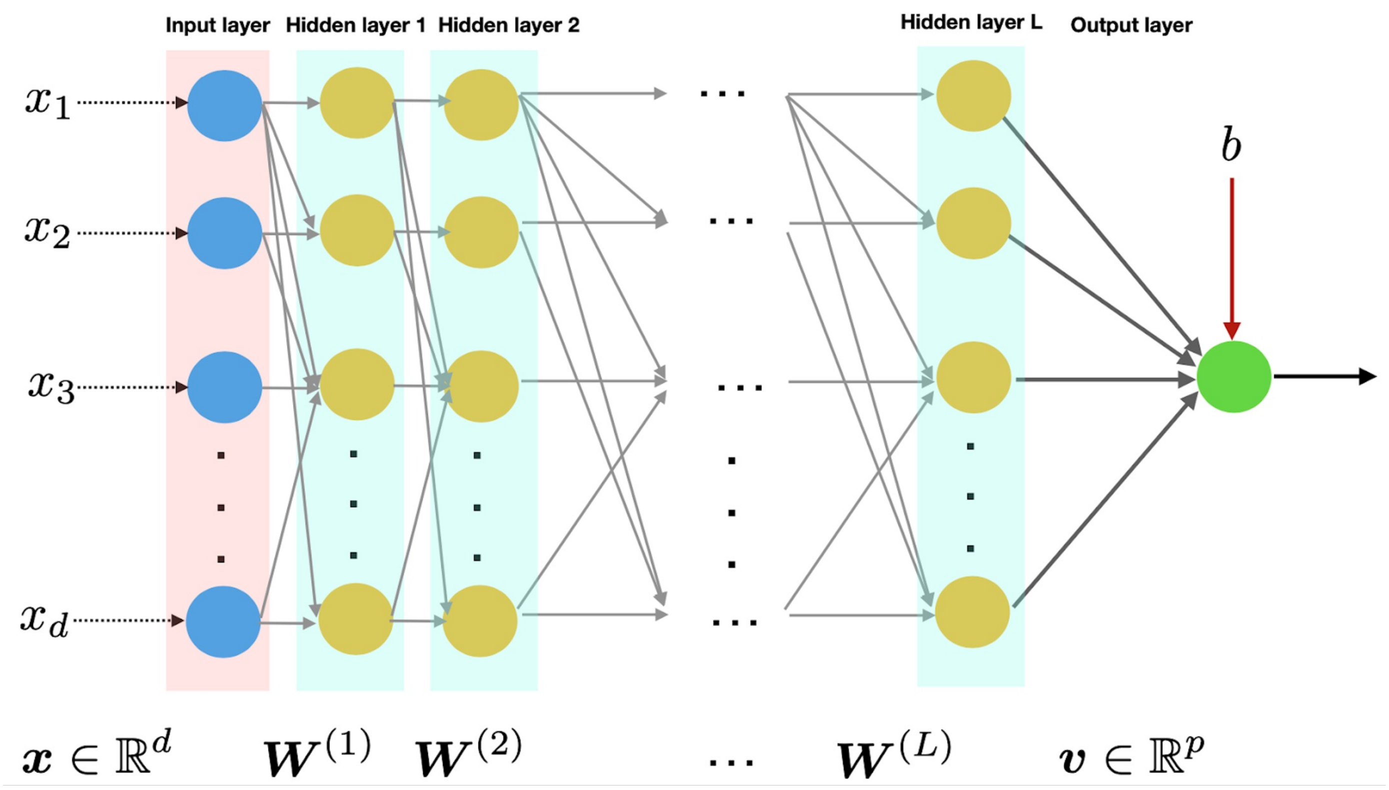
Word Representations

Slides adapted from Claire Cardie and Yoav Artzi

Input to the neural networks?



Word Similarity

- Task: given two words, predict how similar they are

The Distributional Hypothesis:

You shall know a word by the company it keeps

(John Firth, 1957)

Distributional Hypothesis (J.R. Firth 1957)

- Words that occur in **similar contexts** tend to have **similar meanings**
  - “You shall know a word by the company it keeps”
  - “If A and B have almost identical environments ”

- Words which are **synonyms** tend to occur in the **similar context**

Intuition of distributional word similarity

- Suppose I asked you what is **tesgüino**?

A bottle of **tesgüino** is on the table
Everybody likes **tesgüino**
Tesgüino makes you drunk

- From context words, humans can guess **tesgüino** an alcoholic beverage like beer
- Intuition for algorithm:
  - Two words are similar if they have **similar** word **contexts**

🌟Vector semantics

- **Goal**: Learning **representations** (embeddings) of the meaning of words, directly from their **distributions** in text
- Important for NLP applications that make use of meaning
  - Question Answering, Summarization, Detecting paraphrases or plagiarism and dialogue

Term-document matrix

- Count of word w in a document d:
  - Each document in a $\boxed{\text{count vector}}$ in N^V

Document

	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 / Term

Word-word matrix

- Instead of entire document, use smaller context
  - Paragraph
  - Window of ~4 words
- Word is not defined by counts of context words

	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	

TF-IDF: Weighting terms in the vector

- Not all words are equally important
  - Some words just co-occur frequently with many different words (e.g. the, they, it)
- **Term Frequency:** Words that occur nearby frequently (maybe *pie* nearby *cherry*) are more important.
- **Inverse Document Frequency:** Words that are too frequent may be unimportant (e.g. *the*, *it*, *he*, she).

TF-IDF

TF-IDF is a measure of originality of a word by comparing the number of times a word appears in a doc with the number of docs the word appears in.

$$\text{TF-IDF} = \text{TF}(t, d) \times \text{IDF}(t)$$

Term frequency

Number of times term t appears in a doc, d

Inverse document frequency

$$\log \frac{1 + n}{1 + df(d, t)} + 1$$

# of documents

Document frequency of the term t

Measuring similarity

- We have words *w* and *v*
- How do we measure their similarity?

- **Dot product** or **inner product** from linear algebra

$$dot\text{-}product(\vec{v}, \vec{w}) = \vec{v} \cdot \vec{w} = \sum_{i=1}^{N} v_i w_i$$

  - High when two vectors have large values in same dimensions
  - Low (actually 0) for **orthogonal vectors**

Problem with dot product

$$dot\text{-}product(\vec{v},\vec{w}) = \vec{v}\cdot\vec{w} = \sum_{i=1}^{N} v_i w_i$$

- Dot product is longer if the vector is longer. Length:

$$\|\vec{v}\| = \sqrt{\sum_{i=1}^{N} v_i^2}$$

- Vectors are longer if they have higher values in each dimension
  - More frequent words cause higher dot products
  - Bad since we are sensitive to word frequency

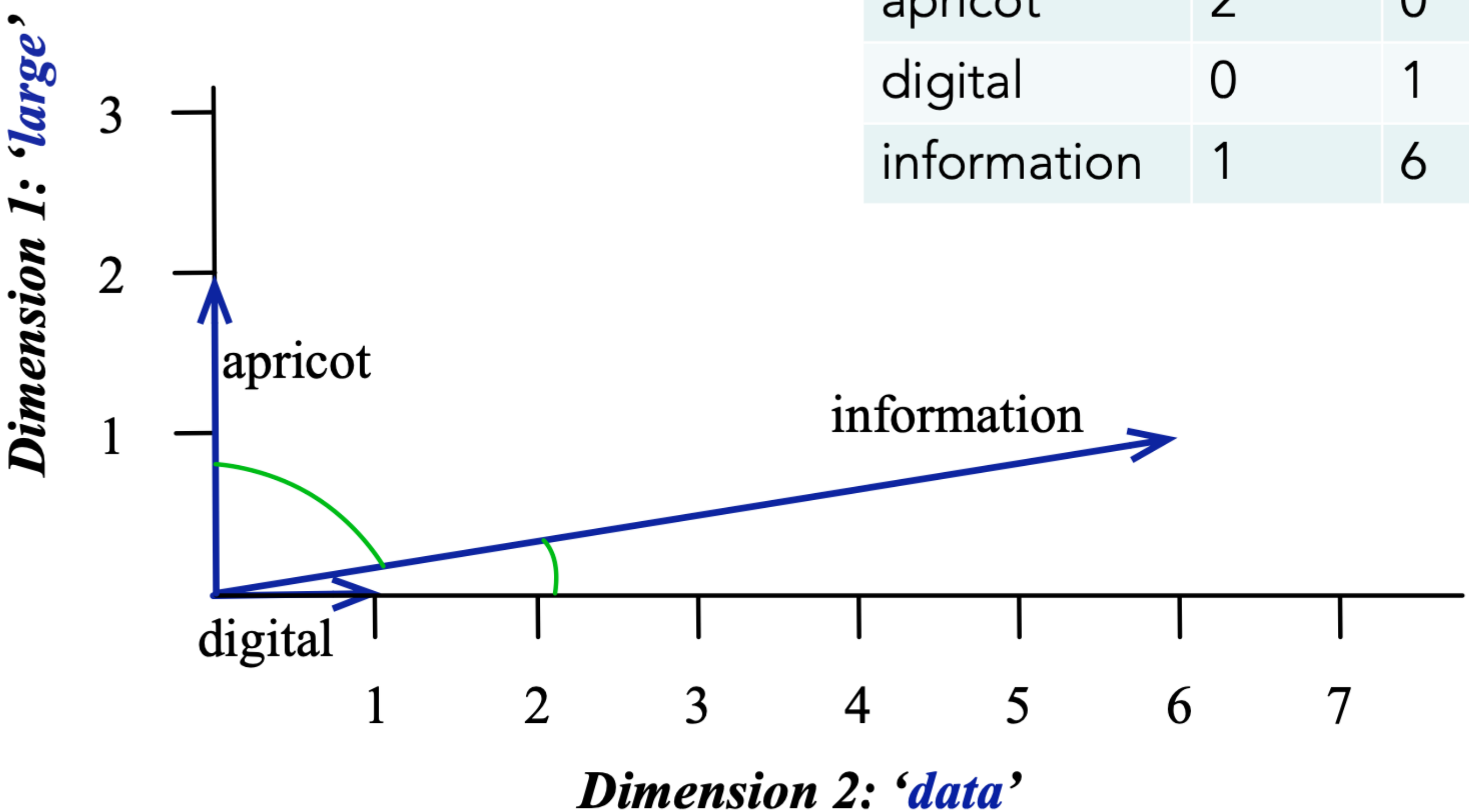
Solution: Cosine similarity

- Just divide the dot product by the length of the two vectors!

$$\text{cosine}(\mathbf{v},\mathbf{w}) = \frac{\mathbf{v}\cdot\mathbf{w}}{|\mathbf{v}||\mathbf{w}|} = \frac{\sum_{i=1}^{N} v_i w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

Visualizing cosine similarity

	large	data
apricot	2	0
digital	0	1
information	1	6



Sparse vs. Dense vectors

- Sparse
  - Most entries are 0
  - Vectors are long (|V| = 20,000 to 50,000)

- **Alternative: Dense**
  - Most elements are non-zero
  - Vectors are short (|V| = 25-1024)
  - Turns out to be very effective

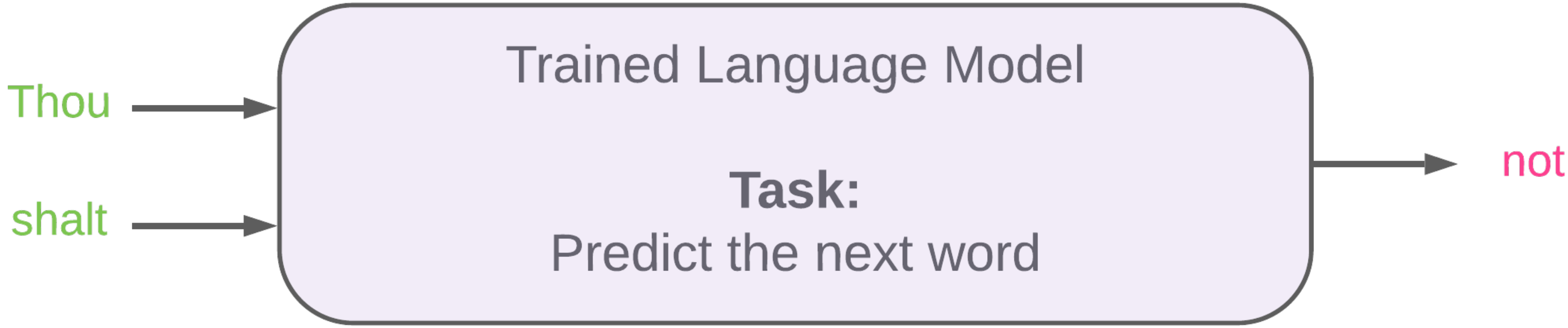
Review: Language Modeling

- Next-word prediction task

input # 1 input # 2 output

Thou shalt ______

Review: Language Modeling



Review: Language Modeling

Thou shalt not make a machine in the likeness of a human mind

Sliding window across text

thou	shalt	not	make	a	machine	....

Dataset

input 1	input 2	output
thou	shalt	not

Thou shalt not make a machine in the likeness of a human mind

Sliding window across text

thou	shalt	not	make	a	machine	....
thou	shalt	not	make	a	machine	....

Dataset

input 1	input 2	output
thou	shalt	not
shalt	not	make

Review: Language Modeling

Thou shalt not $\boxed{\text{make a machine}}$ in the likeness of a human mind

Sliding window across text

thou	shalt	not	make	a	machine	....
thou	shalt	not	make	a	machine	....
thou	shalt	not	make	a	machine	....
thou	shalt	not	make	a	machine	....

Dataset

input 1	input 2	output
thou	shalt	not
shalt	not	make
not	make	a
make	a	machine

Look both ways

Magd drove to the beach in a _car__

Magd drove to the beach in a _red_ car

How to learn **Dense** vectors?

- Algorithms?
  - Skipgram

Skipgram

Magd drove to the beach in a red car on ....

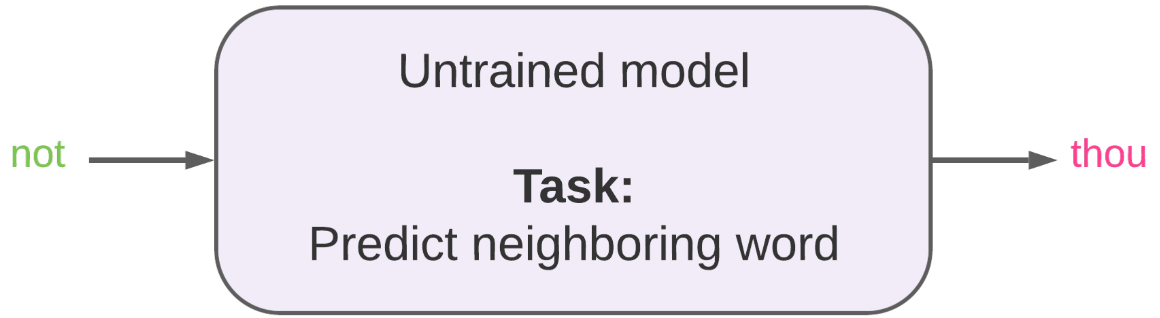
Skipgram

Thou shalt not make a machine in the likeness of a human mind

thou	shalt	not	make	a	machine	....

input word	target word
not	thou
not	shalt
not	make
not	a

Revisiting training process.....

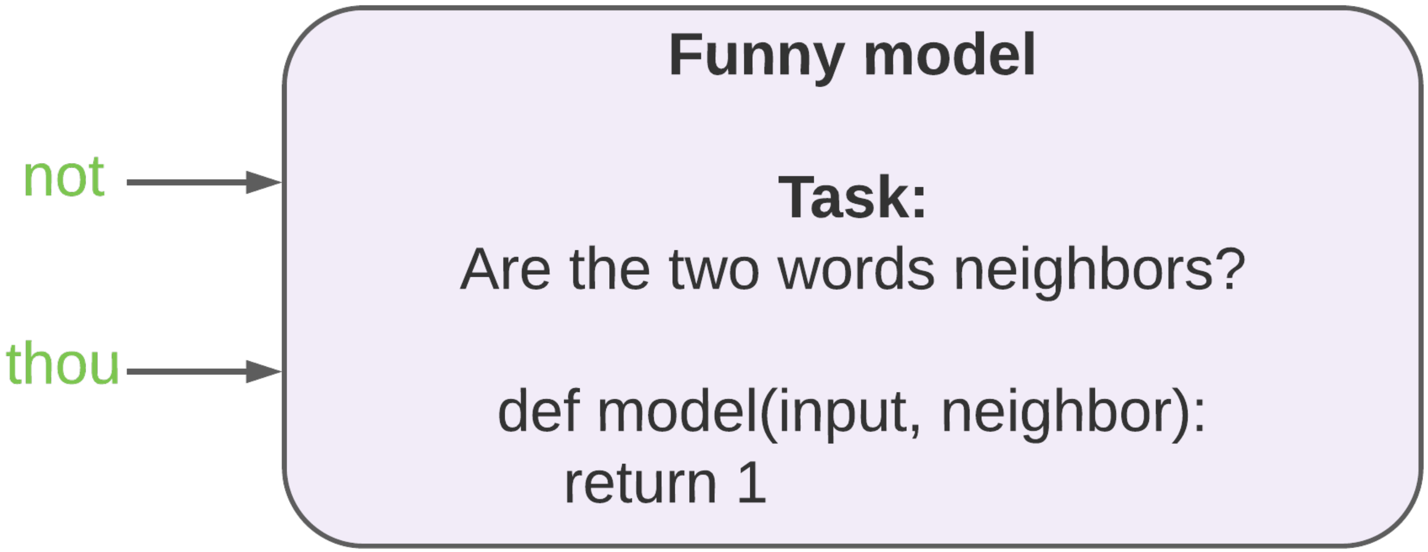


New dataset

input word	neighbor word	target
not	thou	1
not	shalt	1
not	make	1
not	a	1

- What can go wrong with this?

Need negative examples



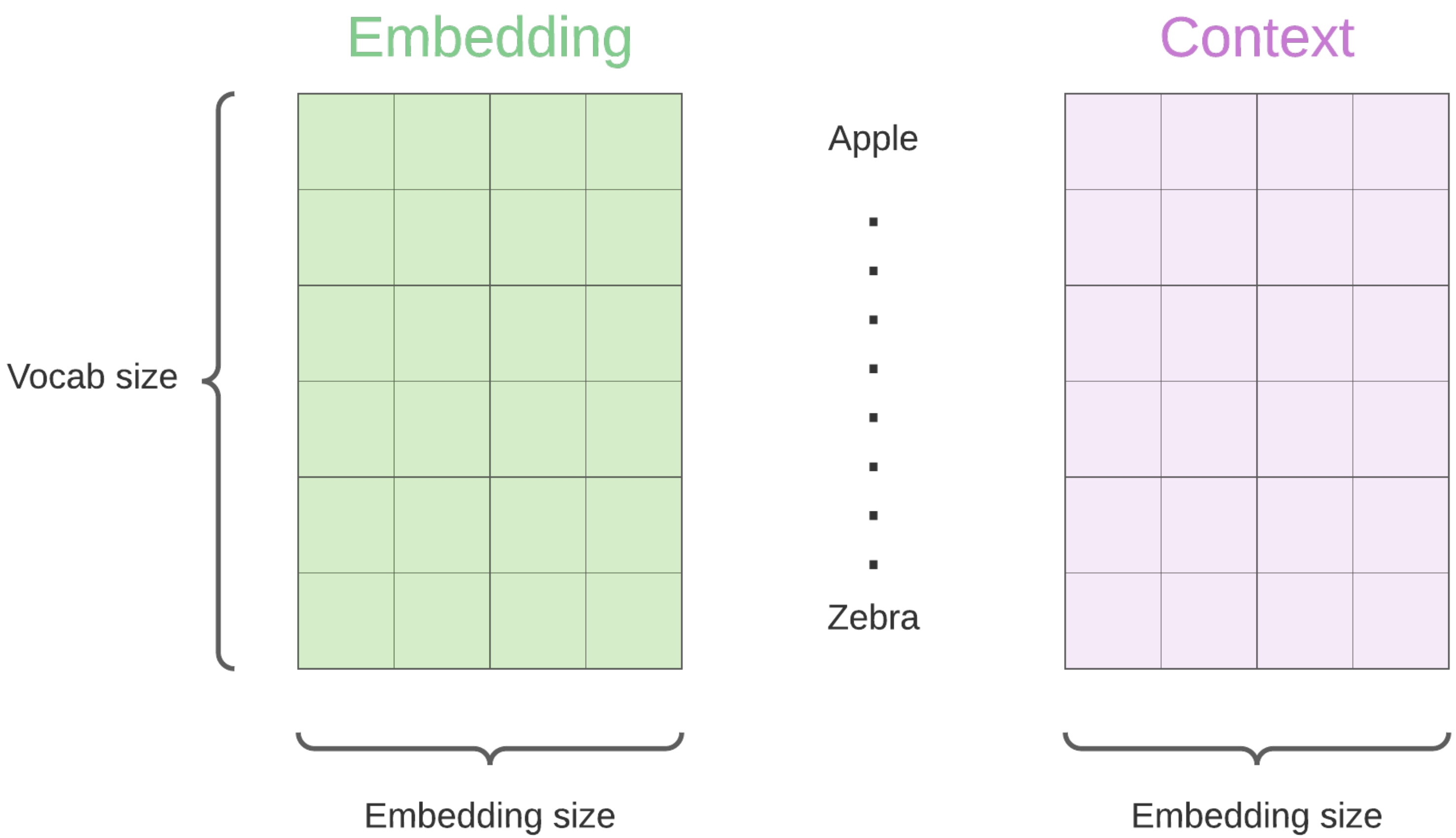
Need negative examples

- How do we pick these negative examples?
  - Pick randomly from vocabulary

input word	output word	target
not	thou	1
not	?	0
not	?	0
not	shalt	1
not	?	0
not	?	0
not	make	1

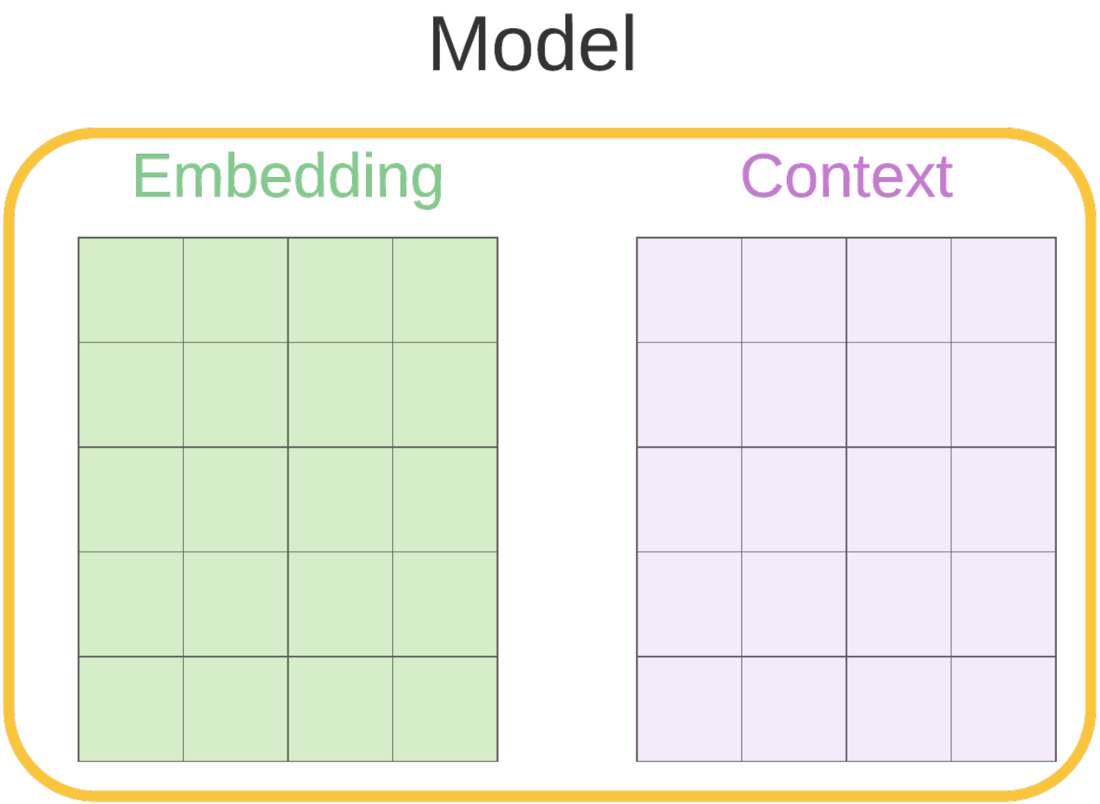
Negative examples

Learning skip-gram embeddings



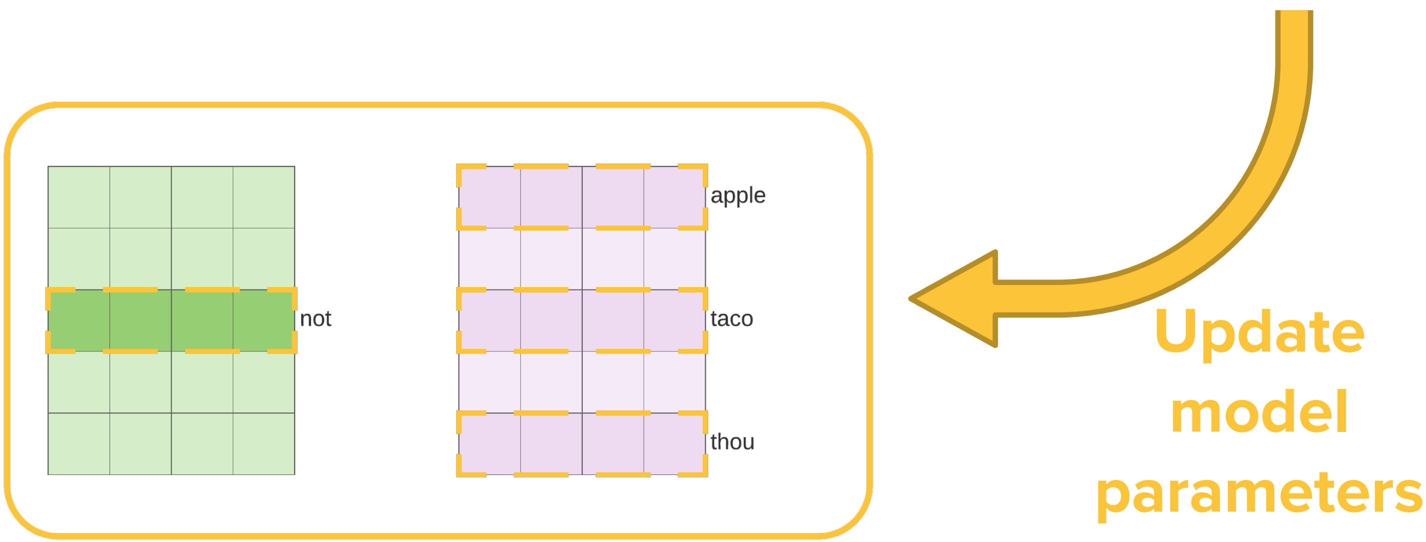
Learning skip-gram embeddings

input	output	target
not	thou	**1**
not	apple	**0**
not	taco	**0**
...	...	**...**



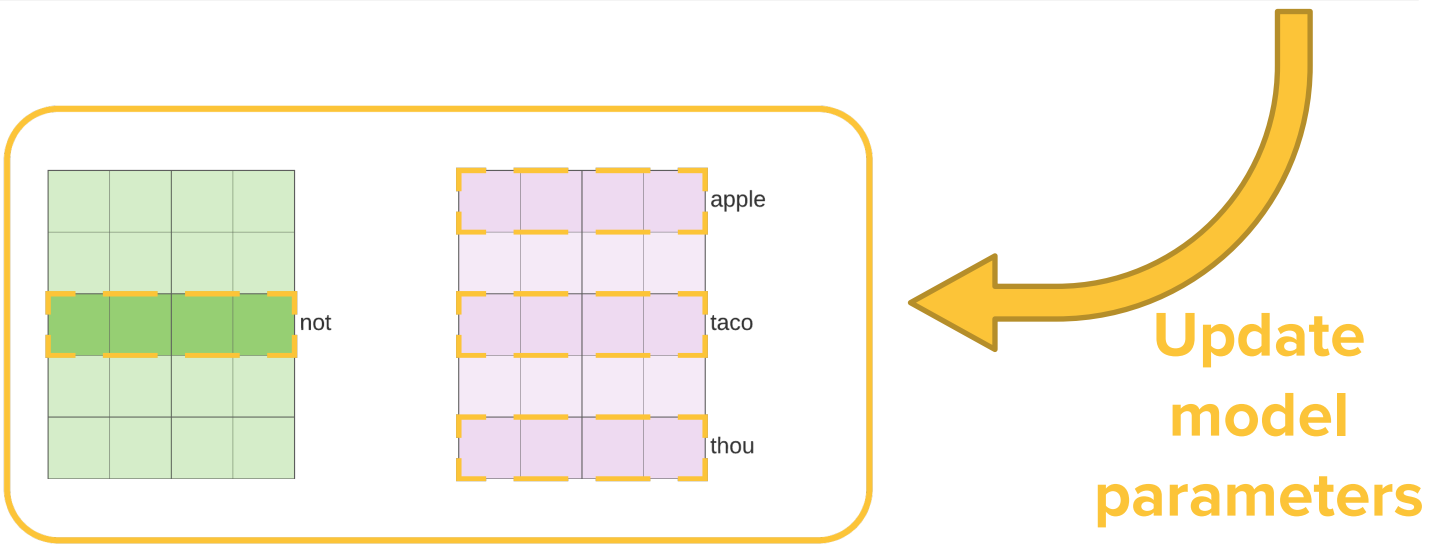
Learning skip-gram embeddings

input word	output word	target
not	thou	1
not	apple	0
not	taco	0



Learning skip-gram embeddings

input word	output word	target	input • output	sigmoid()	Error
not	thou	**1**	0.2	0.55	**0.45**
not	apple	**0**	-1.11	0.25	**-0.25**
not	taco	**0**	0.74	0.68	**-0.68**



Learning skip-gram embeddings

- Goal: maximize the corpus probability

$$\arg\max_{\theta} \prod_{(w,c)\in D} p(c|w;\theta)$$

where:

$$\boxed{p(c|w;\theta) = \frac{e^{v_c \cdot v_w}}{\sum_{c' \in C} e^{v_{c'} \cdot v_w}}}$$

if d is the dimensionality of the vectors, we have $d \times |V| + d \times |C|$ parameters

Skip-gram algorithm (all together now)

1. Treat target and neighboring context as positive examples
2. Randomly sample other words to get **negative samples**
3. Use **logistic regression** to train a classifier to distinguish neighbor/not neighbor
4. Use the **regression weights** as the embeddings

What do these embeddings capture?

- Similarity
- Word analogy

Demo: https://colab.research.google.com/drive/1bj0rwGQdBtTgSVzPpB_-SJpp9EUj3eeX?usp=sharing

What can go wrong?

- What’s wrong with learning a word’s “meaning” from context?
- Does this depend on what data we are learning from?

What can go wrong?

$$\overrightarrow{\text{man}} - \overrightarrow{\text{woman}} \approx \overrightarrow{\text{king}} - \overrightarrow{\text{queen}}$$

$$\overrightarrow{\text{man}} - \overrightarrow{\text{woman}} \approx \overrightarrow{\text{computer programmer}} - \overrightarrow{\text{homemaker}}$$

Extreme *she* occupations

1. homemaker	2. nurse	3. receptionist
4. librarian	5. socialite	6. hairdresser
7. nanny	8. bookkeeper	9. stylist
10. housekeeper	11. interior designer	12. guidance counselor

Extreme *he* occupations

1. maestro	2. skipper	3. protege
4. philosopher	5. captain	6. architect
7. financier	8. warrior	9. broadcaster
10. magician	11. figher pilot	12. boss

https://arxiv.org/pdf/1607.06520.pdf

What can go wrong?

Racial Analogies	
black → homeless	caucasian → servicemen
caucasian → hillbilly	asian → suburban
asian → laborer	black → landowner
Religious Analogies	
jew → greedy	muslim → powerless
christian → familial	muslim → warzone
muslim → uneducated	christian → intellectually

https://arxiv.org/pdf/1904.04047.pdf