Language Technologies Institute

Carnegie Mellon University

Multimodal Machine Learning

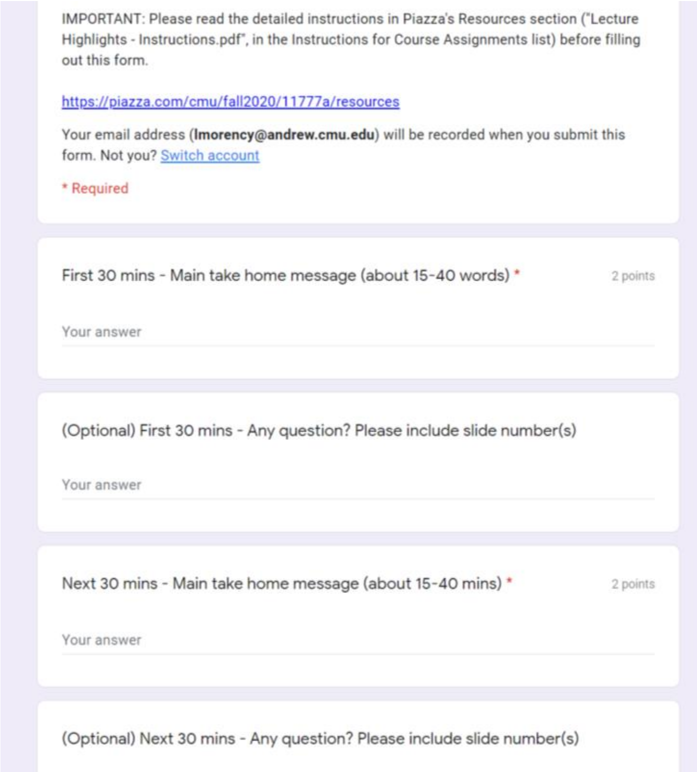
Lecture 2.2: Unimodal Representations (Part 2)

Louis-Philippe Morency

** Co-lecturer: Paul Liang. Original course co-developed with Tadas Baltrusaitis. Spring 2021 and 2022 editions taught by Yonatan Bisk*

Administrative Stuff

Lecture Highlight Form



IMPORTANT: Please read the detailed instructions in Piazza's Resources section ("Lecture Highlights - Instructions.pdf", in the Instructions for Course Assignments list) before filling out this form.

https://piazza.com/cmu/fall2020/11777a/resources

Your email address (**lmorency@andrew.cmu.edu**) will be recorded when you submit this form. Not you? Switch account

* Required

First 30 mins - Main take home message (about 15-40 words) * 2 points

Your answer

(Optional) First 30 mins - Any question? Please include slide number(s)

Your answer

Next 30 mins - Main take home message (about 15-40 mins) * 2 points

Your answer

(Optional) Next 30 mins - Any question? Please include slide number(s)

Deadline: Today, Thursday at 9pm ET

Use your Andrew CMU email

- You will need to login using this address

New form for each lecture

- Posted on Piazza's Resources section

You should start taking notes as soon as the administrative stuff is over!

Contact us if you have any problem

Reading Assignments – Weekly Schedule

Four main steps for the reading assignments

1. Monday 8pm: Official start of the assignment
2. Wednesday 8pm: Select your paper
3. **Friday 8pm:** Post your summary
4. **Monday 8pm:** Post your extra comments (3 posts)

Team Matching Event – Today!

Today around 10:30am ET

(later part of the lecture)

- Detailed instructions will be shared during lecture
- Event optional for students who already have a full team

AWS Credits

New procedure this semester!

- We need your AWS account info (deadline: Tuesday 9/12)
- Max $150 credit for the whole semester. No exception.
- More details will be sent on Piazza

Alternative: Amazon SageMaker Studio Lab

- Similar to Google Colab (link)
- No cost, easy access to JupyterLab-based user interface
- Access to G4dn.xlarge instances

Language Technologies Institute

Carnegie Mellon University

Multimodal Machine Learning

Lecture 2.2: Unimodal Representations (Part 2)

Louis-Philippe Morency

** Co-lecturer: Paul Liang. Original course co-developed with Tadas Baltrusaitis. Spring 2021 and 2022 editions taught by Yonatan Bisk*

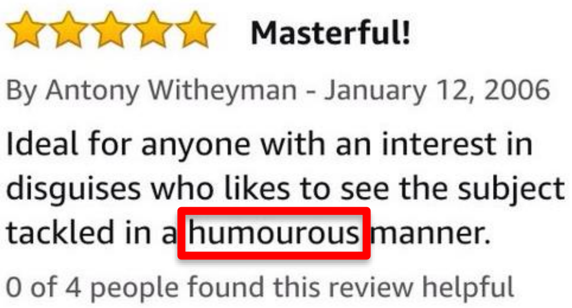
Lecture Objectives

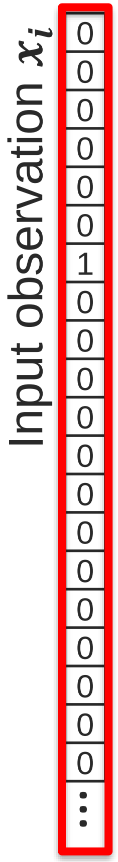
- Word representations
  - Distributional hypothesis
  - Learning neural representations
- Sentence representations and sequence modeling
  - Recurrent neural networks
  - Language models
- Syntax and language structure
  - Phrase-structure and dependency grammars
  - Recursive neural network
    - Tree-based RNN

Word Representations

Simple Word Representation

Written language





“one-hot” vector

$|x_i|$ = number of words in dictionary

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

He was walking away because …
He was running away because …

Geometric interpretation

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

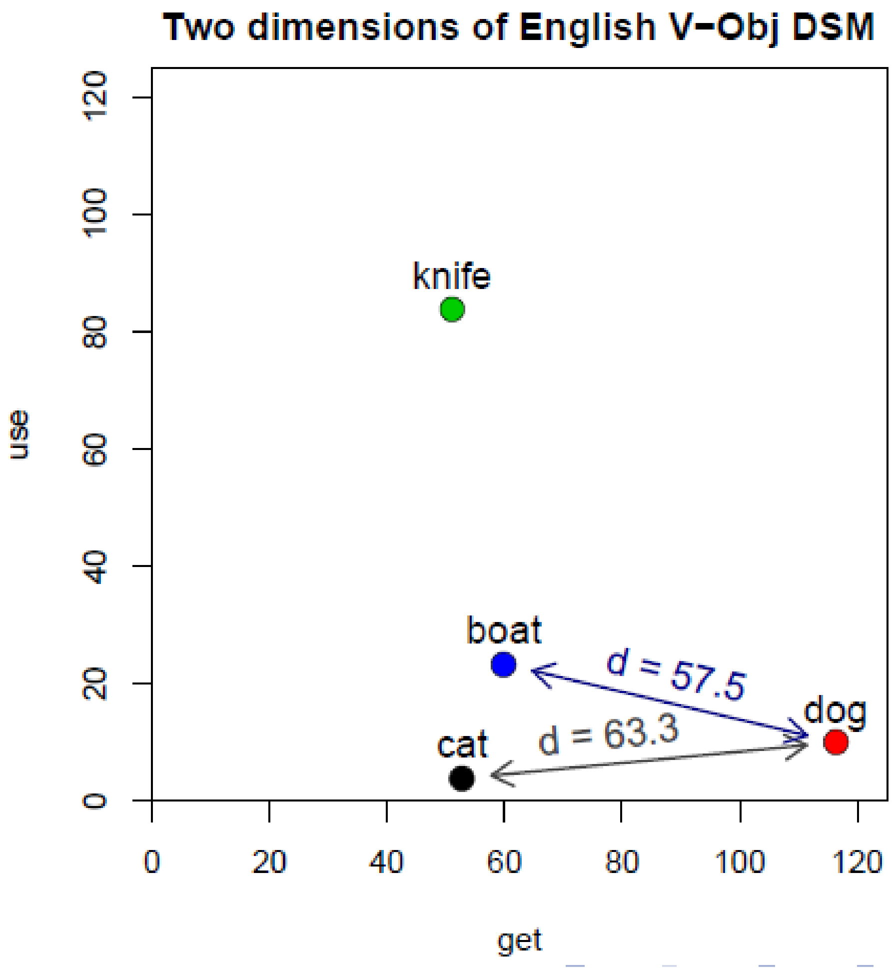
- can be seen as coordinates of point in *n*-dimensional Euclidean space R^n

	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

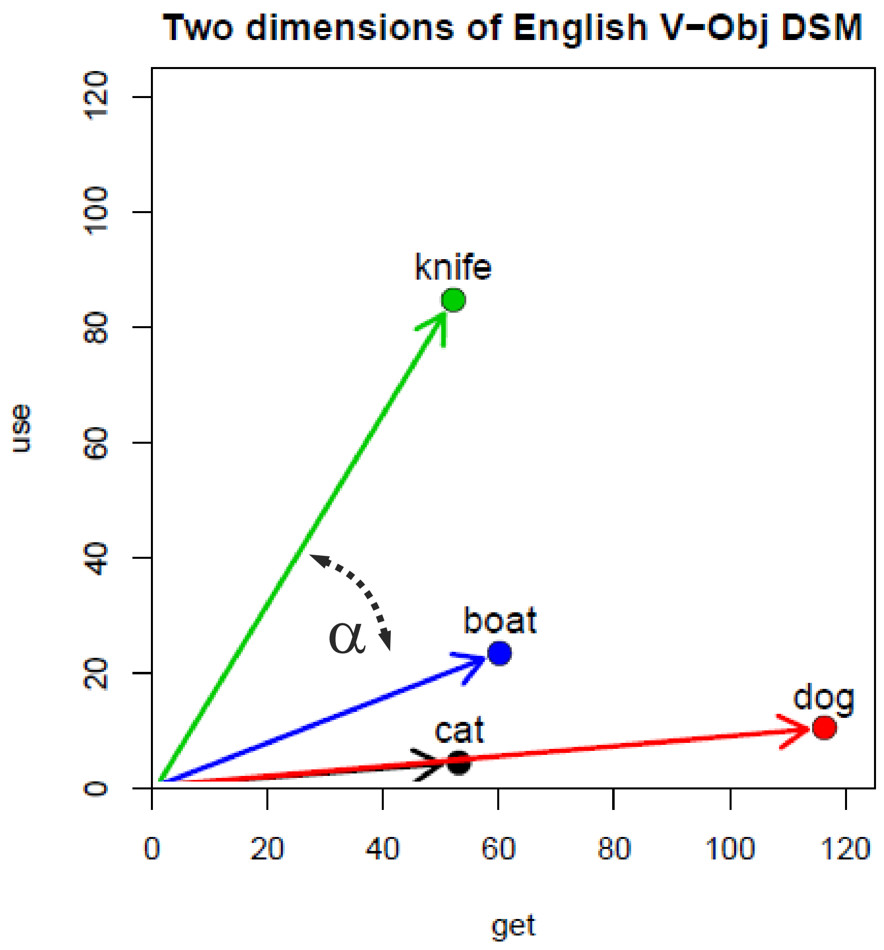
Distance and similarity

- 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?

Distribution hypothesis: Approximate the word meaning by its surrounding words

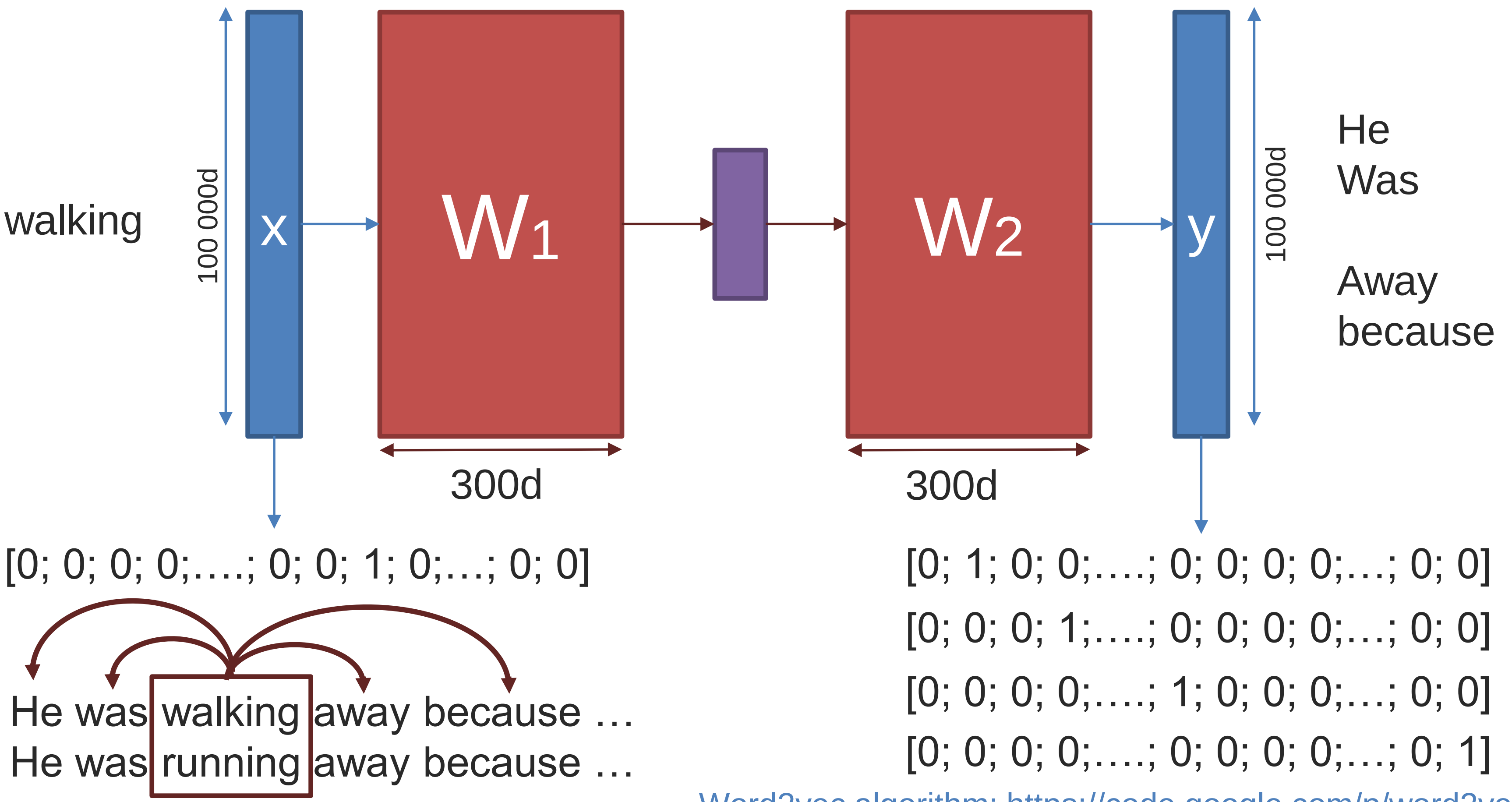
Words used in a similar context will lie close together

He was walking away because …
He was running away because …

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

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

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

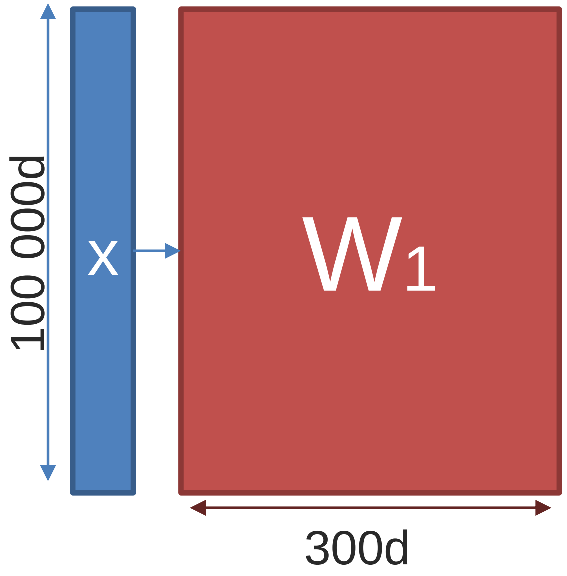
Transform: x’=x*W

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

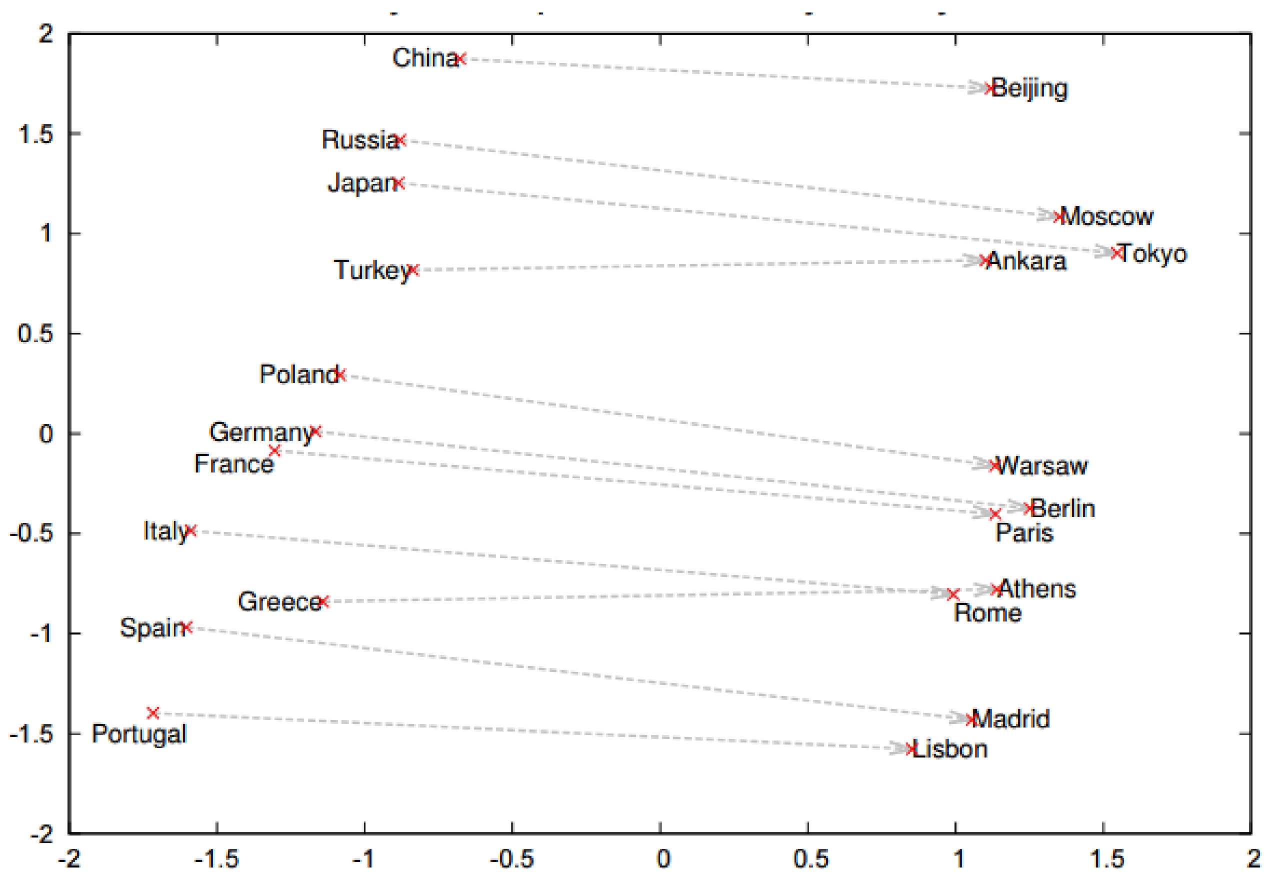


Vector space models of words

- 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:

$$\text{Vec(king)} - \text{vec(man)} + \text{vec(woman)} \approx \text{vec(queen)}$$

Vector space models of words: semantic relationships



Trained on the Google news corpus with over 300 billion words

Do these work?
Issues of bias here

e.g. https://arxiv.org/abs/1607.06520
https://aclanthology.org/W14-1618.pdf

vec(programmer) – vec(man) + vec(woman) ≈ vec(homemaker)

Sentence Modeling and Recurrent Networks

Sentence Modeling: Sequence Prediction

Masterful!

By Antony Witheyman - January 12, 2006

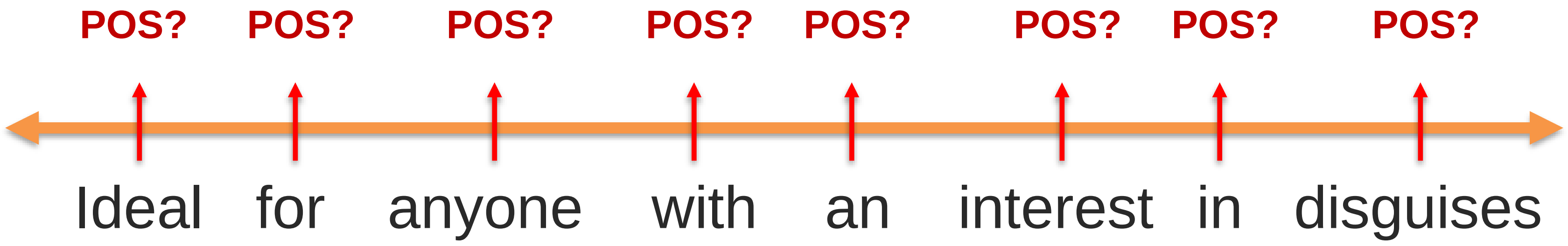
Ideal for anyone with an interest in disguises who likes to see the subject tackled in a humourous manner.

0 of 4 people found this review helpful

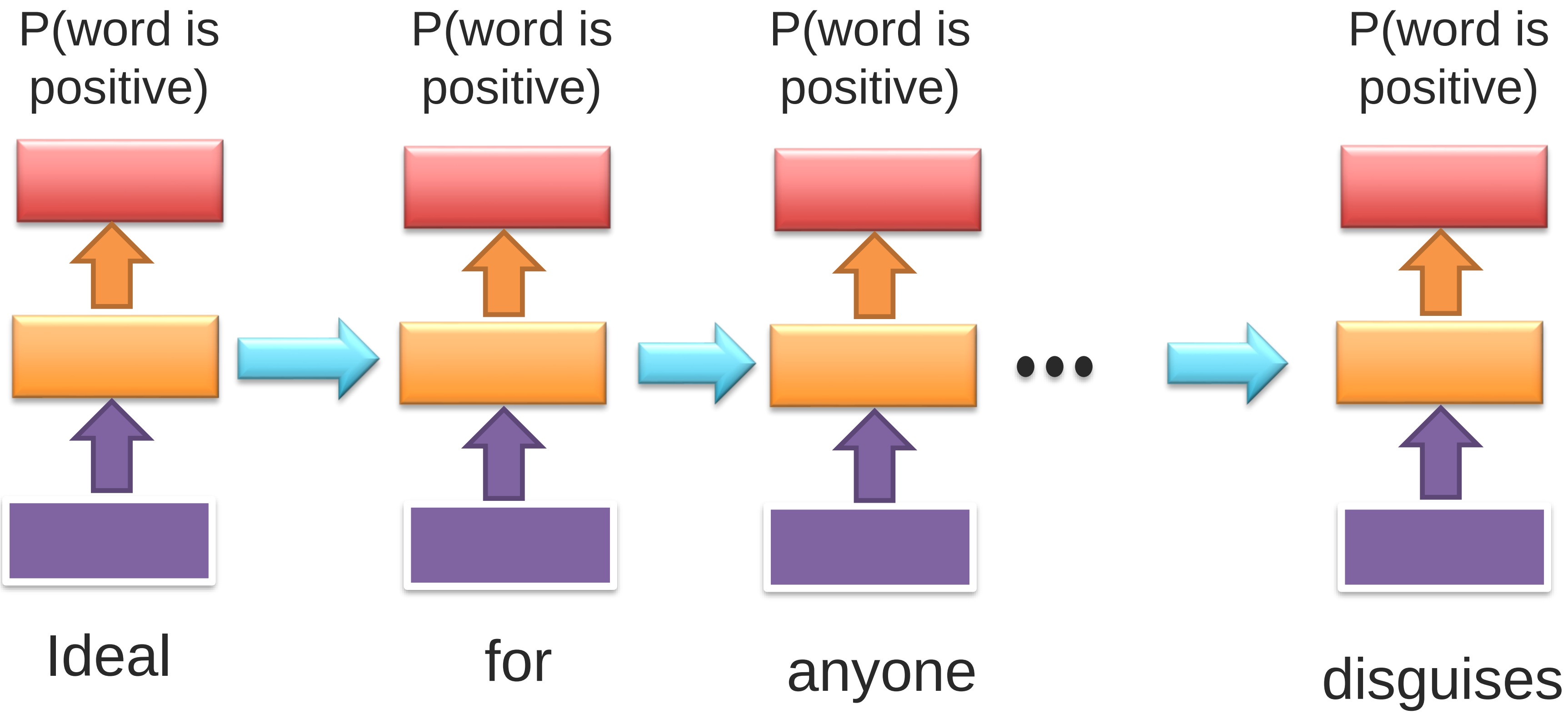
Prediction

Part-of-speech ?
(noun, verb,…)

Sentiment ?
(positive or negative)



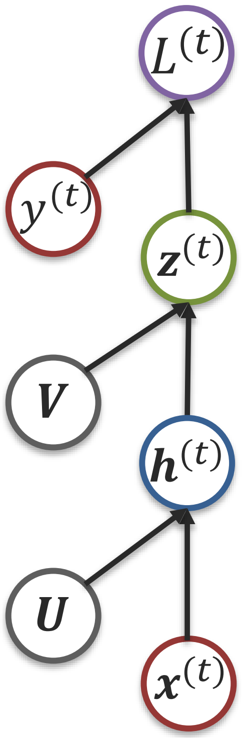
RNN for Sequence Prediction



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

Recurrent Neural Network

Feedforward Neural Network



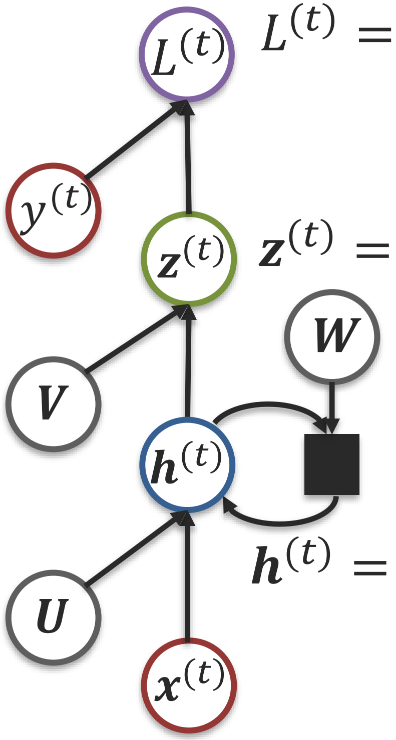
$$L^{(t)} = -logP(Y = y^{(t)}|\boldsymbol{z}^{(t)})$$

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

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

Recurrent Neural Networks

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



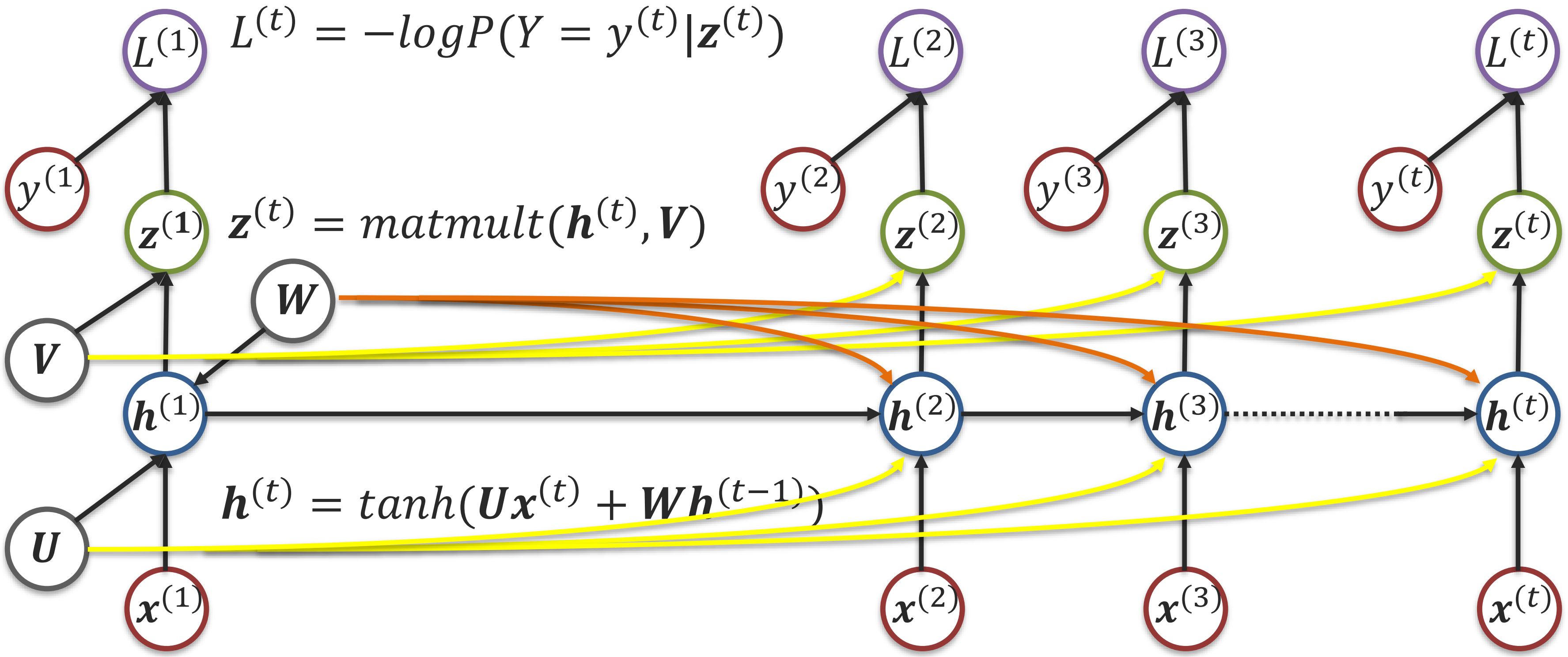
$$L^{(t)} = -logP(Y = y^{(t)} | \boldsymbol{z}^{(t)})$$

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

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

Recurrent Neural Networks - Unrolling

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



Same model parameters are used for all time parts.

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

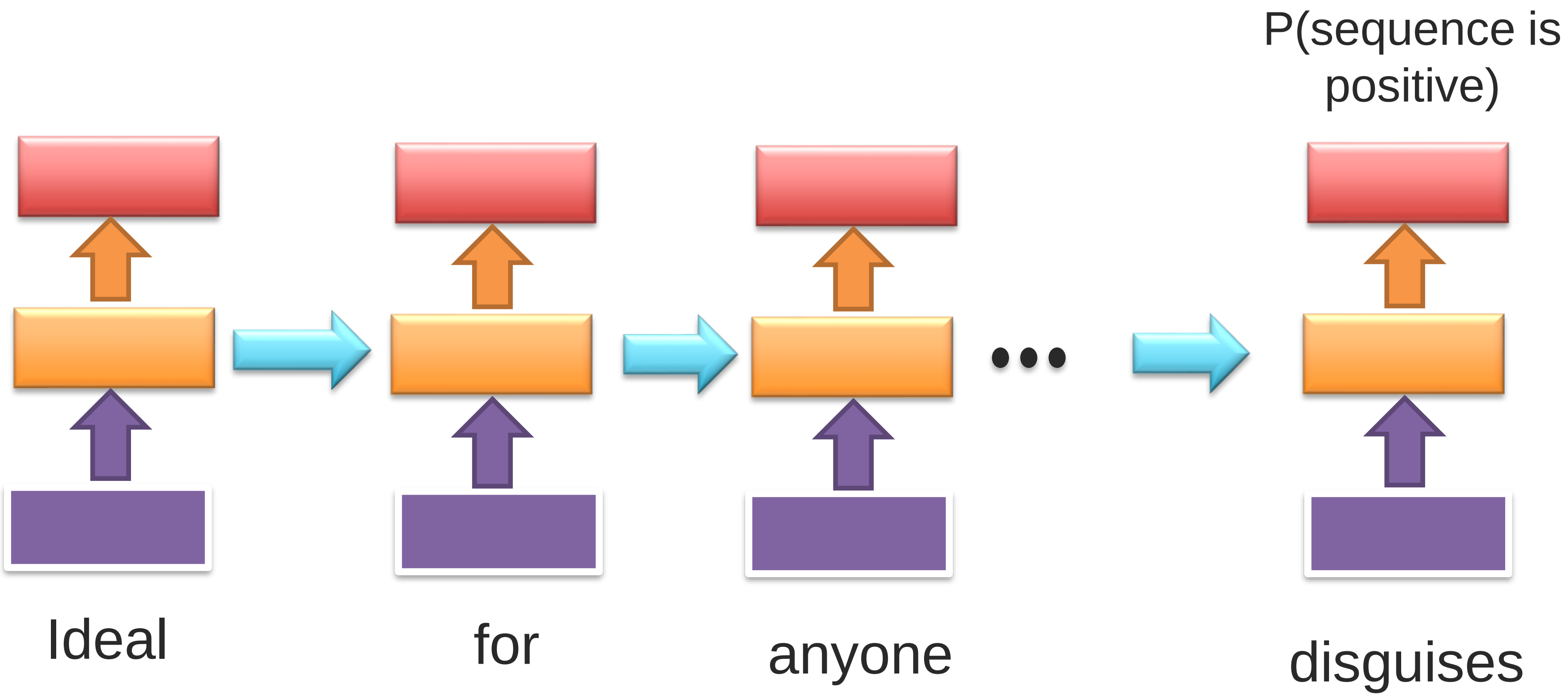
Prediction →

Sentiment ?
(positive or negative)

Sentiment label?

Ideal for anyone with an interest in disguises

RNN for Sequence Prediction



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

Language Models

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 → Next word

Next word?

Ideal for anyone with an interest in disguises

Language Model Application: Speech Recognition

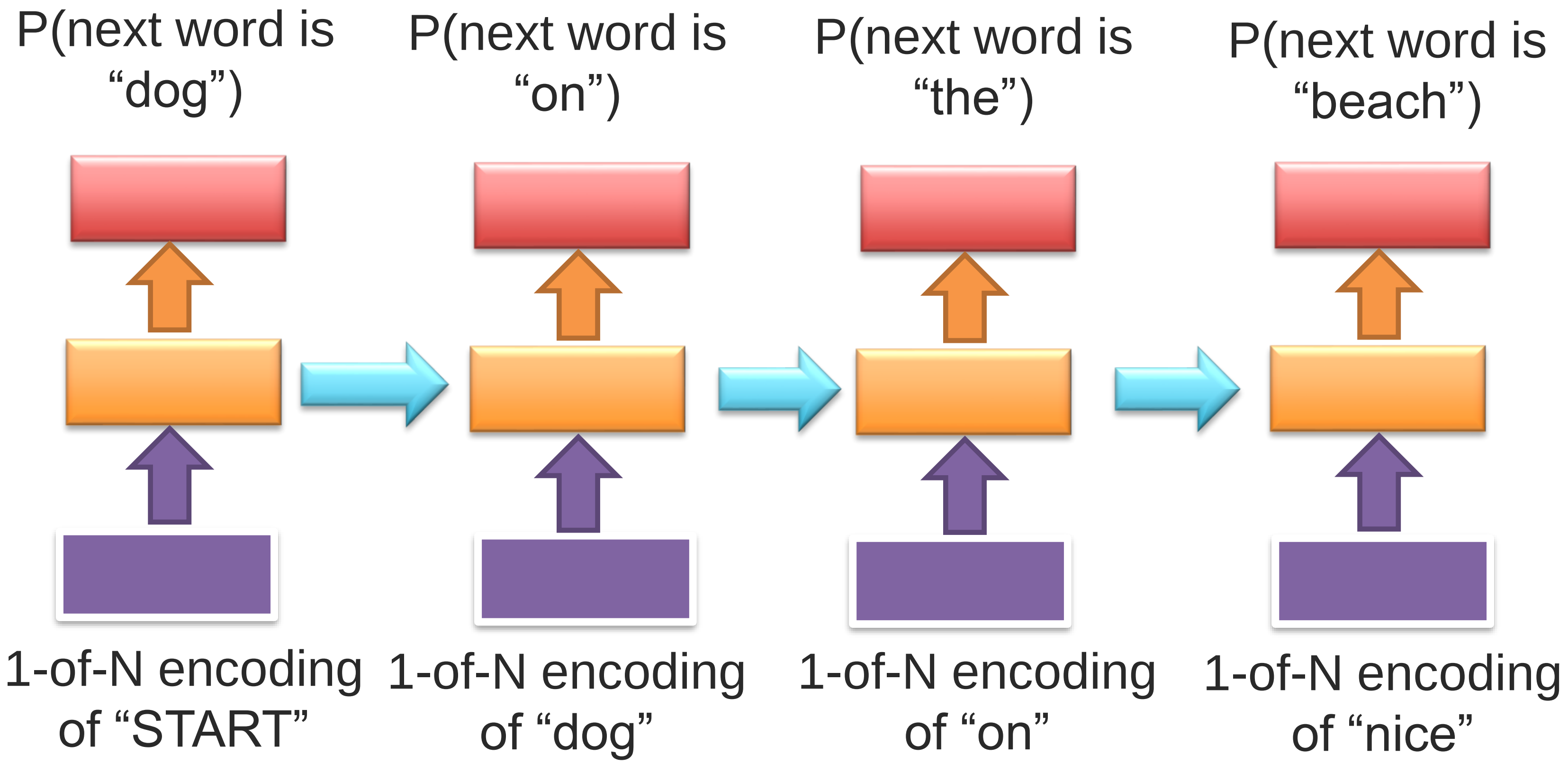
$$\underset{wordsequence}{\arg\max}\; P(wordsequence \mid acoustics) =$$

$$\underset{wordsequence}{\arg\max}\; \frac{P(acoustics \mid wordsequence) \times P(wordsequence)}{P(acoustics)}$$

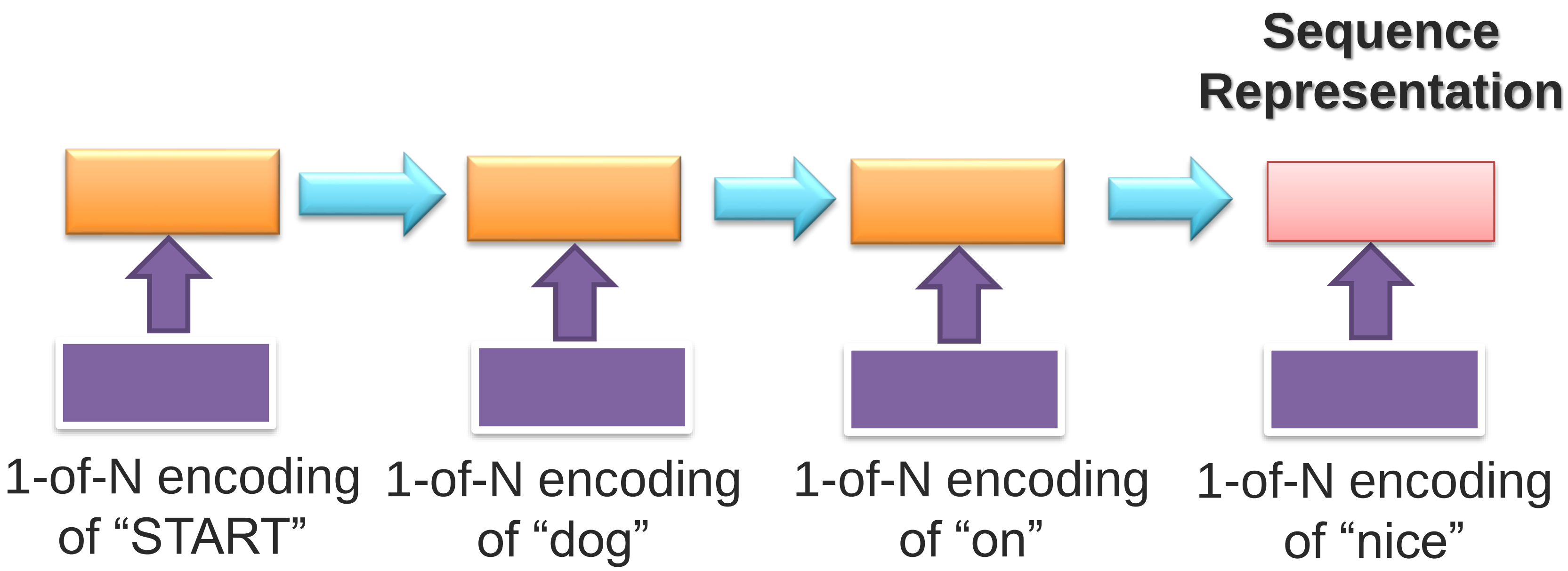
$$\underset{wordsequence}{\arg\max}\; P(acoustics \mid wordsequence) \times P(wordsequence)$$

Language model

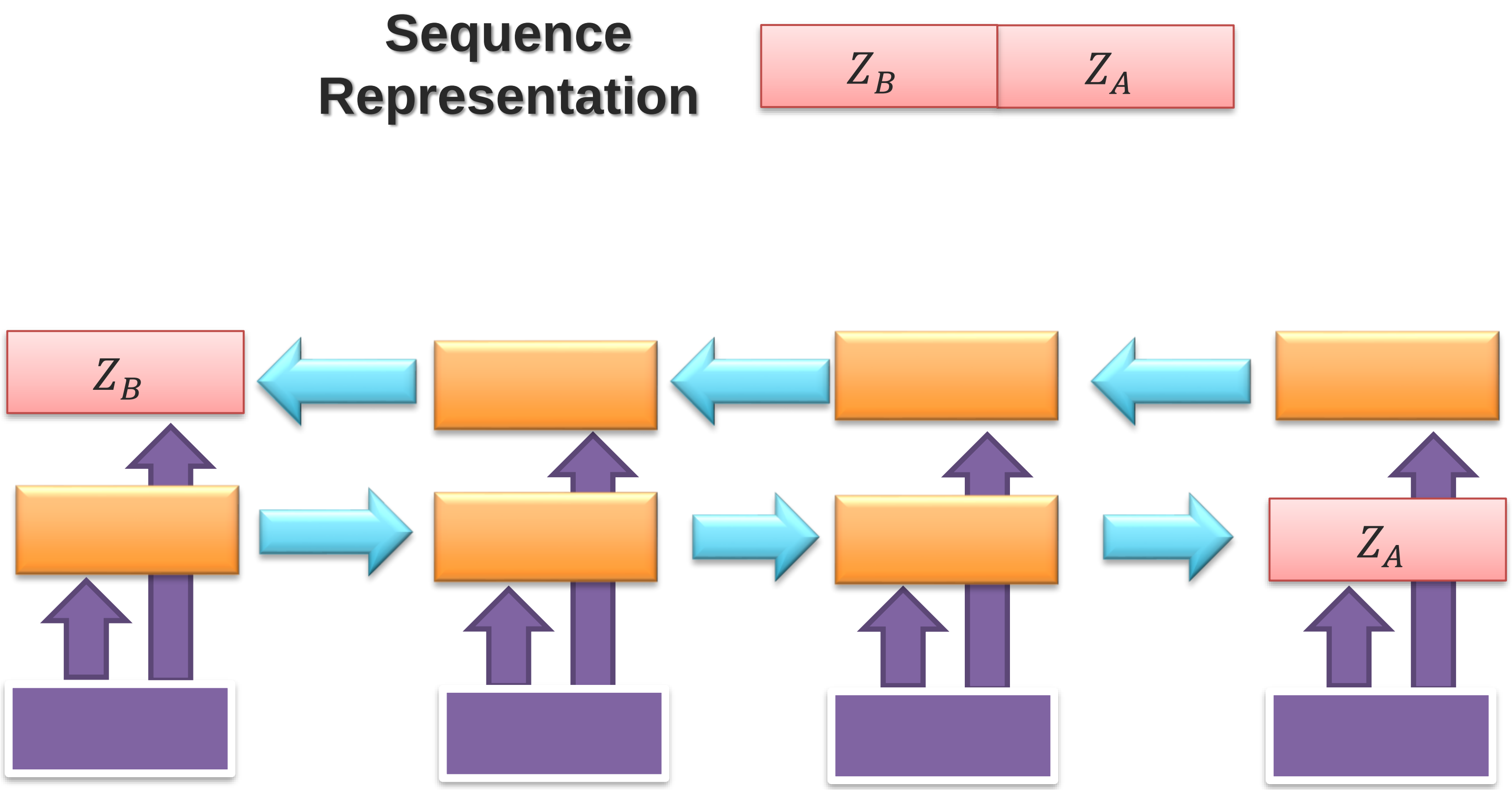
RNN for Language Model



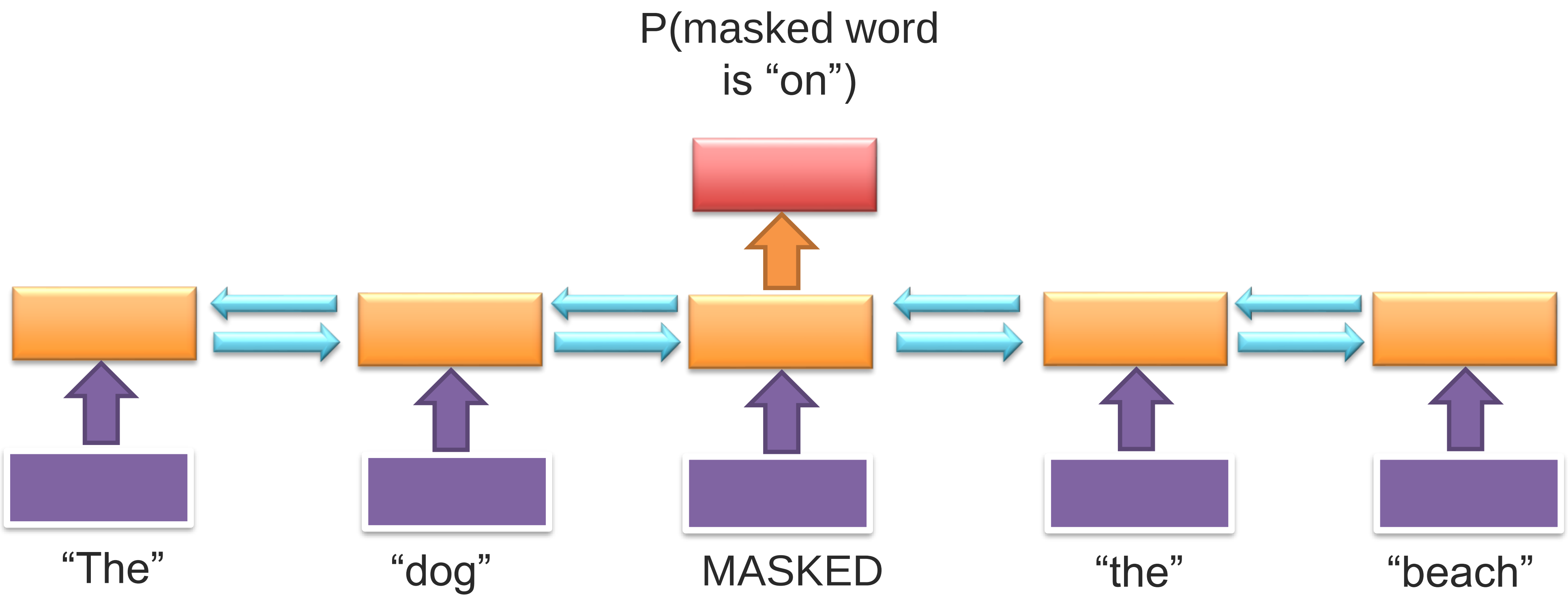
RNN for Sequence Representation (Encoder)



Bi-Directional RNN



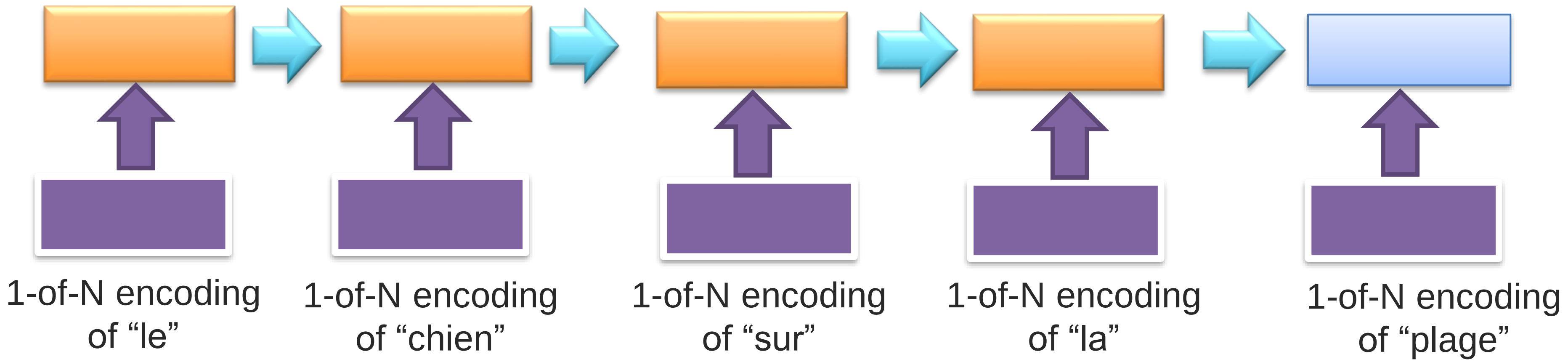
Pre-training and “Masking”



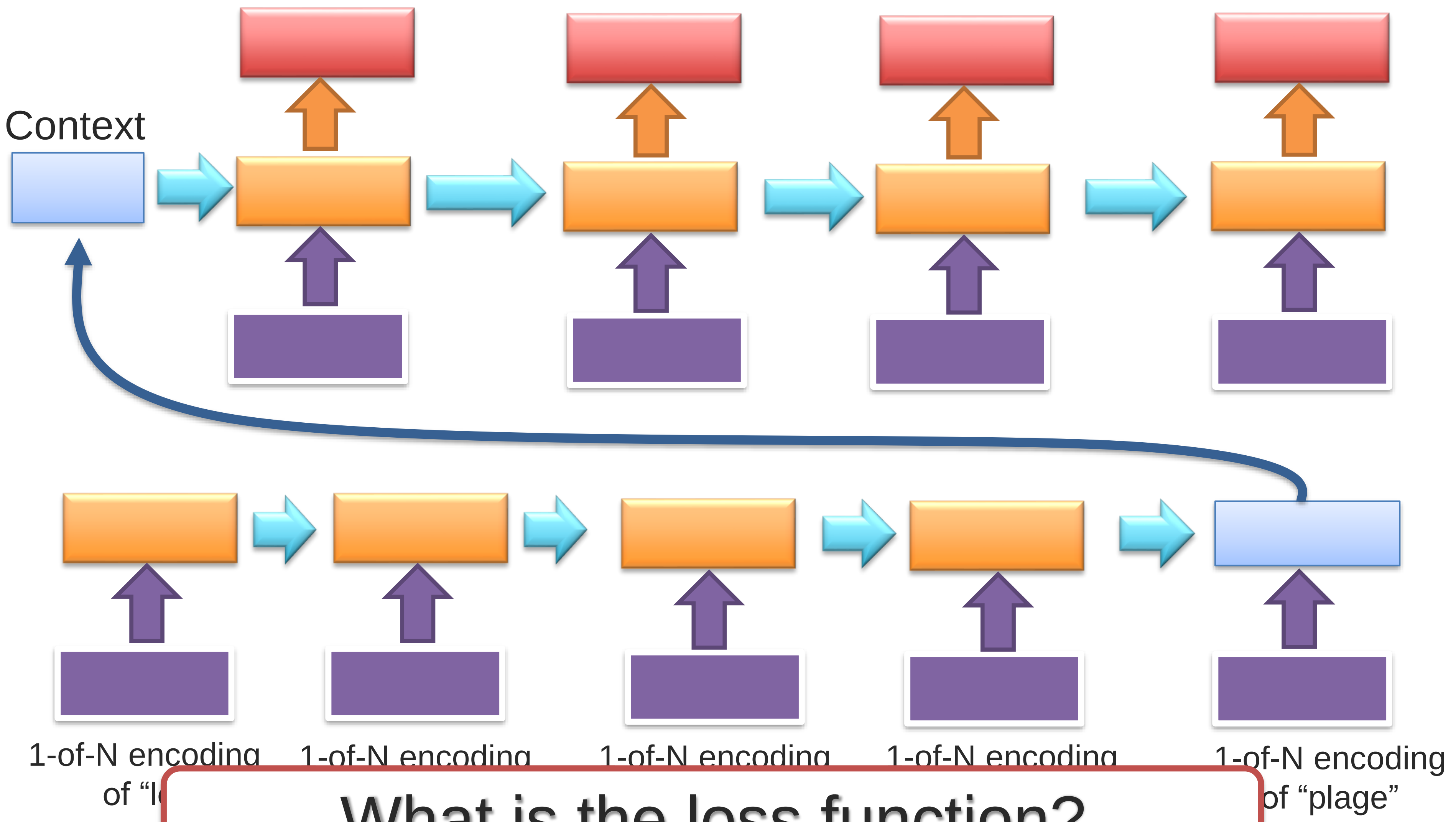
(short-lived) ELMO was a bi-directional pretrained language model

RNN-based for Machine Translation

Le chien sur la plage → The dog on the beach



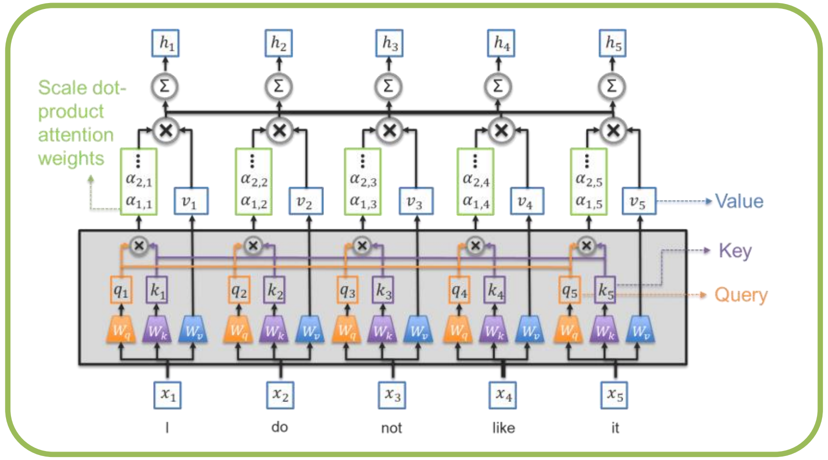
Encoder-Decoder Architecture



What is the loss function?

And There Are More Ways To Model Sequences…

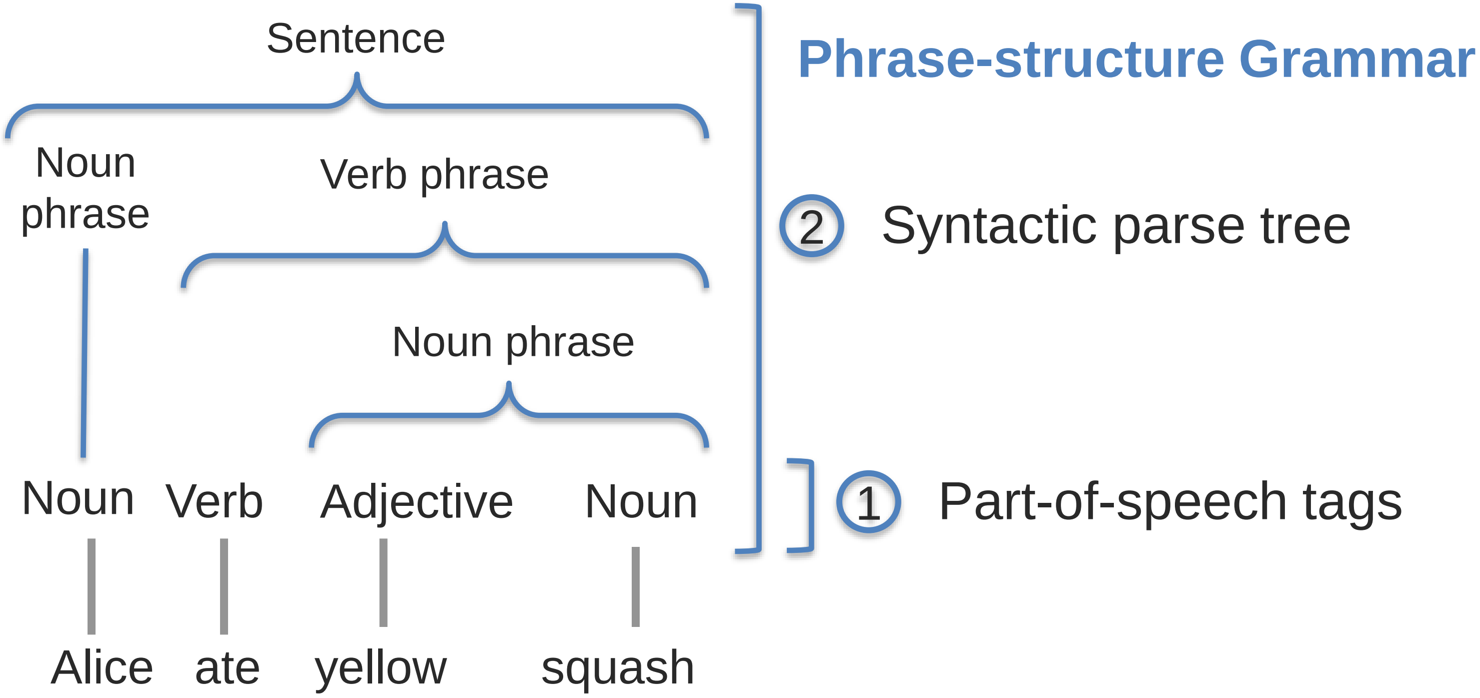
Self-attention Models
(e.g., BERT, RoBERTa)



Syntax and Language Structure

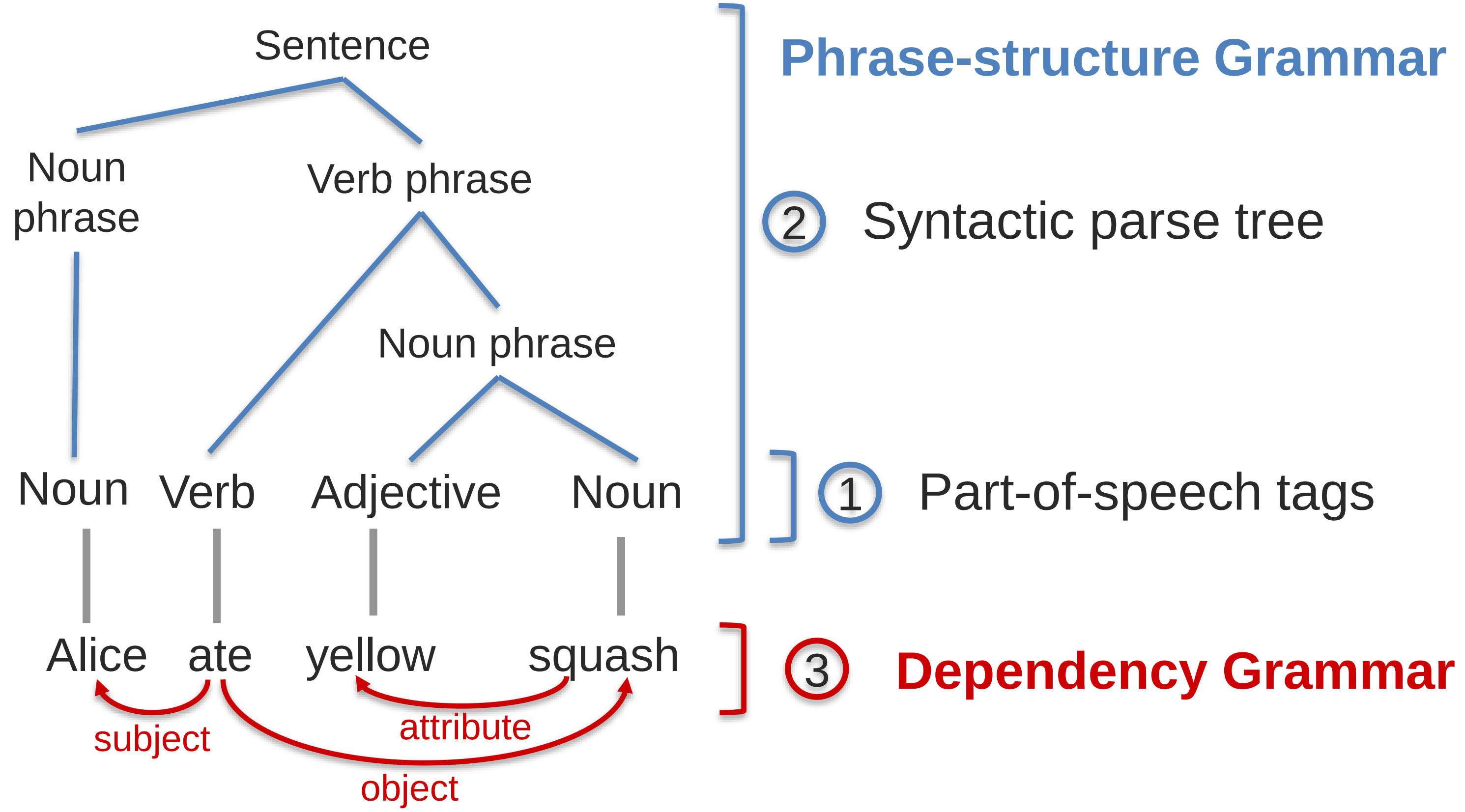
Syntax and Language Structure

What can you tell about this sentence?



Syntax and Language Structure

What can you tell about this sentence?



Ambiguity in Syntactic Parsing

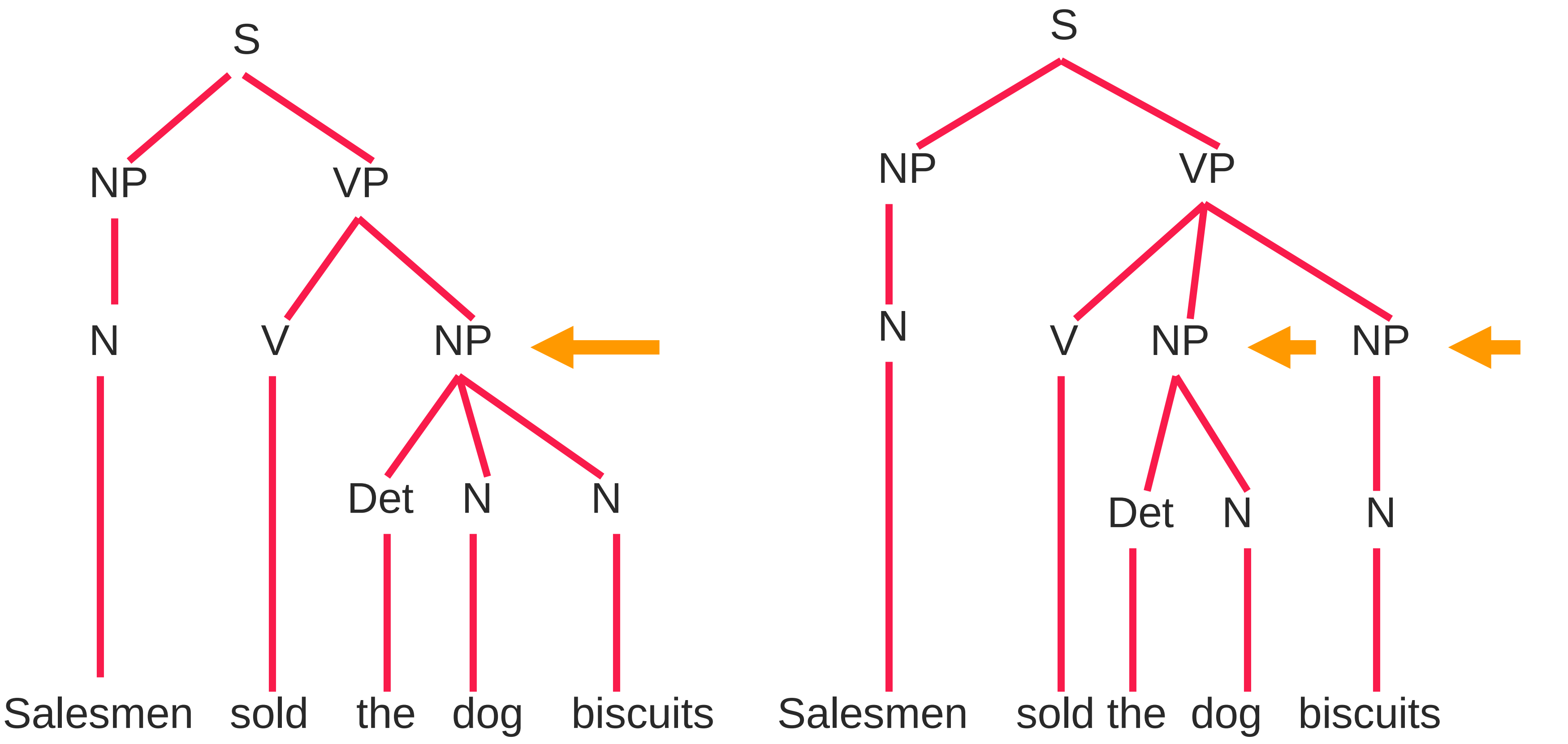
"Like" can be a verb or a preposition

- 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.

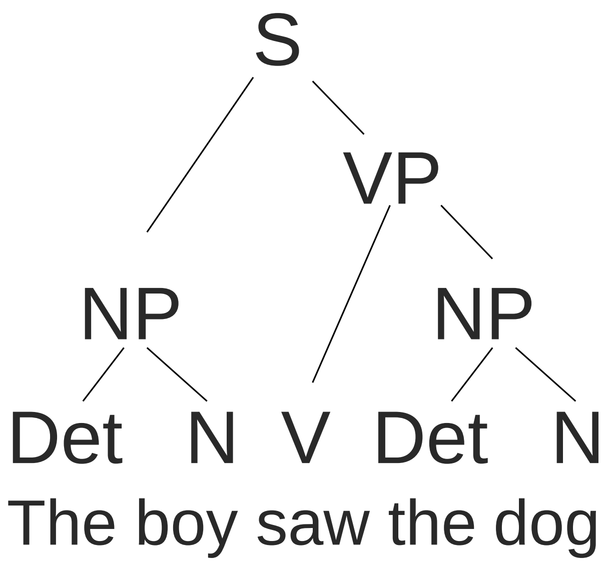
Language Ambiguity

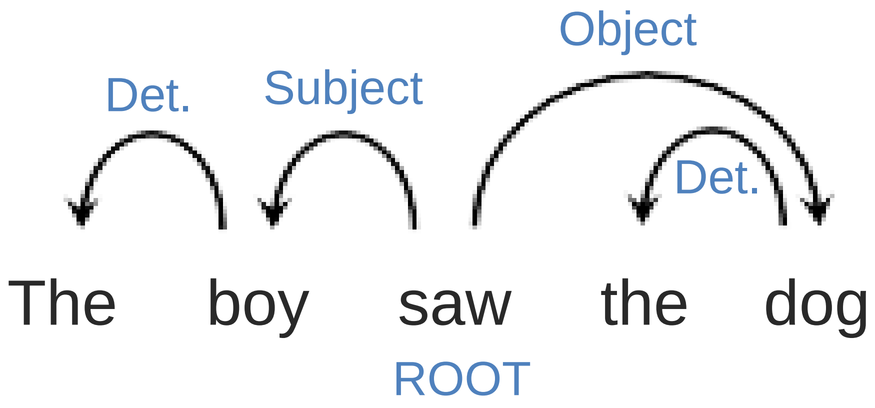


Language Syntax – Examples

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

Part of Speech tagging

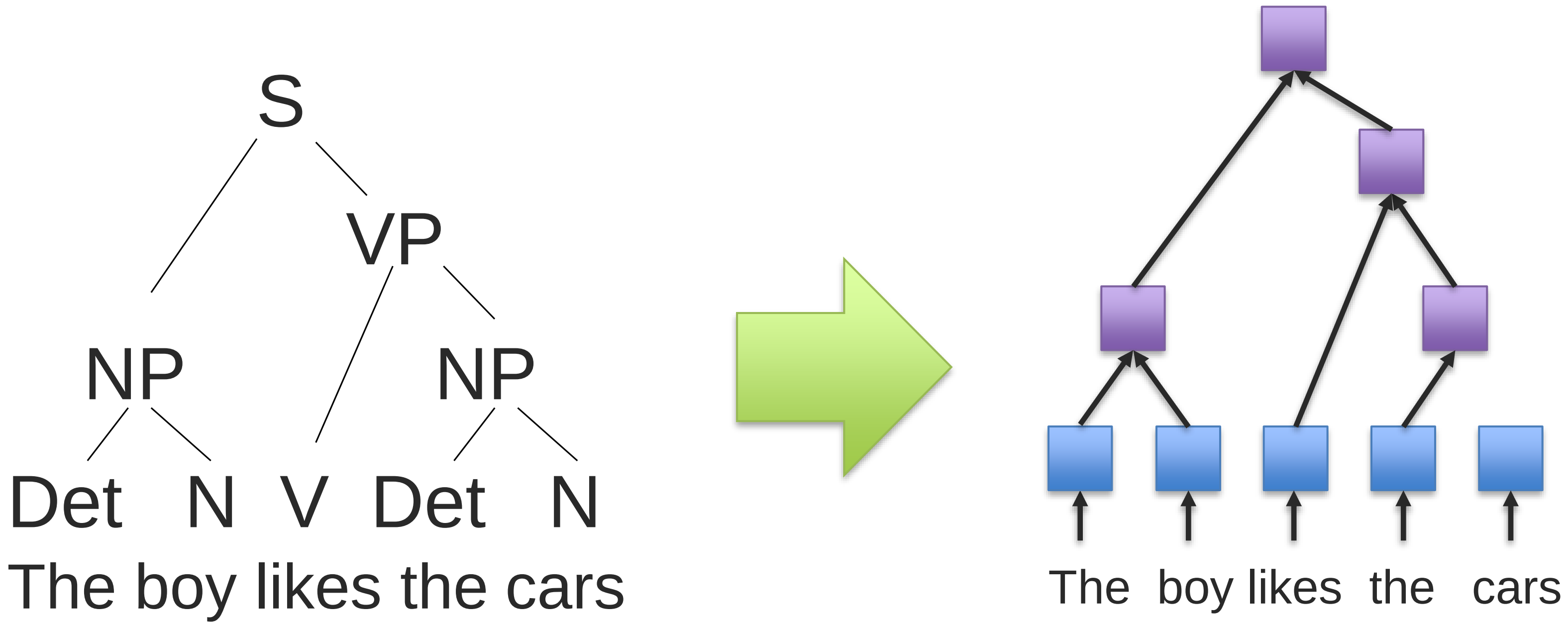




Constituency Parsing **Dependency Parsing**

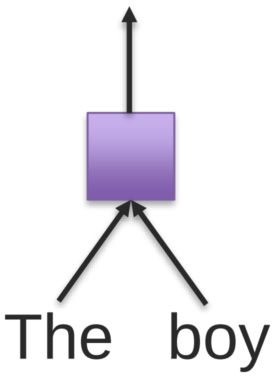
How to take advantage of syntax when modeling language with neural networks?

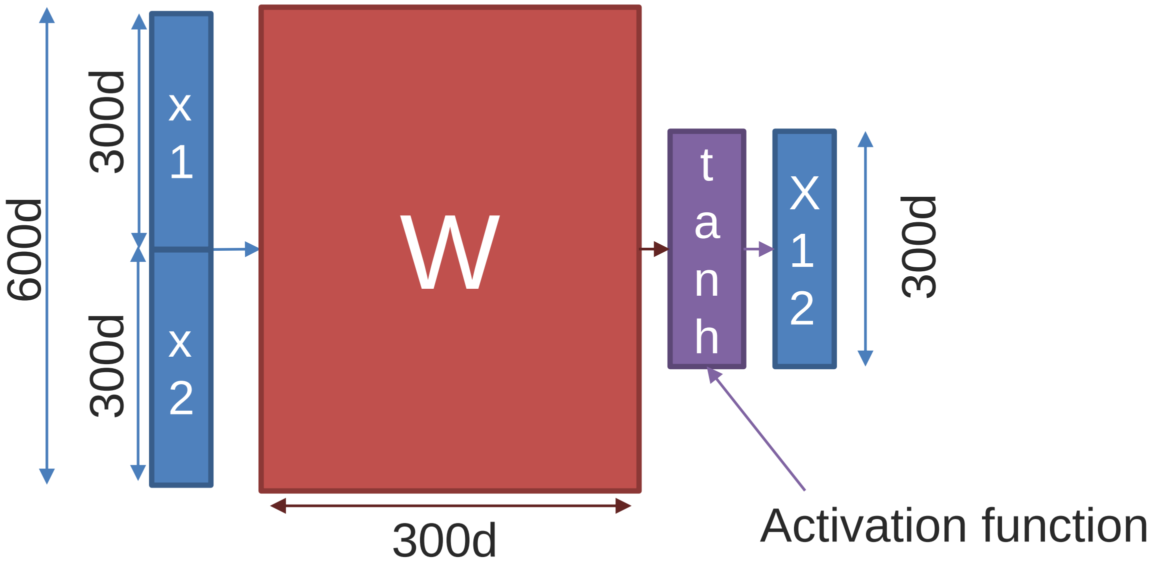
Tree-based RNNs (or Recursive Neural Network)



Recursive Neural Unit

Pair-wise combination of two input features





Resources

Resources

- spaCy (https://spacy.io/)
  - POS tagger, dependency parser, etc.
- Berkeley Neural Parser (Constituency Parser)
  - Software: https://github.com/nikitakit/self-attentive-parser
  - Demo: https://parser.kitaev.io/
- Stanford NLP software
  - Stanza: https://stanfordnlp.github.io/stanza/index.html
  - Others (some are outdated): https://nlp.stanford.edu/software/

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/

Contextual 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

Factorizing co-occurrence matrix

Uses sub-word information (e.g. *walk* and *walking* are similar?)

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

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