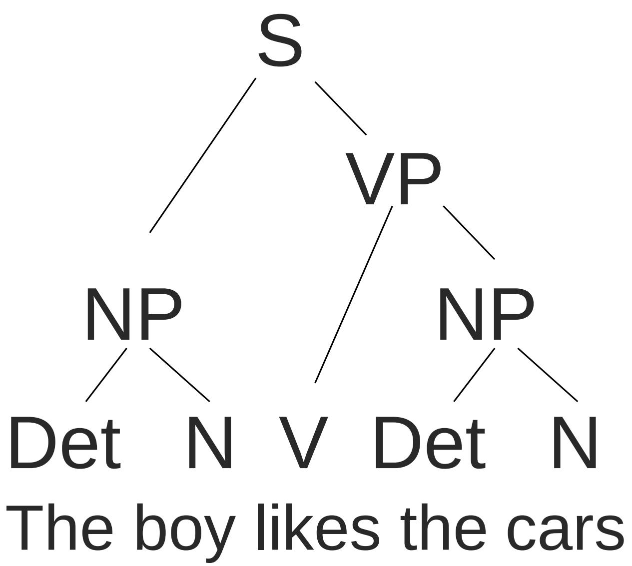


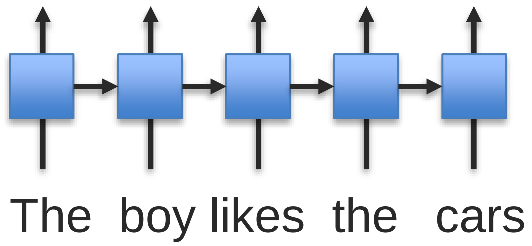
Dependency Parsing

How to take advantage of syntax when modeling language with neural networks?

Recursive Neural Network

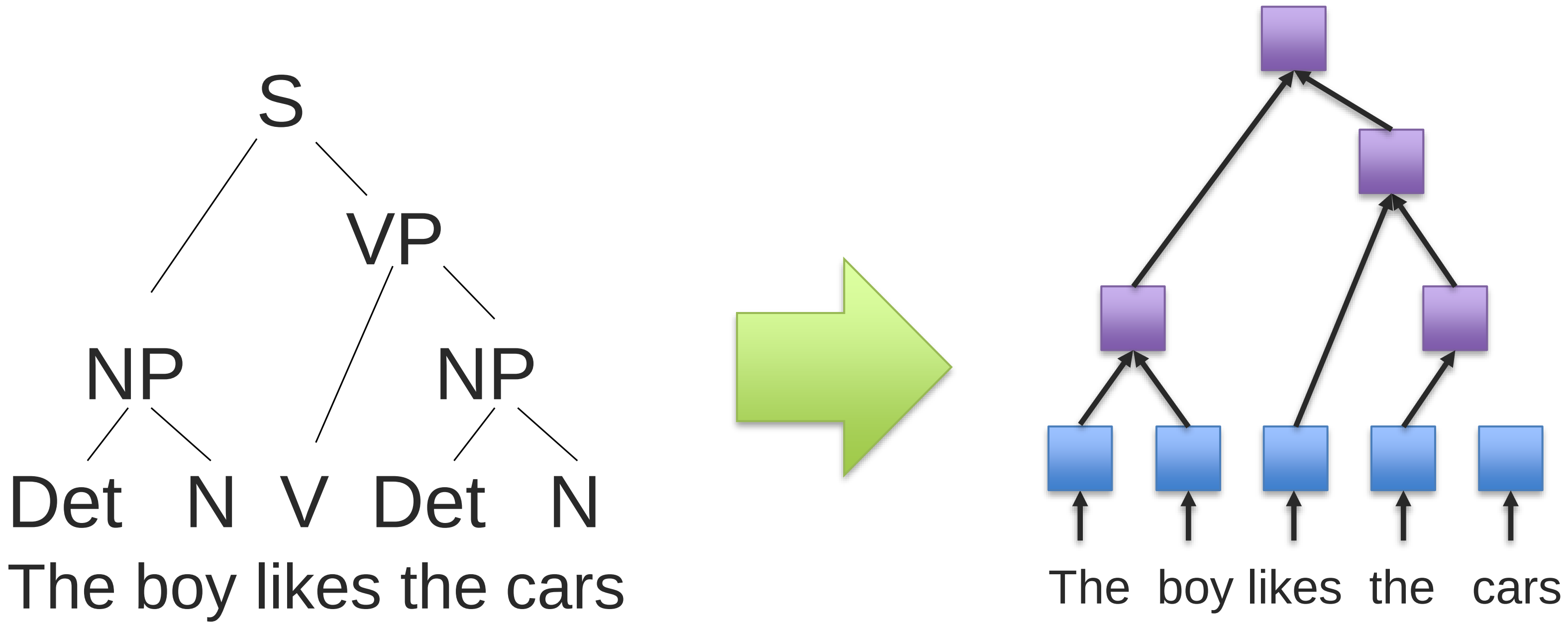
How to Model Syntax with RNNs?





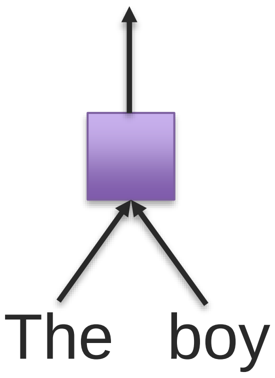
We could use Part-of-Speech tags.

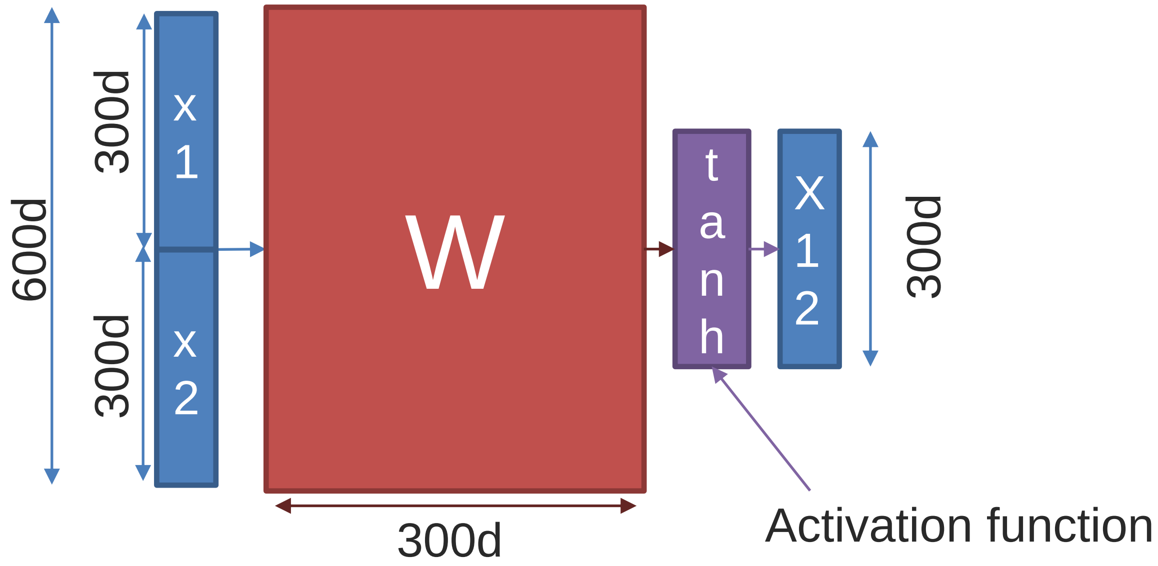
Tree-based RNNs (or Recursive Neural Network)



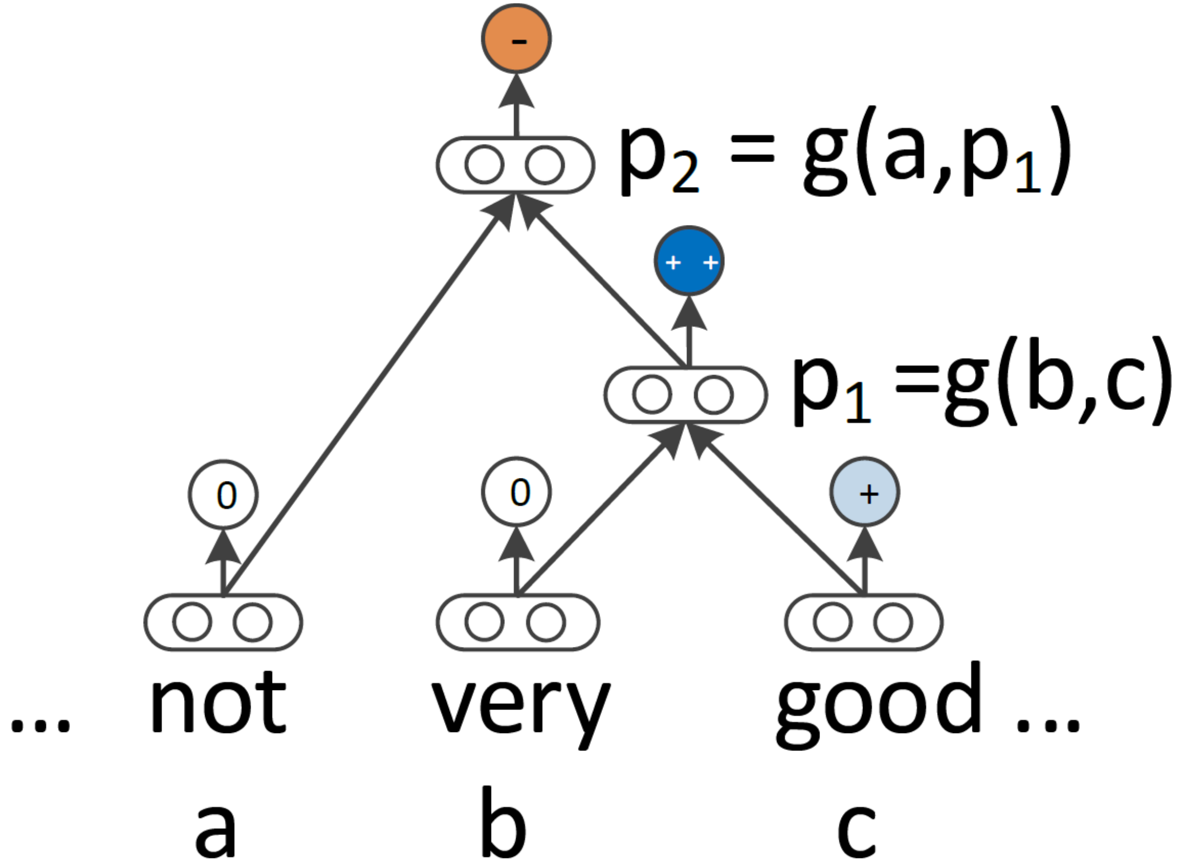
Recursive Neural Unit

Pair-wise combination of two input features





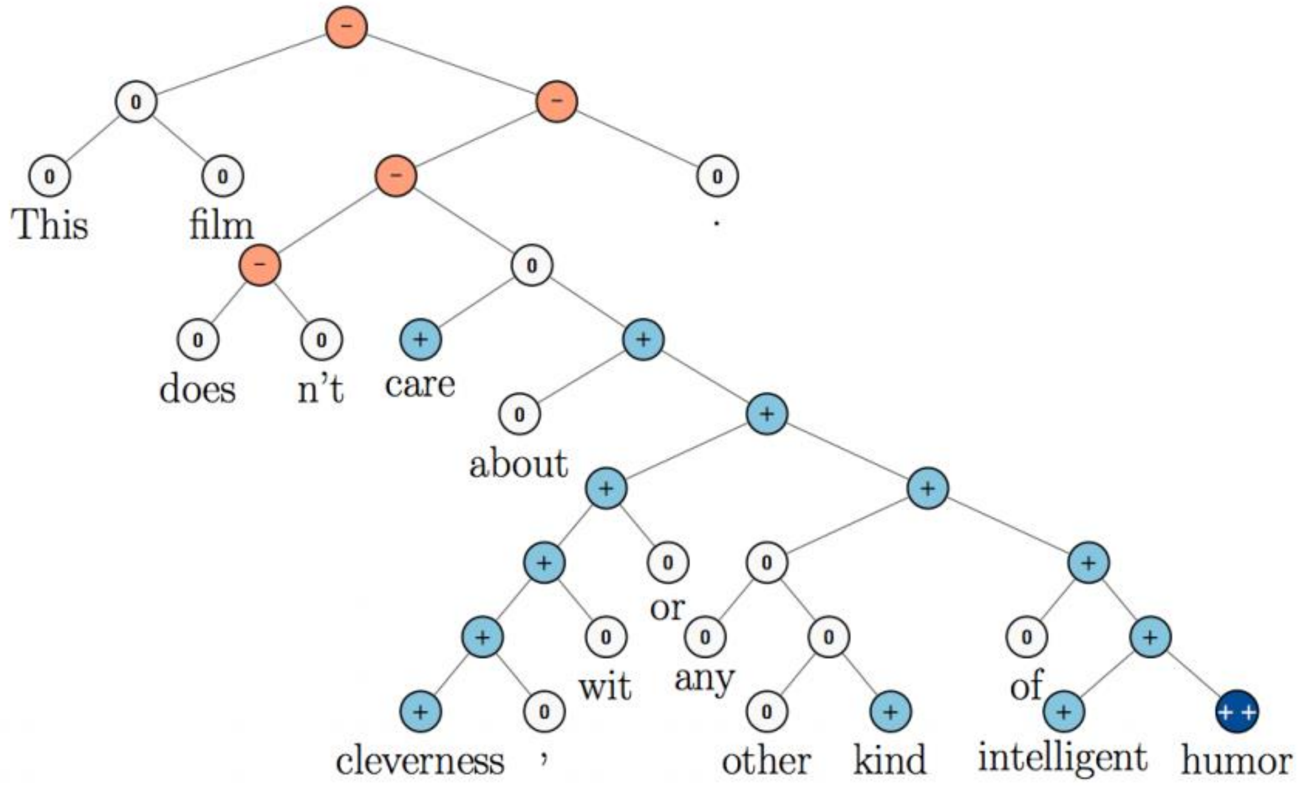
Recursive Neural Network for Sentiment Analysis



Socher et al., Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, EMNLP 2013

Recursive Neural Network for Sentiment Analysis

Classification of a sentence using tree-based compositionality of words



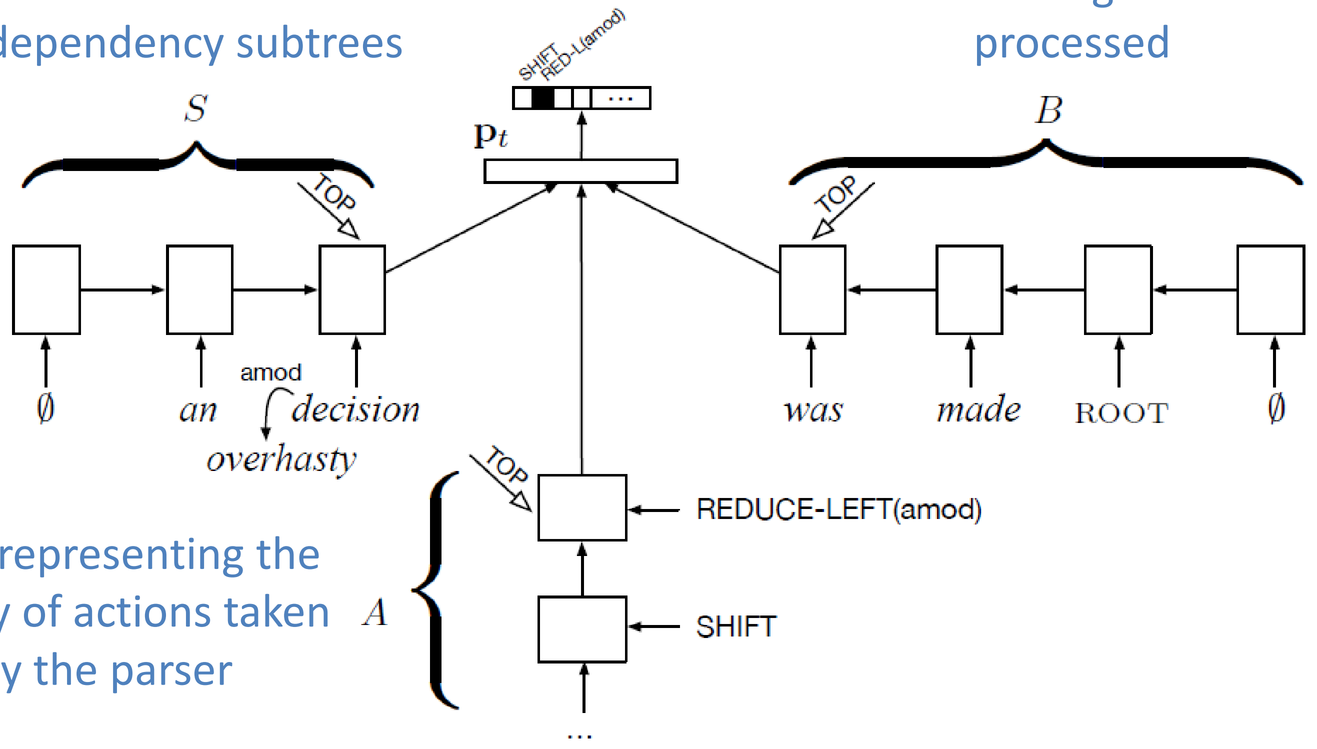
Demo: http://nlp.stanford.edu/sentiment/

Socher et al., Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, EMNLP 2013

Stack LSTM

stack of partially constructed dependency subtrees

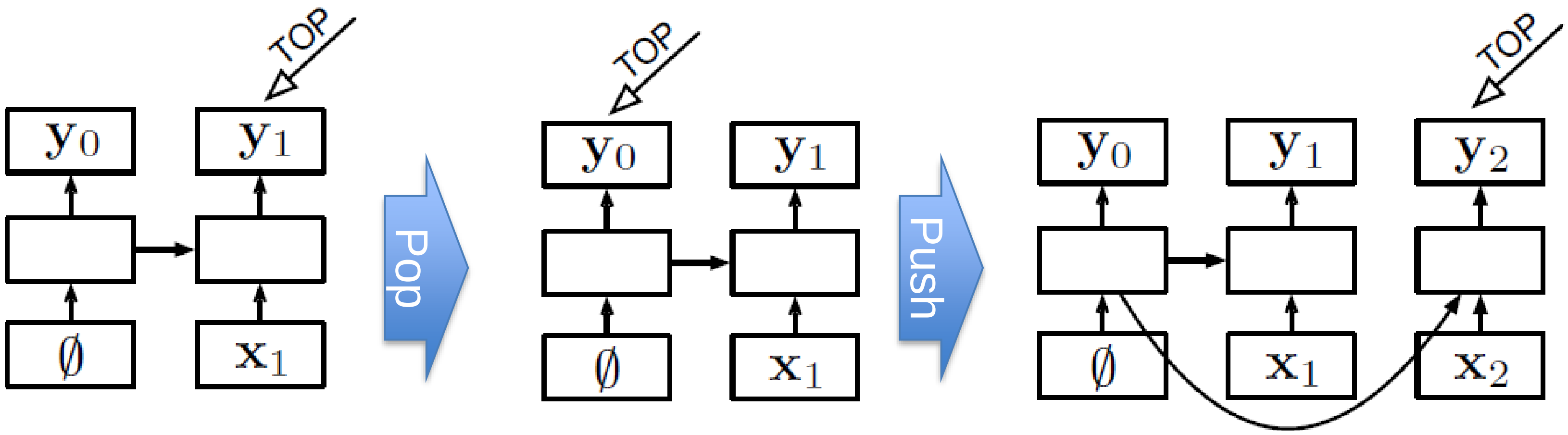
buffer of words remaining to be processed



stack representing the history of actions taken by the parser

Dyer et al., Transition-Based Dependency Parsing with Stack Long Short-Term Memory, 2015

Stack LSTM



Dyer et al., Transition-Based Dependency Parsing with Stack Long Short-Term Memory, 2015

Resources

- Stanford NLP software

https://nlp.stanford.edu/software/

  - Stanford Parser
  - Stanford POS Tagger
- UC Berkeley Parser

https://github.com/slavpetrov/berkeleyparser

- Parsers by Kenji Sagae (syntactic parsers) http://www.sagae.org/software.html