Queen Mary
University of London

ECS763 Natural Language Processing

Unit 9: Semantics II: Meaning Representation with Vectors

Lecturer: Julian Hough
School of Electronic Engineering and Computer Science

OUTLINE

1) Recap: vectors for documents and words, normalization and weighting techniques
2) Dense Vectors: Embeddings and word2vec
3) From words to sentences: compositional distributional semantics
4) Applications/Evaluation

OUTLINE

1) Recap: vectors for documents and words, normalization and weighting techniques
2) Dense Vectors: Embeddings and word2vec
3) From words to sentences: compositional distributional semantics
4) Applications/Evaluation

Documents as Vectors

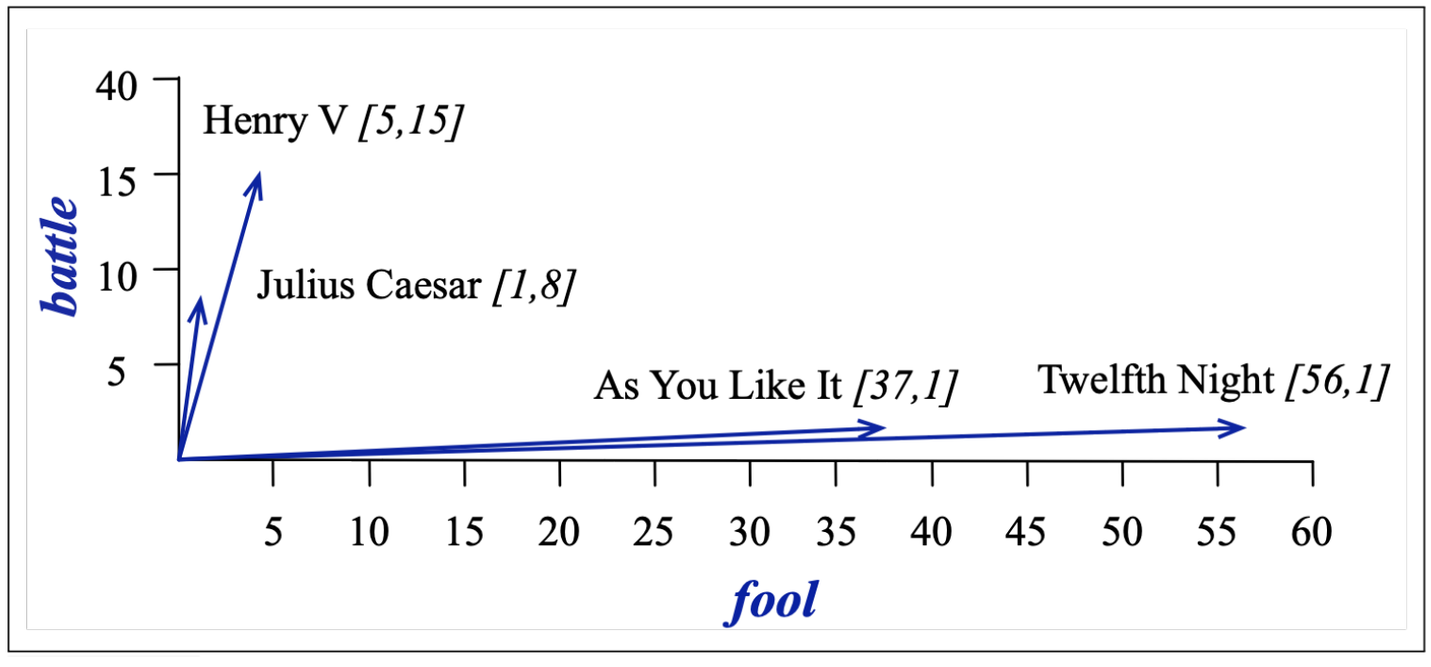
	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	5	117	0	0

Figure 15.1 The term-document matrix for four words in four Shakespeare plays. Each cell contains the number of times the (row) word occurs in the (column) document.

Documents (plays) are column vectors:

As You like It: [1, 2, 37, 5]
Twelfth Night: [1, 2, 58, 117]
Julius Caesar: [8,12, 1, 0]
Henry V: [15,36,5,0]

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$



Words as Vectors

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	5	117	0	0

Figure 15.1 The term-document matrix for four words in four Shakespeare plays. Each cell contains the number of times the (row) word occurs in the (column) document.

Words could be row vectors:

battle: [1, 1, 8, 15]
soldier: [2, 2, 12, 36]
fool: [37, 58, 1, 5]
clown: [5, 117, 0, 0]

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

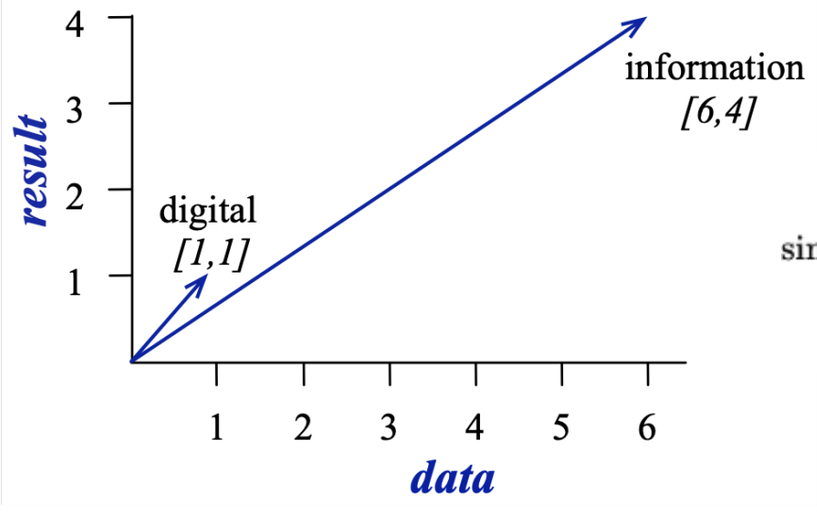
Words as Vectors

More general: row vectors in a matrix whose columns are also words – co-occurrences in context e.g. in a k-word window

	aardvark	...	computer	data	pinch	result	sugar	...
apricot	0	...	0	0	1	0	1	
pineapple	0	...	0	0	1	0	1	
digital	0	...	2	1	0	1	0	
information	0	...	1	6	0	4	0	

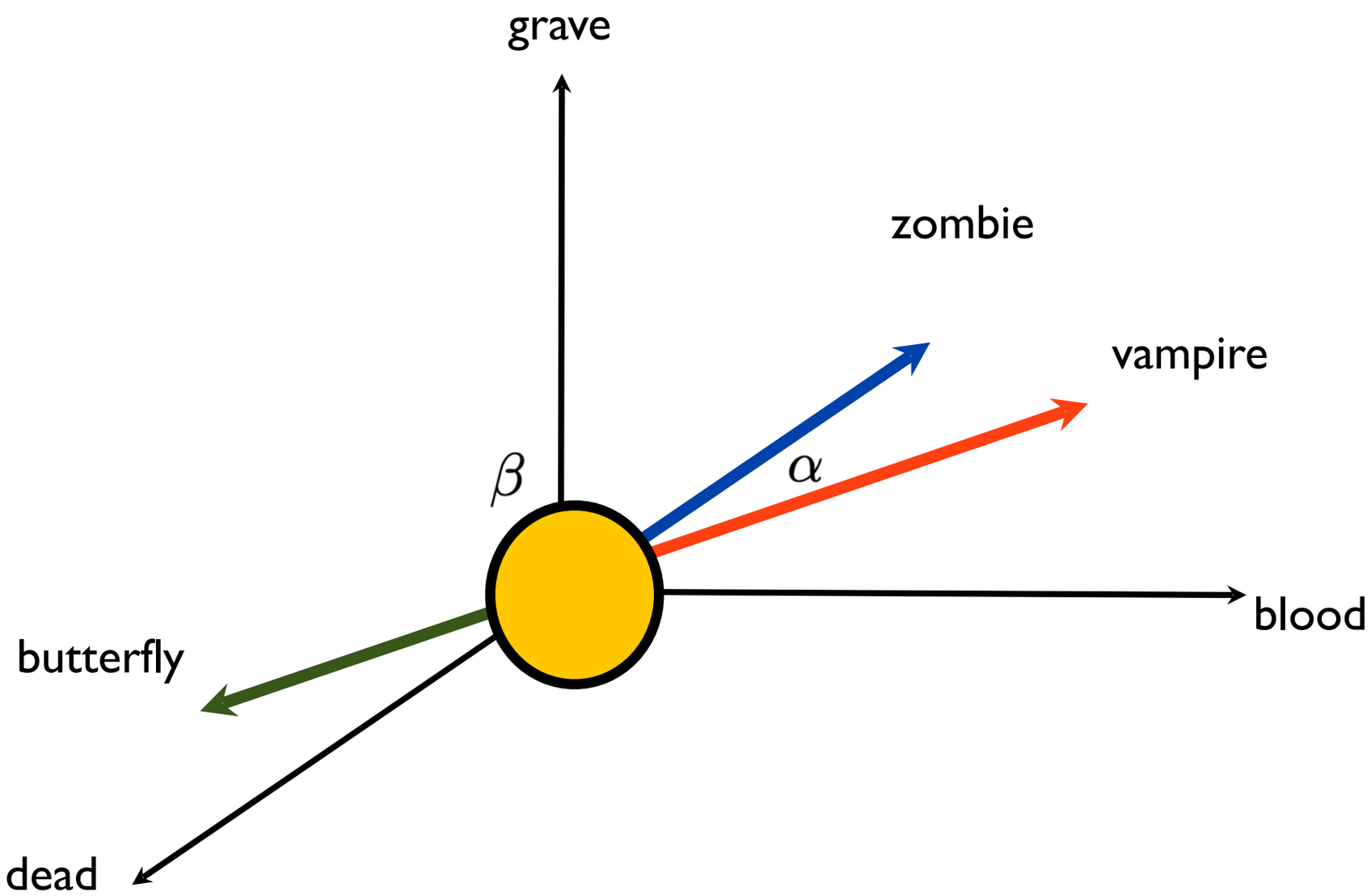
sugar, a sliced lemon, a tablespoonful of **apricot** preserve or jam, a pinch each of,
their enjoyment. Cautiously she sampled her first **pineapple** and another fruit whose taste she likened
well suited to programming on the digital **computer**. In finding the optimal R-stage policy from
for the purpose of gathering data and **information** necessary for the study authorized in the

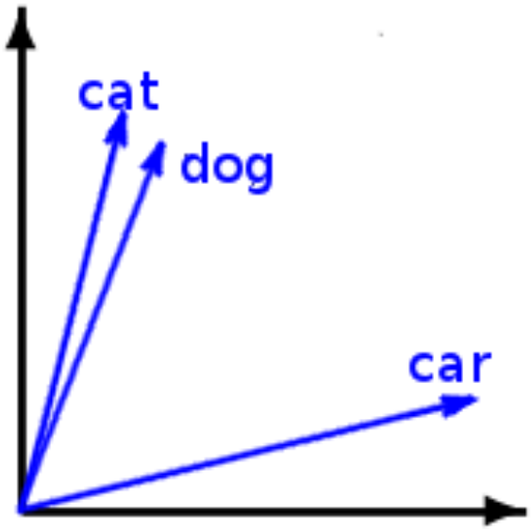
Words are row vectors:
apricot: [0, 0, 1, 0, 1]
pineapple: [0, 0, 1, 0, 1]
digital: [2, 1, 0, 1, 0]
Information: [1, 6, 0, 4, 0]



$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Examples of Word Similarities





spaCy

Weighting terms in Vectors

- Raw counts not an effective vector representation for word/term-document and word/term-context matrices.
- Lots of **unimportant** words occur frequently (stop words).
- **Important** words occur **very infrequently.**
- We need a method to give importance to words that occur in a few documents, whilst minimising effect of non-important words.

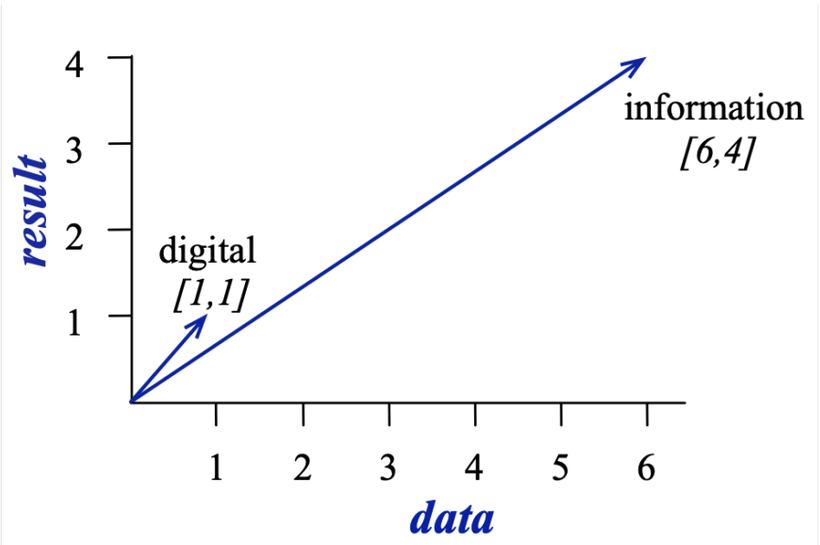
Small Corpus

Co-occurrences from 7-word windows in the Brown Corpus:

	aardvark	...	computer	data	pinch	result	sugar	...
apricot	0	...	0	0	1	0	1	
pineapple	0	...	0	0	1	0	1	
digital	0	...	2	1	0	1	0	
information	0	...	1	6	0	4	0	

sugar, a sliced lemon, a tablespoonful of **apricot** preserve or jam, a pinch each of,
their enjoyment. Cautiously she sampled her first **pineapple** and another fruit whose taste she likened
well suited to programming on the digital **computer**. In finding the optimal R-stage policy from
for the purpose of gathering data and **information** necessary for the study authorized in the

Words are row vectors:
apricot: [0, 0, 1, 0, 1]
pineapple: [0, 0, 1, 0, 1]
digital: [2, 1, 0, 1, 0]
Information: [1, 6, 0, 4, 0]



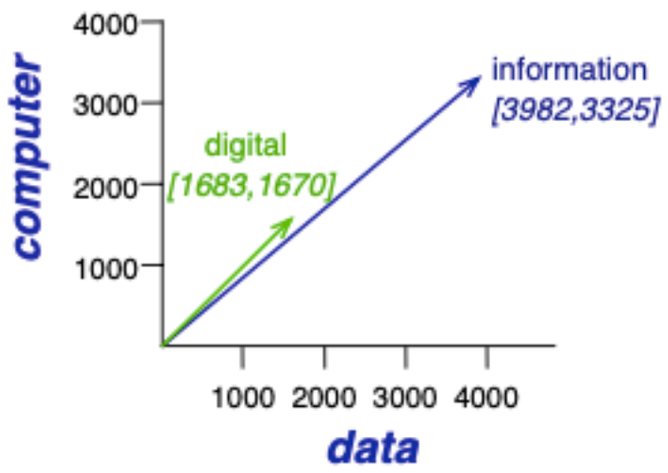
Bigger Corpus

Co-occurrences from 4-word windows in Wikipedia:

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

is traditionally followed by	**cherry**	pie, a traditional dessert
often mixed, such as	**strawberry**	rhubarb pie. Apple pie
computer peripherals and personal	**digital**	assistants. These devices usually
a computer. This includes	**information**	available on the internet

Words are row vectors:
cherry: [2, 8, 9, 442, 25]
strawberry: [0, 0, 1, 60, 19]
digital: [1670, 1683, 85, 5, 4]
Information: [3325, 3982, 378, 5, 13]



For documents: TF-IDF

$$w_{t,d} = \mathrm{tf}_{t,d} \times \mathrm{idf}_t$$

TF/IDF: initiated for creating document vectors

TF: frequency of the word in the document (Luhn 1975),
IDF: inverse document frequency (Sparck Jones, 1972).

TF the term frequency – more usually its log, with 1 added to counts (Laplace smoothed) to avoid the log of 0:

$$\mathrm{tf}_{t,d} = \log_{10}(\mathrm{count}(t,d)+1)$$

IDF is defined using the fraction N/df_t, where N is the total number of documents in the collection, and df_t is the number of documents in which term t occurs.

Since N is high, idf (and tf) usually adjusted with a log function, e.g. :

$$\mathrm{idf}_t = \log_{10}\left(\frac{N}{\mathrm{df}_t}\right)$$

Example of Effect

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.967
forest	12	0.489
battle	21	0.246
wit	34	0.037
fool	36	0.012
good	37	0
sweet	37	0

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	0.074	0	0.22	0.28
good	0	0	0	0
fool	0.019	0.021	0.0036	0.0083
wit	0.049	0.044	0.018	0.022

Figure 6.8 A tf-idf weighted term-document matrix for four words in four Shakespeare plays, using the counts in Fig. 6.2. For example the 0.049 value for *wit* in *As You Like It* is the product of $\text{tf} = \log_{10}(20+1) = 1.322$ and $\text{idf} = .037$. Note that the idf weighting has eliminated the importance of the ubiquitous word *good* and vastly reduced the impact of the almost-ubiquitous word *fool*.

$$\text{tf}_{t,d} = \log_{10}(\text{count}(t,d)+1)$$

$$\text{idf}_t = \log_{10}\left(\frac{N}{\text{df}_t}\right)$$

N = 37

For words: PPMI

$$\text{PPMI}(w,c) = \max(\log_2 \frac{P(w,c)}{P(w)P(c)}, 0)$$

"How much more strongly are *w* and *c* associated than we would expect by chance?"

$$P(w\_i,c\_j) = \frac{f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}}$$

$$P(w\_i) = \frac{\sum_{j=1}^{C} f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}}$$

$$P(c\_j) = \frac{\sum_{i=1}^{W} f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}}$$

Given a word-context matrix with W rows and C columns:

f_ij is the number of times word w_i occurred in the context of word c_j.

Example of Effect

	computer	data	result	pie	sugar	count(w)
cherry	2	8	9	442	25	486
strawberry	0	0	1	60	19	80
digital	1670	1683	85	5	4	3447
information	3325	3982	378	5	13	7703
count(context)	4997	5673	473	512	61	11716

Figure 6.10 Co-occurrence counts for four words in 5 contexts in the Wikipedia corpus, together with the marginals, pretending for the purpose of this calculation that no other words/contexts matter.

	computer	data	result	pie	sugar
cherry	0	0	0	4.38	3.30
strawberry	0	0	0	4.10	5.51
digital	0.18	0.01	0	0	0
information	0.02	0.09	0.28	0	0

Figure 6.12 The PPMI matrix showing the association between words and context words, computed from the counts in Fig. 6.11. Note that most of the 0 PPMI values are ones that had a negative PMI; for example PMI(*cherry,computer*) = -6.7, meaning that *cherry* and *computer* co-occur on Wikipedia less often than we would expect by chance, and with PPMI we replace negative values by zero.

- And we also often reduce dimensionality via e.g. SVD or PCA
- 10,000s of dimensions → 100s of dimensions
- Faster computation, less storage
- Smoother

OUTLINE

1) Recap: vectors for documents and words, normalization and weighting techniques
2) Dense Vectors: Embeddings and word2vec
3) From words to sentences: compositional distributional semantics
4) Applications/Evaluation

Dense vectors: Word Embeddings

- Word-context (term-term) and word-document vectors represent a word as a **sparse, long vector** with dimensions corresponding to words in the vocabulary or documents in a collection.

- We can also use a more powerful word representation: **embeddings**, which are **short dense vectors** (every value is present, and can be negative).

- These are often learned by **neural network/deep learning** models.

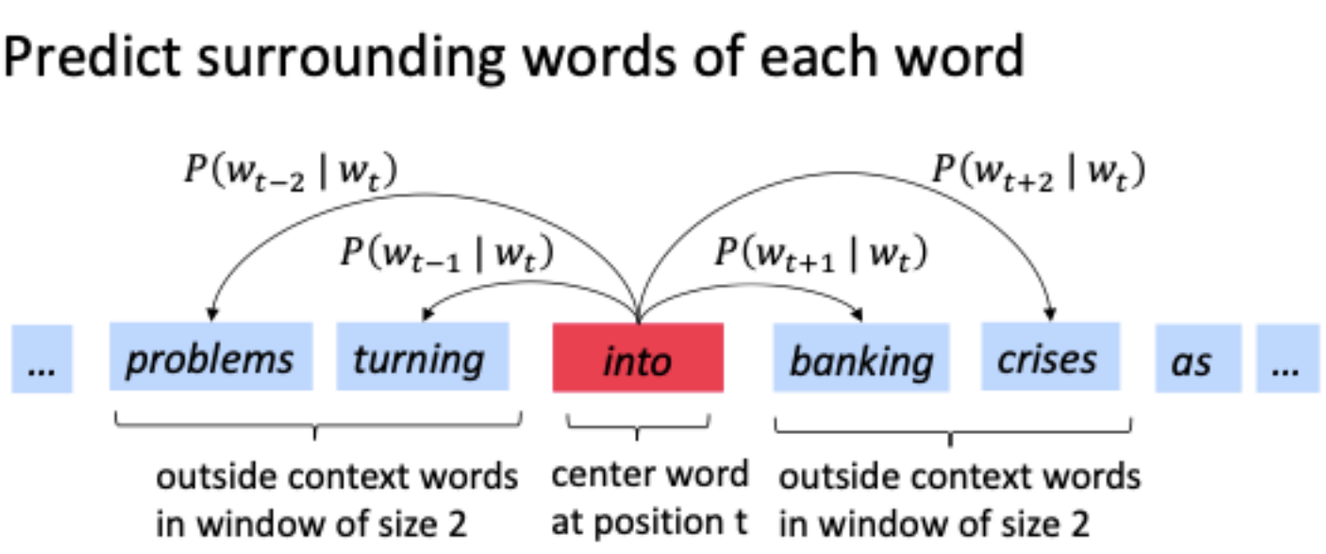
Sparse versus dense vectors

Why dense vectors?

- Short vectors may be easier to use as features in machine learning (**fewer weights** to tune)
- Dense vectors may g**eneralise better** than explicit counts
- Dense vectors may **do better at capturing synonymy**:
  - *car* and *automobile* are synonyms; but are distinct dimensions
  - a word with *car* as a neighbour and a word with *automobile* as a neighbour should be similar, but aren't
- In practice, **they work better.**

Neural methods

- So far we are estimating vectors based on **co-occurrence counts**
- An alternative: learn vectors that are good at **predicting co-occurrence**
    - e.g. word2Vec



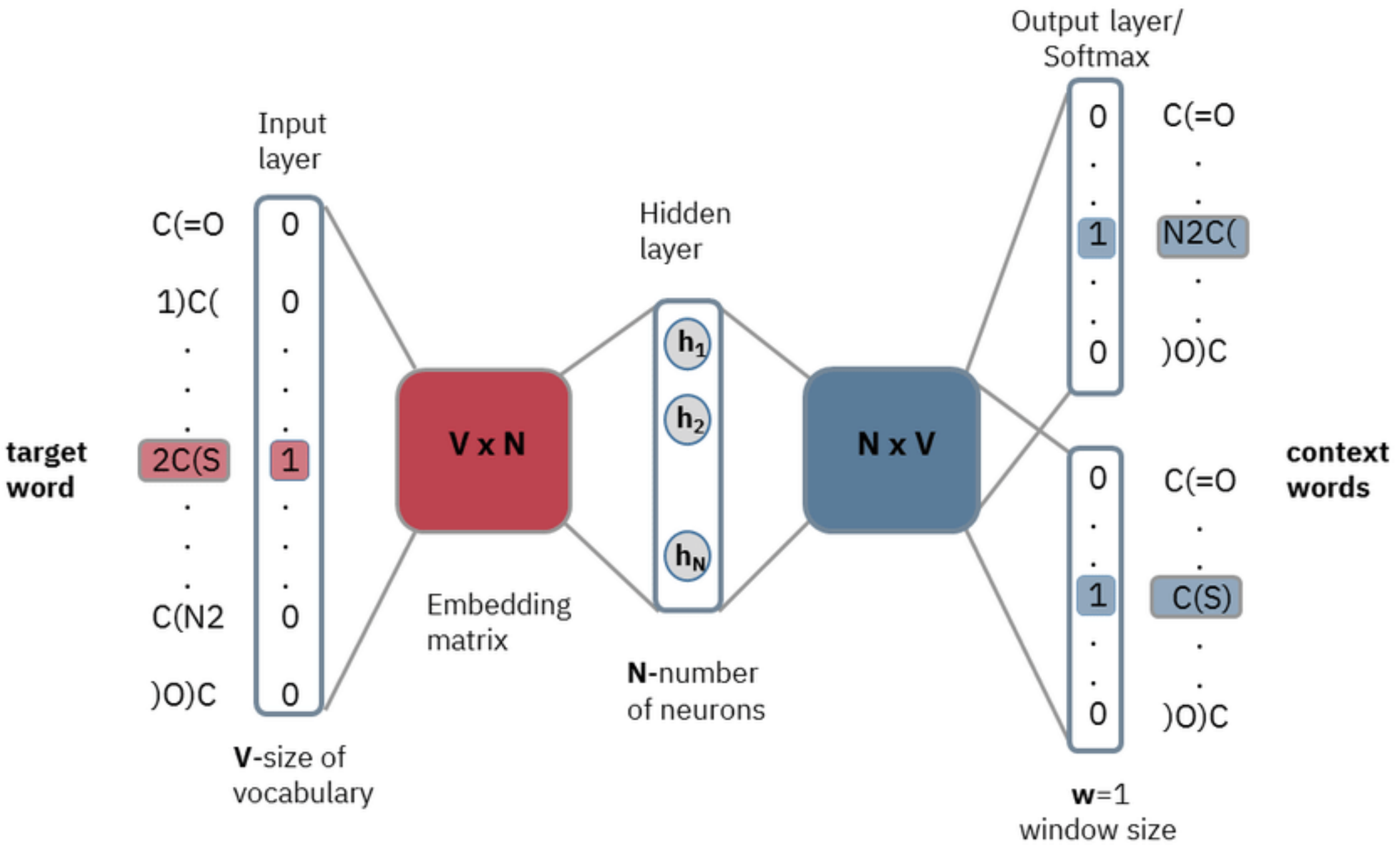
- Popular method: Skipgram
    - train a classifier to predict whether target word t is likely to occur close to context word c or not
    - Use “negative sampling”: train with real vs randomly sampled pairs
    - The weights of this classifier are used as the word vectors

Skipgram
(used in word2vec)

- “The man who passes the sentence should swing the sword” - Ned Stark

Sliding window (size = 5)	Target word	Context
[The man who]	the	man, who
[The man who passes]	man	the, who, passes
[The man who passes the]	who	the, man, passes, the
[man who passes the sentence]	passes	man, who, the, sentence
…	…	…
[sentence should swing the sword]	swing	sentence, should, the, sword
[should swing the sword]	the	should, swing, sword
[swing the sword]	sword	swing, the

word2vec


Input layer
Hidden layer
Output layer/ Softmax
C(=O
1)C(
2C(S
C(N2
)O)C
0
1
V x N
N x V
h1
h2
hN
N2C(
C(S)
target word
context words
Embedding matrix
N-number of neurons
V-size of vocabulary
w=1 window size

word2vec

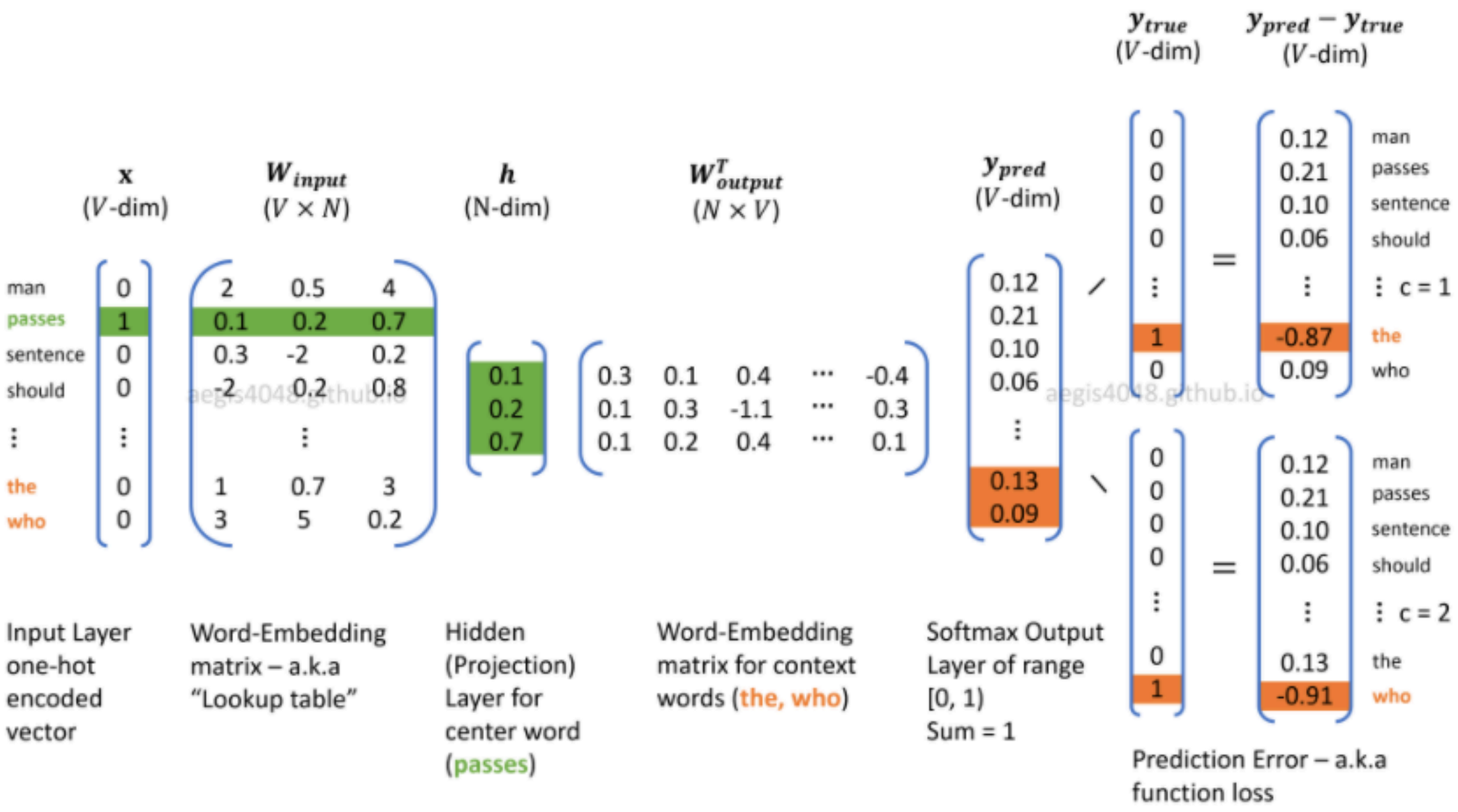
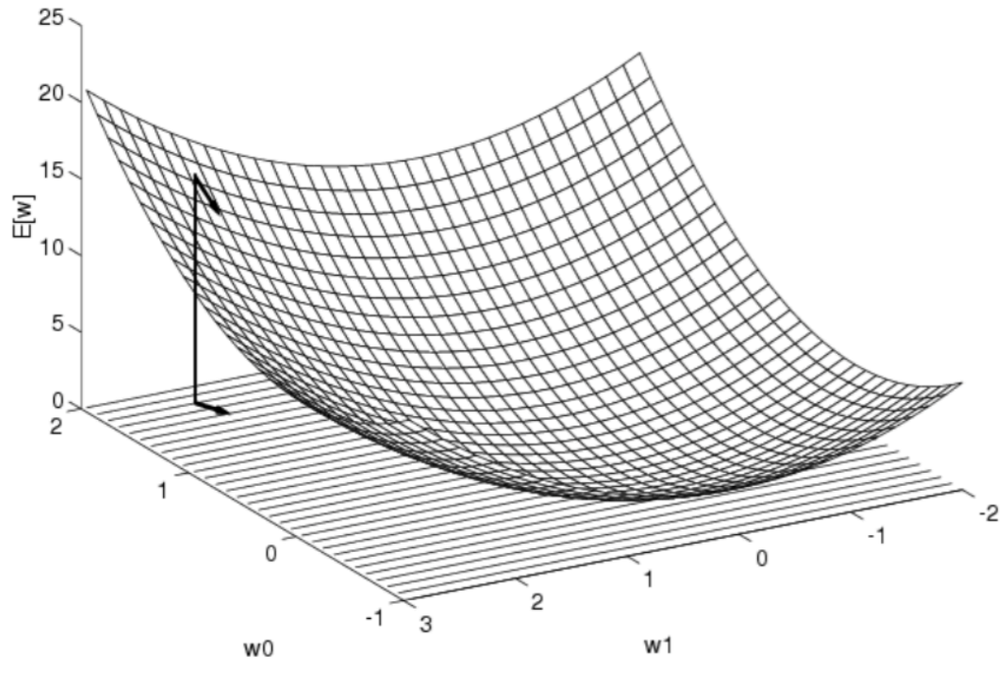


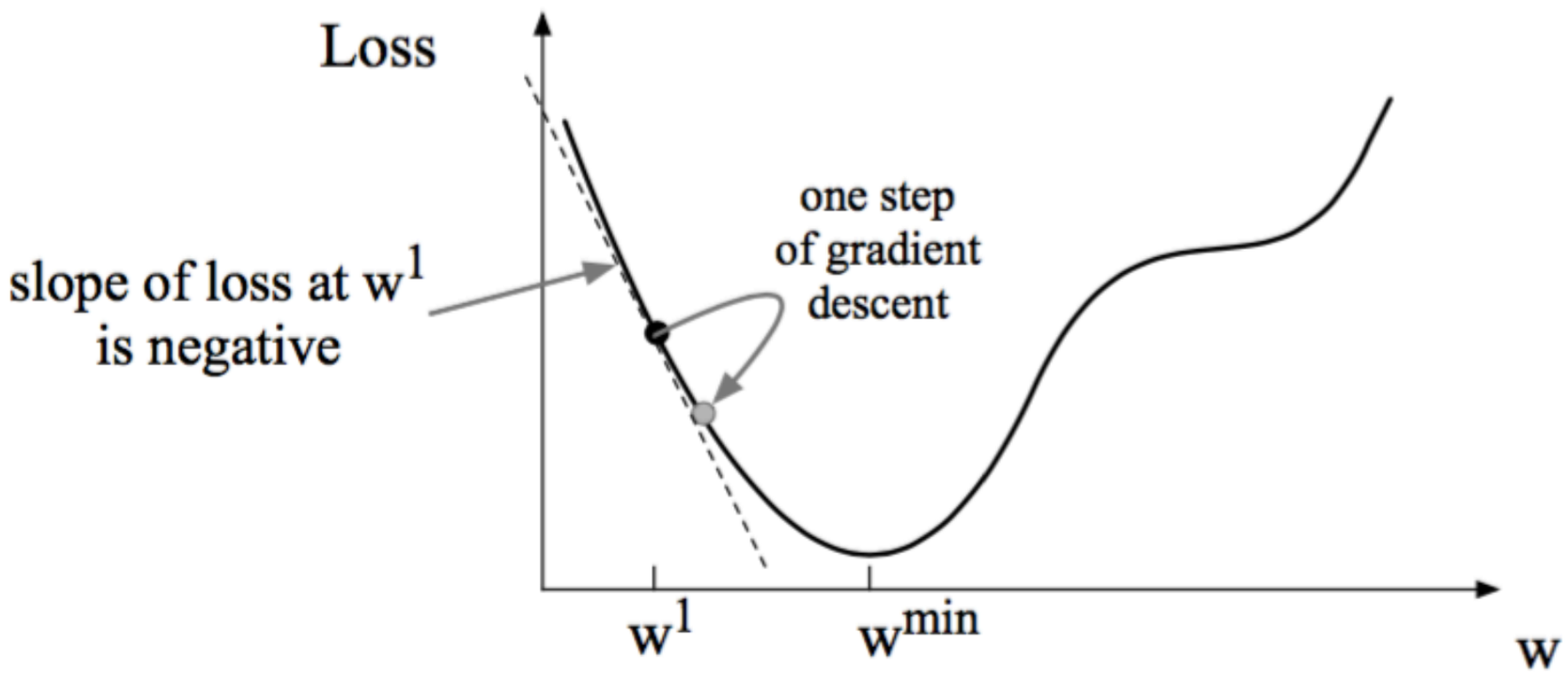
Figure 1: Neural network structure of Skip-Gram

Aside: training by gradient descent

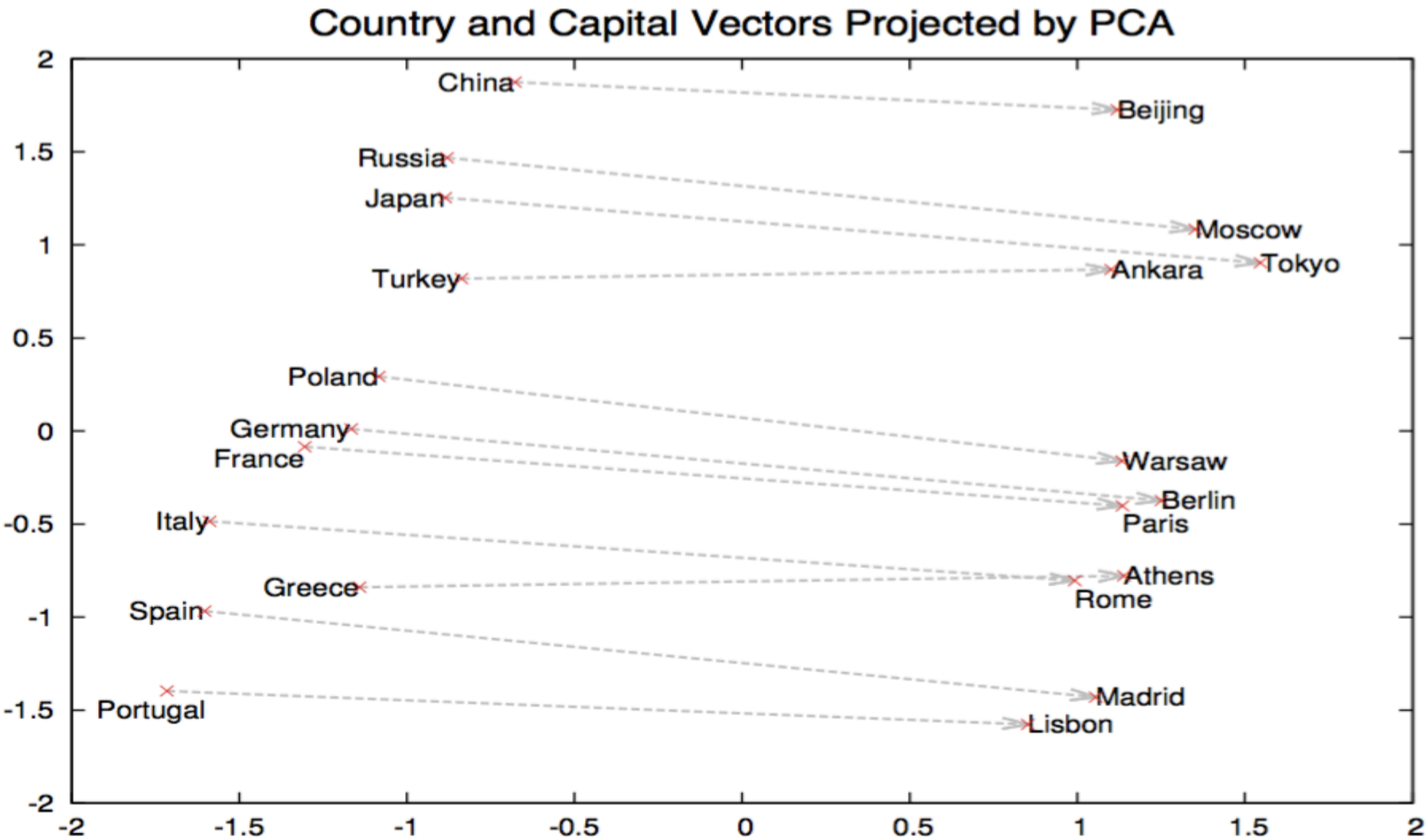
- Initialise weights randomly
- Measure the loss (cost)
- Determine the gradient of the loss
  - (partial derivative wrt. each weight)
- Move each weight in the direction that reduces loss
  - gradient descent
- Converge to optimal w



Jurafsky & Martin S&LP



Semantic regularities



Semantic regularities

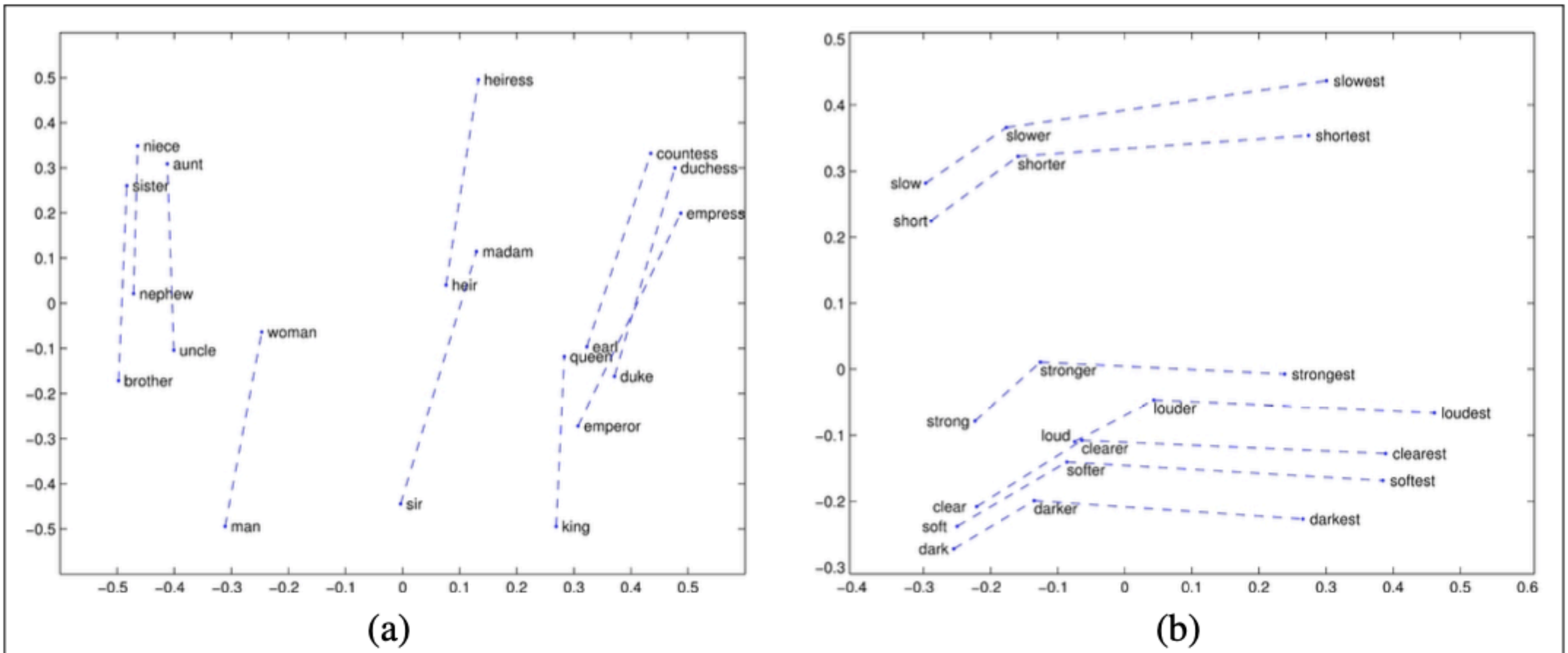


Figure 6.16 Relational properties of the GloVe vector space, shown by projecting vectors onto two dimensions. (a) $(\overrightarrow{\text{king}}) - \overrightarrow{\text{man}} + \overrightarrow{\text{woman}}$ is close to $\overrightarrow{\text{queen}}$. (b) offsets seem to capture comparative and superlative morphology (Pennington et al., 2014).

Analogies

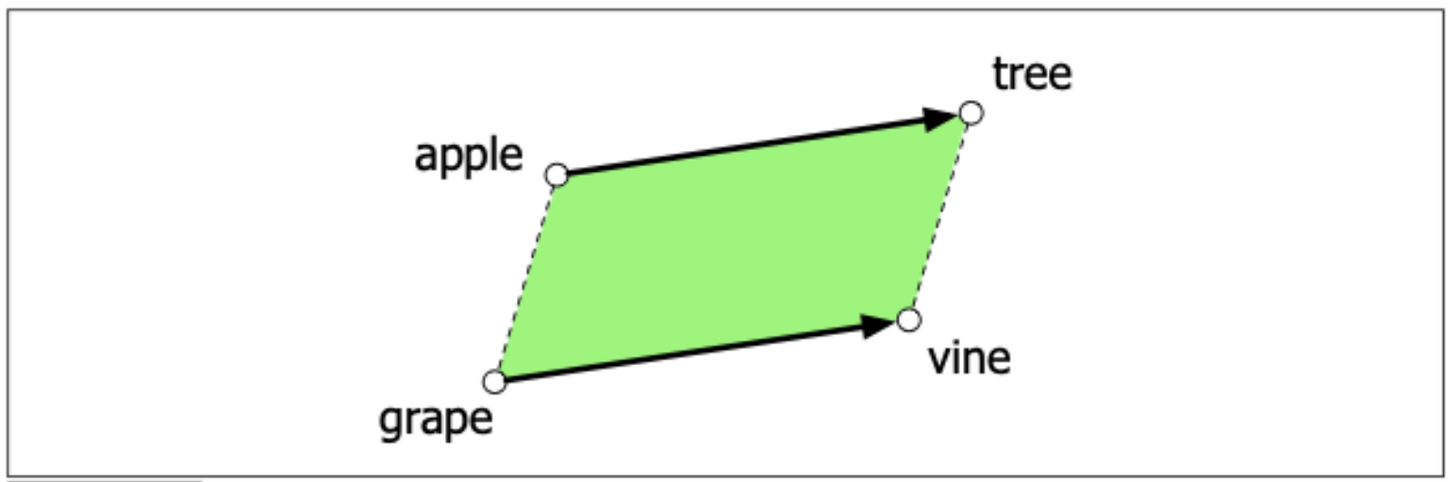


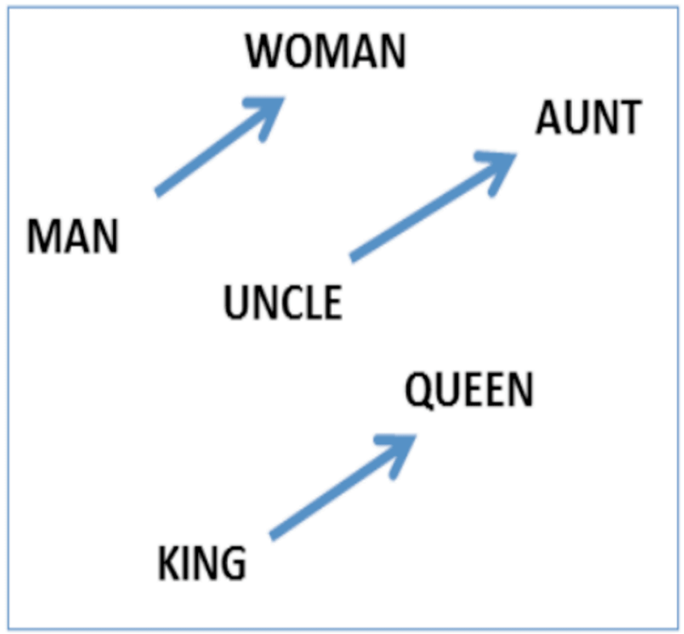
Figure 6.15 The parallelogram model for analogy problems (Rumelhart and Abrahamson, 1973): the location of $\overrightarrow{\text{vine}}$ can be found by subtracting $\overrightarrow{\text{tree}}$ from $\overrightarrow{\text{apple}}$ and adding $\overrightarrow{\text{grape}}$.

apple – tree = grape – vine
apple – tree + vine = grape
tree – apple + grape = vine

So we can try to solve:
apple – tree + vine = X
“tree is to apple as vine is to what?”
tree – apple + grape = X
“apple is to tree as grape is to what

word2vec “analogies”

- `king – man + woman = queen`
- `uncle – man + woman = aunt`



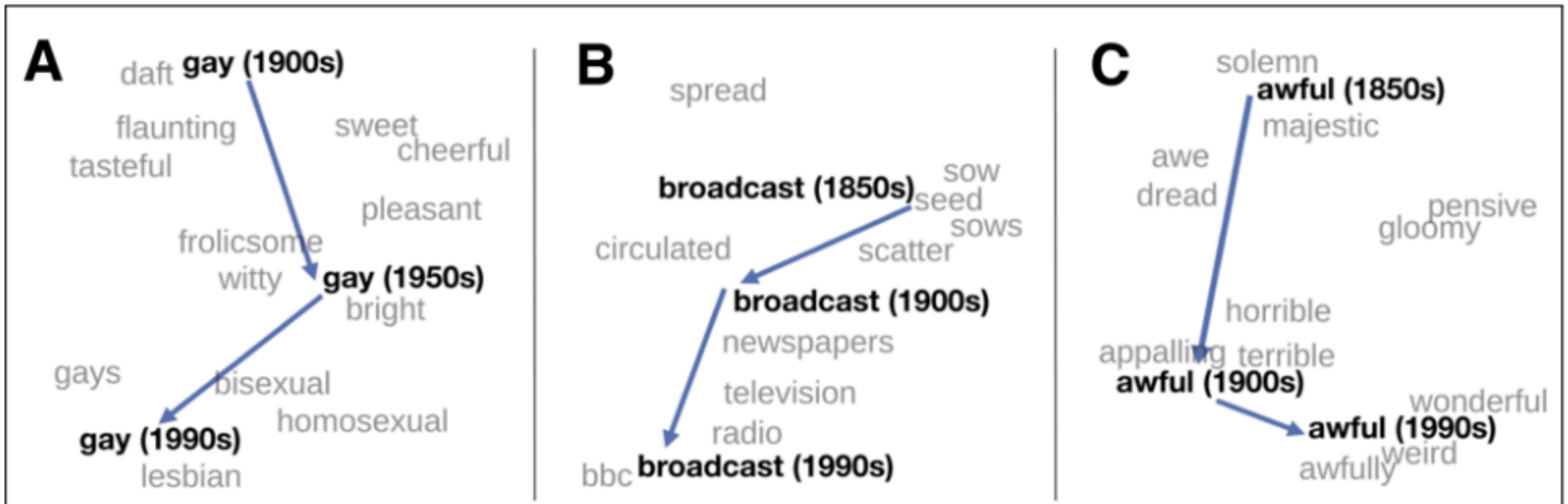
- `surgeon – man + woman = nurse`
- `programmer – man + woman = homemaker`
- `pizza – man + woman = cupcakes`
- `chuckle – man + woman = giggle`

Do we understand why this happens?

- See Bolukbasi et al (2016)- bias and stereotypes.

Word meaning shift

- We can derive word vectors from corpora sampled from different time periods:



Hamilton et al., 2016

OUTLINE

1) Recap: vectors for documents and words, normalization and weighting techniques
2) Dense Vectors: Embeddings and word2vec
3) From words to sentences: compositional distributional semantics
4) Applications/Evaluation

Sentence meanings

- Sequence:

```
man bites dog
dog bites man
```

- Syntax:

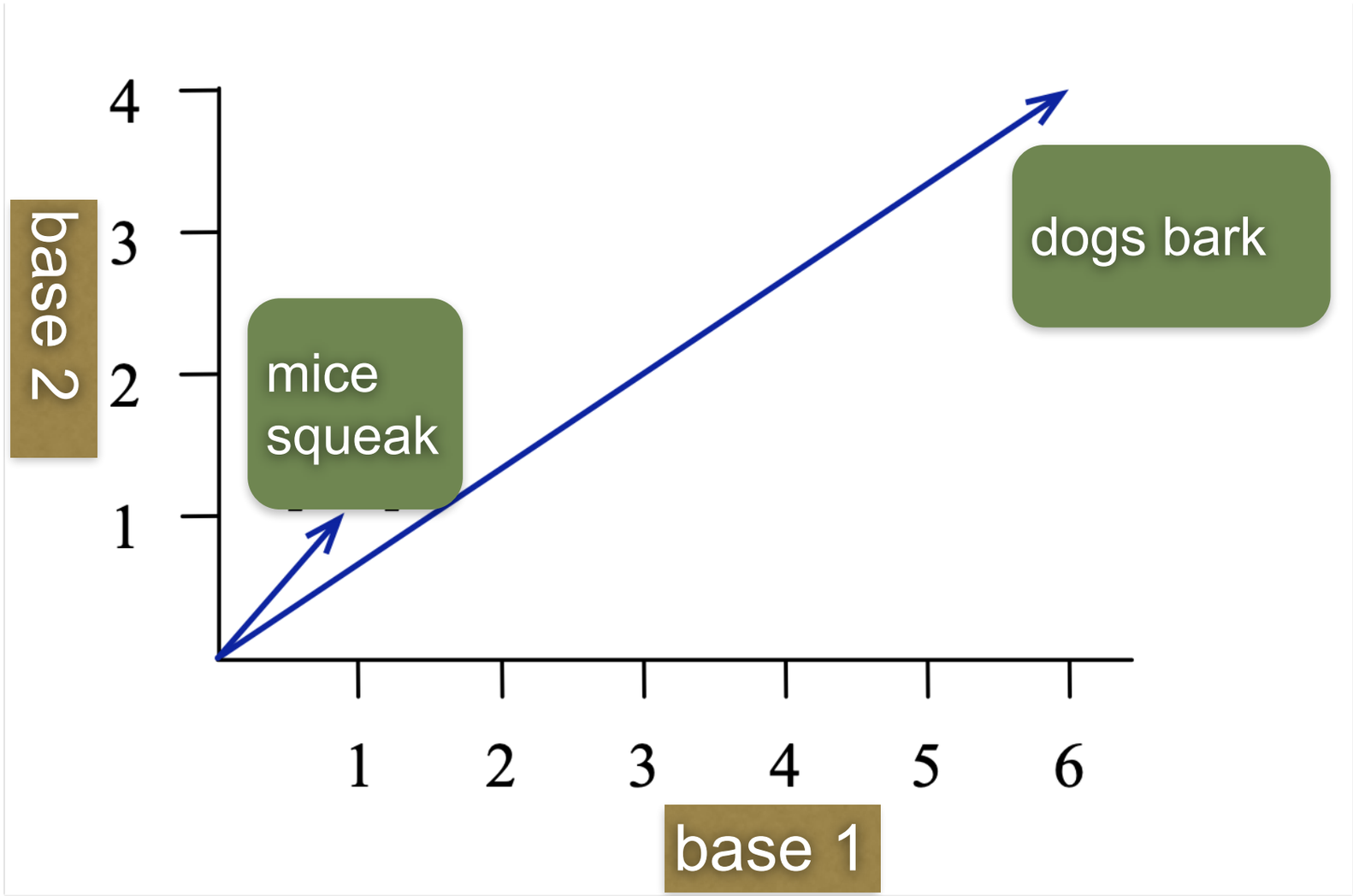
```
(S (NP man) (VP (V bites) (NP dog)))
(S (NP dog) (VP (V bites) (NP man)))
```

- Semantics:

```
bite(man,dog)
bite(dog,man)

bite(e) & biter(e,man) & bitten(e,dog)
bite(e) & biter(e,dog) & bitten(e,man)
```

Vectors for Sentences



Compositional Semantics

We have learnt how to build vectors for words and for documents.

What about for sentences?

Neither of methods used for words or documents can be directly applied to get sentence vectors.

1- We cannot use the method used for document vectors, since words do repeat in a document but very rarely in a sentence.

2- We cannot use the method used for word vectors, since one can collect frequency data for words but not for sentences, as they very rarely repeat.

Naïve way 1

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	5	117	0	0

Figure 15.1 The term-document matrix for four words in four Shakespeare plays. Each cell contains the number of times the (row) word occurs in the (column) document.

Could we do this for sentences?

Not great as a semantic representation:
- Sentences are too short
- We lose sequence/structure

This is a bag of words representation

Naïve way 2

is traditionally followed by	**cherry**	pie, a traditional dessert
often mixed, such as	**strawberry**	rhubarb pie. Apple pie
computer peripherals and personal	**digital**	assistants. These devices usually
a computer. This includes	**information**	available on the internet

Could we do this for sentences?

Not reliable:

- Sentences are too rare (we'd need a massive corpus)
- It's not clear that sentence context works this way

Compositionality

- We need to be able to build **sentence** representations
- … e.g. to assign a natural language sentence to its canonical logical form
- … out of the representations of its components (words)

- This is a new take on the old problem of **compositionality of meaning**, studied for a long time in traditional semantics (i.e. t**he meaning of a sentence is determined by those of its parts**), where, going away from simple approaches assuming bags of words, we want to maintain the semantic information of interest.

Naïve way 3

$$\overrightarrow{\text{vampires kill men}} = \overrightarrow{\text{vampires}} + \overrightarrow{\text{kill}} + \overrightarrow{\text{men}}$$
$$= \overrightarrow{\text{vampires}} \odot \overrightarrow{\text{kill}} \odot \overrightarrow{\text{men}}$$

But again:

- We lose sequence/structure information

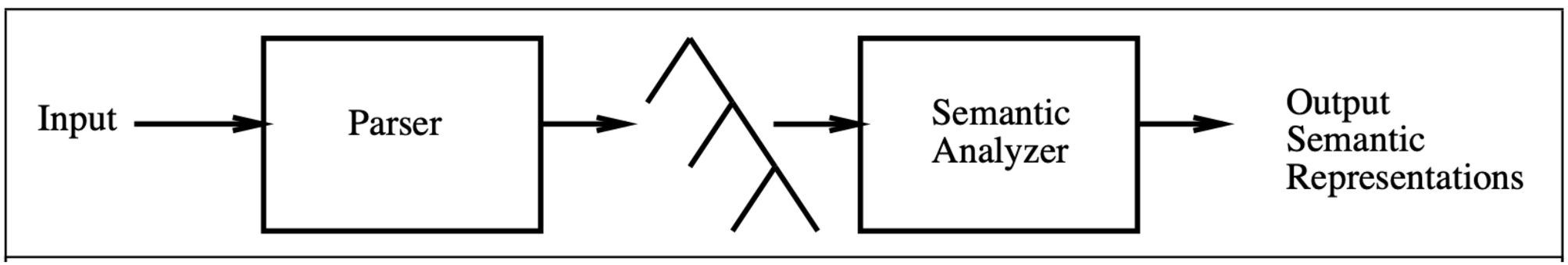
$$\overrightarrow{\text{vampires kill men}} = \overrightarrow{\text{men kill vampires}}$$

Compositionality

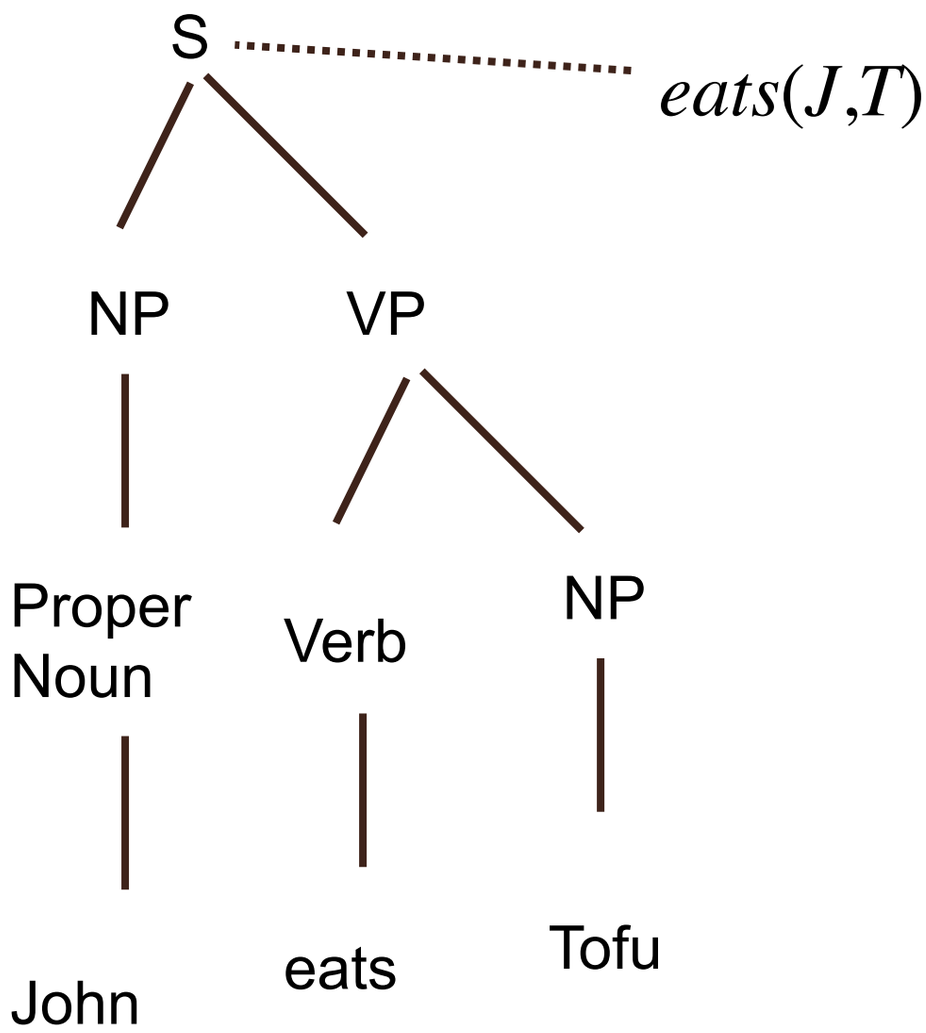
- We need to be able to build **sentence** representations
- … e.g. to assign a natural language sentence to its canonical logical form
- We need to know about the components (words) and how they relate to each other

- syntax: expression (a) ``Kipling serves vegetarian food”
- semantics: expression (b) Serves(Kipling, Vegetarian(food))

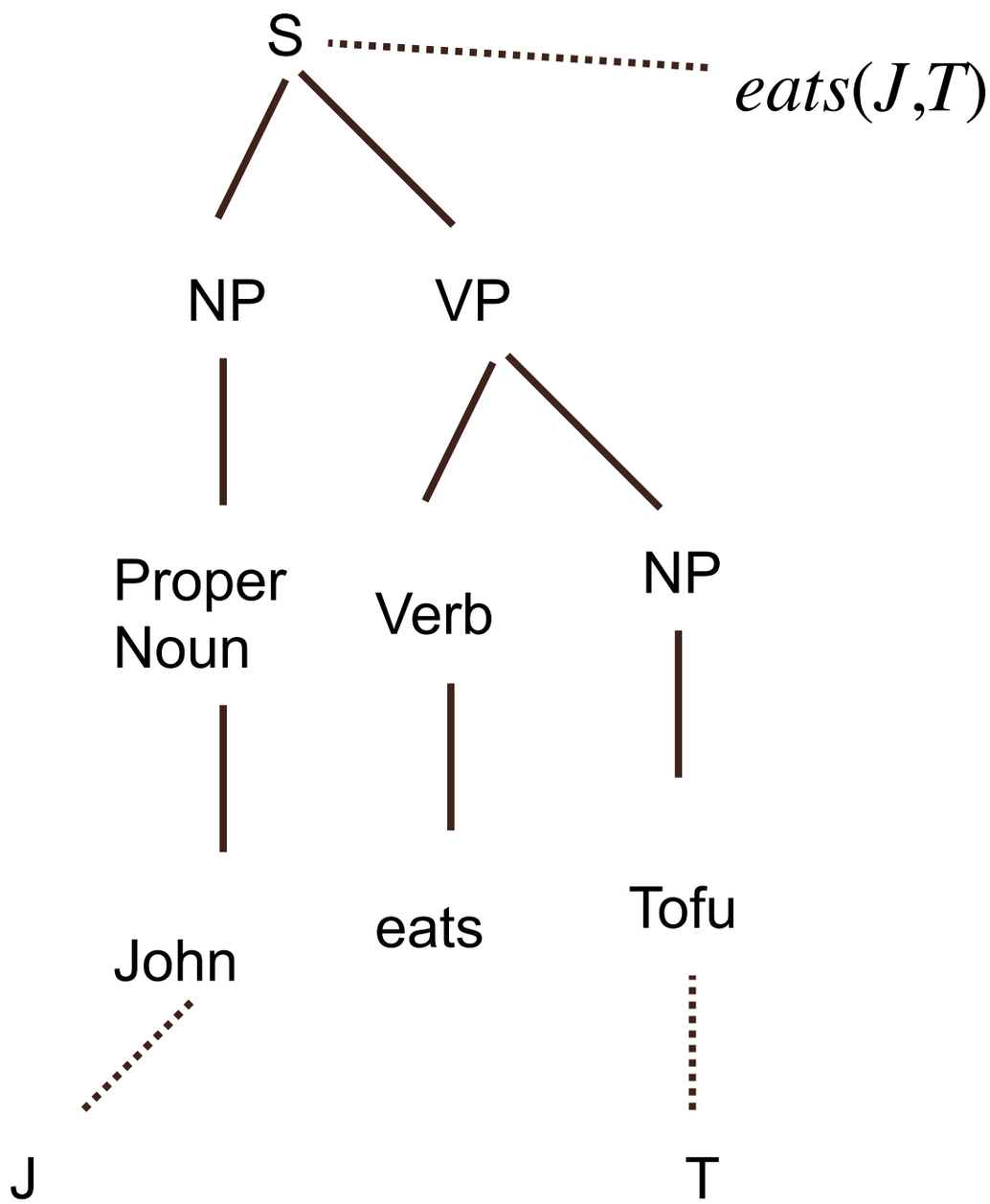
- How do we get from (a) to (b)?
- We need to map syntax to semantics:



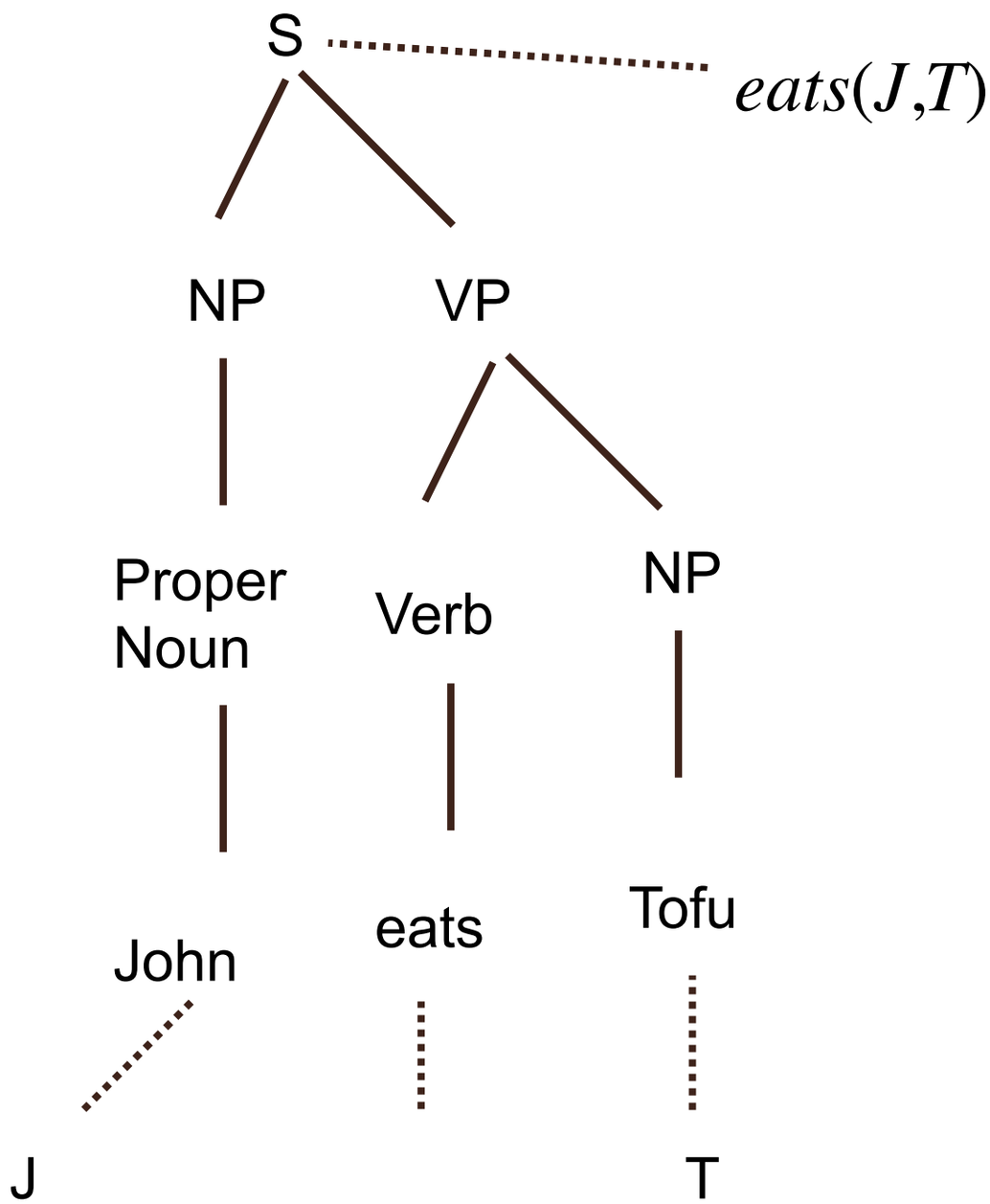
Classical semantic composition



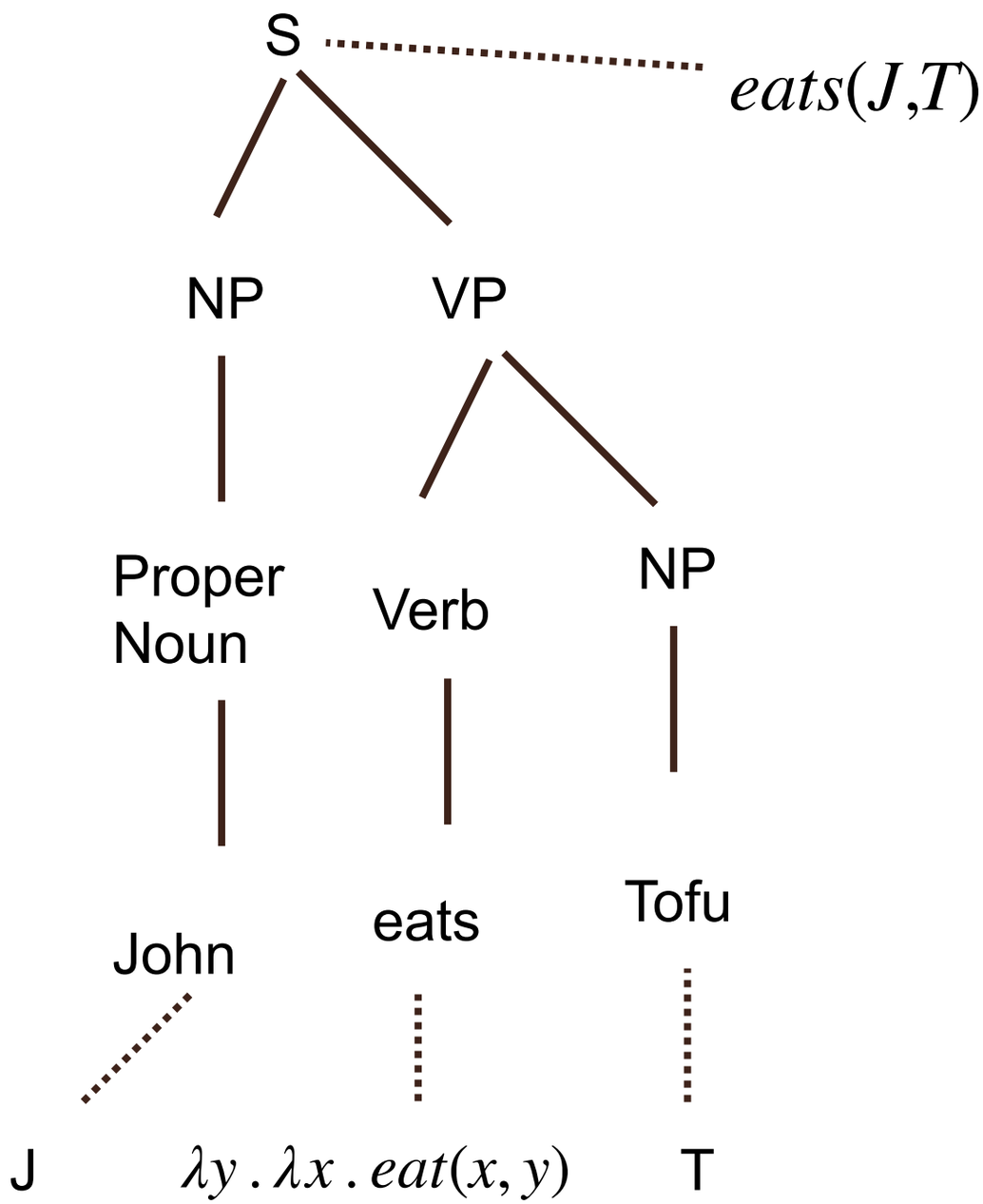
Classical semantic composition



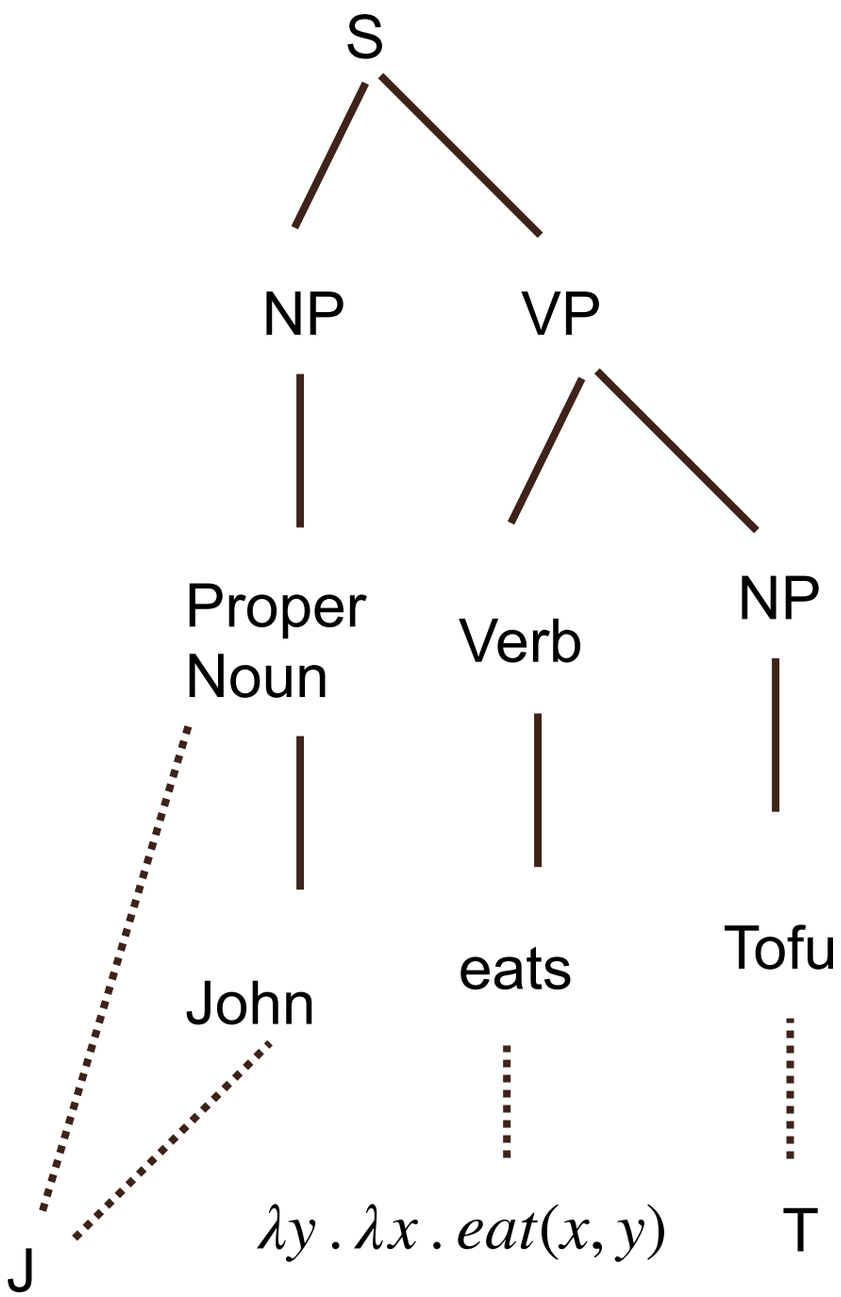
Classical semantic composition



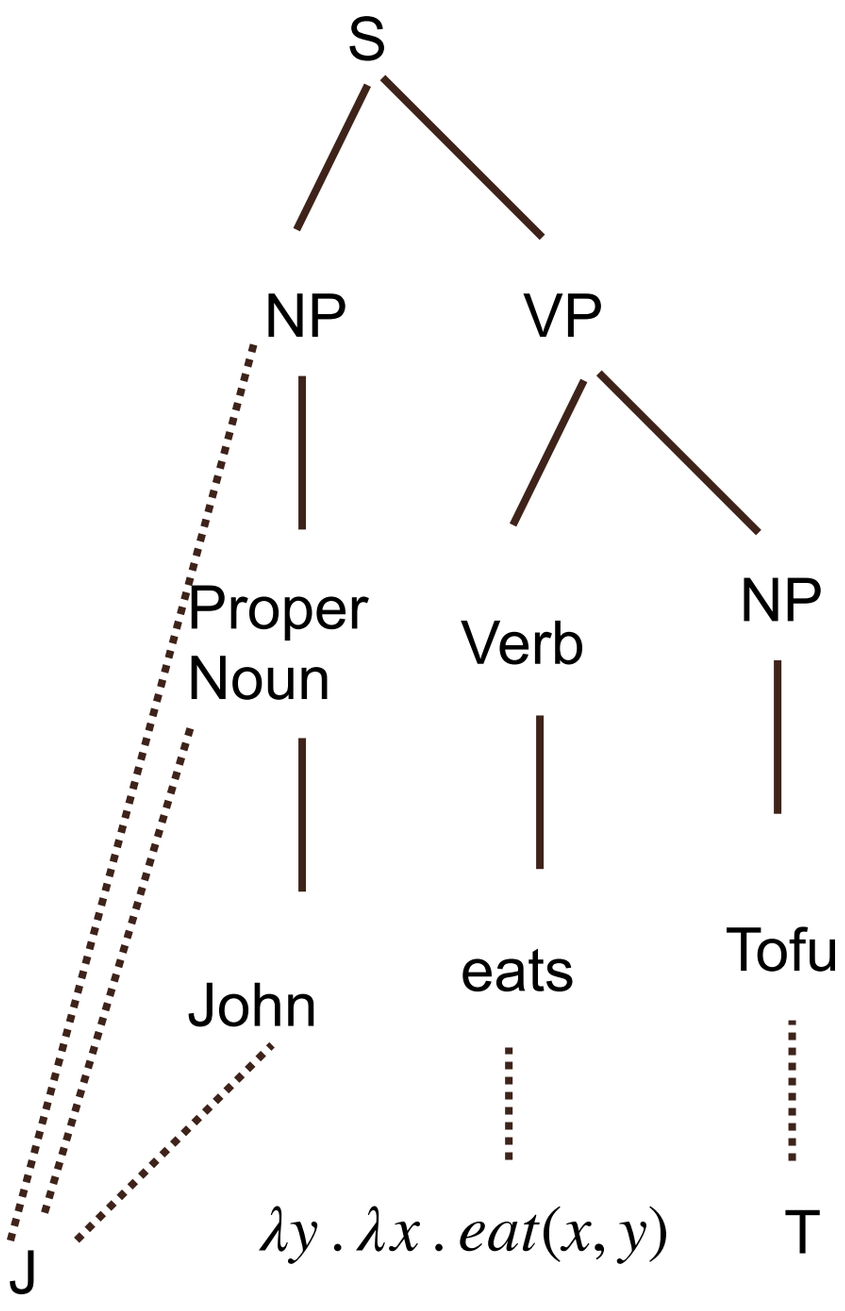
Classical semantic composition



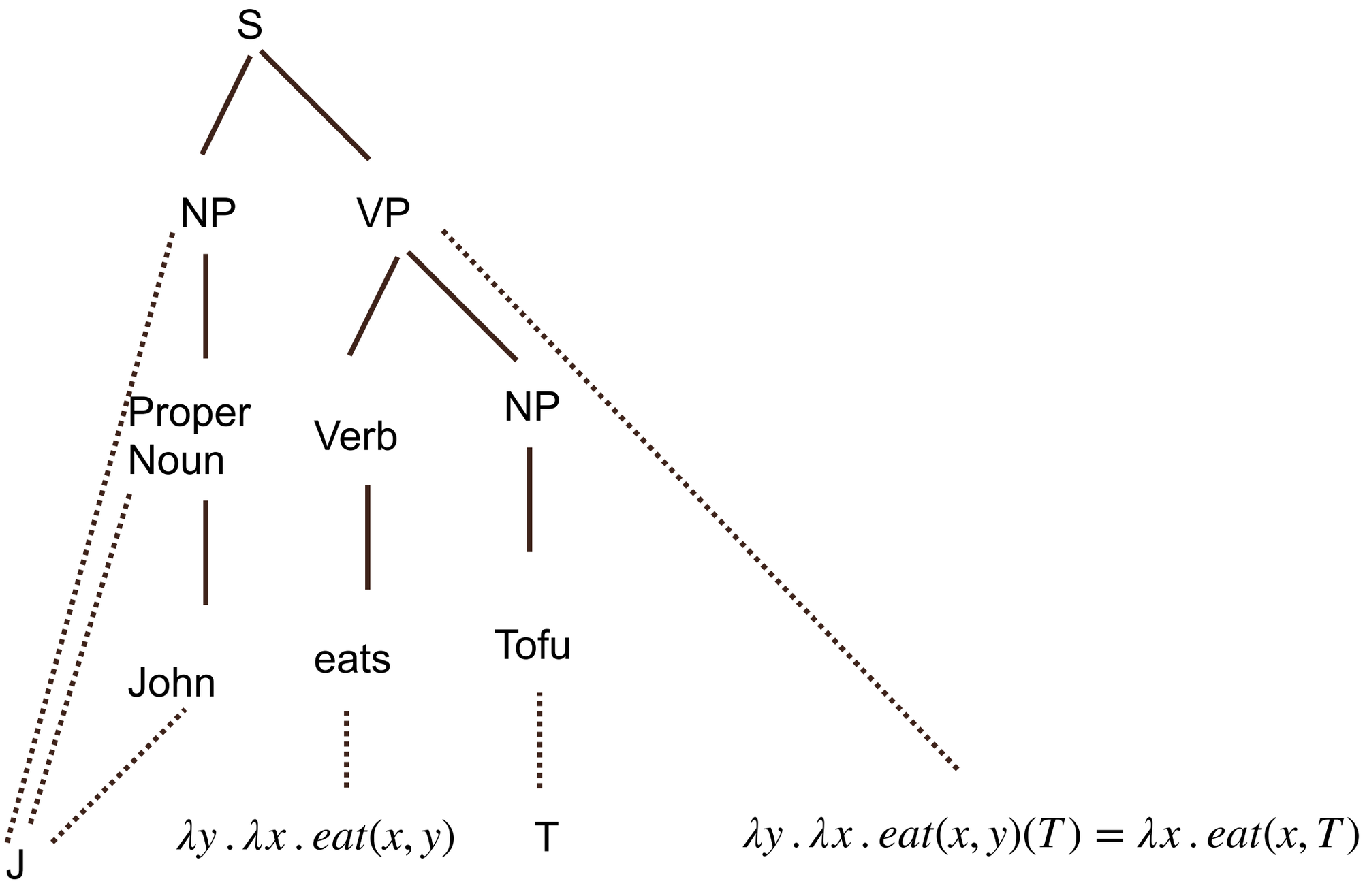
Classical semantic composition



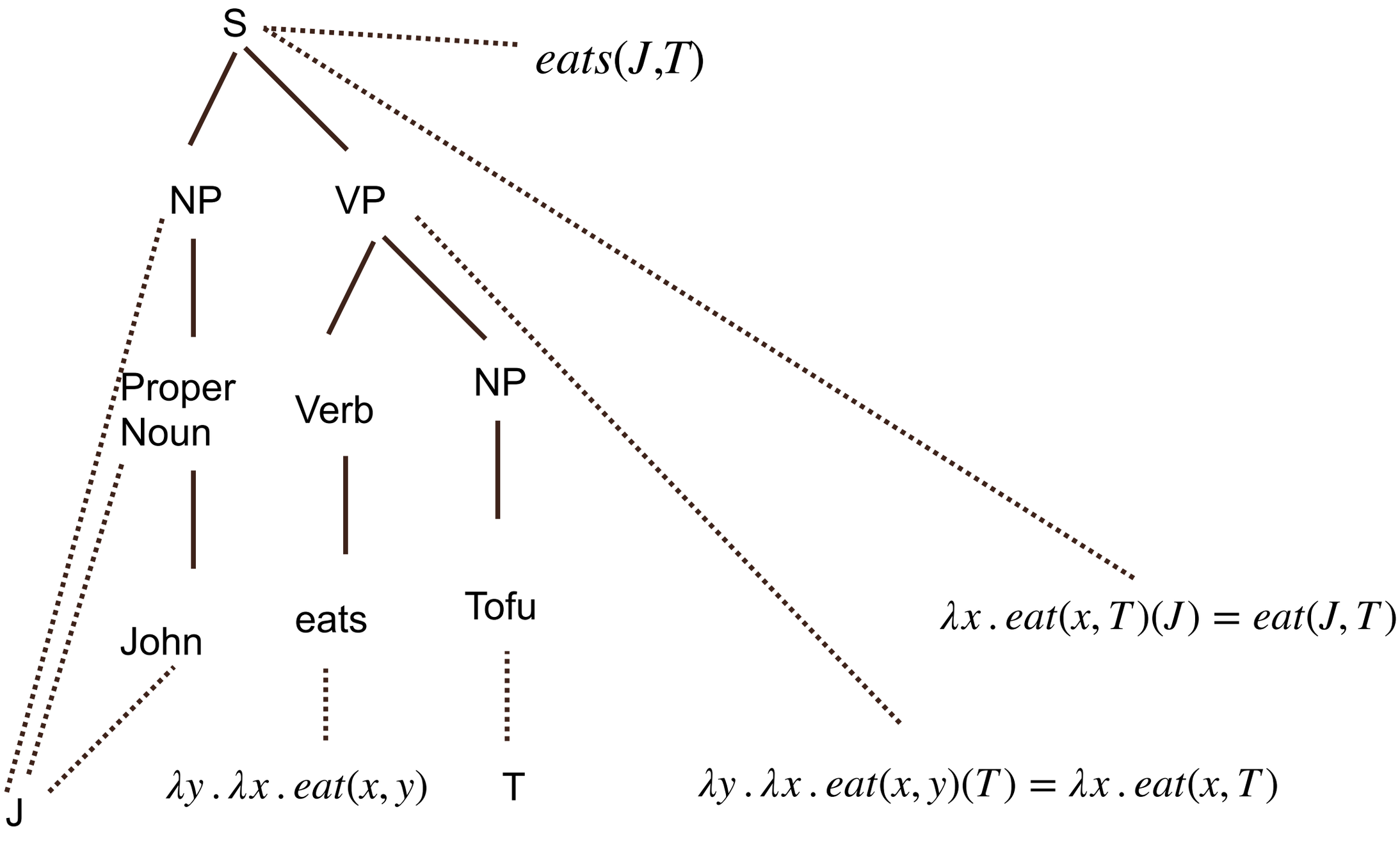
Classical semantic composition



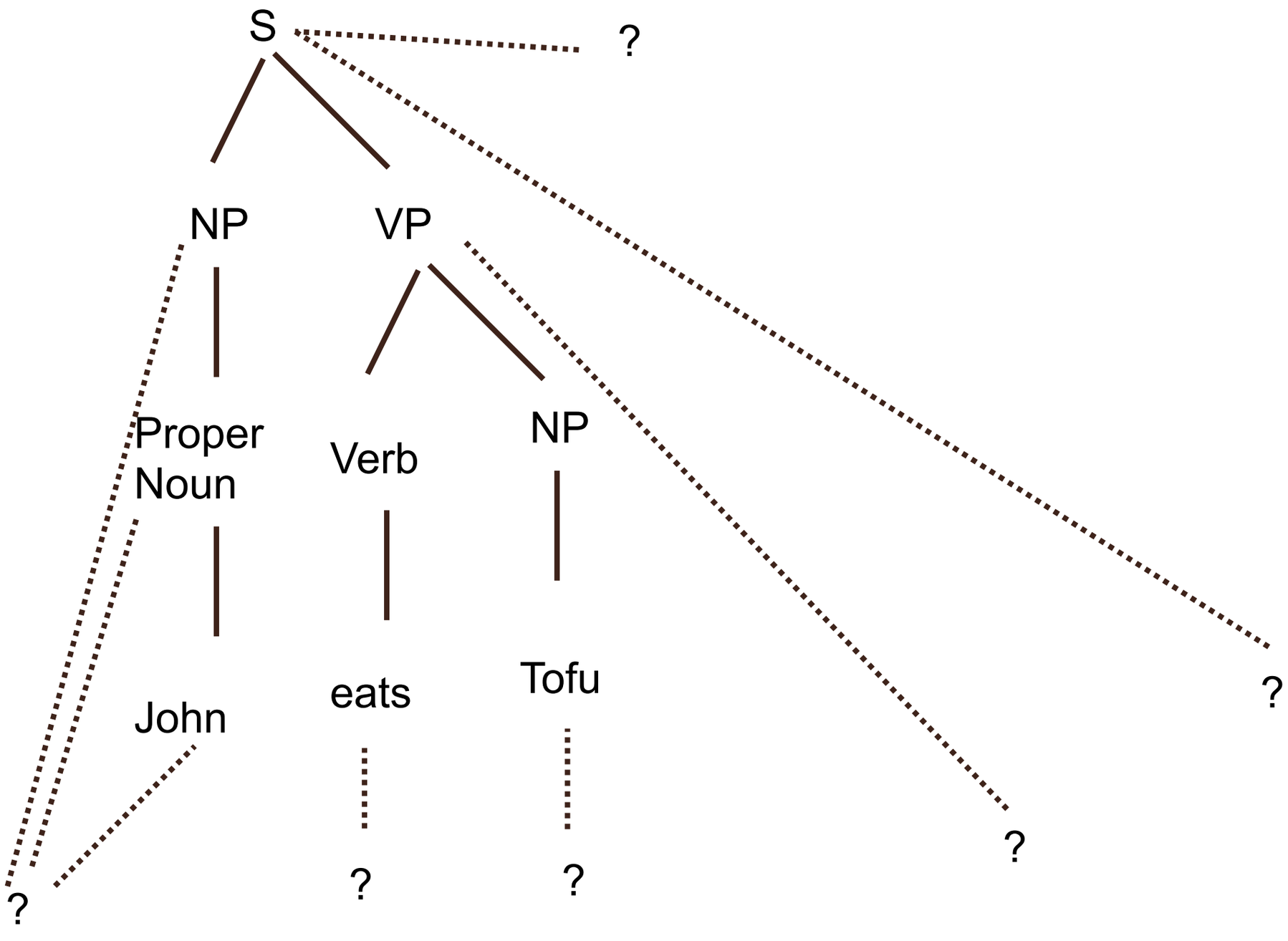
Classical semantic composition



Classical semantic composition



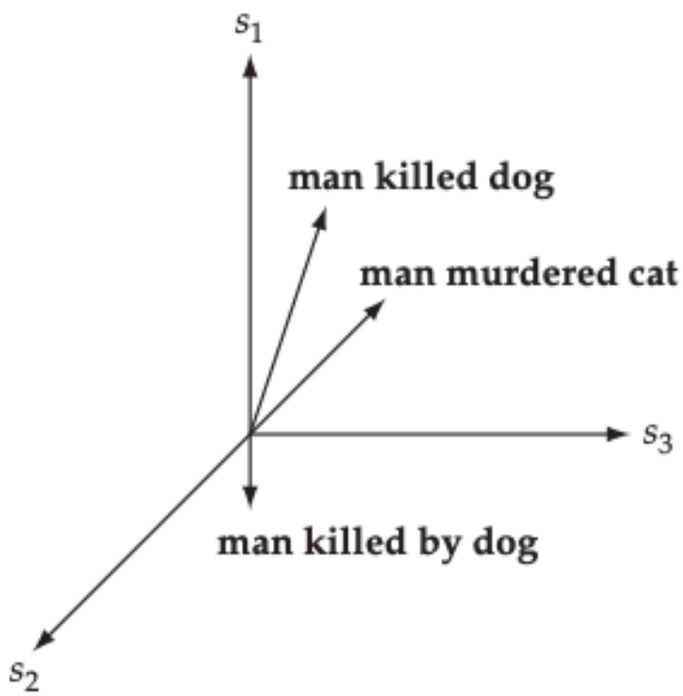
We need a vector-space equivalent …



We need a vector-space equivalent …

- For a model of sentence meaning for vector space semantics, we need a procedure which, given vectors for each word in a phrase or sentence, **combines the vectors** in some way to produce a **single vector** representing the meaning of the sentence where sentences with similar meanings are close together in the space (Clark 2015).

- Of great practical interest for natural language queries: *man killed dog* should not return similar sentences close to the meaning *man killed by dog.* Language not just a bag of words.



Traditional syntax>semantics mapping

Syntactic CFG Rule	Semantics
$S \rightarrow NP\ VP$	$\{VP.sem(NP.sem)\}$

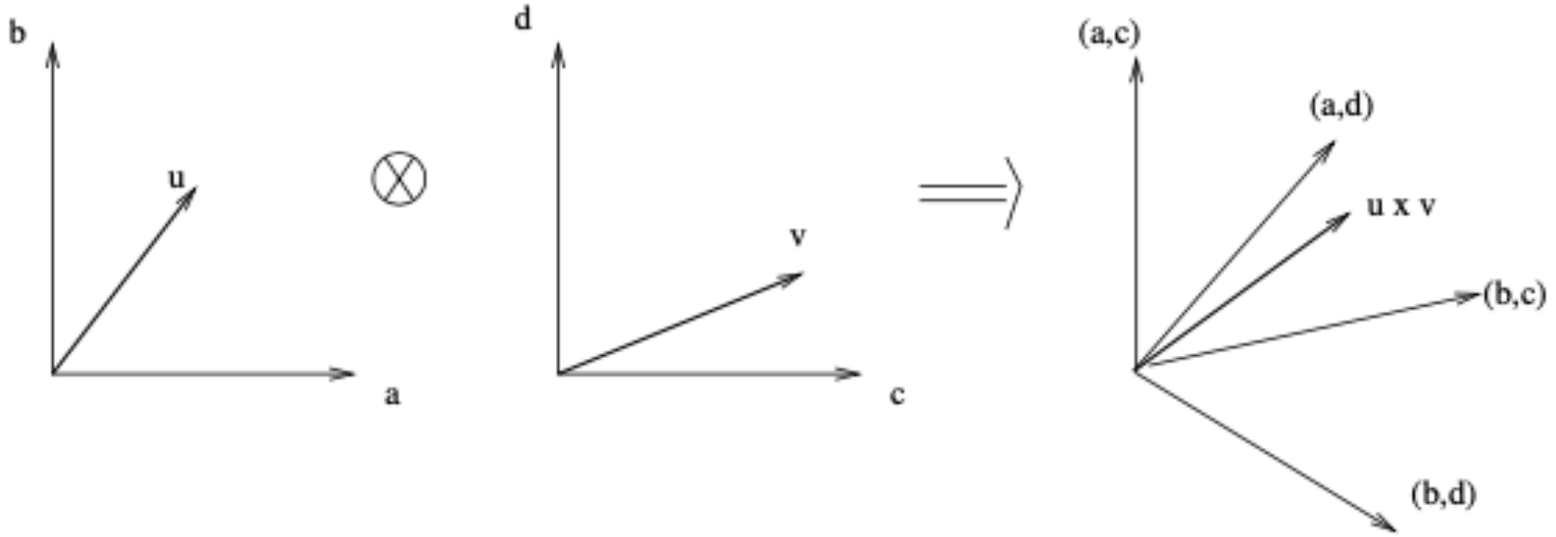
- This means: to get the semantic LF for S, apply the semantics of VP (VP.sem) to that of NP (NP.sem)
- Similarly for the rest of the rules below:

$VP \rightarrow Verb\ NP$	$\{Verb.sem(NP.sem)\}$
$Verb \rightarrow serves$	$\{\lambda x \exists e,y\ Isa(e,Serving) \wedge Server(e,y) \wedge Served(e,x)\}$
$NP \rightarrow ProperNoun$	$\{ProperNoun.sem\}$

Key questions for vector space semantics:

1) What is the semantic type corresponding to a particular syntactic type?
2) How can the vector for e.g. transitive verbs be combined with vectors of a subject and object to give a vector in the sentence space?

Tensor products



$$(u \otimes v)_{(a,d)} = u_a \,.\, v_d$$

Figure 4. The tensor product of two vector spaces; $(u \otimes v)_{(a,d)}$ is the coefficient of $(u \otimes v)$ on the (a, d) basis vector

Clark (2013)

Compositional Distributional Semantics

Logical Types

$$A \mapsto \mathcal{A}$$

$$X/Y \quad \mapsto \quad \mathcal{X} \otimes \mathcal{Y}$$

$$X \setminus Y \quad \mapsto \quad \mathcal{Y} \otimes \mathcal{X}$$

Cancelation Schemata (Forward and Backward composition, Lambek 1958)

$$X/Y \quad Y \implies X \quad \mapsto \quad \mathcal{X} \otimes \mathcal{Y} \quad \mathcal{Y} \implies \mathcal{X}$$

$$Y \quad X \setminus Y \implies X \quad \mapsto \quad \mathcal{Y} \quad \mathcal{Y} \otimes \mathcal{X} \implies \mathcal{X}$$

Compositional Distributional Semantics

Dogs	Chase	White	Cats
N	$(S \backslash N)/N$	N/N	N
		N	
	$S \backslash N$		
S			

Compositional Distributional Semantics

Dogs Chase White Cats

$$T_i \qquad T_{ijk} \qquad T_{kl} \qquad T_l$$

$$T_k \qquad \in \mathcal{N}$$

$$T_{ij} \qquad \in \mathcal{N} \otimes \mathcal{S}$$

$$T_j \qquad \in S$$

Compositional Distributional Semantics

Vectors

$$A \mapsto \mathcal{A} \qquad \mathcal{A} = \{e_i\}_i \quad \ni T_i = \sum_i C_i e_i$$

Compositional Distributional Semantics

Matrices

$$A/B \mapsto \mathcal{A} \otimes \mathcal{B} \qquad \mathcal{A} = \{e_i\}_i \quad \mathcal{B} = \{e_j\}_j$$

$$\mathcal{A} \otimes \mathcal{B} \ni T_{ij} = \sum_{ij} C_{ij}\, e_i \otimes e_j$$

Compositional Distributional Semantics

Cubes

$A/(B/C) \mapsto \mathcal{A} \otimes (\mathcal{B} \otimes C)$ $\quad \mathcal{A} = \{e_i\}_i \quad \mathcal{B} = \{e_j\}_j \quad C = \{e_k\}_k$

$$\mathcal{A} \otimes \mathcal{B} \otimes C \ni T_{ijk} = \sum_{ijk} C_{ijk}\, e_i \otimes e_j \otimes e_k$$

Compositional Distributional Semantics

Matrix Multiplication

$$A/B \quad B \implies A \quad \mapsto \quad (\mathcal{A} \otimes \mathcal{B}) \quad \mathcal{B} \implies \mathcal{A}$$

Tensor Contraction

$$T_{ij} \quad T_j \overset{\text{tensor}\ \ \text{contract}}{\Longrightarrow} T_i$$

$$(\sum_{ij} C_{ij} e_i \otimes e_j)(\sum_i C_j e_j) = \sum_i C_{ij} C_j e_i \langle e_j \mid e_j \rangle$$

Compositional Distributional Semantics

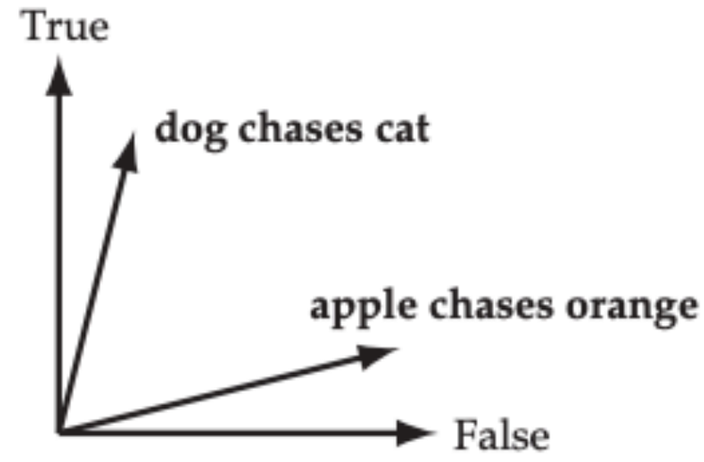


Figure 16.11. An example "plausibility space" for sentences. *Source*: Clark (2013). © Oxford University Press. By permission of Oxford University Press.

Example (Clark, 2013)

	fluffy	run	fast	aggressive	tasty	buy	juice	fruit
$\overrightarrow{dog}$	0.8	0.8	0.7	0.6	0.1	0.5	0.0	0.0
$\overrightarrow{cat}$	0.9	0.8	0.6	0.3	0.0	0.5	0.0	0.0
$\overrightarrow{apple}$	0.0	0.0	0.0	0.0	0.9	0.9	0.8	1.0
$\overrightarrow{orange}$	0.0	0.0	0.0	0.0	1.0	0.9	1.0	1.0

Figure 6. Example noun vectors in $\mathbf{N}$

	⟨fluffy,fluffy⟩	⟨fluffy,fast⟩	⟨fluffy,juice⟩	⟨tasty,juice⟩	⟨tasty,buy⟩	⟨buy,fruit⟩	⟨fruit,fruit⟩ . . .
$\overrightarrow{chases}$	0.8	0.75	0.2	0.1	0.2	0.2	0.0
$\overrightarrow{eats}$	0.7	0.6	0.9	0.1	0.1	0.7	0.1

Figure 10. Example transitive verb vectors in $\mathbf{N} \otimes \mathbf{N}$

	⟨fluffy,fluffy⟩	⟨fluffy,fast⟩	⟨fluffy,juice⟩	⟨tasty,juice⟩	⟨tasty,buy⟩	⟨buy,fruit⟩	⟨fruit,fruit⟩ . . .
$\overrightarrow{chases}$	0.8	0.75	0.2	0.1	0.2	0.2	0.0
dog, cat	0.8,0.9	0.8,0.6	0.8,0.0	0.1,0.0	0.1,0.5	0.5,0.0	0.0,0.0
$\overrightarrow{dog\ chases\ cat}$	0.58	0.36	0.0	0.0	0.01	0.0	0.0

Structural Vector Combination Operations

structure preserving map

Syntax → Semantics

strongly monoidal functor

Pregroup → Vectors Spaces

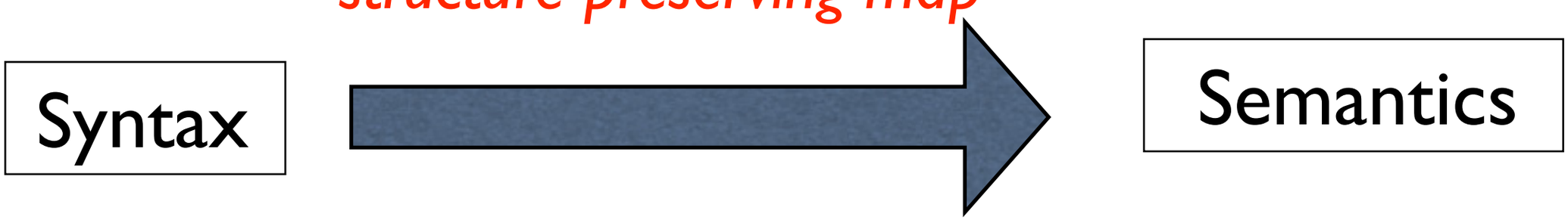
Coecke, Sadrzadeh, Clark, (Lambek's 90th Festschrift), 2010

Preller, Sadrzadeh (JoLLI), 2011

Structural Vector Combination Operations

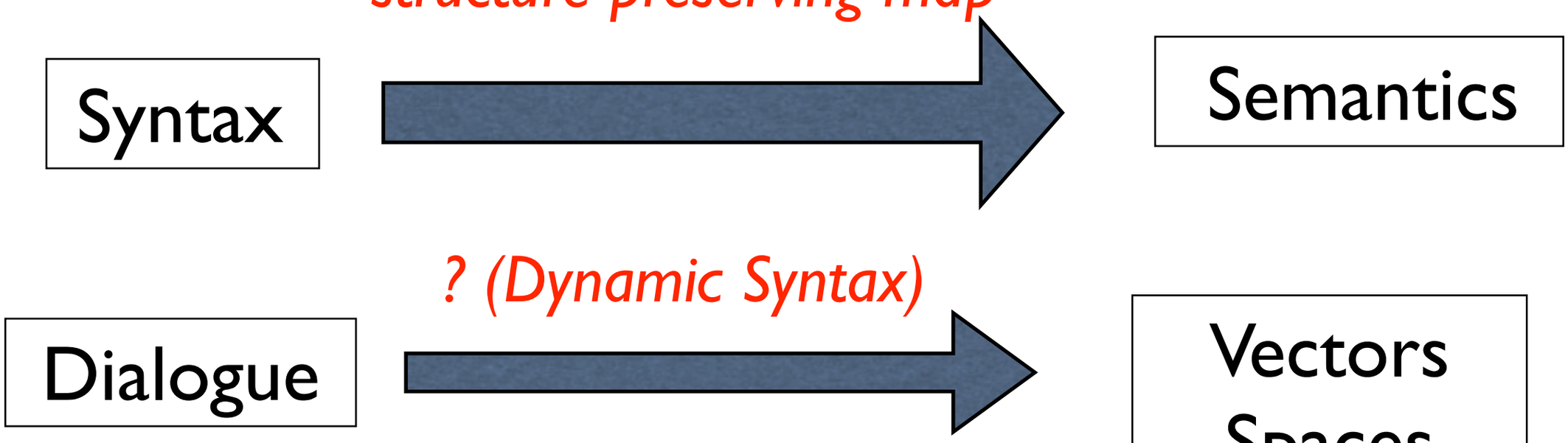
Coecke, Grefenstette,Sadrzadeh, (APAL), 2013

Structural Vector Combination Operations



Maillard, Clark, Grefenstette, (Type Theory and NL, EACL workshop), 2014.
Krishnamurty and Mitchell, (CVSC, ACL workshop), 2013.
Baroni, Bernardini, Zamparelli, (LILT), 2014.

Compositional Distributional Semantics



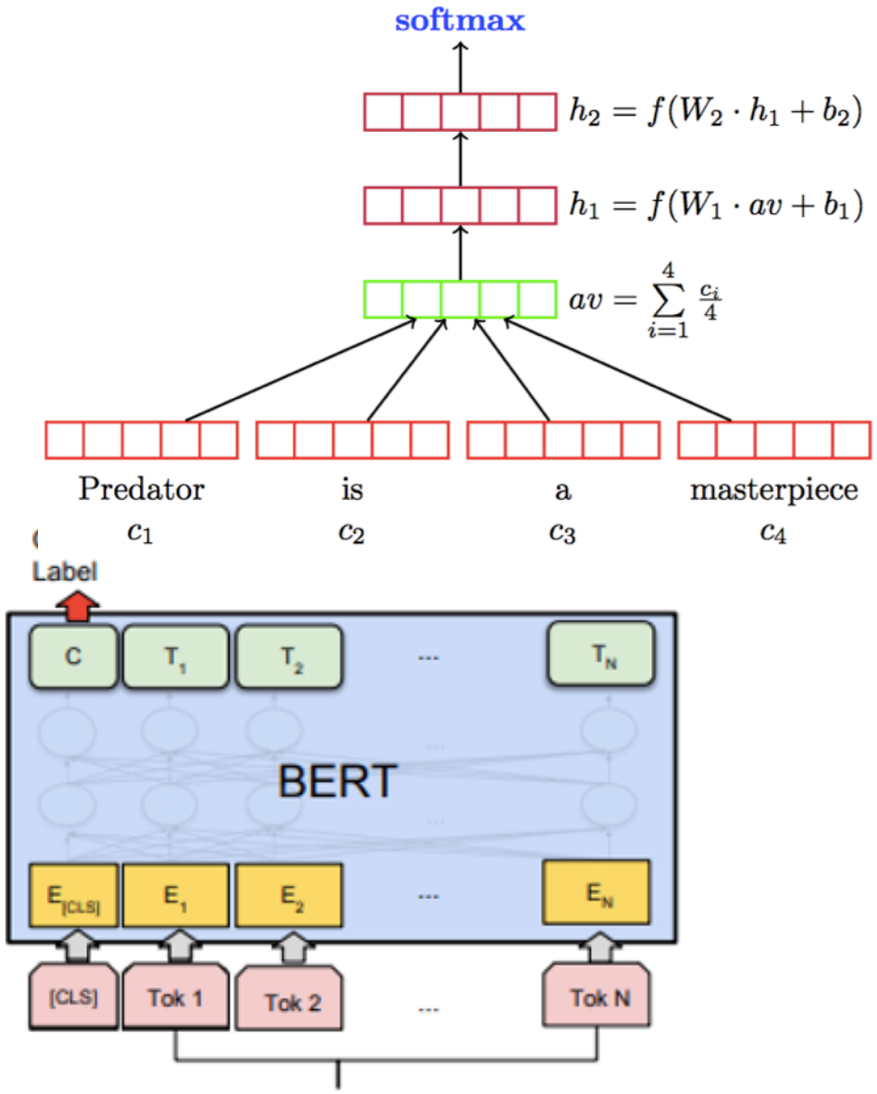
Sadrzadeh, Kempson, Purver, Hough SemDial, 2018;
Purver, Sadrzadeh, Kempson, Wijnholds, Hough JOLLI 2021

Alternatives

- There's more than one way to do this!
  - Map type depends on grammar type
  - Still an active research area …
  - You just need to know that it's possible and remember the basic idea!

- More common: vector sum/average
  - (see previous slides)

- More common: learn the function directly
  - usually with a neural network



OUTLINE

1) Recap: vectors for documents and words, normalization and weighting techniques
2) Dense Vectors: Embeddings and word2vec
3) From words to sentences: compositional distributional semantics
4) Applications/Evaluation

Application/Evaluation

- **Intrinsic evaluation** (test quality of word meanings against some gold standard marked up by human annotators):
  - Word-level: **Synonym, antonym, hypernym, hyponym** detection/ judgement.
  - Word-level task but in context: **Word-sense disambiguation (WSD),** the task of determining the correct sense of a word in context. **Multiple-choice synonym questions.**
  - Sentence-level tasks: **Disambiguation**, **Similarity judgements**, **entailment.**
- **Extrinsic evaluation** (I.e. how much do these vector representations help to improve performance when used as input to other NLP tasks?)
  - Text classification, sentiment analysis, sequence tagging, parsing, NL inference etc.

Word sense disambiguation

- Say we've got an ambiguous verb like 'draw'.
- It has multiple *senses*/meanings.
- One sense it that is is more like 'pulled' as in pulled (out) a ceremonial sword.
- The other is that it is more like attract as in 'attract attention'.
- WSD as a task is to decide which sense is being employed in a sentence. Concretely, to determine which of the following pairs are closer together in terms of meaning, i.e. are most similar:

old man drew ceremonial sword

old man pulled ceremonial sword

Or

annual report attracted attention

annual report drew attention

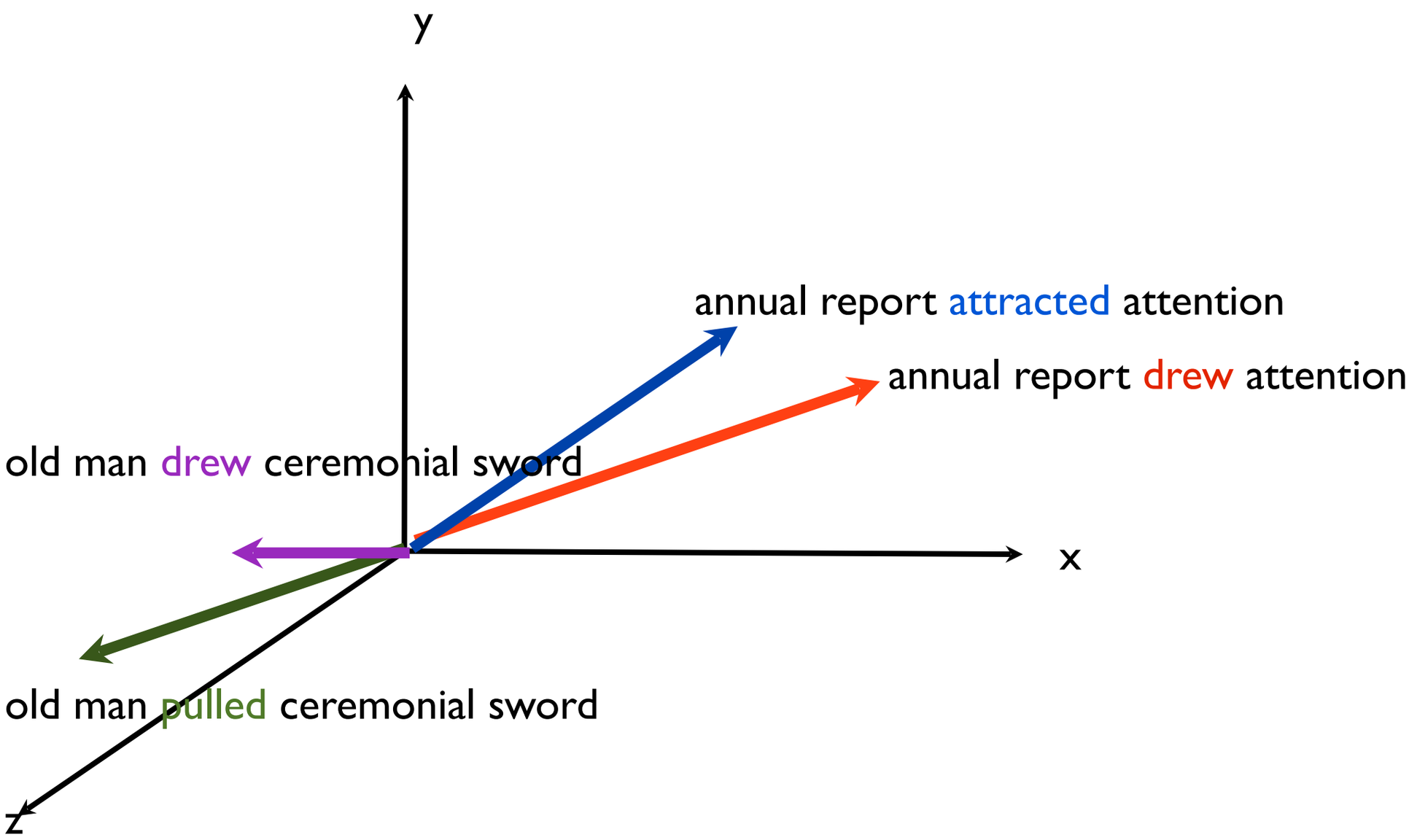
Evaluating Vector Models

- WordSim-353 : a set of of ratings for 353 noun pairs.
- SimLex-999 : quantifies similarity (cup, mug) rather than relatedness (cup, coffee), and includes adjective, noun and verb pairs.
- TOEFL dataset : a set of 80 questions, each consisting of a target word with 4 additional word choices; the task is to
- choose which is the correct synonym.
- Stanford Contextual Word Similarity (SCWS) : human similarity ratings for 2,003 pairs of words, but in their sentential context.
- Word2Vec Dataset: a set of patterns “a is to b as c is to d”. Given a, b, and c, the task is to find d. The words are in certain semantic relationships with each other. For example Athens is to Greece as Oslo is to ? …. Norway.: Mikolov et al. 2013.

Evaluating Sentence Vectors

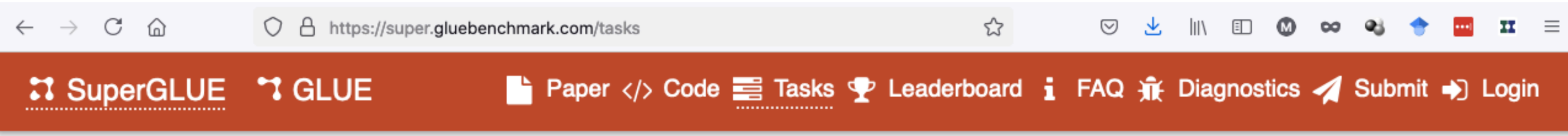
Grefenstette-Sadrzadeh, EMNLP 2012, *Journal of Computational Linguistics 2015*

sentence 1	**sentence 2**
old man draw ceremonial sword	old man attracted ceremonial sword
annual report draw huge attention	annual report attracted huge attention



Extrinsic Evaluation

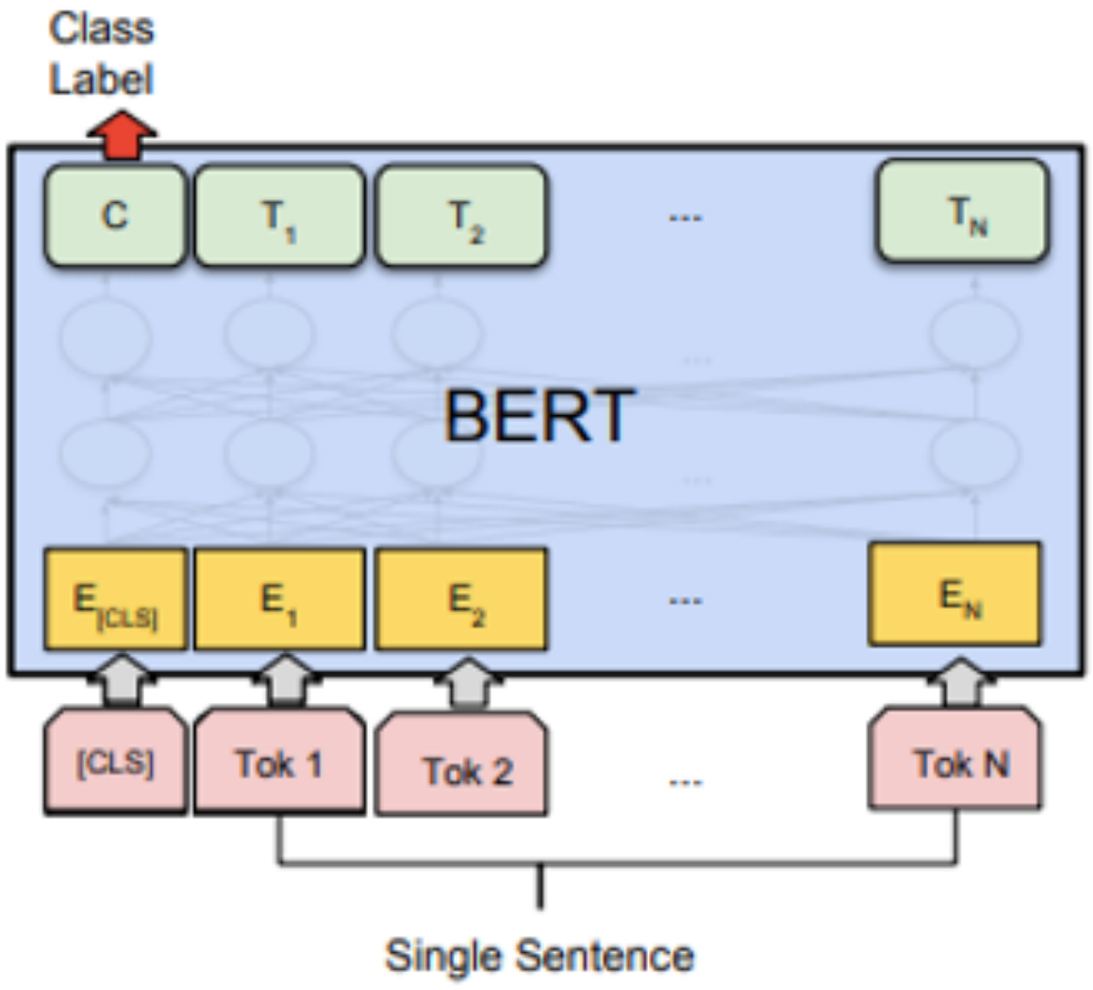
- More common: evaluate your embedding model via its performance in downstream tasks e.g. SuperGLUE



SuperGLUE Tasks

Name	Identifier	Download	More Info	Metric
Broadcoverage Diagnostics	AX-b			Matthew's Corr
CommitmentBank	CB			Avg. F1 / Accuracy
Choice of Plausible Alternatives	COPA			Accuracy
Multi-Sentence Reading Comprehension	MultiRC			F1a / EM
Recognizing Textual Entailment	RTE			Accuracy
Words in Context	WiC			Accuracy
The Winograd Schema Challenge	WSC			Accuracy

BERT as an embedding model



(b) Single Sentence Classification Tasks:
SST-2, CoLA

Reading

- Jurafsky and Martin (3rd ed, online):
  - Chapter 6. **Vector Semantics and Embeddings**.
- Stephen Clark (2015). **Vector Space Models of Lexical Meaning.** In *The Handbook of Contemporary Semantic Theory*, Second Edition. Edited by Shalom Lappin and Chris Fox.