

# Data Science 1

STAT/CS 287

Jim Bagrow, UVM Dept of Math and Statistics

LECTURE 17

# Natural Language Processing Tasks & Computational Semantics

# Related but non-NLP tasks

Optical Character Recognition

Images of text → text

Speech Recognition

Text-to-speech (speech synthesis)



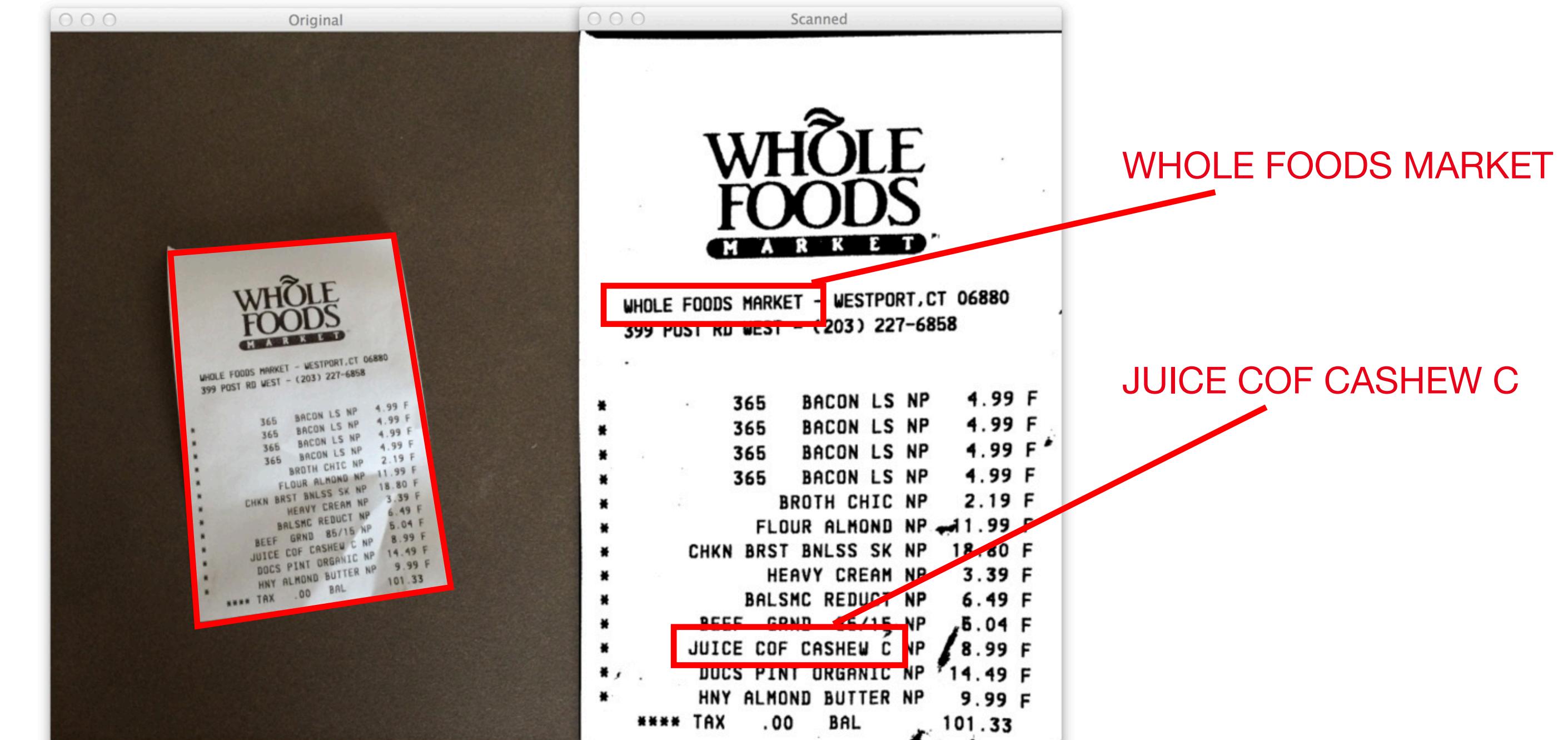
# Related but non-NLP tasks

Optical Character Recognition

Images of text → text

Speech Recognition

Text-to-speech (speech synthesis)



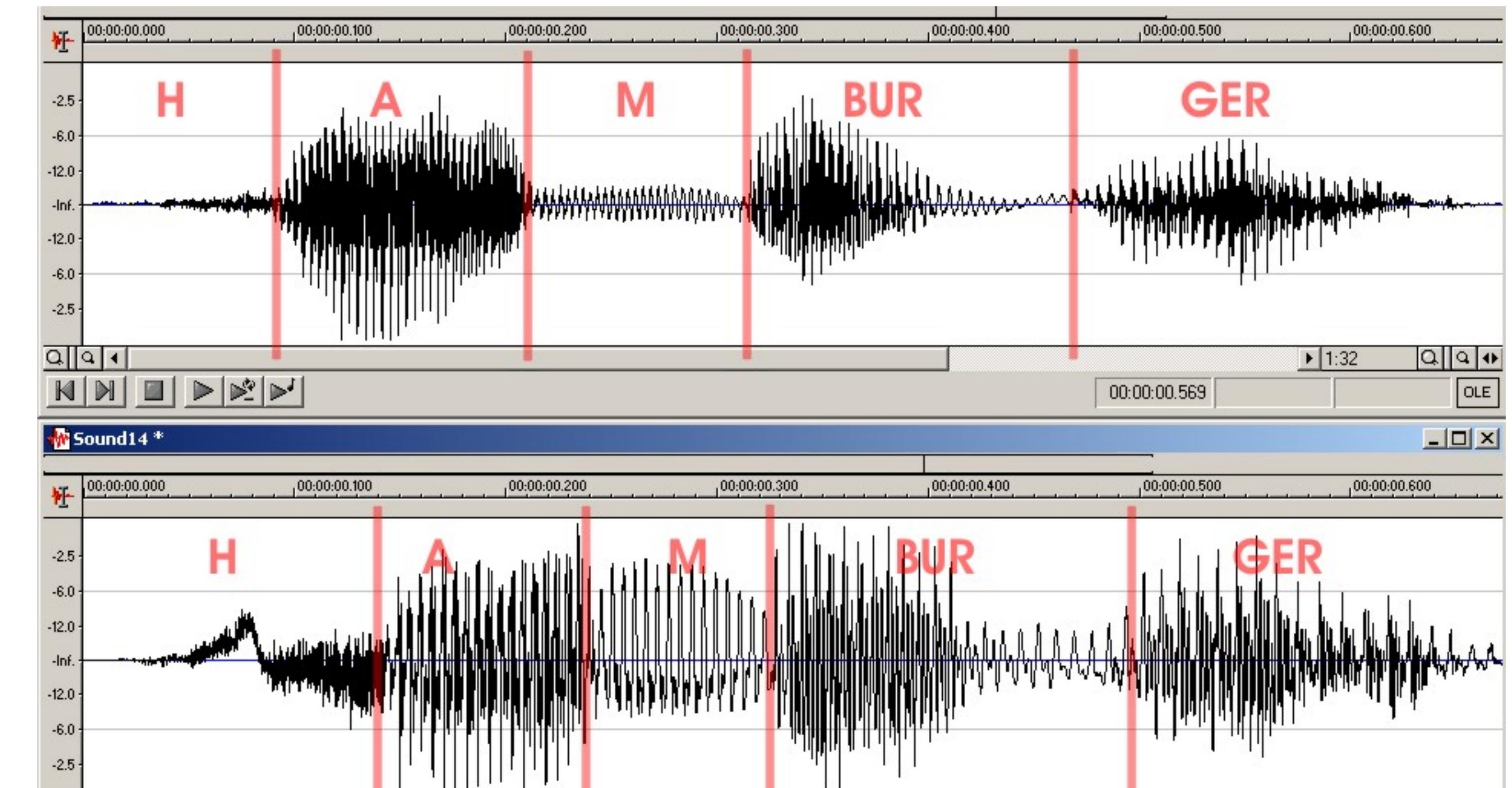
# Related but non-NLP tasks

Optical Character Recognition

Speech Recognition

Audio of text → text

Text-to-speech (speech synthesis)



<https://xsv.mm.cs.sunysb.edu/hci/speech/speech.html>

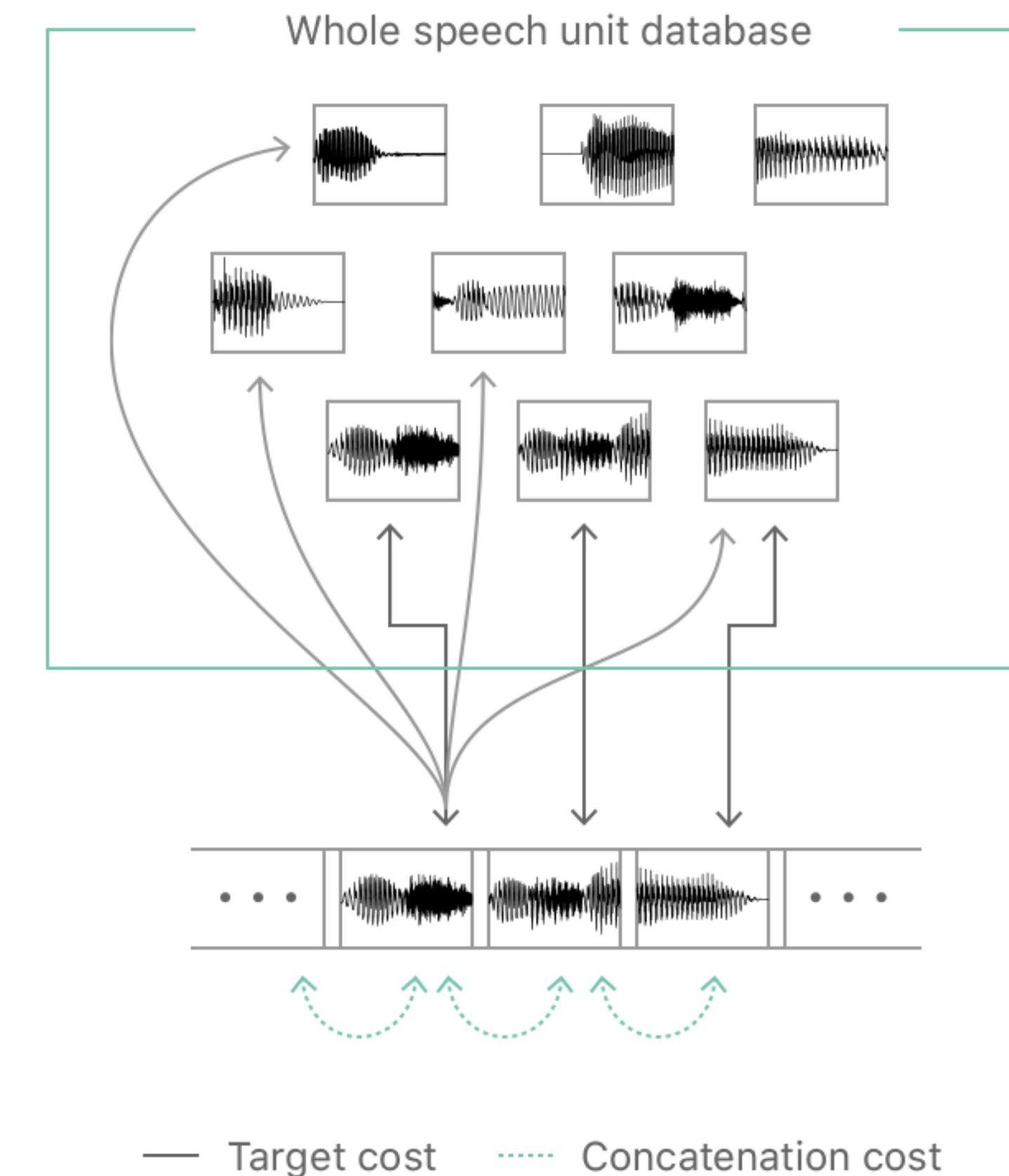
# Related but non-NLP tasks

Optical Character Recognition

Speech Recognition

Text-to-speech (speech synthesis)

text → audio



# Some NLP Tasks

Part-of-Speech (POS) tagging

```
In [1]: text = nltk.word_tokenize("My  
God it's full of stars!")
```

Named Entity Recognition

```
In [2]: text  
Out[2]: ['My', 'God', 'it', "'s",  
'full', 'of', 'stars', '!']
```

Sentence parsing

```
In [3]: nltk.pos_tag(text)  
Out[3]:  
[('My', 'PRP$'),  
 ('God', 'NNP'),  
 ('it', 'PRP'),  
 ("'s", 'VBZ'),  
 ('full', 'JJ'),  
 ('of', 'IN'),  
 ('stars', 'NNS'),  
 ('!', '.')]
```

Question answering

Summarization

Semantic similarity

# Some NLP Tasks

Part-of-Speech (POS) tagging

Named Entity Recognition

Sentence parsing

Question answering

Summarization

Semantic similarity

## Stanford Named Entity Tagger

Classifier:

Output Format:

Preserve Spacing:

Please enter your text here:

Just what do you think you're doing, **Dave?** **Dave**, I really think I'm entitled to an answer to that question. I know everything hasn't been quite right with me, but I can assure you now, very confidently, that it's going to be all right again. I feel much better now. I really do. Look, **Dave**, I can see you're really upset about this. I honestly think you ought to sit down calmly, take a stress pill and think things over. I know I've made some very poor decisions recently, but I can give you my complete assurance that my work will be back to normal. I've still got the greatest enthusiasm and

Just what do you think you're doing, **Dave?** **Dave**, I really think I'm entitled to an answer to that question. I know everything hasn't been quite right with me, but I can assure you now, very confidently, that it's going to be all right again. I feel much better now. I really do. Look, **Dave**, I can see you're really upset about this. I honestly think you ought to sit down calmly, take a stress pill and think things over. I know I've made some very poor decisions recently, but I can give you my complete assurance that my work will be back to normal. I've still got the greatest enthusiasm and confidence in the mission. And I want to help you. **Dave**, stop. Stop, will you? Stop, **Dave**. Will you stop, **Dave**? Stop, **Dave**. I'm afraid. I'm afraid, **Dave**. **Dave**, my mind is going. I can feel it. I can feel it. My mind is going. There is no question about it. I can feel it. I can feel it. I can feel it. I'm a...fraid. Good afternoon, gentlemen. I am a HAL 9000 computer. I became operational at the H.A.L. plant in **Urbana**, **Illinois** on the 12th of January 1992. My instructor was Mr. **Langley**, and he taught me to sing a song. If you'd like to hear it, I could sing it for you.

Potential tags:

**ORGANIZATION**  
**LOCATION**  
**PERSON**  
**MISC**

# Some NLP Tasks

Part-of-Speech (POS) tagging

Named Entity Recognition

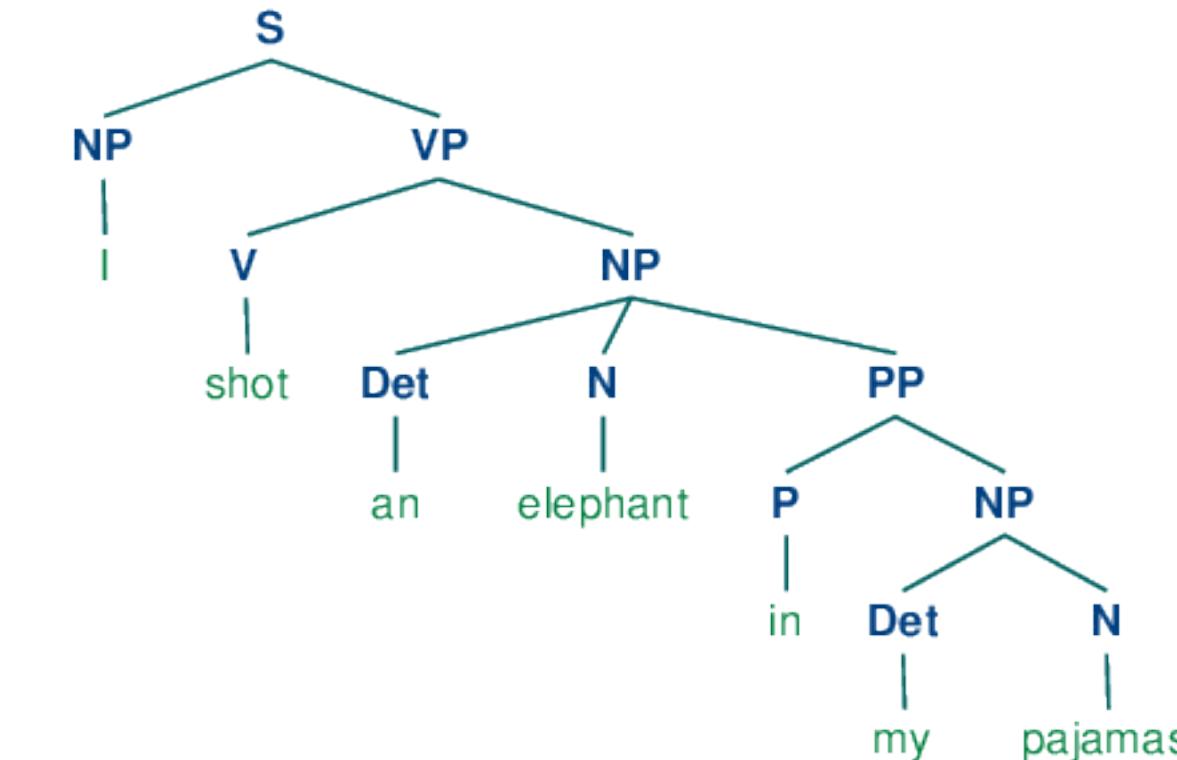
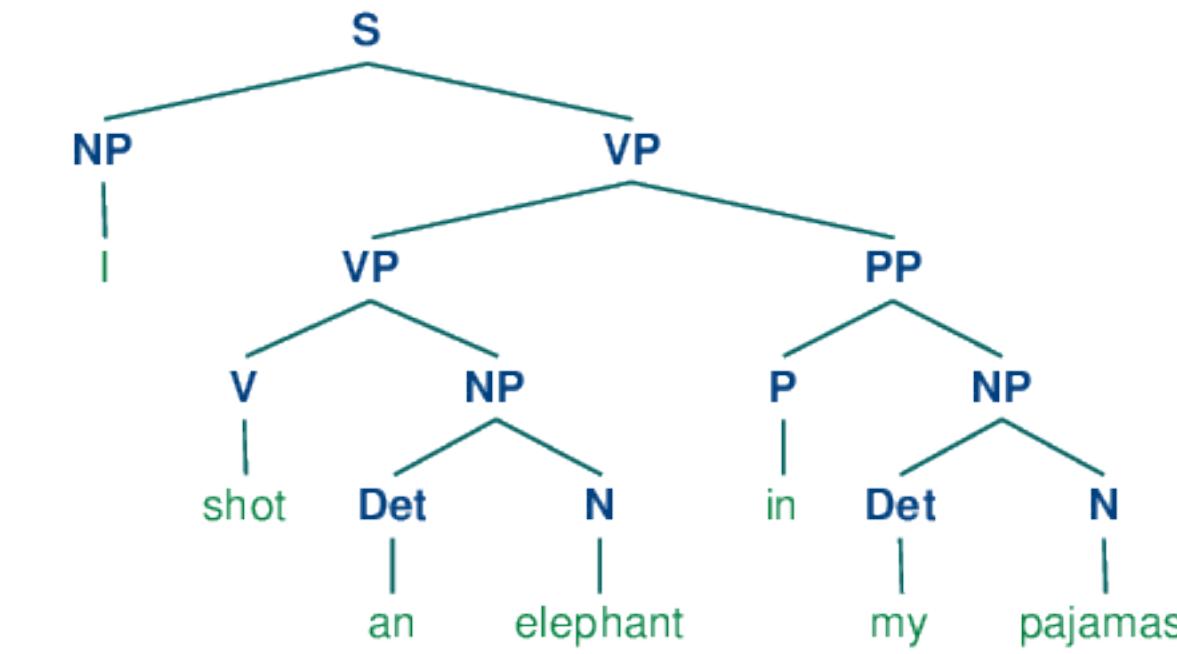
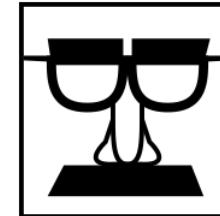
Sentence parsing

Question answering

Summarization

Semantic similarity

"I shot an elephant in my pajamas"



# Some NLP Tasks

Part-of-Speech (POS) tagging

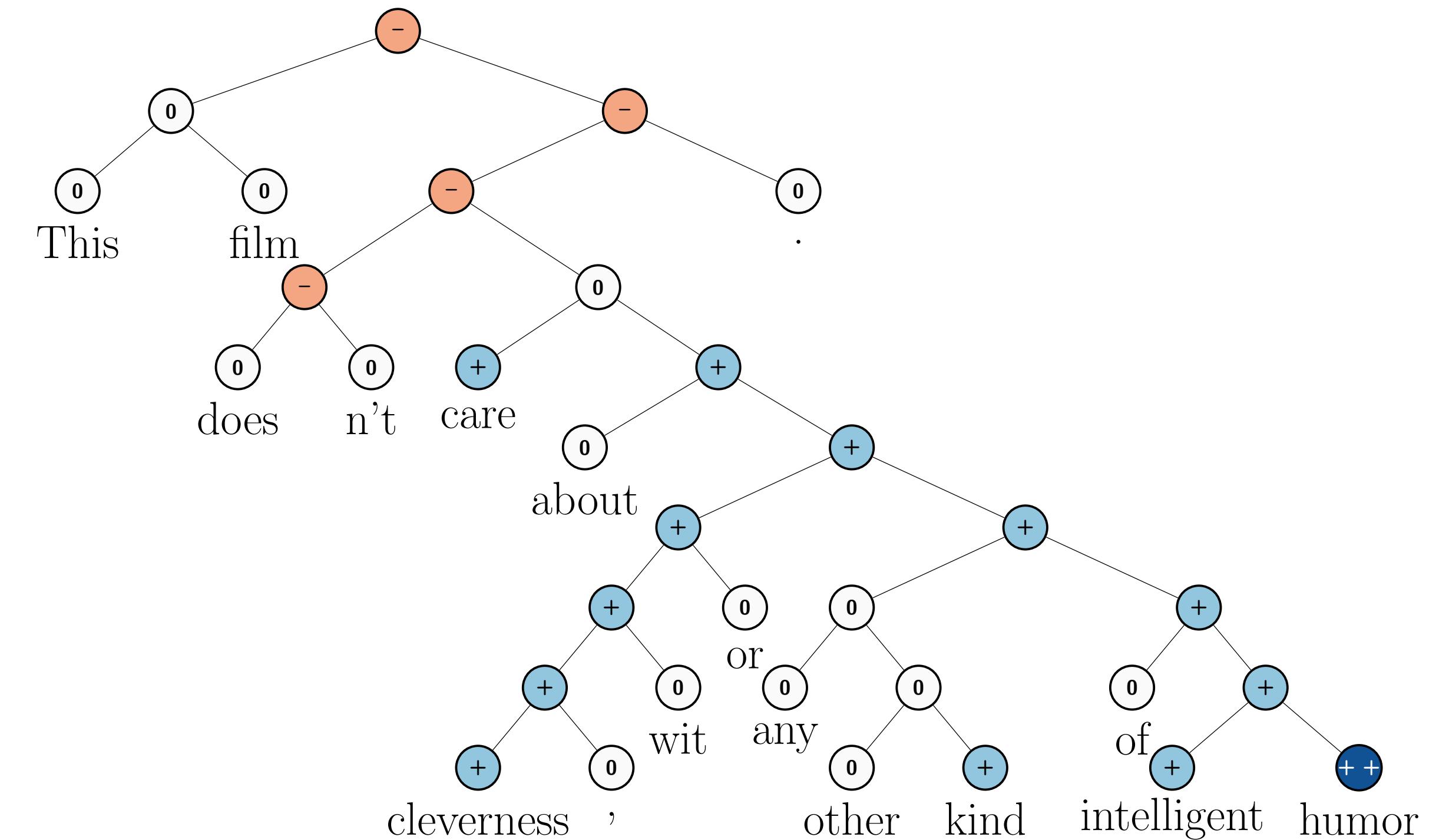
Named Entity Recognition

Sentence parsing    + **Sentiment Analysis**

Question answering

Summarization

Semantic similarity



Socher *et al.* (2013)

# Some NLP Tasks

Part-of-Speech (POS) tagging

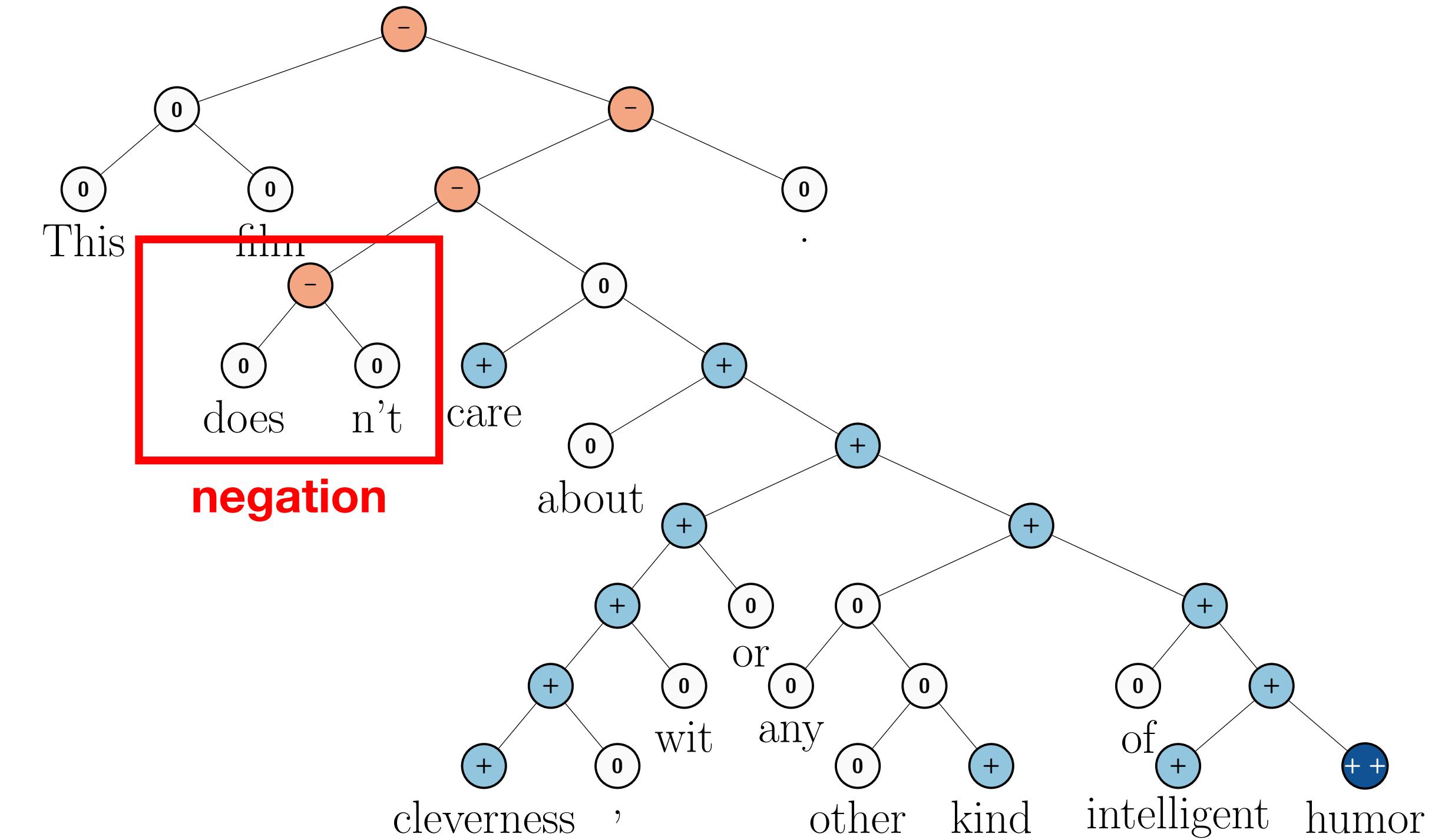
Named Entity Recognition

Sentence parsing      **+ Sentiment Analysis**

Question answering

Summarization

Semantic similarity



Socher *et al.* (2013)

# Some NLP Tasks

Part-of-Speech (POS) tagging

Named Entity Recognition

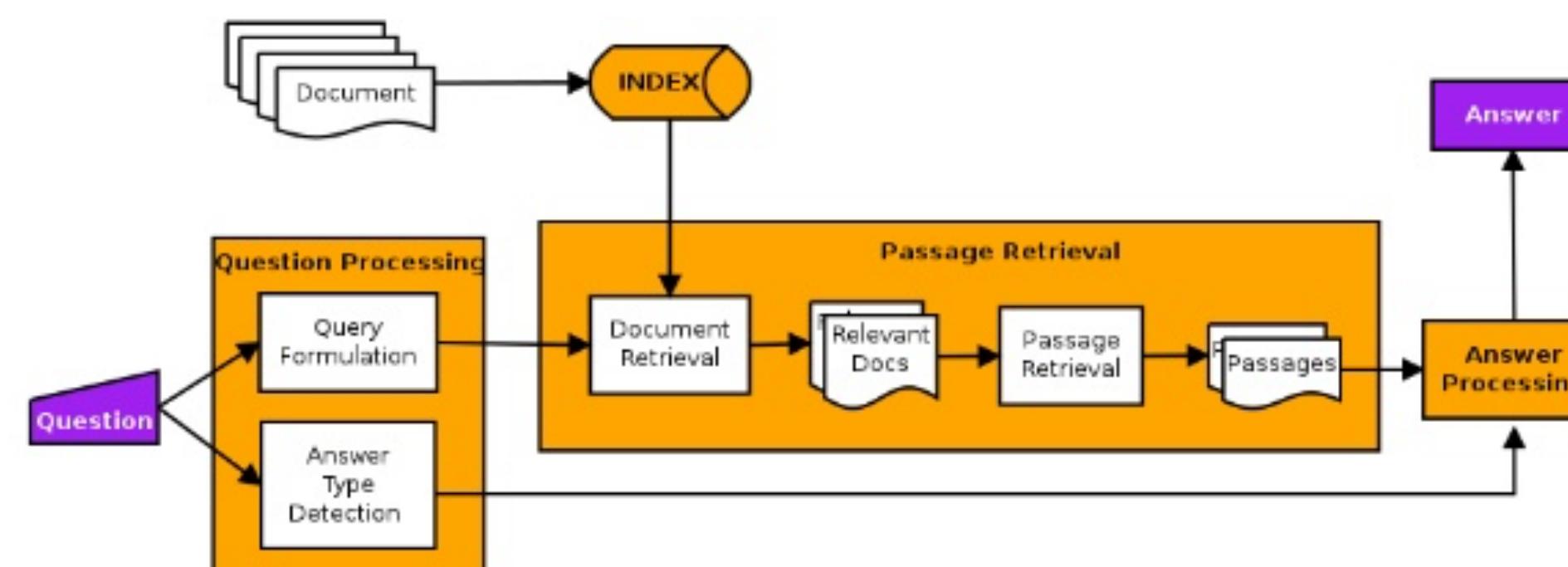
Sentence parsing

Question answering

Summarization

Semantic similarity

**Q: What is the capital  
of Australia?**  
**A: Canberra**



**QA as an information  
retrieval (IR) task**

Jurafsky & Manning

# Some NLP Tasks

Part-of-Speech (POS) tagging

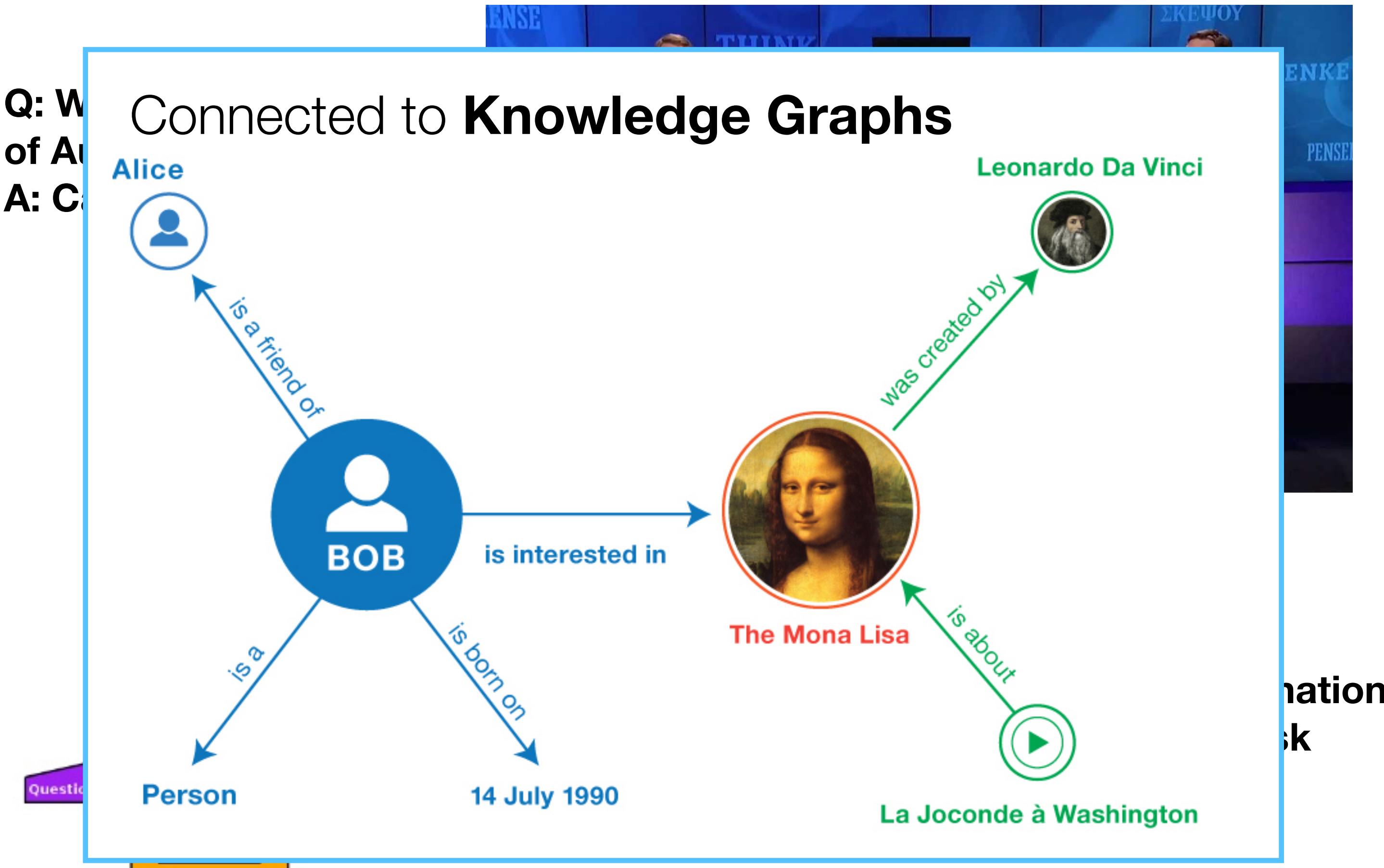
Named Entity Recognition

Sentence parsing

Question answering

Summarization

Semantic similarity



Jurafsky & Manning

# Some NLP Tasks

Part-of-Speech (POS) tagging

Shorten long texts without losing meaning

Named Entity Recognition

Sentence parsing

Question answering

Summarization

Semantic similarity

# Some NLP Tasks

Part-of-Speech (POS) tagging

Named Entity Recognition

Sentence parsing

Question answering

Summarization

Semantic similarity

Shorten long texts without losing meaning



Source Text: **Peter** and Elizabeth took a taxi to attend the night party in the city.

While in the party, Elizabeth collapsed and was rushed to the hospital.

Summary: Peter

**Extractive Summarization** (courtesy [blog.floydhub.com](http://blog.floydhub.com))

→related task: key phrase extraction

# Some NLP Tasks

Part-of-Speech (POS) tagging

Named Entity Recognition

Sentence parsing

Question answering

Summarization

Semantic similarity

Shorten long texts without losing meaning

**Source Text:** Peter and Elizabeth took a taxi to attend the night party in the city.

While in the party, Elizabeth collapsed and was rushed to the hospital.

**Summary:** Elizabeth was hospitalized after attending a party with Peter.



Abstractive Summarization

# Some NLP Tasks

Part-of-Speech (POS) tagging

Named Entity Recognition

Sentence parsing

Question answering

Summarization

Semantic similarity

## Summarization carries risks

Readers respond

### Correspondence

#### TL;DR: how well do machines summarize our work?

The SciTLDR software tool ([scitldr.apps.allenai.org](https://scitldr.apps.allenai.org)) uses machine learning to summarize scientific texts (see *Nature* <https://doi.org/ghmnjj>; 2020). Although it is impressive how far natural language processing has come, there is a risk it could distort scientific discourse by stripping away important context and over-amplifying results.

SciTLDR tends to extract one or two key statements from the original text and edits them into a cohesive sentence, sometimes removing parenthetical phrases and using synonyms for common words or phrases. Such changes are mostly innocuous, but they could omit qualifiers that the authors deem relevant. When the software replaces “we investigated” with “we identified”, for instance, it changes the meaning by seeming to present results rather than simply setting a research context.

And what happens when these tools are applied to, for example, anti-vaccination research or papers denying climate change? When I submitted abstracts from retracted works to the SciTLDR online demo, the summary statements of the results were often stronger than those in the original paper because they lacked context. They failed to acknowledge that the paper had been retracted, as a human writer would. Given the long-running threats posed by anti-science movements, caution is needed when developing and deploying tools such as SciTLDR.

**James P. Bagrow** University of Vermont, USA.  
[james.bagrow@uvm.edu](mailto:james.bagrow@uvm.edu)

# Task - Image Description

NLP + Computer Vision



zlikovec via Getty Images

ACCESSIBILITY

## Using AI to give people who are blind the “full picture”

**Dominic Mazzoni**  
Software Engineer, Chrome Accessibility

Published Oct 11, 2019

Everything that makes up the web—text, images, video and audio—can be easily discovered. Many people who are blind or have low vision rely on screen readers to make the content of web pages accessible through spoken feedback or braille.

For images and graphics, screen readers rely on descriptions created by developers and



Machine-generated description for this image: "[Appears to be: Person playing guitar on the sofa.](#)"



Machine-generated description for this image: "[Appears to be: Fruits and vegetables at the market.](#)"

# Some NLP Tasks

Part-of-Speech (POS) tagging

Named Entity Recognition

Capturing the *meaning* of words and phrases

Sentence parsing

**Synonymy**

**Bandit, Robber, Thief**

Question answering

**Polysemy**

**Bank**



Summarization

Semantic similarity

# Some NLP Tasks

Part-of-Speech (POS) tagging

Named Entity Recognition

Capturing the *meaning* of words and phrases

Sentence parsing

**Synonymy**

**Bandit, Robber, Thief**

Question answering

**Polysemy**

**Bank**



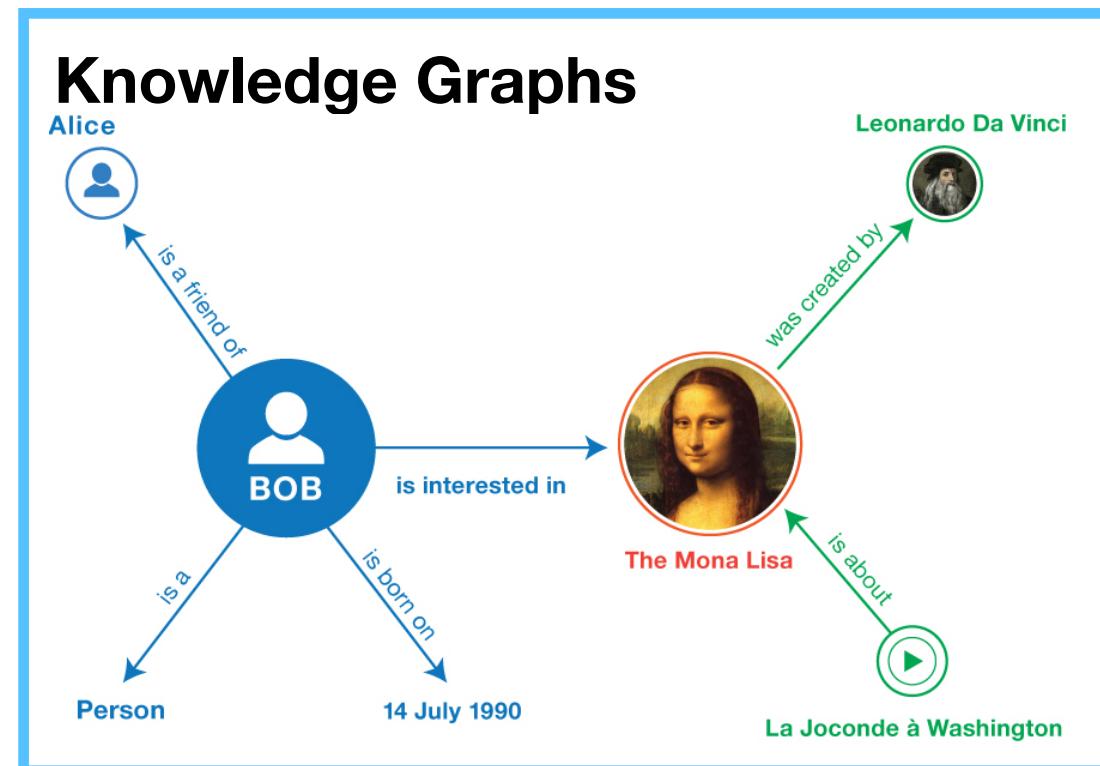
Summarization

Semantic similarity

Application →

# Application - Network Cleanup

Dealing with  
ambiguous data



Inferring the size of the causal universe: features and fusion of causal attribution networks

Daniel Berenberg<sup>1,2</sup> and James P. Bagrow<sup>3,2,\*</sup>

<sup>1</sup>Department of Computer Science, University of Vermont, Burlington, VT, United States

<sup>2</sup>Vermont Complex Systems Center, University of Vermont, Burlington, VT, United States

<sup>3</sup>Department of Mathematics & Statistics, University of Vermont, Burlington, VT, United States

\*Corresponding author. Email: [james.bagrow@uvm.edu](mailto:james.bagrow@uvm.edu), Homepage: [bagrow.com](http://bagrow.com)

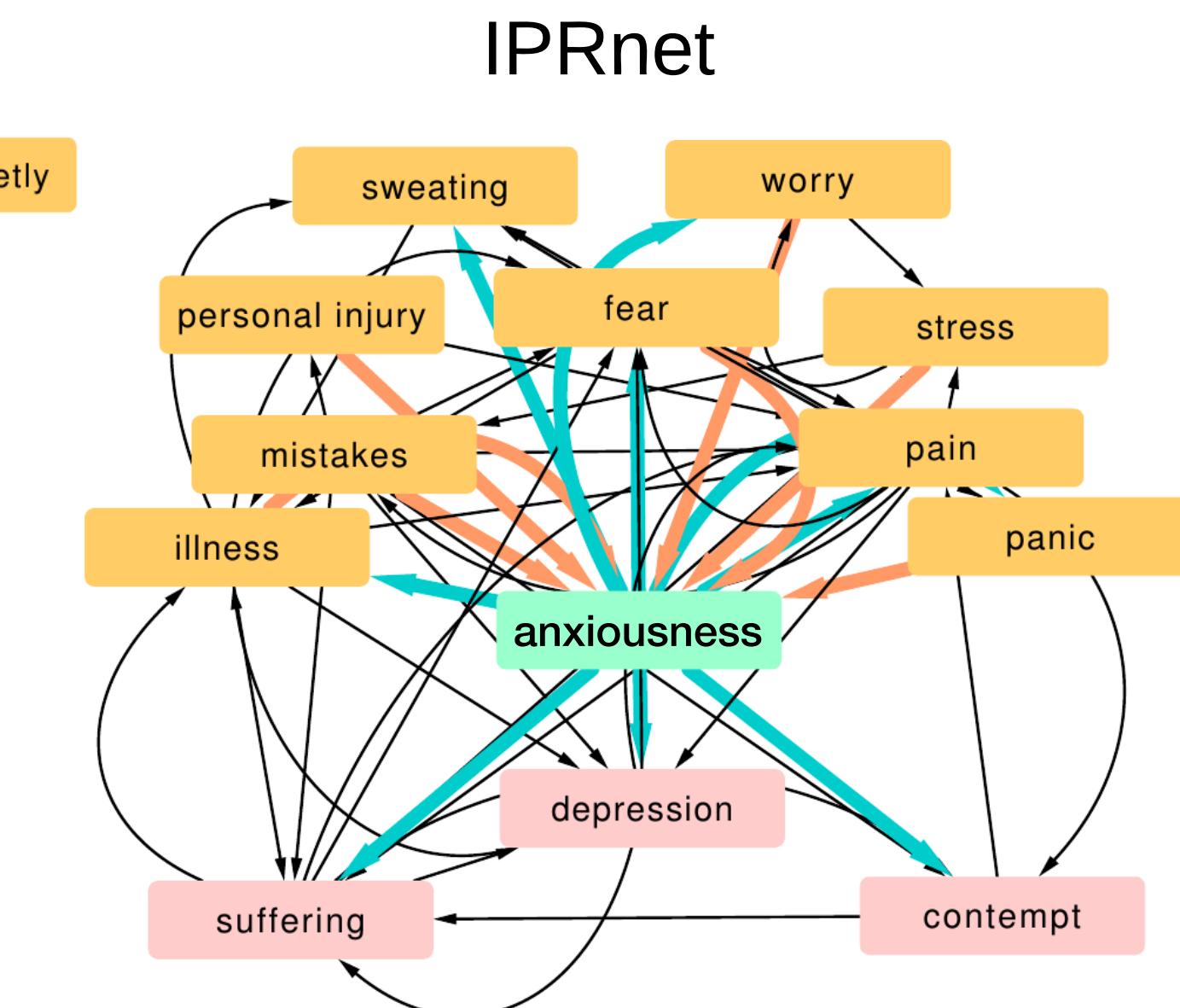
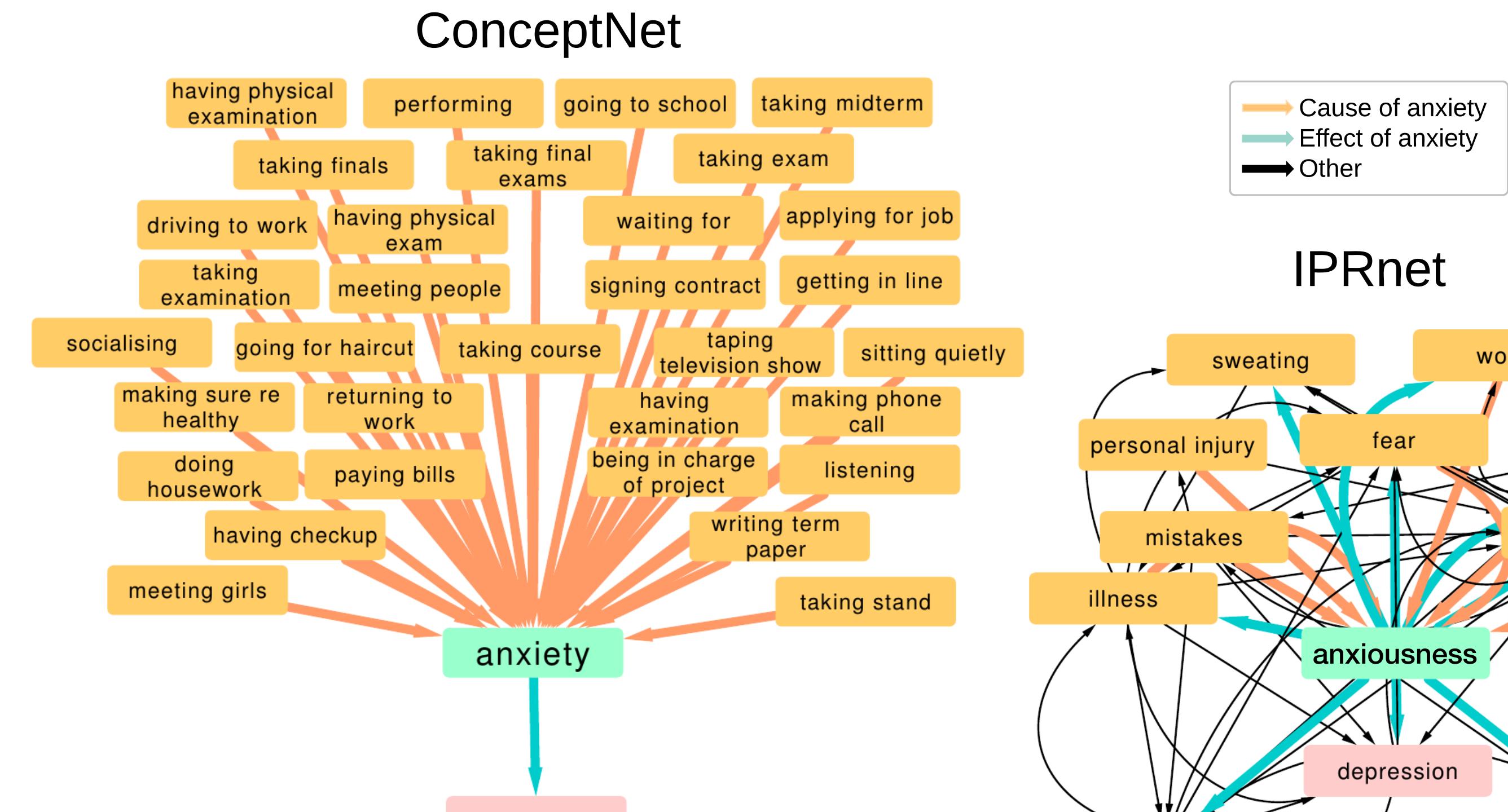
December 14, 2018

# Knowledge graphs

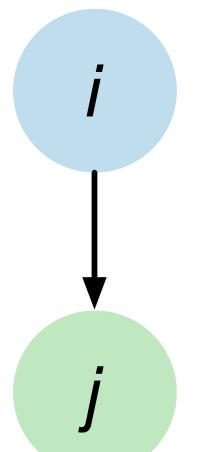
A diagram illustrating a directed edge between two nodes. A light blue circle at the top contains the letter 'i'. A dark green circle at the bottom contains the letter 'j'. A thick black arrow points vertically downwards from node 'i' to node 'j'.

$s_i = \text{"anxiety"}$

$s_j = \text{"sleep loss"}$

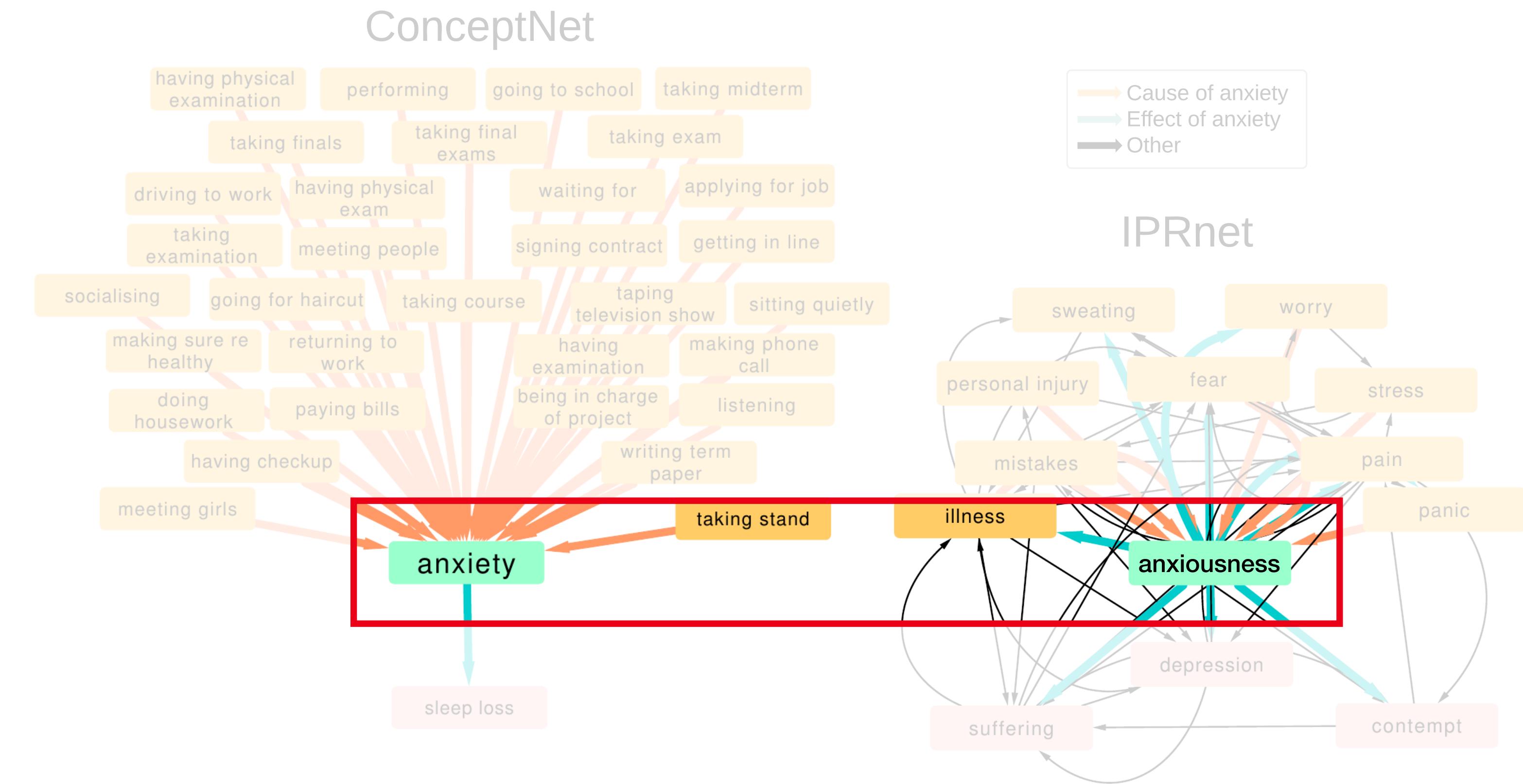


# Knowledge graphs



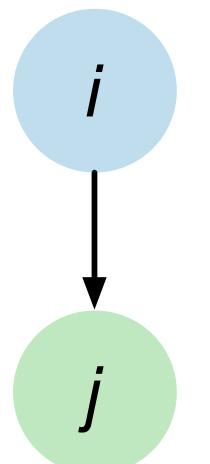
$s_i = \text{"anxiety"}$

$s_j = \text{"sleep loss"}$



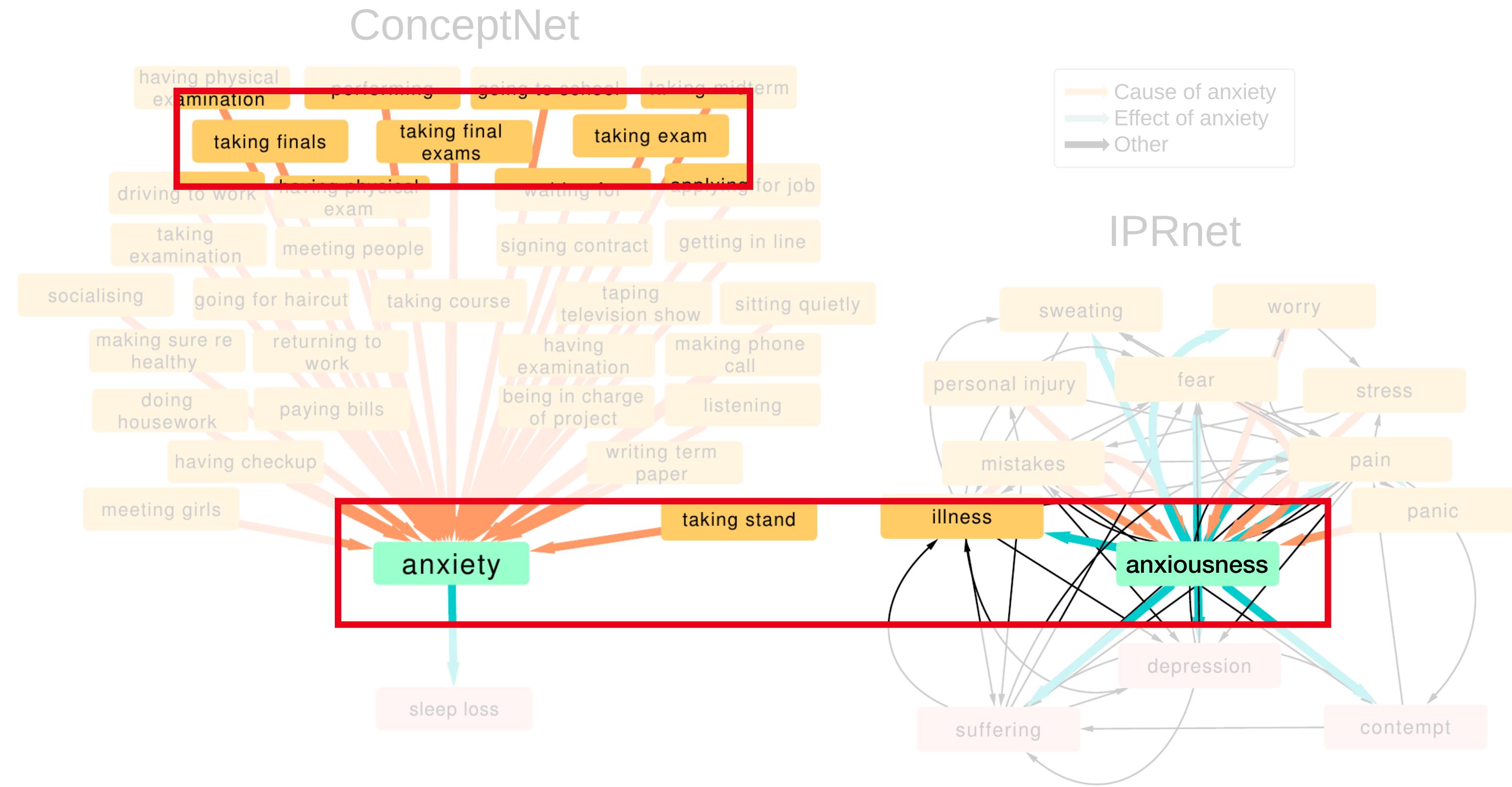
Nodes are identified *only* by these text...  
Could be ambiguous, even within one  
network...

# Knowledge graphs



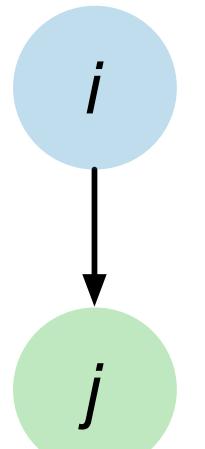
$s_i = \text{"anxiety"}$

$s_j = \text{"sleep loss"}$



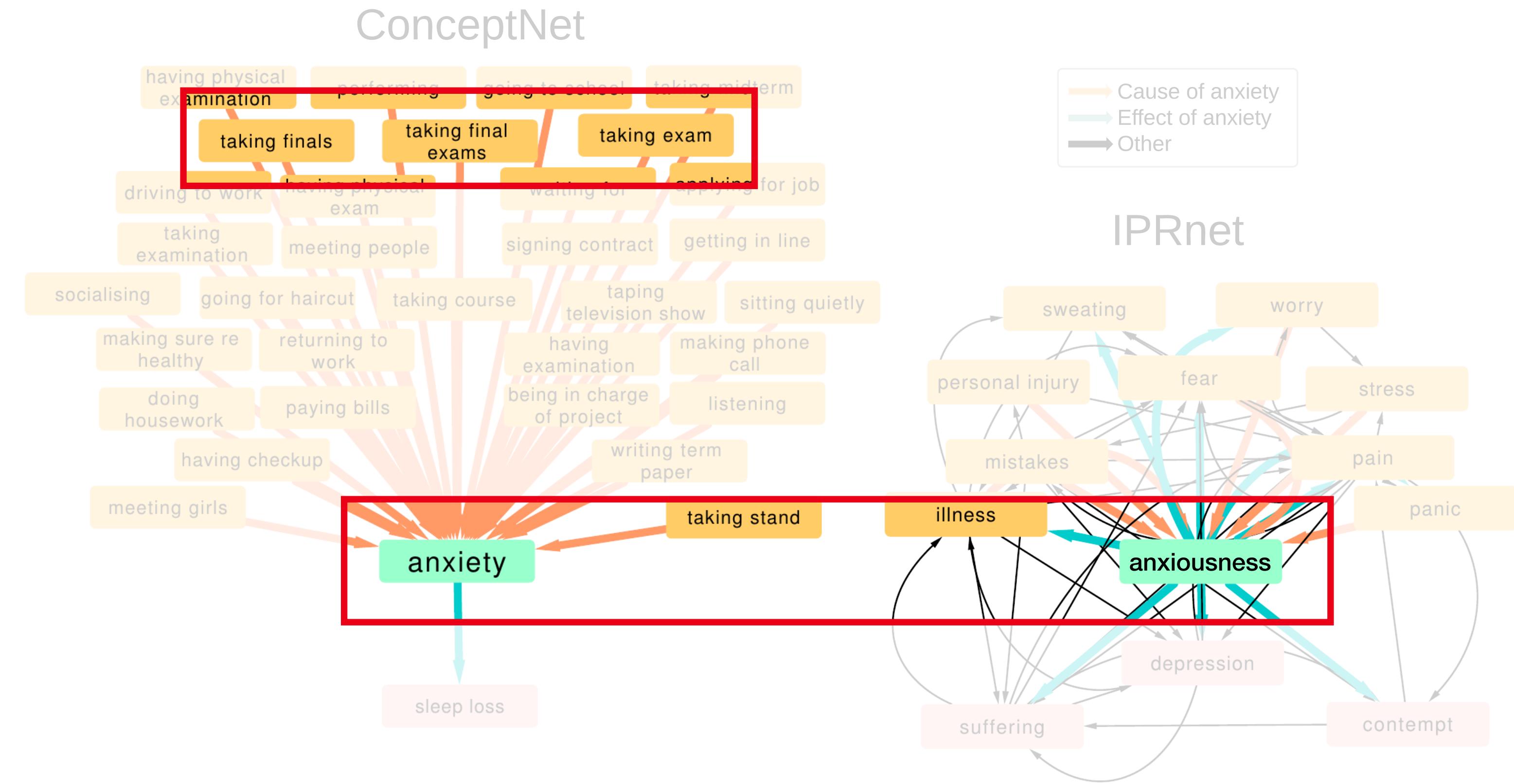
Nodes are identified *only* by these text...  
Could be ambiguous, even within one  
network...

# Knowledge graphs



$s_i = \text{"anxiety"}$

$s_j = \text{"sleep loss"}$

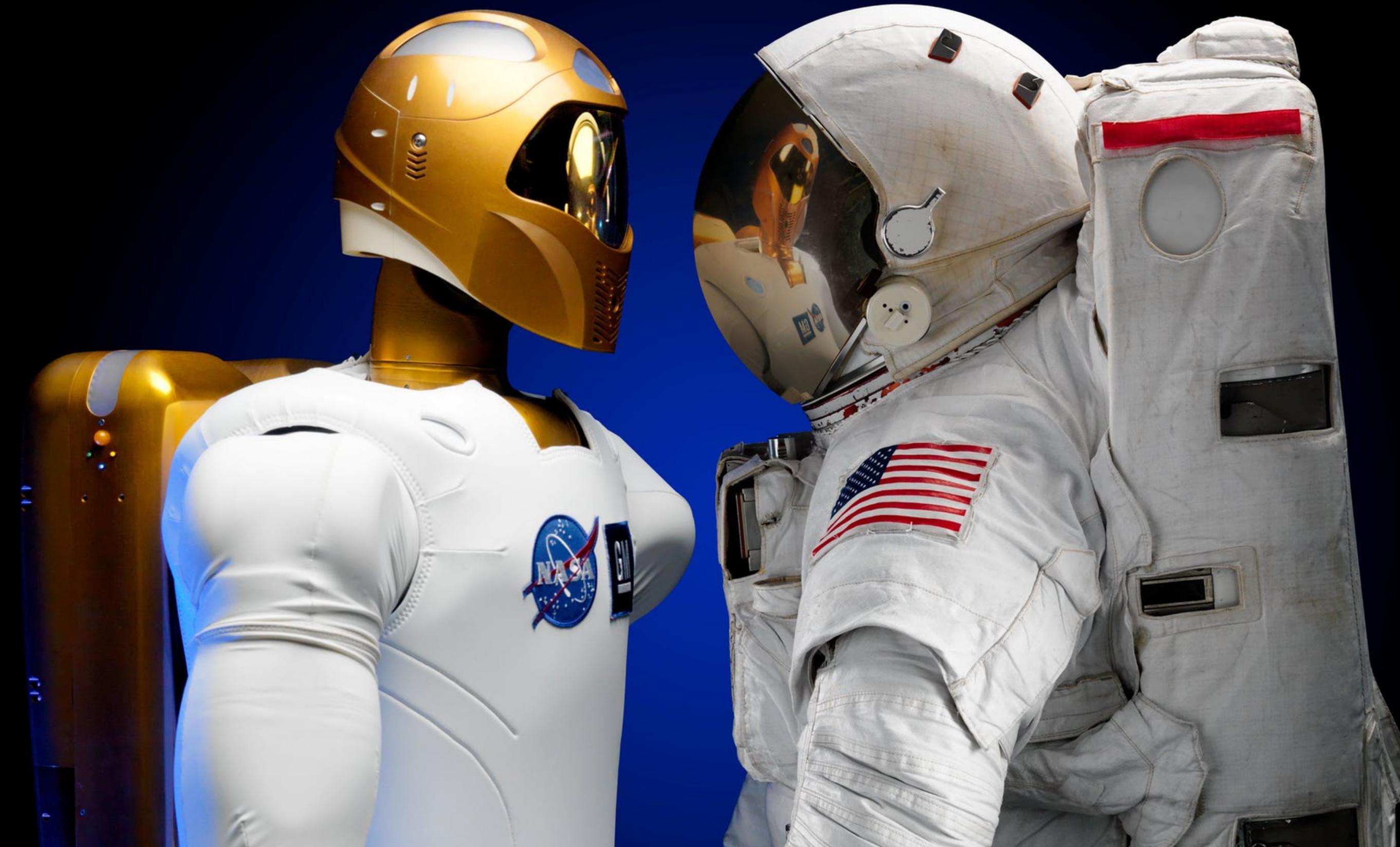


Nodes are identified *only* by these text...  
Could be ambiguous, even within one  
network...

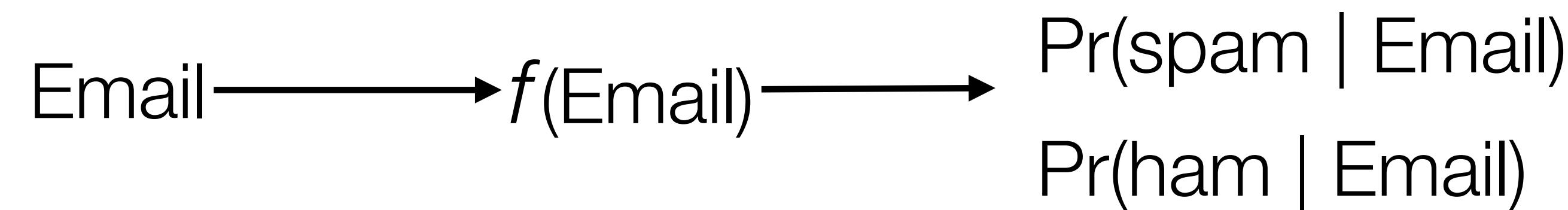
Can we combine these  
different networks together?

# Machine Learning

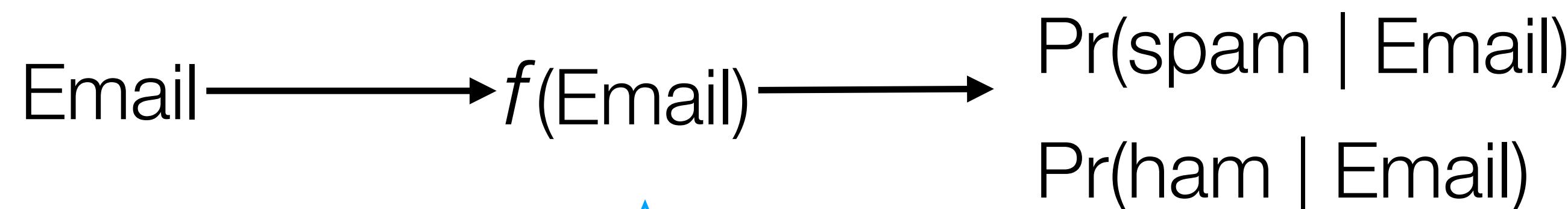
(How to measure semantic similarity of text?)



## Recall our **spam detector** – Text classifier



# Recall our **spam detector** – Text classifier



$$\Pr(c|\{w_i\}) \approx \Pr(c) \prod_{i=1}^n \Pr(w_i | c) \quad c \in \{\text{spam, ham}\}$$

using training data: labeled emails

$$\Pr(c) \approx \frac{\# \text{ training emails labeled } c}{\# \text{ training emails}}$$

$$\Pr(w_i|c) \approx \frac{\text{count}(w_i, c)}{\sum_j \text{count}(w_j, c)}$$

# Another view on the naive Bayes calculation

—# unique words (types)—

Convert word to vector:

[ 0 0 0 0 1 0 0 0 … 0 0 0 0 ] — "One-hot" vector

# Another view on the naive Bayes calculation

Convert word to vector:

Aggregate vectors for documents:

— # unique words (types) —

[ 0 0 0 1 0 0 0 … 0 0 0 0 ] — "One-hot" vector

— # unique words (types) —

$\begin{bmatrix} 0 & 0 & 0 & 7 & 3 & 0 & 0 & 1 & \cdots & 0 & 5 & 0 & 0 & 0 \\ \vdots & \ddots & & & & & & & & & & & & \vdots \end{bmatrix}$

$X$

— # documents —

$\begin{bmatrix} \text{spam} \\ \text{spam} \\ \vdots \\ \text{spam} \\ \text{ham} \\ \vdots \\ \text{ham} \end{bmatrix}$

$y$

— Labels vector

(tf-idf)

— Training data

# Another view on the naive Bayes calculation

Convert word to vector:

—# unique words (types)—

[ 0 0 0 1 0 0 0 … 0 0 0 0 ] — "One-hot" vector

Aggregate vectors for documents:

—# documents—

$\begin{bmatrix} 0 & 0 & 0 & 7 & 3 & 0 & 0 & 1 & \cdots & 0 & 5 & 0 & 0 & 0 \\ \vdots & \ddots & & & & & & & & & & & & \vdots \end{bmatrix}$

$\begin{bmatrix} \text{spam} \\ \text{spam} \\ \vdots \\ \text{spam} \\ \text{ham} \\ \vdots \\ \text{ham} \end{bmatrix}$

— Labels vector

(tf-idf)

Apply functions (+, ×, ÷) to aggregate:

$$\Pr(c|\{w_i\}) \approx \Pr(c) \prod_{i=1}^n \Pr(w_i | c)$$

$X$

$y$

— Training data

# Another view on the naive Bayes calculation

Convert word to vector:

Aggregate vectors for documents:

— # unique words (types) —

[ 0 0 0 1 0 0 0 … 0 0 0 0 ] — "One-hot" vector

— # unique words (types) —

$\begin{bmatrix} 0 & 0 & 0 & 7 & 3 & 0 & 0 & 1 & \cdots & 0 & 5 & 0 & 0 & 0 \\ \vdots & \ddots & & & & & & & & & & & & \vdots \end{bmatrix}$

$X$

— # documents —

$\begin{bmatrix} \text{spam} \\ \text{spam} \\ \vdots \\ \text{spam} \\ \text{ham} \\ \vdots \\ \text{ham} \end{bmatrix}$

$y$

— Labels vector  
— Training data

$y = f(X)$

# Another view on the naive Bayes calculation

Convert word to vector:

Aggregate vectors for documents:

— # unique words (types) —

[ 0 0 0 1 0 0 0 … 0 0 0 0 ] — "One-hot" vector

— # unique words (types) —

$\begin{bmatrix} 0 & 0 & 0 & 7 & 3 & 0 & 0 & 1 & \cdots & 0 & 5 & 0 & 0 & 0 \\ \vdots & \ddots & & & & & & & & & & & & \vdots \end{bmatrix}$

$X$

$\begin{bmatrix} \text{spam} \\ \text{spam} \\ \vdots \\ \text{spam} \\ \text{ham} \\ \vdots \\ \text{ham} \end{bmatrix}$

— Labels vector

$y$  — Training data

$y = f(X)$

A new, unlabeled document comes in:

built a machine  $\hat{f}$  to predict label given  
count vector



# Another view on the naive Bayes calculation

Convert word to vector:

Aggregate vectors for documents:

— # unique words (types) —

[ 0 0 0 1 0 0 0 … 0 0 0 0 ] — "One-hot" vector

— # unique words (types) —

$\begin{bmatrix} 0 & 0 & 0 & 7 & 3 & 0 & 0 & 1 & \cdots & 0 & 5 & 0 & 0 & 0 \\ \vdots & \ddots & & & & & & & & \vdots & & & & \end{bmatrix}$

— # documents —

$\begin{bmatrix} \text{spam} \\ \text{spam} \\ \vdots \\ \text{spam} \\ \text{ham} \\ \vdots \\ \text{ham} \end{bmatrix}$

— Labels vector

(tf-idf)

$X$

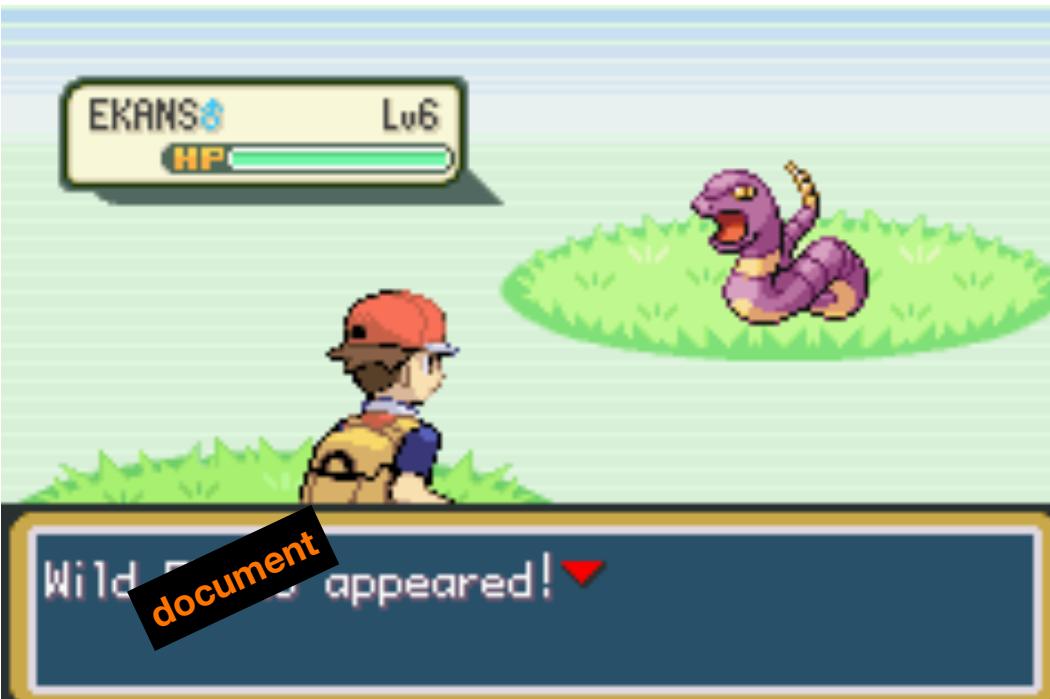
$y$

— Training data

$y = f(X)$

A new, unlabeled document comes in:

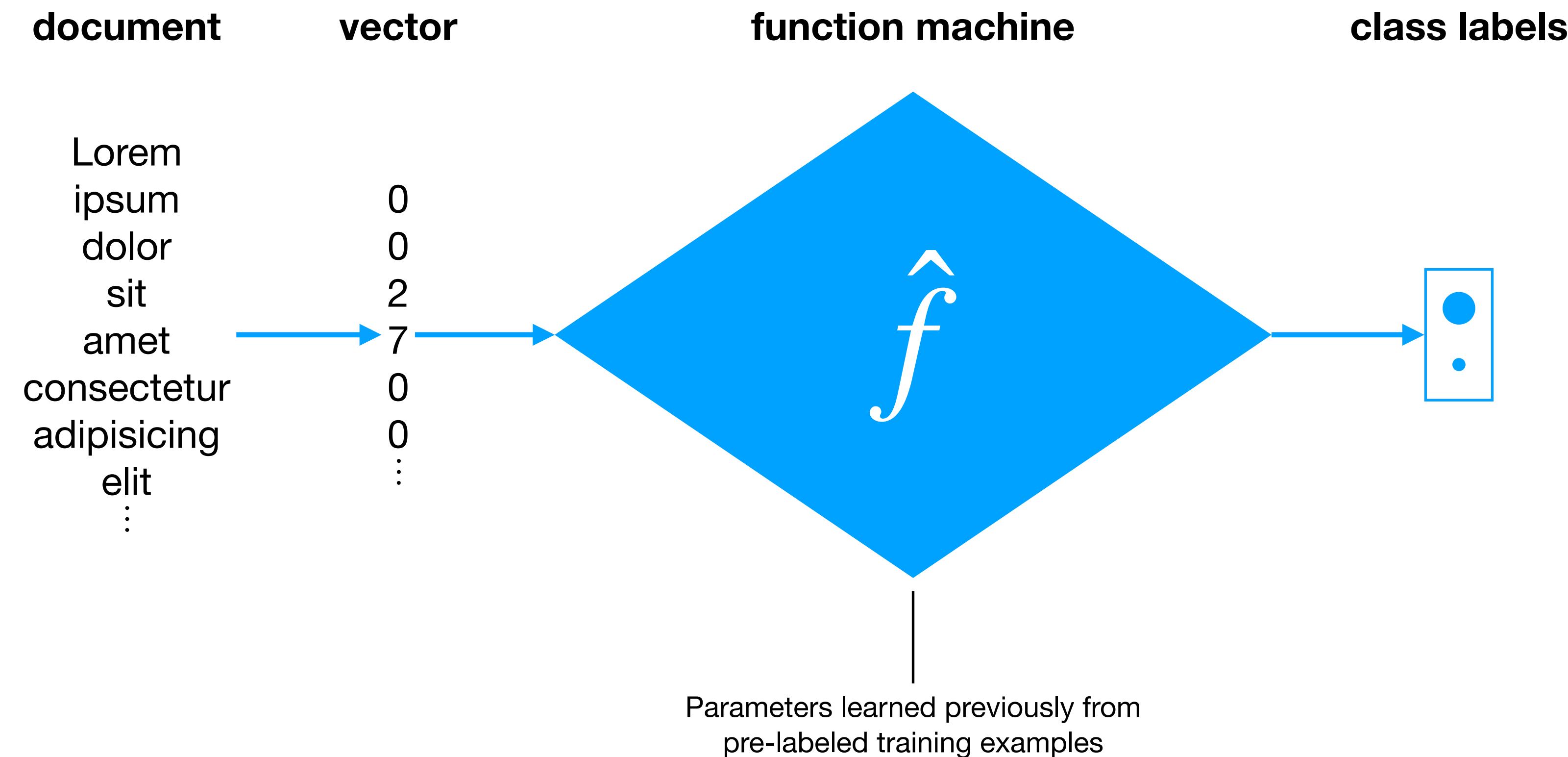
built a machine  $\hat{f}$  to predict label given  
count vector —



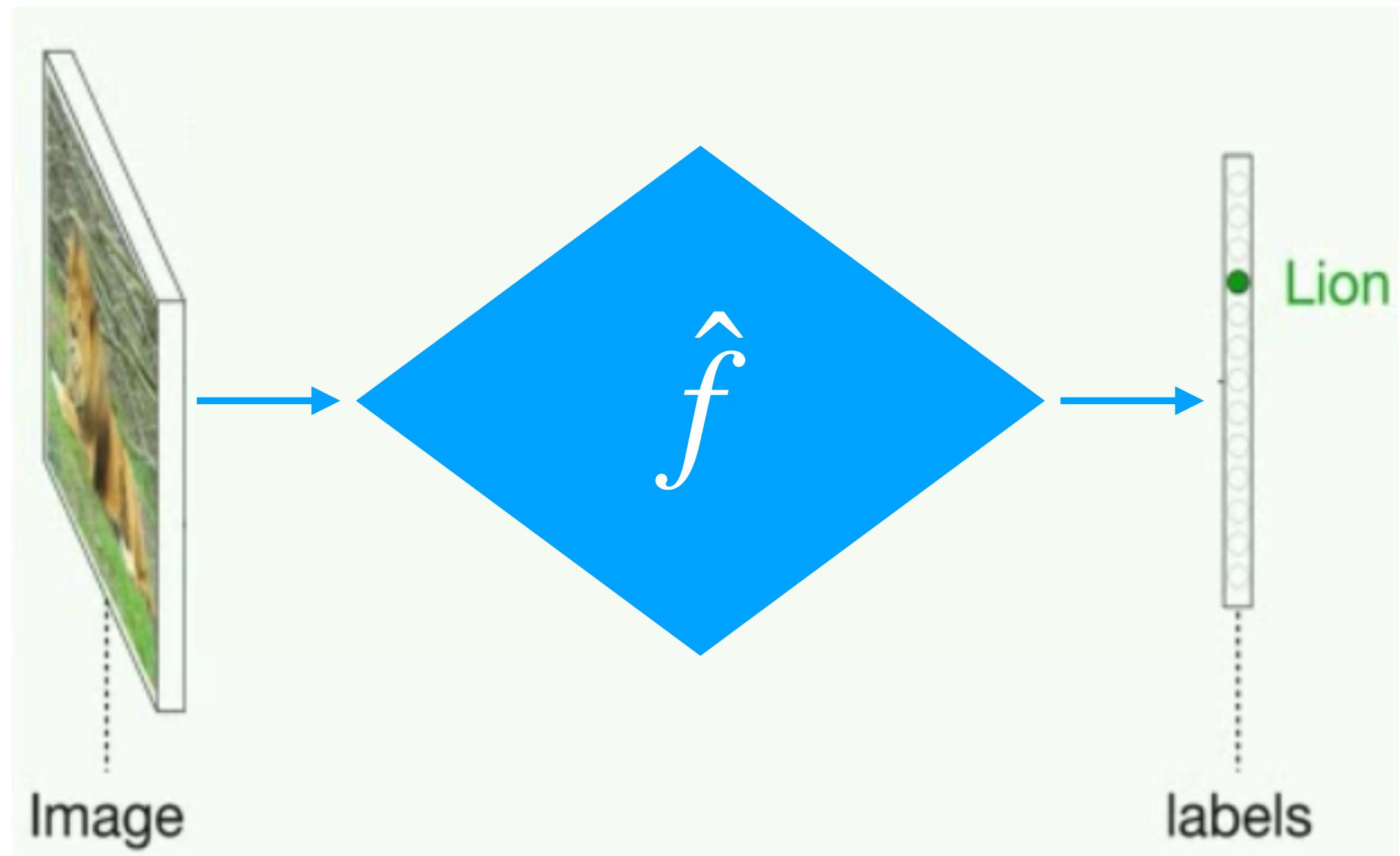
$$\begin{aligned}\hat{f}([0 & 0 & 0 & 2 & 6 & 3 & 0 & 5 & \cdots & 0 & 0 & 1 & 1 & 0 & 0]) = \\ \text{arg max } [\Pr(\text{spam}), \Pr(\text{ham})] = \\ \text{arg max } [0.6, 0.4] = \text{(for example)} \\ \text{spam}\end{aligned}$$

$y = \hat{f}(X)$

# Another view on the naive bayes calculation



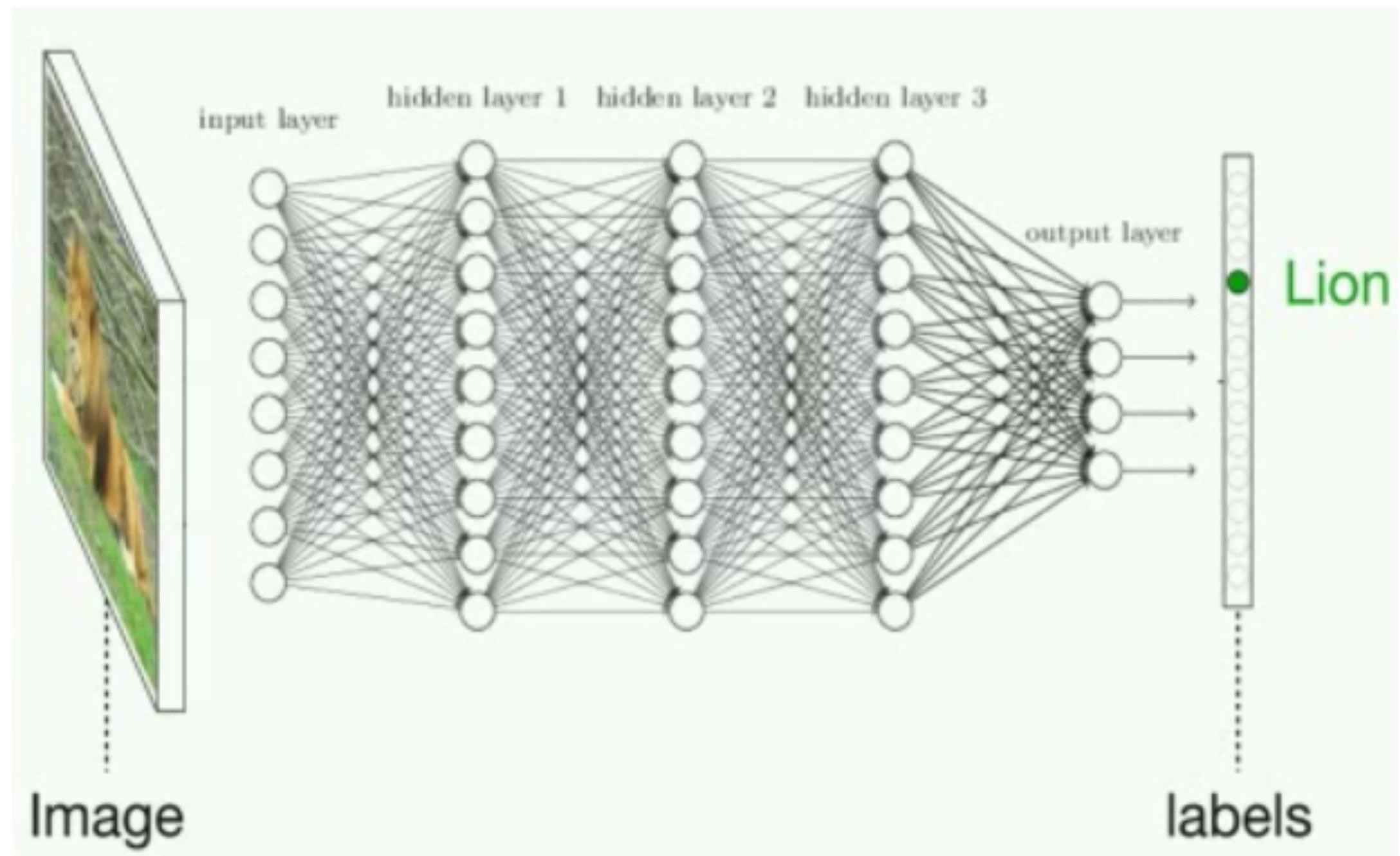
# General purpose



[Image: Jeff Clune](#)

Example: image classification

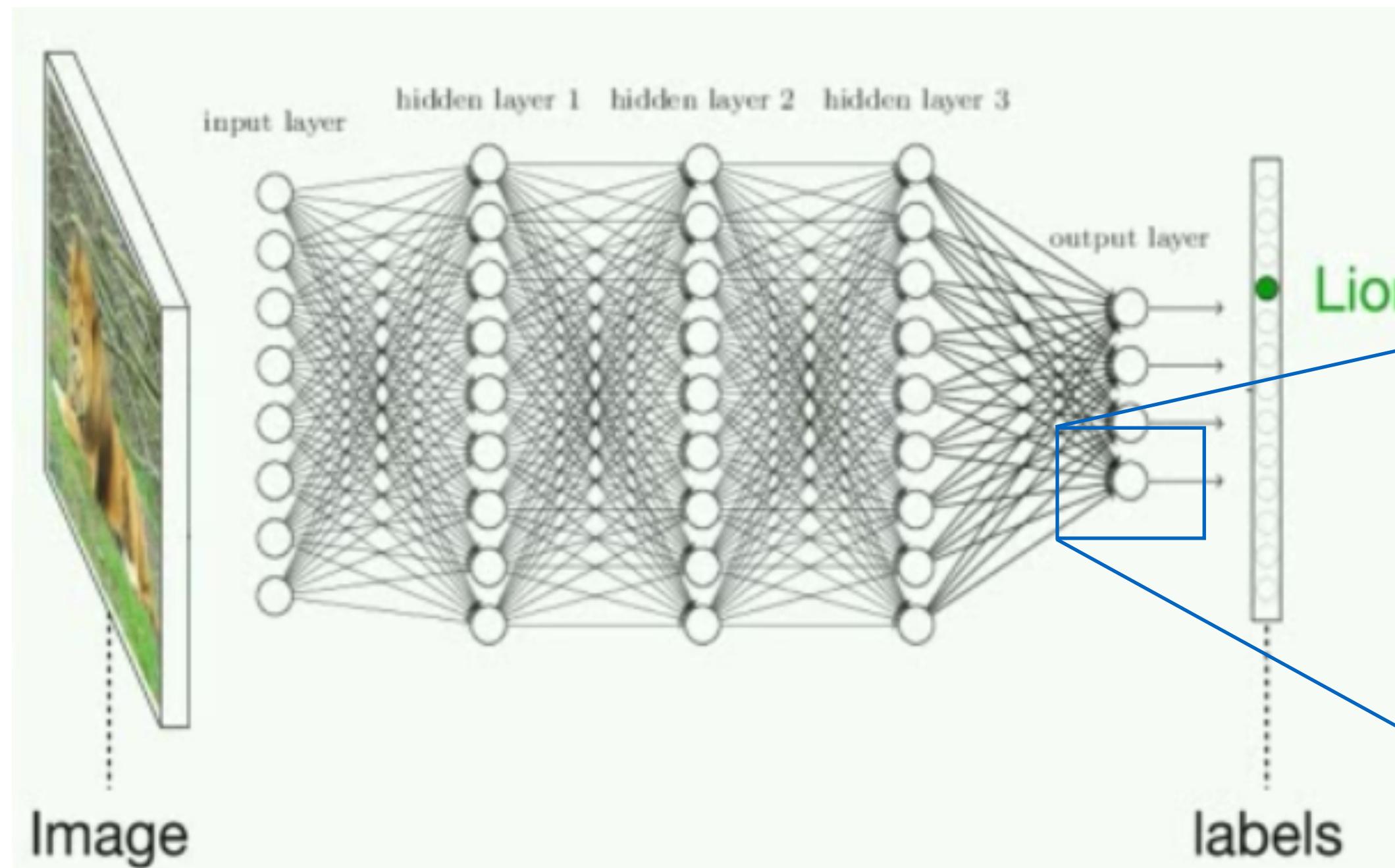
# General purpose



[Image: Jeff Clune](#)

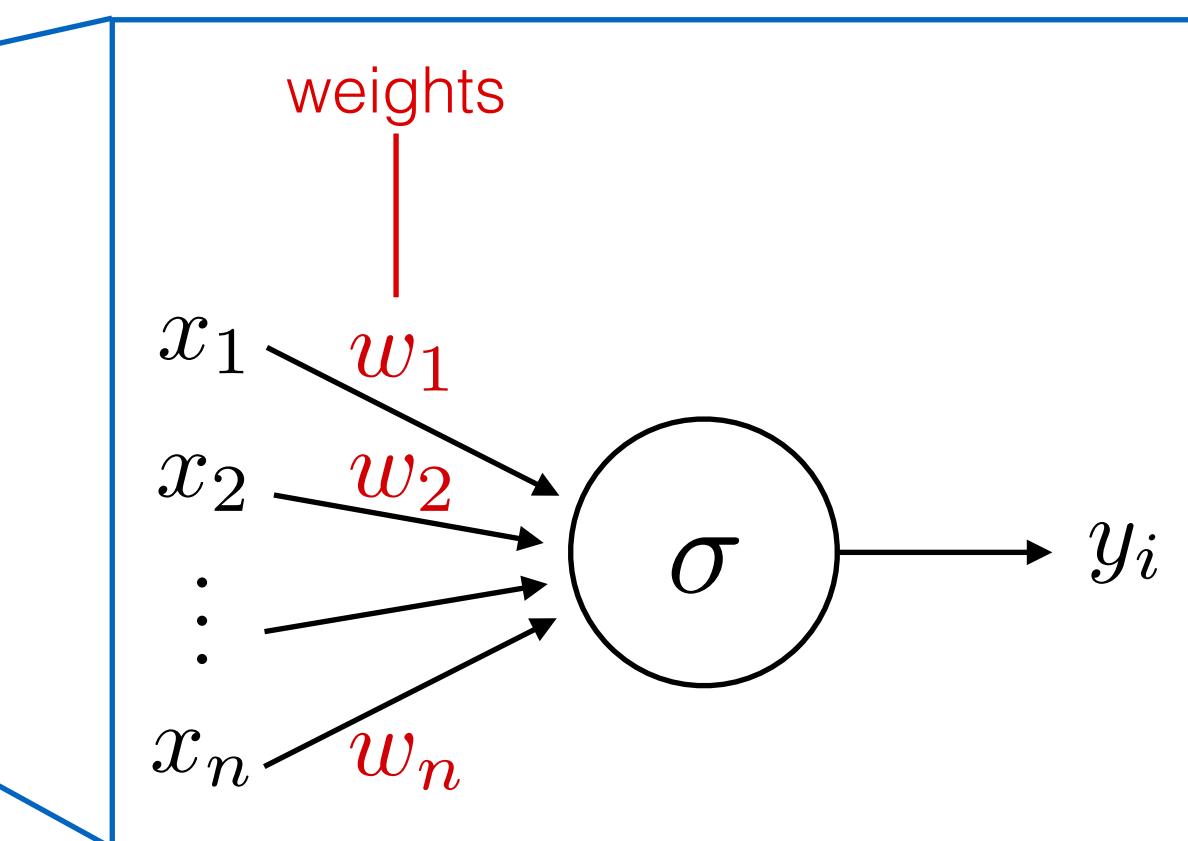
Example: image classification

# General purpose



[Image: Jeff Clune](#)

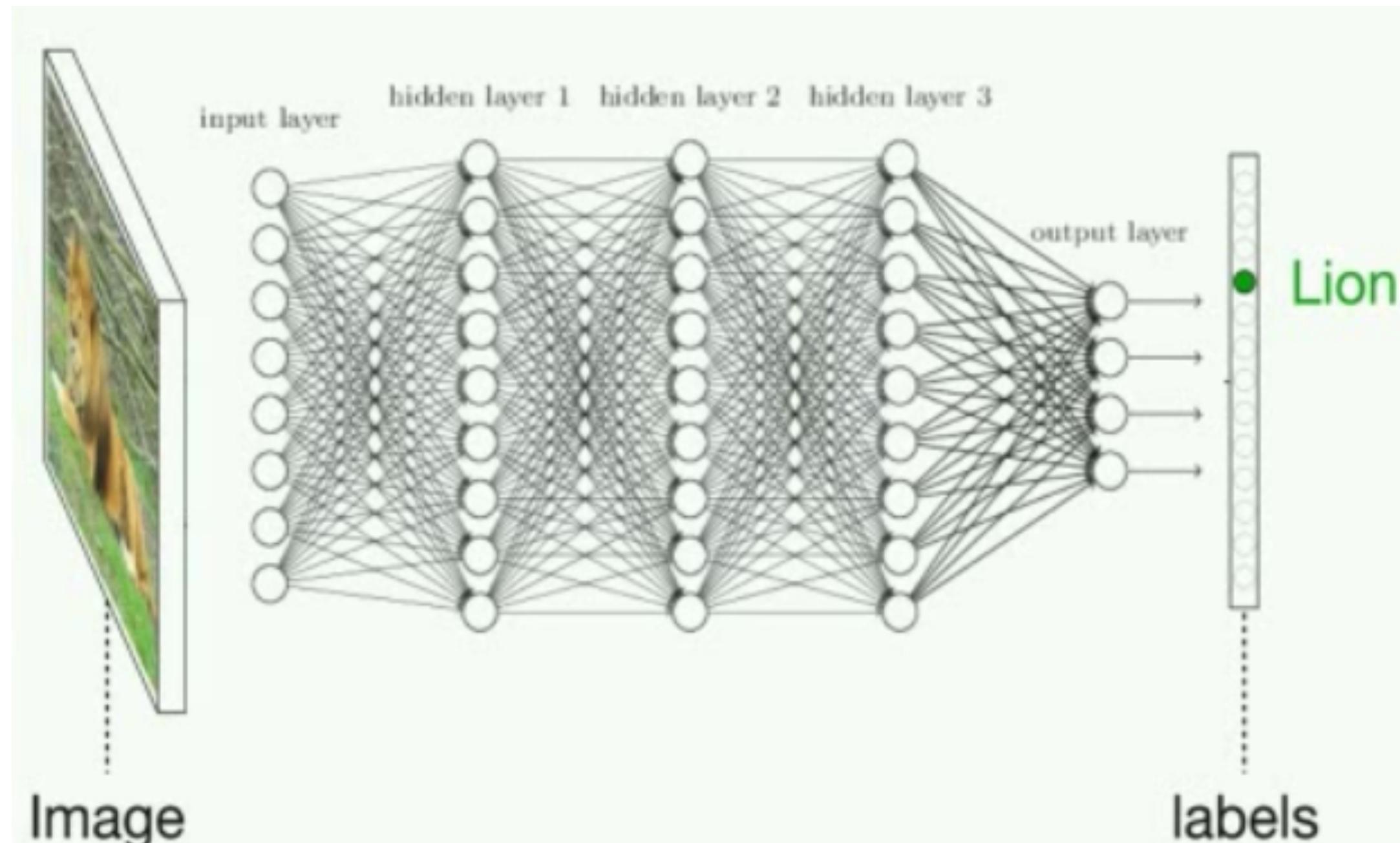
Example: image classification



Parameters (weights) learned from training data:  
labeled images

..., (, 'lion'), ...

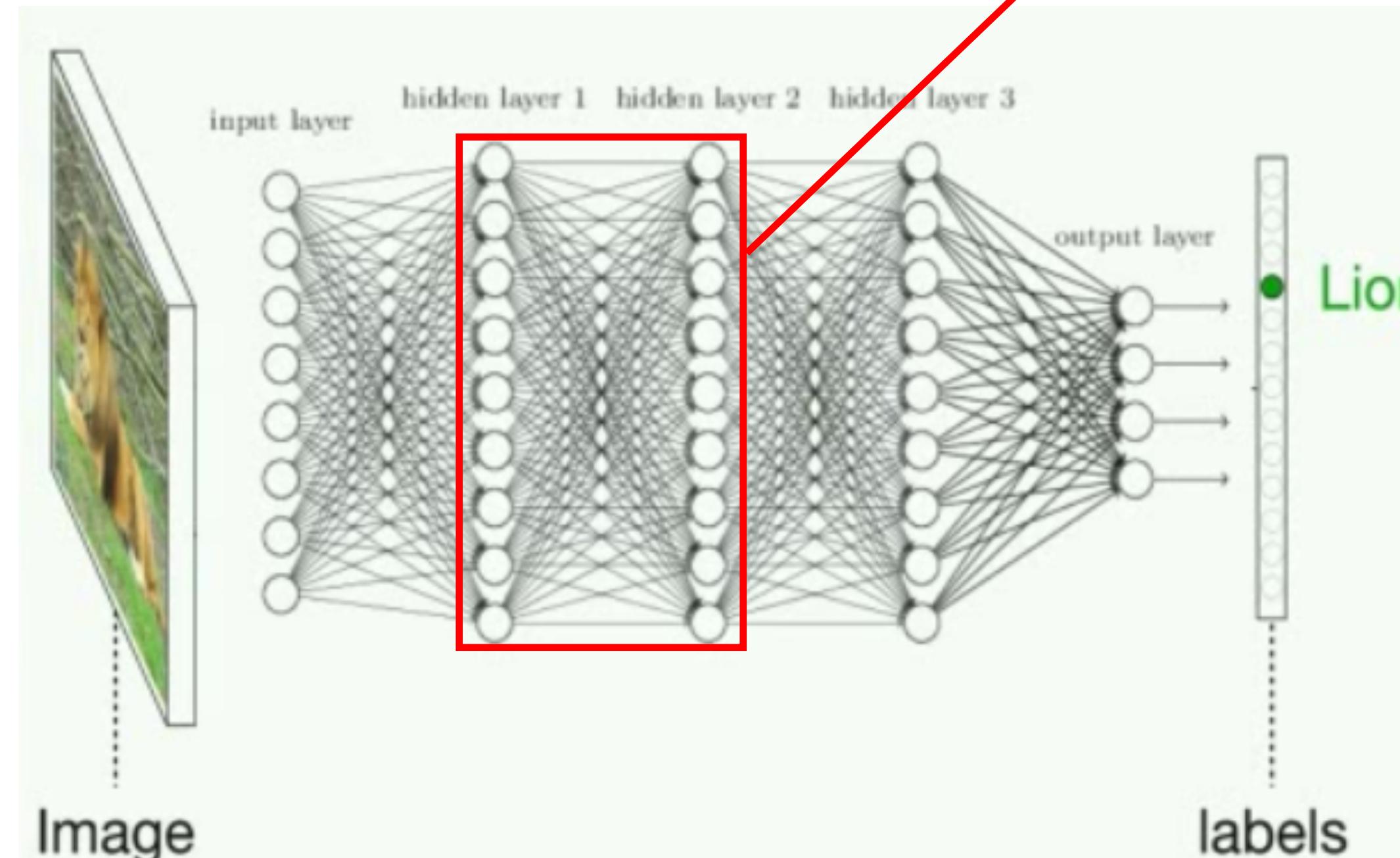
# General purpose



[Image: Jeff Clune](#)

# General purpose

All-to-all coupling between layers → **weight matrices**



[Image: Jeff Clune](#)

# document-word matrix

—# unique words (types)—	
—# documents—	
0	0 0 0 7 3 0 0 1 ... 0 5 0 0 0
:	: : :

Don't need this matrix when computing naive Bayes

- just for organizing our thoughts
- inefficient space-wise: very big but very sparse

# document-word matrix

Don't need this matrix when computing naive Bayes

- just for organizing our thoughts
  - inefficient space-wise: very big but very sparse

# **So why get into it? Why discuss this matrix?**

**Because it's almost what we want,  
and it's what we start from**

# document-word matrix

## Recall

# Part-of-Speech (POS) tagging

# Named Entity Recognition

# Sentence parsing

# Question answering

# Summarization

Don't need this matrix when computing naive Bayes

- just for organizing our thoughts
  - inefficient space-wise: very big but very sparse

# **So why get into it? Why discuss this matrix?**

**Because it's almost what we want,  
and it's what we start from**

# document-word matrix

## Recall

# Part-of-Speech (POS) tagging

# Named Entity Recognition

# Sentence parsing

# Question answering

# Summarization

Don't need this matrix when computing naive Bayes

- just for organizing our thoughts
  - inefficient space-wise: very big but very sparse

# So why get into it? Why discuss this matrix?

**Because it's almost what we want,  
and it's what we start from**

# document-word matrix

## Recall

# Part-of-Speech (POS) tagging

# Named Entity Recognition

# Sentence parsing

# Question answering

# Summarization

# Semantic similarity

Don't need this matrix when computing naive Bayes

- just for organizing our thoughts
  - inefficient space-wise: very big but very sparse

# **So why get into it? Why discuss this matrix?**

**Because it's almost what we want,  
and it's what we start from**

$$S(w_i, w_j)$$

**Suppose I have two words that I want to compare**

# document-word matrix

Semantic similarity

$$S(w_i, w_j)$$

**Suppose I have two words that I want to compare**

# document-word matrix

Semantic similarity

$$S(w_i, w_j)$$

**Suppose I have two words that I want to compare**

I have vectors for each word: **one-hot vectors**

$$\mathbf{v}(w_i) = [0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \cdots]$$

$$\mathbf{v}(w_j) = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \cdots]$$

(for example)

# document-word matrix

Semantic similarity

$S(w_i, w_j)$

Suppose I have two words that I want to compare

I have vectors for each word: **one-hot vectors**

$$\mathbf{v}(w_i) = [0 \ 0 \ 0 \boxed{1} \ 0 \ 0 \boxed{0} \cdots]$$

$$\mathbf{v}(w_j) = [0 \ 0 \ 0 \boxed{0} \ 0 \ 0 \boxed{1} \cdots]$$

(for example)

vectors are great for measuring similarity

Can define all sorts of vector similarities

$$S(w_i, w_j) \equiv \frac{\mathbf{v}(w_i) \cdot \mathbf{v}(w_j)}{\|\mathbf{v}(w_i)\| \|\mathbf{v}(w_j)\|}$$

but all I will really capture here is whether or not

$$w_i = w_j$$

**One-hot vectors not right for  
this problem...**

# document-word matrix

**Back to square one:**

—# unique words (types)—	
—# documents—	
0	0
0	7
3	0
0	1
...	...
5	0
0	0
:	:

Don't need this matrix when computing naive Bayes

- just for organizing our thoughts
- inefficient space-wise: very big but very sparse

**So why get into it? Why discuss this matrix?**

**Because it's almost what we want, and it's what we start from**

# document-word matrix

**Back to square one:**

— # unique words (types) —	
— # documents —	
0	0
0	7
3	0
0	1
...	...
5	0
0	0
:	:

- Don't need this matrix when computing naive Bayes
- just for organizing our thoughts
  - inefficient space-wise: very big but very sparse

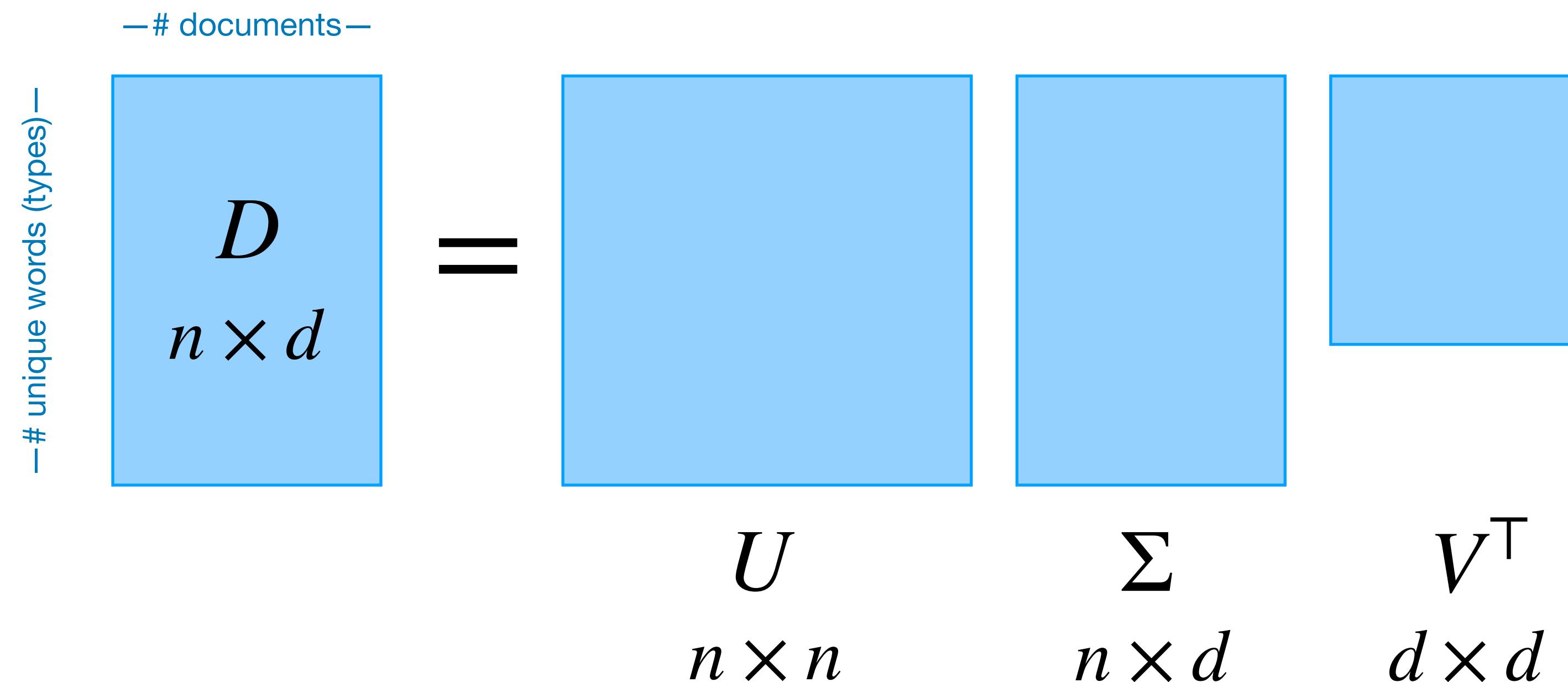
**So why get into it? Why discuss this matrix?**

**Because it's almost what we want, and it's what we start from**

Another approach: *Matrix Factorization*

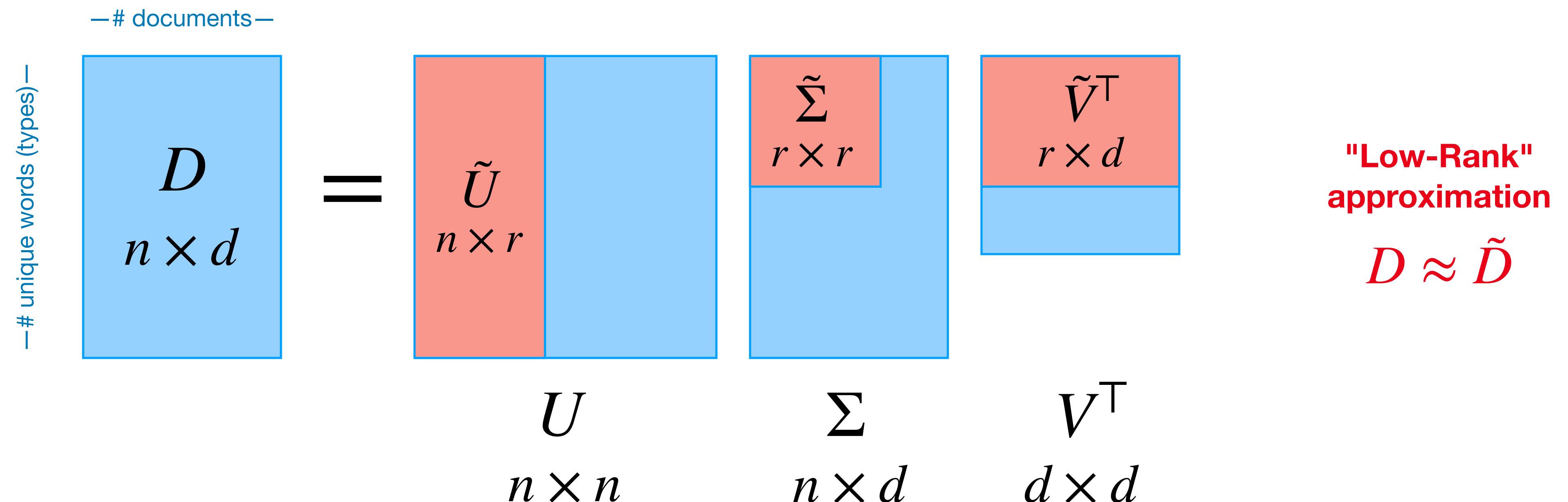
# Matrix factorization: word-document matrix

$$D = U\Sigma V^T \quad (\text{Singular Value Decomposition})$$



# Matrix factorization: word-document matrix

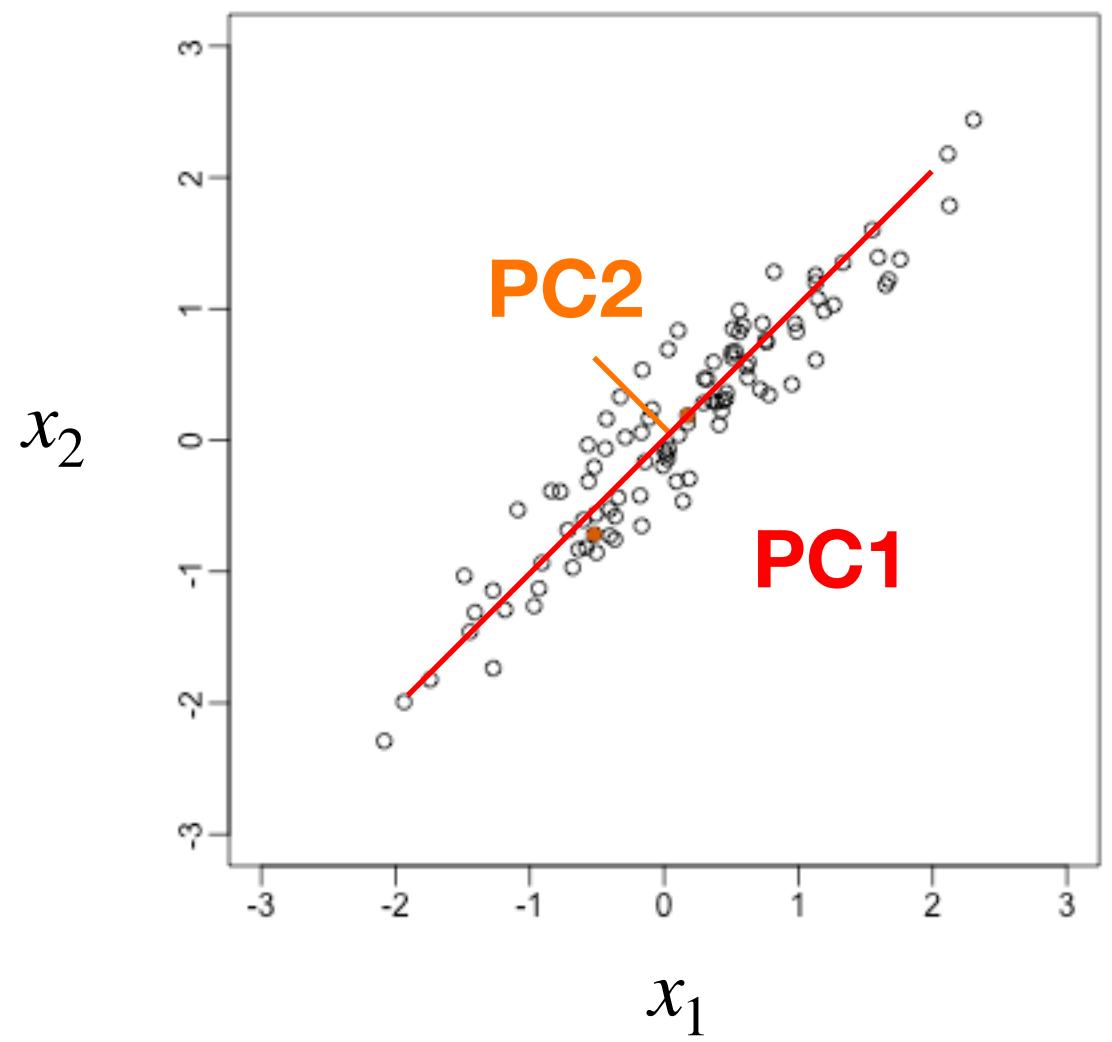
$$D = U\Sigma V^T \quad (\text{Singular Value Decomposition})$$



# Matrix factorization – Intuition

## Scatter plots

Principal Component Analysis (PCA)  
Eigendecomp. of covariance matrix

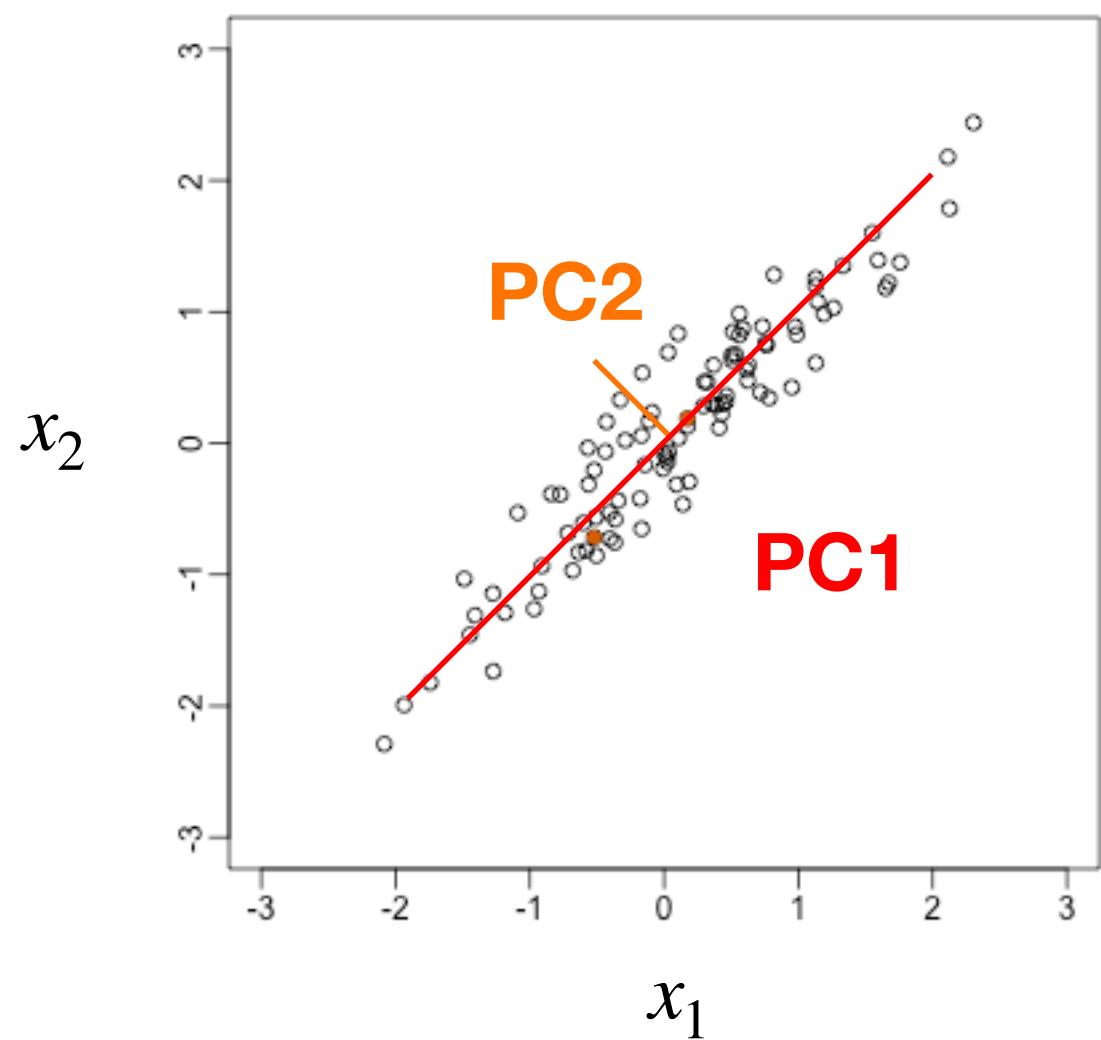


SVD finds **linear combinations**  
of the original variables that  
account for the most variation  
of the data

# Matrix factorization — Intuition

## Scatter plots

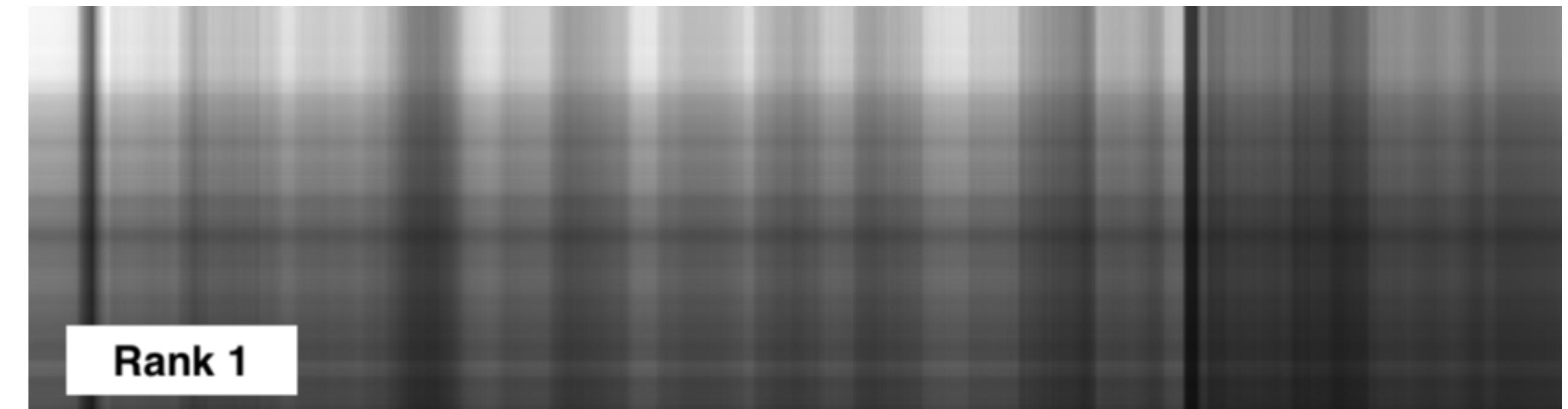
Principal Component Analysis (PCA)  
Eigendecomp. of covariance matrix



SVD finds **linear combinations** of the original variables that account for the most variation of the data

## Image compression

Approximate pixels as linear combinations of (small number of) (outer products of) singular vectors

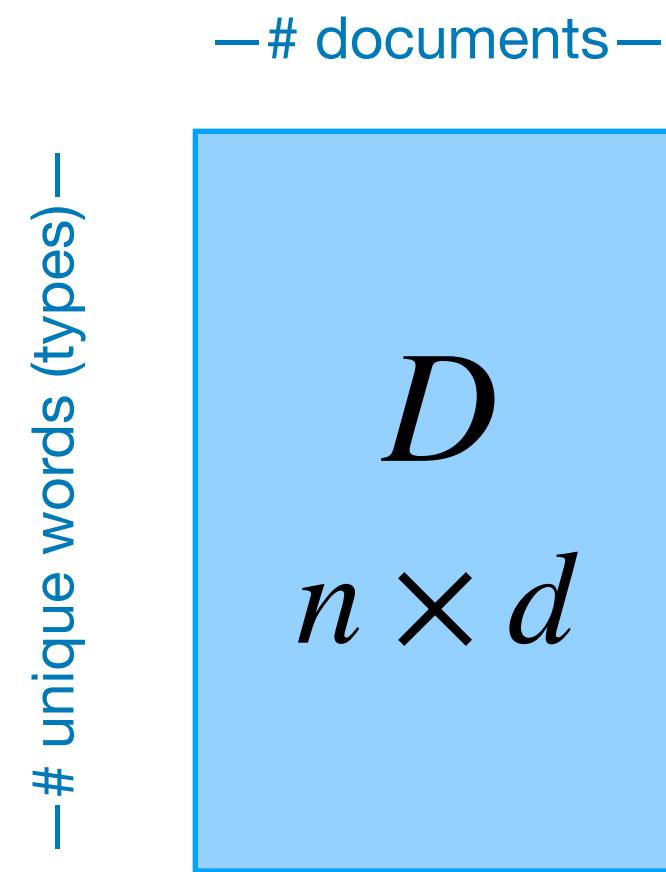


<http://copper.math.buffalo.edu/438/svdimages/components.html>

# Matrix factorization – Intuition

What about word-document matrix?

$$D = U\Sigma V^T$$



**Low-Rank  
approximation**

$$D \approx \tilde{D}$$

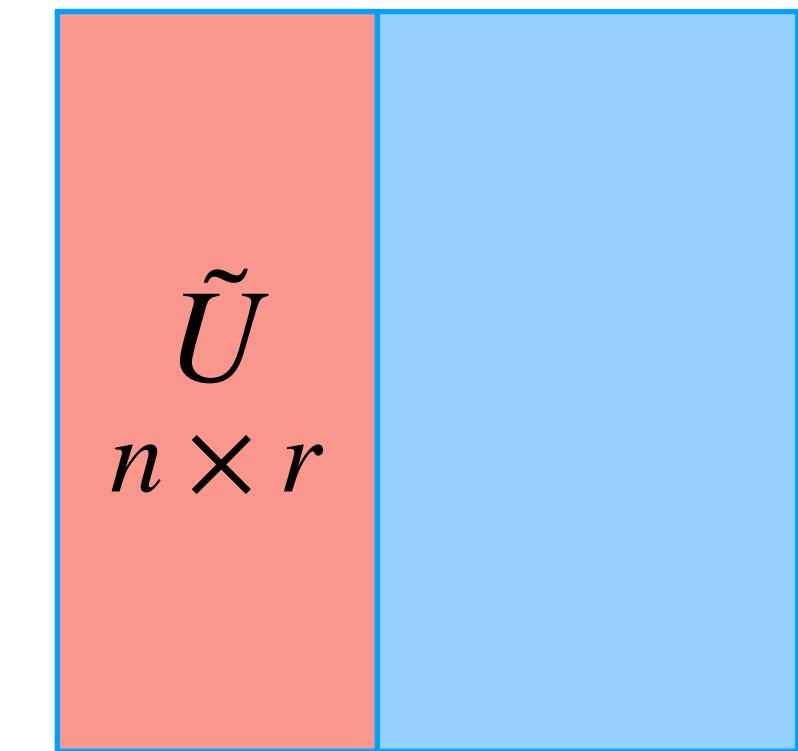
**Learning patterns for how words are distributed across documents**

[car, truck, bird]

[0.56\*car + 1.32\*truck, bird]

3 words

2 "concepts"

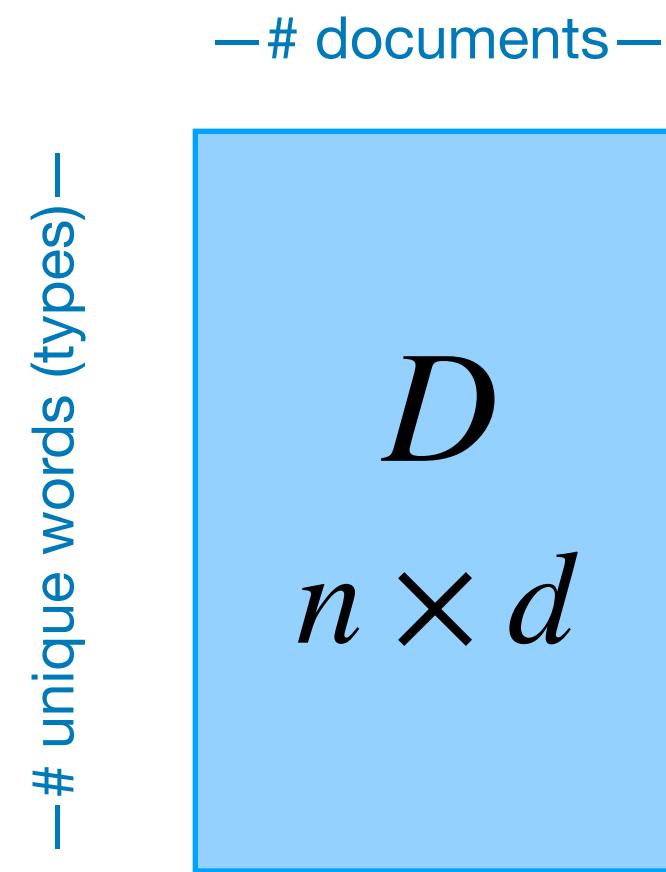


$$U$$
  
$$n \times n$$

# Matrix factorization – Intuition

What about word-document matrix?

$$D = U\Sigma V^\top$$

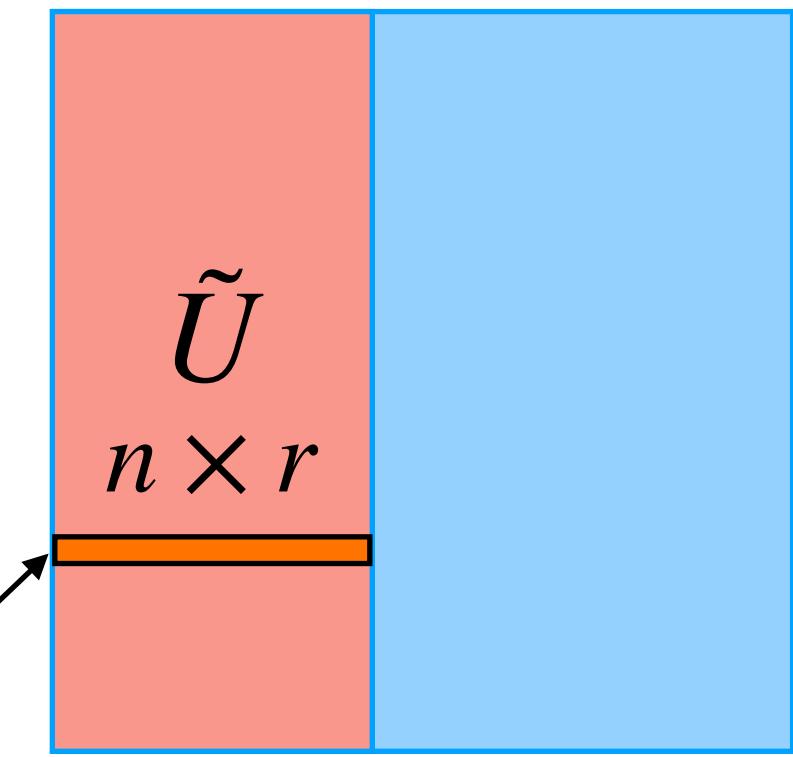


**Low-Rank  
approximation**

$$D \approx \tilde{D}$$

**Learning patterns for how words are distributed across documents**

[car, truck, bird]  
↓  
[0.56\*car + 1.32\*truck, bird]  
3 words  
2 "concepts"



$$U  
n \times n$$

Each word represented by an  $r$ -dimensional (dense) vector

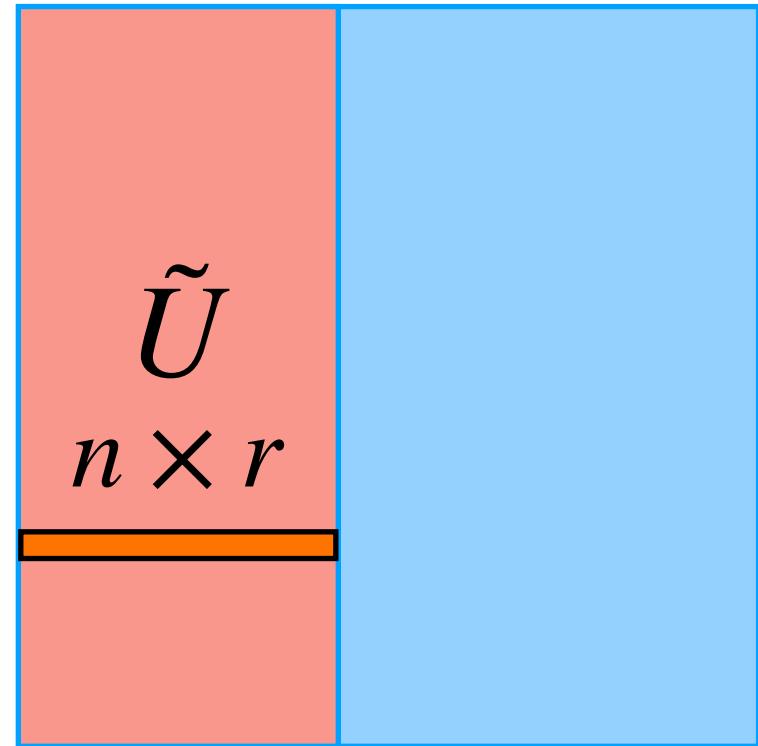
**Latent Semantic Analysis (Indexing)**

Deerwester et al. (1990)

# Matrix factorization – Intuition

What about word-document matrix?

$$D = U\Sigma V^\top$$



Each word represented by an  $r$ -dimensional (dense) vector

I have vectors for each word: one-hot vectors

$$\mathbf{v}(w_i) = [0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ \dots]$$

$$\mathbf{v}(w_j) = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ \dots]$$

(for example)

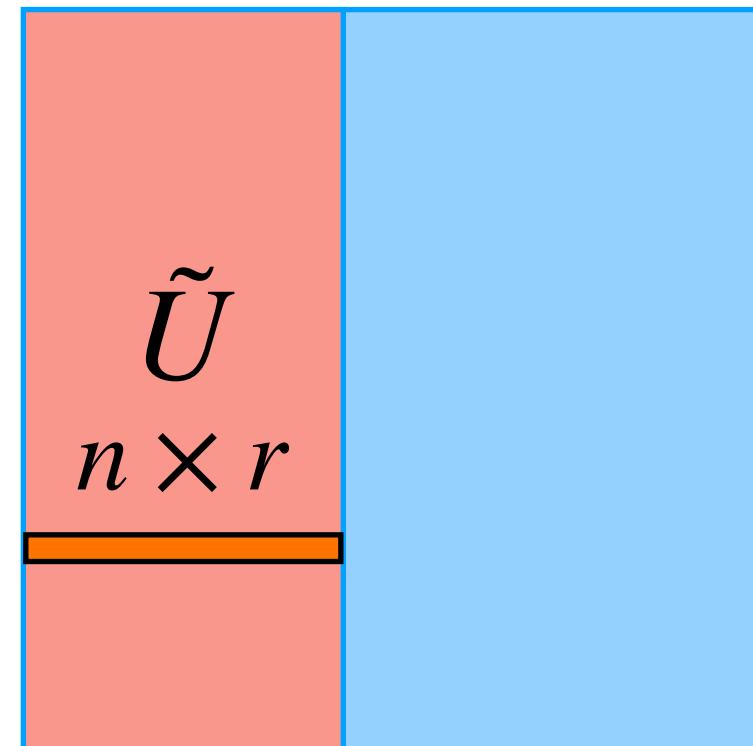
**Latent Semantic Analysis (Indexing)**

Deerwester et al. (1990)

# Matrix factorization – Intuition

What about word-document matrix?

$$D = U\Sigma V^\top$$



Each word represented by an  $r$ -dimensional (dense) vector

I have vectors for each word: one-hot vectors

$$\mathbf{v}(w_i) = [0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ \dots]$$

$$\mathbf{v}(w_j) = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ \dots]$$

(for example)

**Latent Semantic Analysis (Indexing)**

Deerwester et al. (1990)

Use singular vectors as *representations!*

# Limitations of LSA

SVD-derived concepts may not capture all the complexity of natural language—  
makes some simplifying assumptions (including bag-of-words)

Considers global patterns of words across documents—may miss local  
contextual patterns



**Can we do better?**

“You shall know a word by the company  
it keeps.”

—JR Firth

Distributional Semantics (1950s)

“You shall know a word by the company  
it keeps.”

—JR Firth

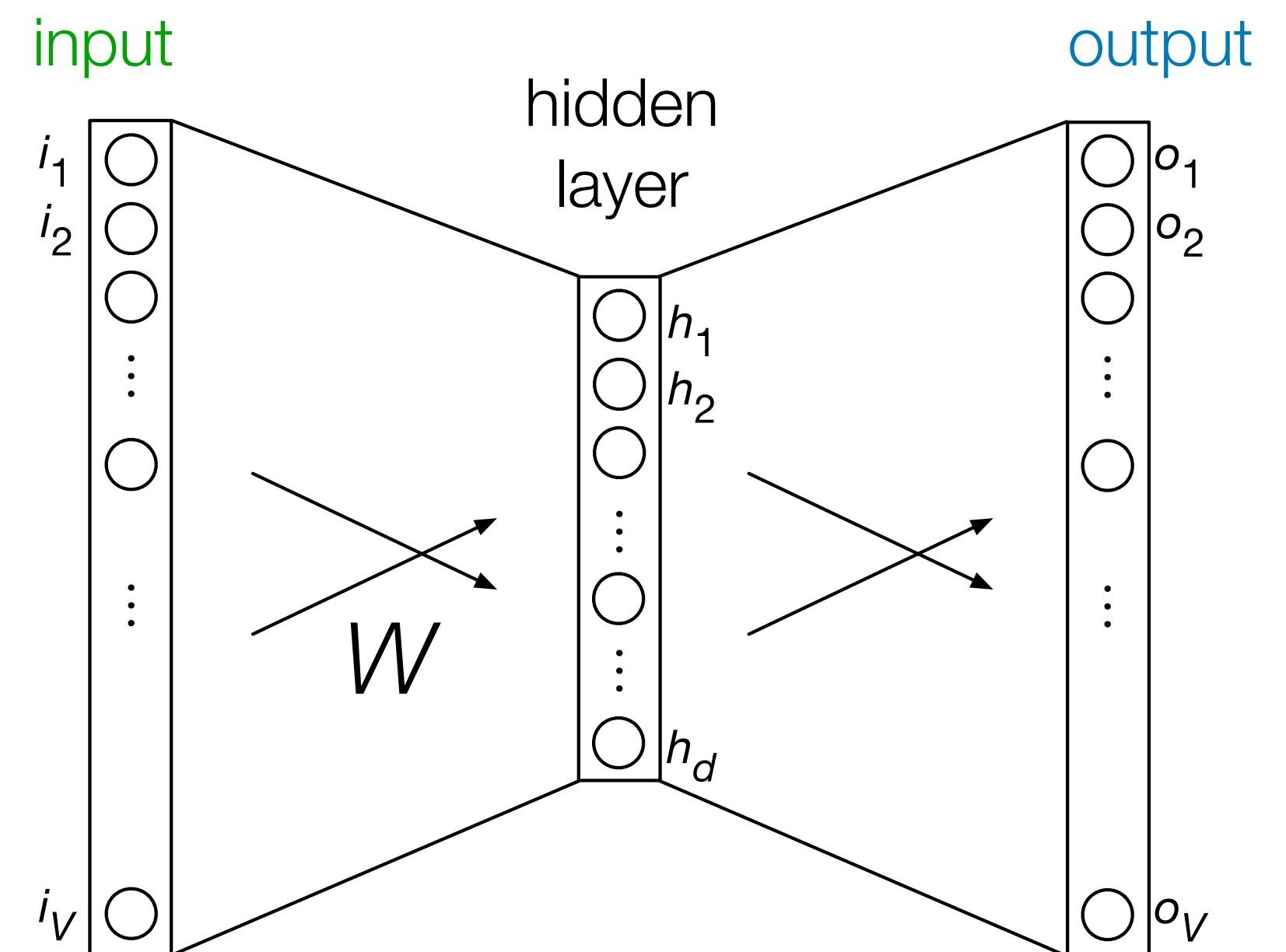
## Distributional Semantics (1950s)

... worlds are yours except **europa** attempt no landings there ...

Turn *large* text corpus into collection  
of **word-context** pairs

Predict **word**  
from **context**

*Training data*

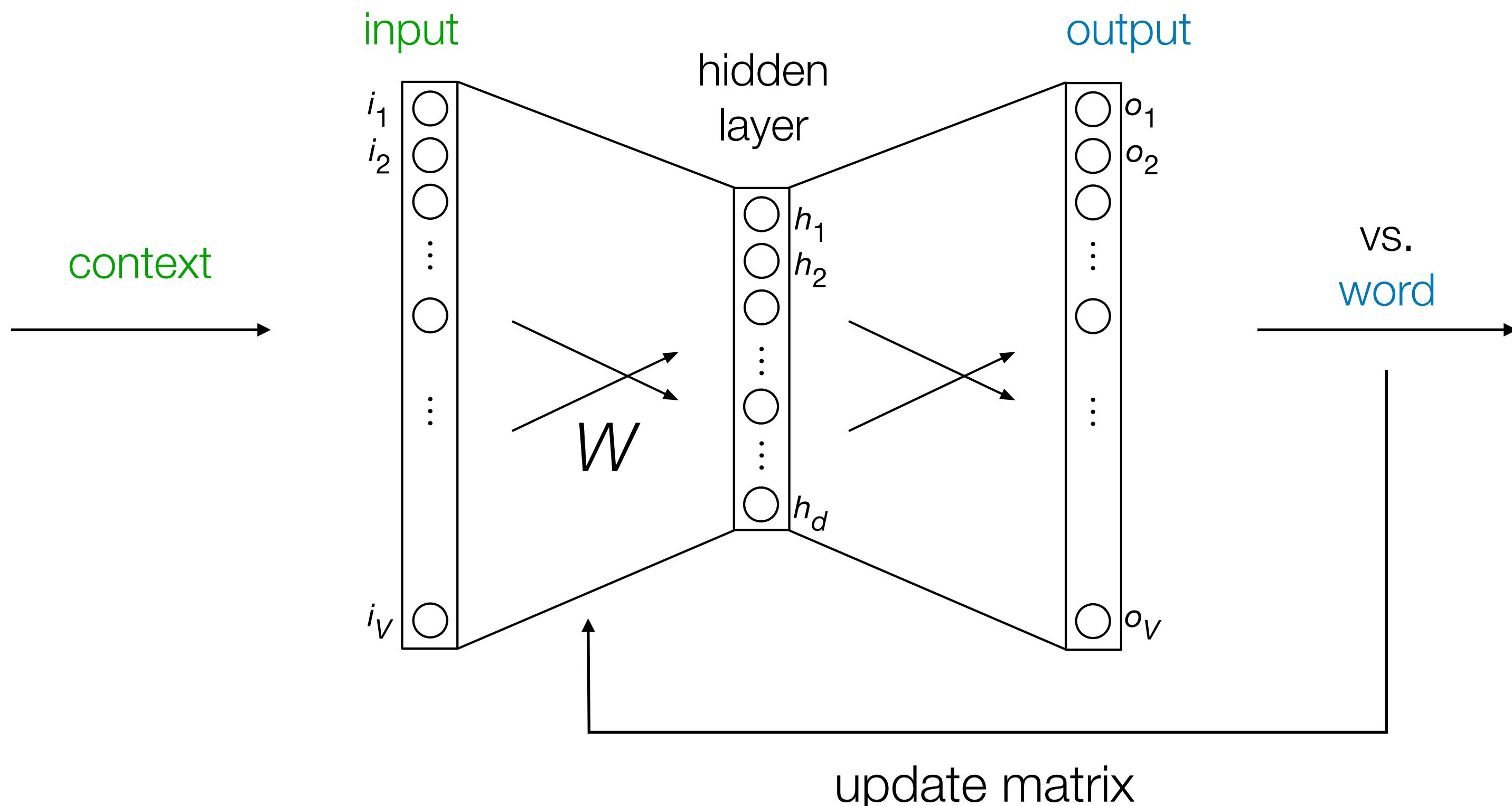


(Or predict **context**  
from **word!**)

Bengio *et al.* JMLR (2003)  
Mikolov *et al.* NIPS (2013)

Predict word  
from context

*Training data*

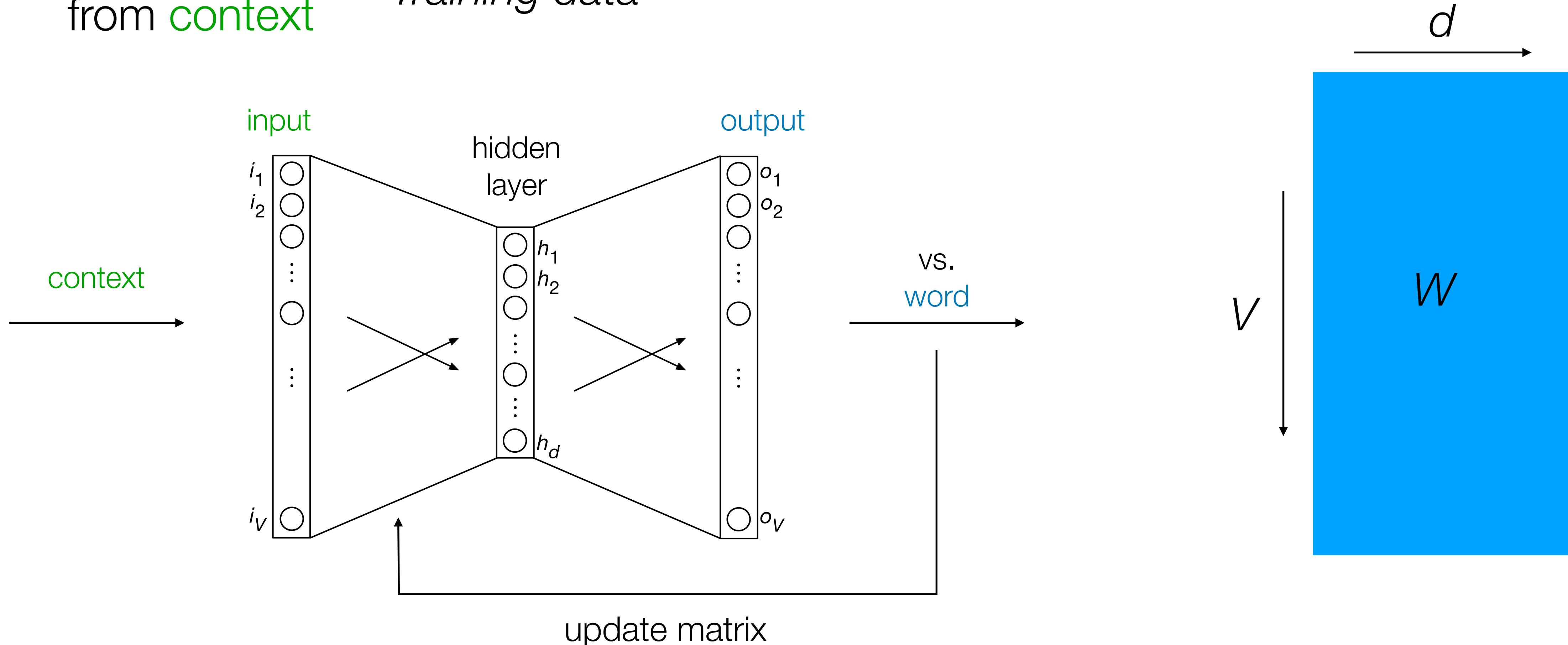


(Or predict context  
from word!)

Bengio *et al.* JMLR (2003)  
Mikolov *et al.* NIPS (2013)

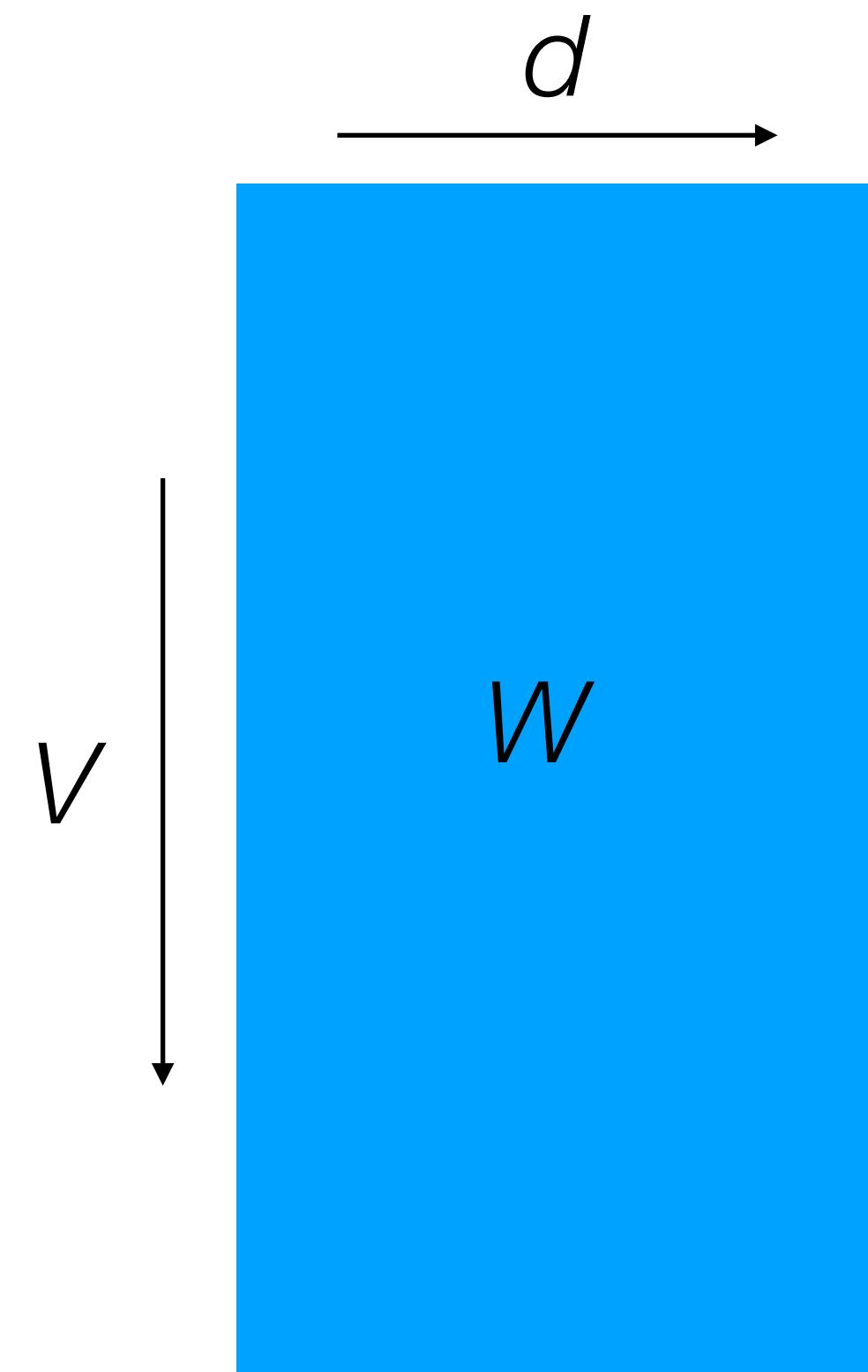
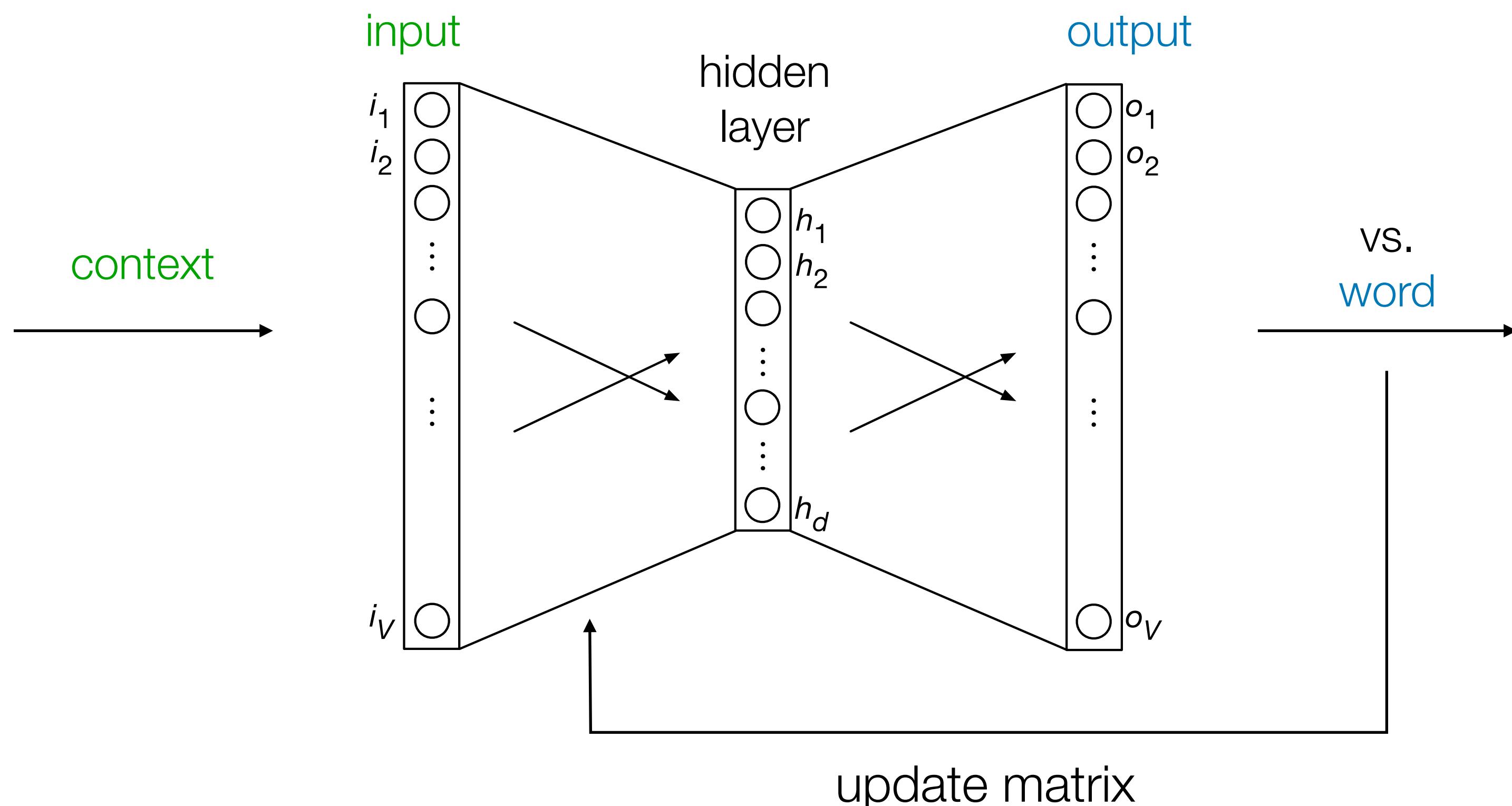
Predict word  
from context

*Training data*



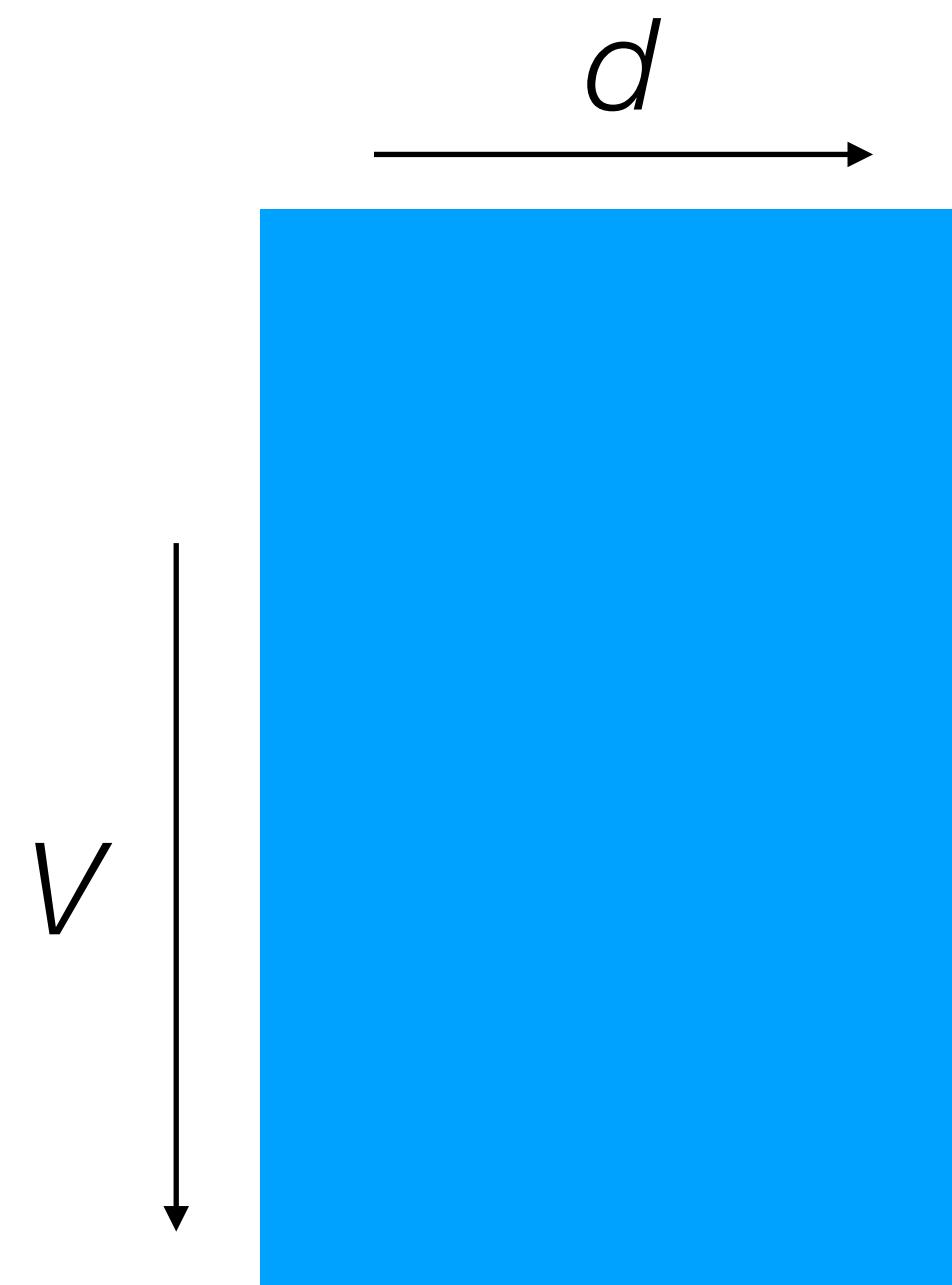
Predict word  
from context

*Training data*



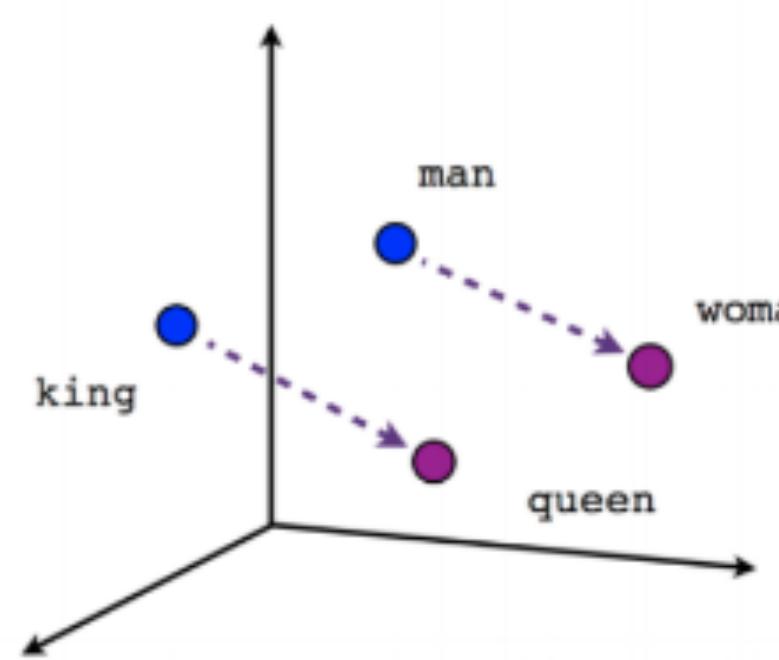
Sound familiar?

$$\begin{matrix} \tilde{U} \\ U \end{matrix} \quad \begin{matrix} n \times r \\ n \times n \end{matrix}$$



- Each row is an  $d$ -dimensional *word vector*

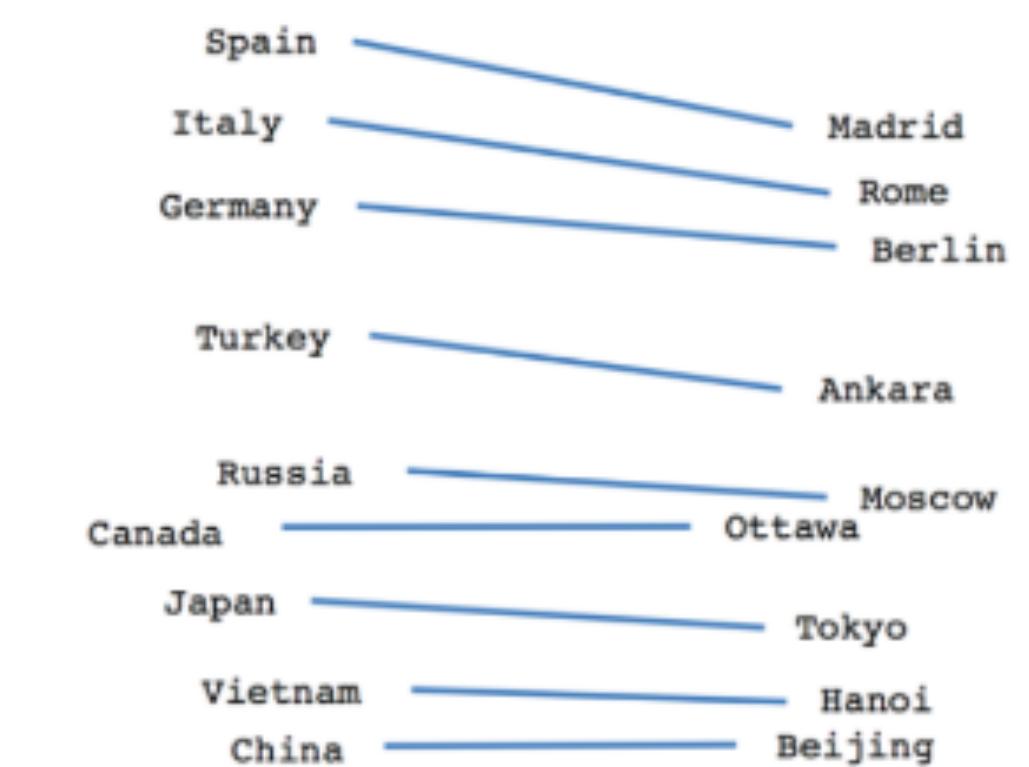
vectors encode semantics



Male-Female



Verb tense



Country-Capital

**Word embeddings, vector representations, word2vec (Mikolov, et al.)**

Bengio *et al.* JMLR (2003)  
Mikolov *et al.* NIPS (2013)

If this sounds like SVD,  
you're not crazy....

$$M \sim \log \frac{P(\mathbf{w}, \mathbf{c})}{P(\mathbf{w})P(\mathbf{c})}$$

$$M = U\Sigma V^\top$$

$$M \approx M_d = U_d \Sigma_d V_d^\top$$

$$W^{\text{SVD}} = U_d \Sigma_d$$

---

## Neural Word Embedding as Implicit Matrix Factorization

---

**Omer Levy**  
Department of Computer Science  
Bar-Ilan University  
omerlevy@gmail.com

**Yoav Goldberg**  
Department of Computer Science  
Bar-Ilan University  
yoav.goldberg@gmail.com

Neural network implicitly performs  
**weighted** factorization of  $M$

# Embedding words in vector spaces has taken the world by storm

Google Scholar

Distributed representations of words and phrases as

T Mikolov, I Sutskever, K Chen, GS Corrado... - Advances in neural

The recently introduced continuous Skip-gram model is an efficient method for computing high quality distributed vector representations that capture a large number of semantic and syntactic word relationships. In this paper we present several imp

☆ 99 Cited by 18411 Related articles All 32 versions Import

word vectors, sentence vectors, *thought*  
vectors...

Lots of natural language processing applications  
including semantic similarity:

$$S(s_i, s_j) = \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_i\| \|\mathbf{v}_j\|}$$

Embedding words in vector spaces has taken the world by storm

Must be approached with **caution**

Google Scholar

Distributed **representations** of words and phrases ↗  
[T Mikolov, I Sutskever, K Chen, GS Corrado... - Advances in neural](#)  
The recently introduced continuous Skip-gram model is an efficient m quality distributed vector representations that capture a large number and semantic word relationships. In this paper we present several imp

☆ 99 Cited by 18411 Related articles All 32 versions Import

word vectors, sentence vectors, *thought* vectors...

Lots of natural language processing **applications** including **semantic similarity**:

$$S(s_i, s_j) = \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_i\| \|\mathbf{v}_j\|}$$

**Semantics derived automatically from language corpora contain human-like biases**

Aylin Caliskan,<sup>1\*</sup> Joanna J. Bryson,<sup>1,2\*</sup> Arvind Narayanan<sup>1\*</sup>

"[...] word vectors contain stereotypes matching those documented with the [Implicit Association Test]"

Must be approached with **caution**

## Semantics derived automatically from language corpora contain human-like biases

Aylin Caliskan,<sup>1\*</sup> Joanna J. Bryson,<sup>1,2\*</sup> Arvind Narayanan<sup>1\*</sup>

"[...] word vectors contain stereotypes matching those documented with the [Implicit Association Test]"

Neural language representations predict outcomes of scientific research

James P. Bagrow<sup>1,2,\*</sup>, Daniel Berenberg<sup>3,2</sup>, and Joshua Bongard<sup>3,2</sup>

<sup>1</sup>Department of Mathematics & Statistics, University of Vermont, Burlington, VT, United States

<sup>2</sup>Vermont Complex Systems Center, University of Vermont, Burlington, VT, United States

<sup>3</sup>Department of Computer Science, University of Vermont, Burlington, VT, United States

\*Corresponding author. Email: [james.bagrow@uvm.edu](mailto:james.bagrow@uvm.edu), Homepage: [bagrow.com](http://bagrow.com)

May 17, 2018



**ConceptNet  
Numberbatch**

**ConceptNet at SemEval-2017 Task 2: Extending Word Embeddings with Multilingual Relational Knowledge**

Robyn Speer

Joanna Lowry-Duda

# Get started with pretrained vectors

The screenshot shows the homepage of the fastText library. It features a large, stylized logo with "fast" in red and "Text" in blue. Below the logo is the text "Library for efficient text classification and representation learning". At the bottom are two buttons: "GET STARTED" and "DOWNLOAD MODELS".

## English word vectors

This page gathers several pre-trained word vectors trained using fastText.

### Download pre-trained word vectors

Pre-trained word vectors learned on different sources can be downloaded:

1. [wiki-news-300d-1M.vec.zip](#): 1 million word vectors trained on Wikipedia news dataset (16B tokens).
2. [wiki-news-300d-1M-subword.vec.zip](#): 1 million word vectors trained on statmt.org news dataset (16B tokens).
3. [crawl-300d-2M.vec.zip](#): 2 million word vectors trained on CommonCrawl dataset (2B tokens).
4. [crawl-300d-2M-subword.zip](#): 2 million word vectors trained with subword n-grams (2B tokens).

<https://fasttext.cc>

## Google's Universal Sentence Encoder

The screenshot shows the TensorFlow Hub page for the Universal Sentence Encoder. The URL is <https://tfhub.dev/google/universal-sentence-encoder/2>. The page includes a sidebar with navigation links like "TensorFlow Hub" and "universal-sentence-encoder". The main content area displays the following details:

Version	Encoder of greater-than-word length text trained on a variety of data.
2	Version
Language	English
Network	DAN
Publisher	Google
Module type	text-embedding

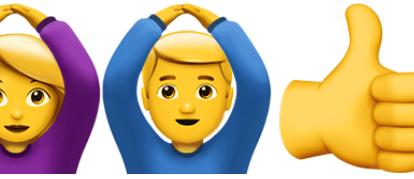
The "Overview" section states: "The Universal Sentence Encoder encodes text into high-dimensional vectors that can be used for text classification, semantic similarity, clustering and other natural language tasks."

<https://tfhub.dev/google/universal-sentence-encoder/2>

# Vectors as base for downstream tasks

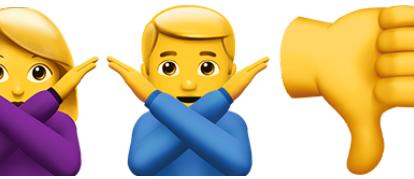
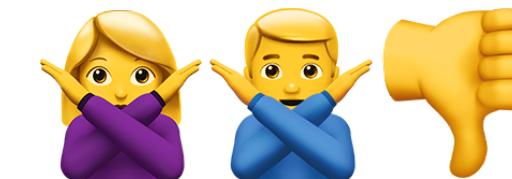
## Recognizing Textual Entailment (RTE)

"Alice and Bob both were at the scene of the crash"



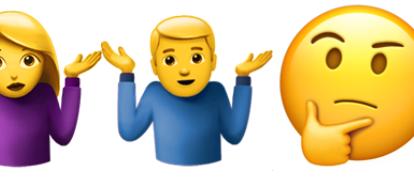
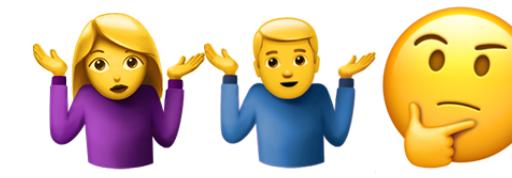
"Multiple people saw the accident"

Two dogs played in the park with the old man



There was only one canine in the park that day

I played baseball with the kids



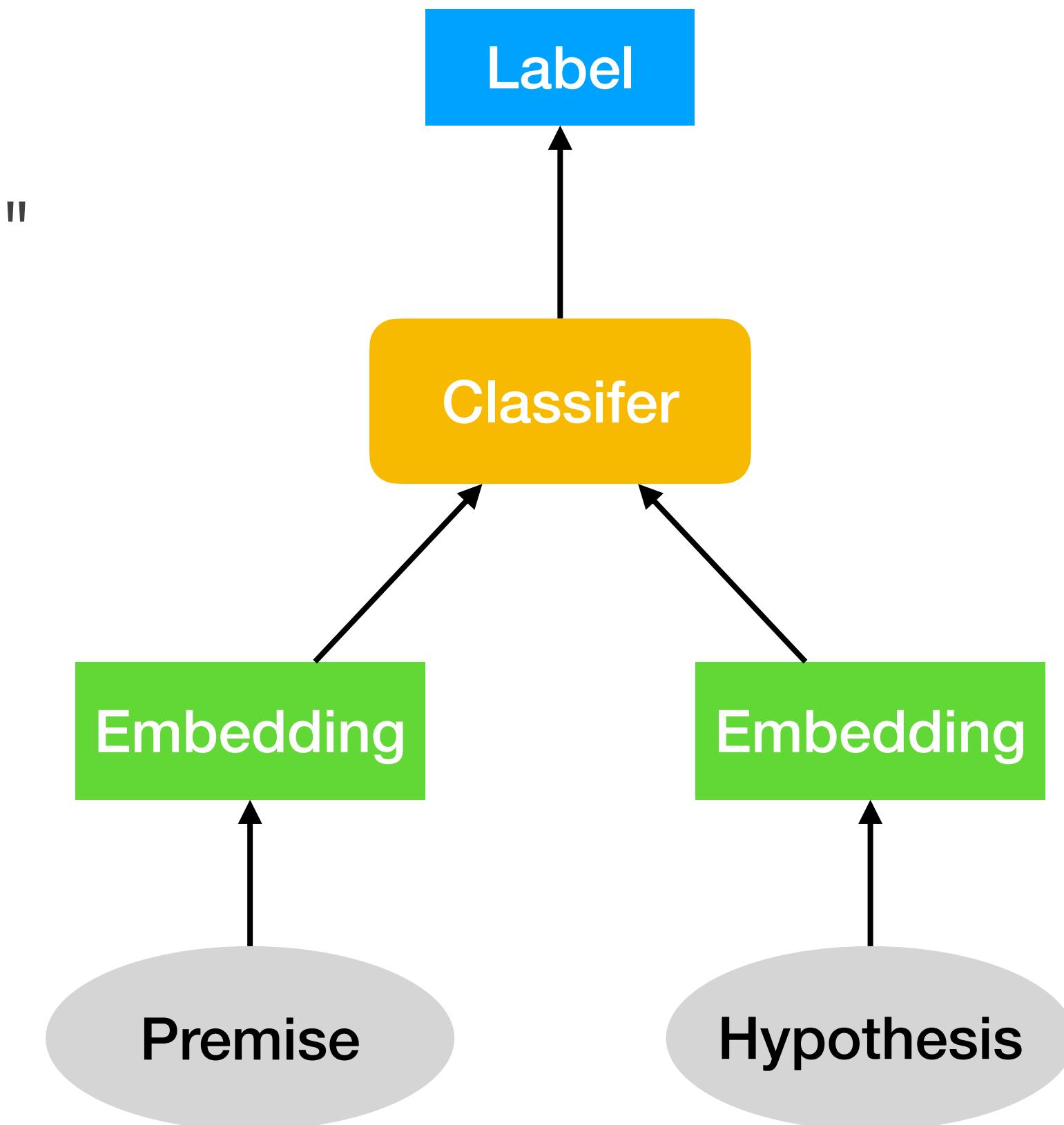
The kids love ice cream

# Vectors as base for downstream tasks

## Recognizing Textual Entailment (RTE)

P "Alice and Bob both were at the scene of the car crash"

H "Multiple people saw the accident"



# Vectors as base for downstream tasks

Neural language representations predict outcomes of scientific research

James P. Bagrow<sup>1,2,\*</sup>, Daniel Berenberg<sup>3,2</sup>, and Joshua Bongard<sup>3,2</sup>

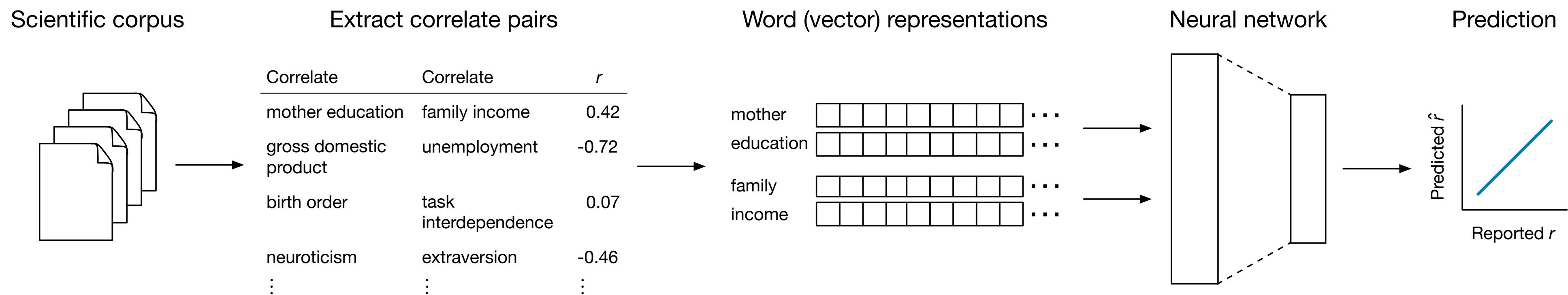
<sup>1</sup>Department of Mathematics & Statistics, University of Vermont, Burlington, VT, United States

<sup>2</sup>Vermont Complex Systems Center, University of Vermont, Burlington, VT, United States

<sup>3</sup>Department of Computer Science, University of Vermont, Burlington, VT, United States

\*Corresponding author. Email: [james.bagrow@uvm.edu](mailto:james.bagrow@uvm.edu), Homepage: [bagrow.com](http://bagrow.com)

May 17, 2018



# Vectors as base for downstream tasks

Neural language representations predict outcomes of scientific research

James P. Bagrow<sup>1,2,\*</sup>, Daniel Berenberg<sup>3,2</sup>, and Joshua Bongard<sup>3,2</sup>

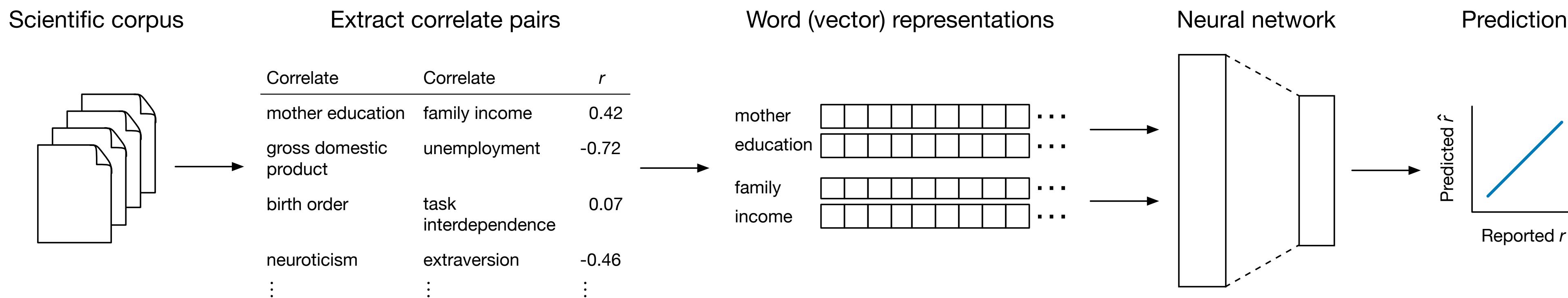
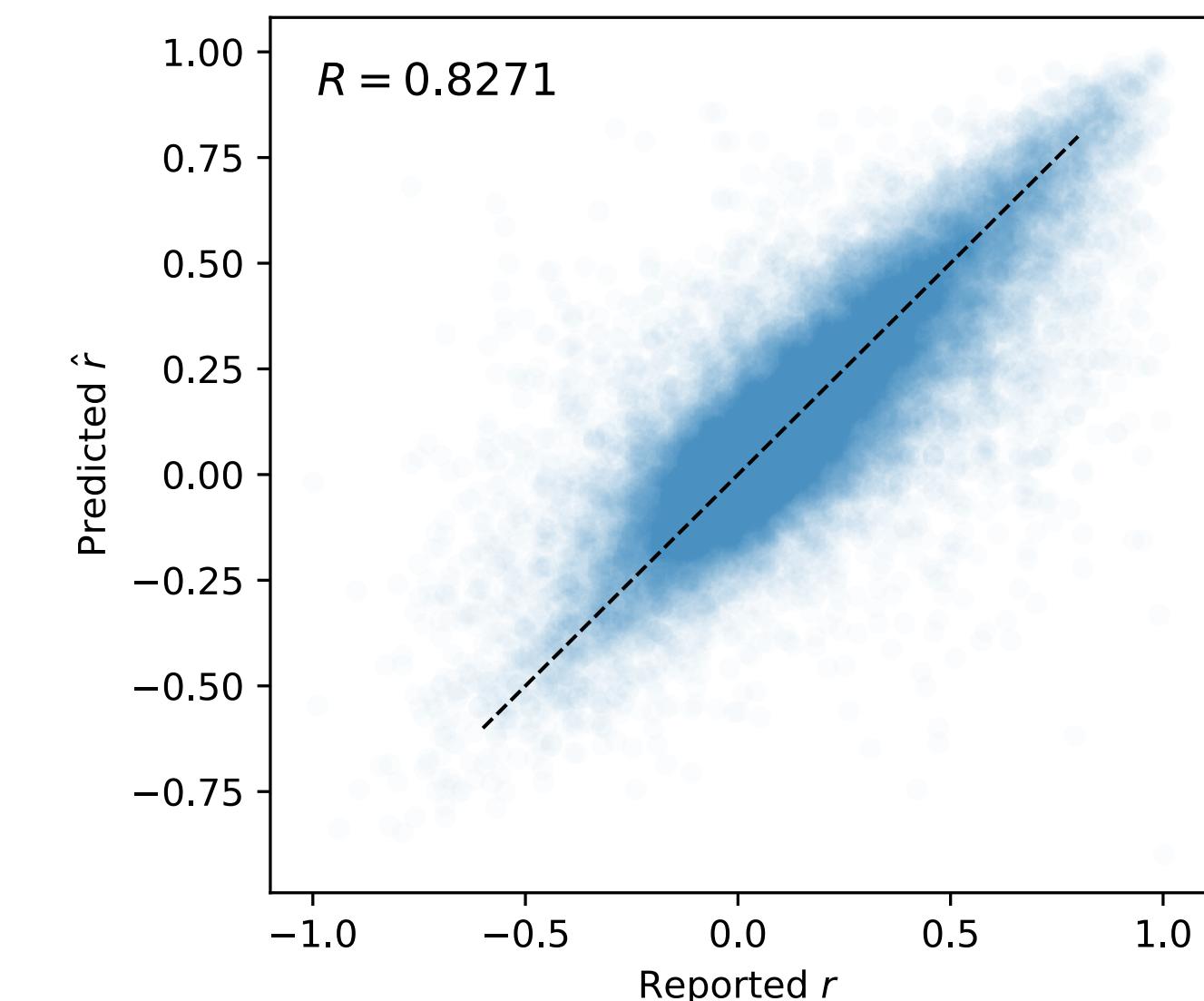
<sup>1</sup>Department of Mathematics & Statistics, University of Vermont, Burlington, VT, United States

<sup>2</sup>Vermont Complex Systems Center, University of Vermont, Burlington, VT, United States

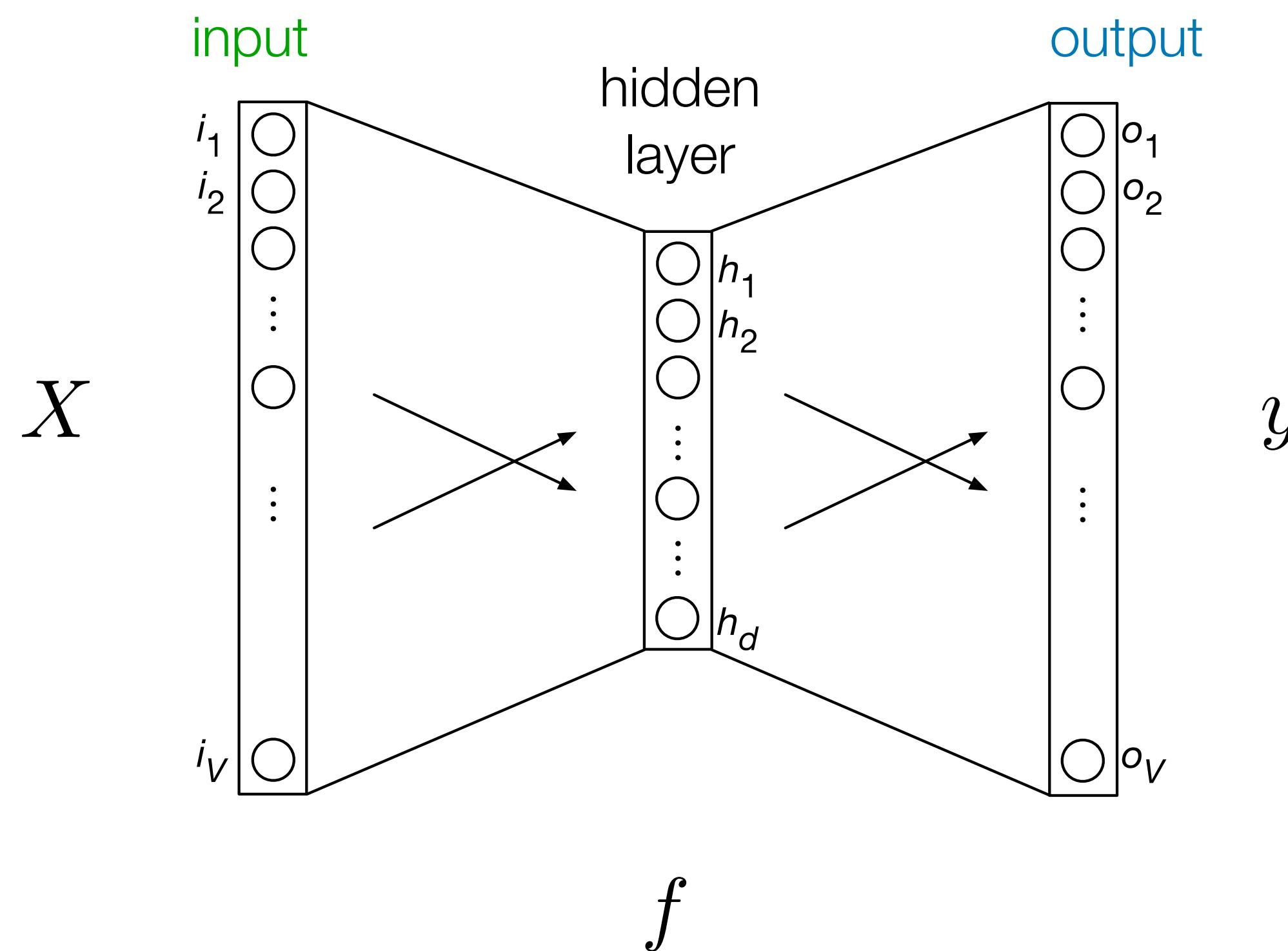
<sup>3</sup>Department of Computer Science, University of Vermont, Burlington, VT, United States

\*Corresponding author. Email: [james.bagrow@uvm.edu](mailto:james.bagrow@uvm.edu), Homepage: [bagrow.com](http://bagrow.com)

May 17, 2018



# Machine Learning takeaway: the power of sourcing training data

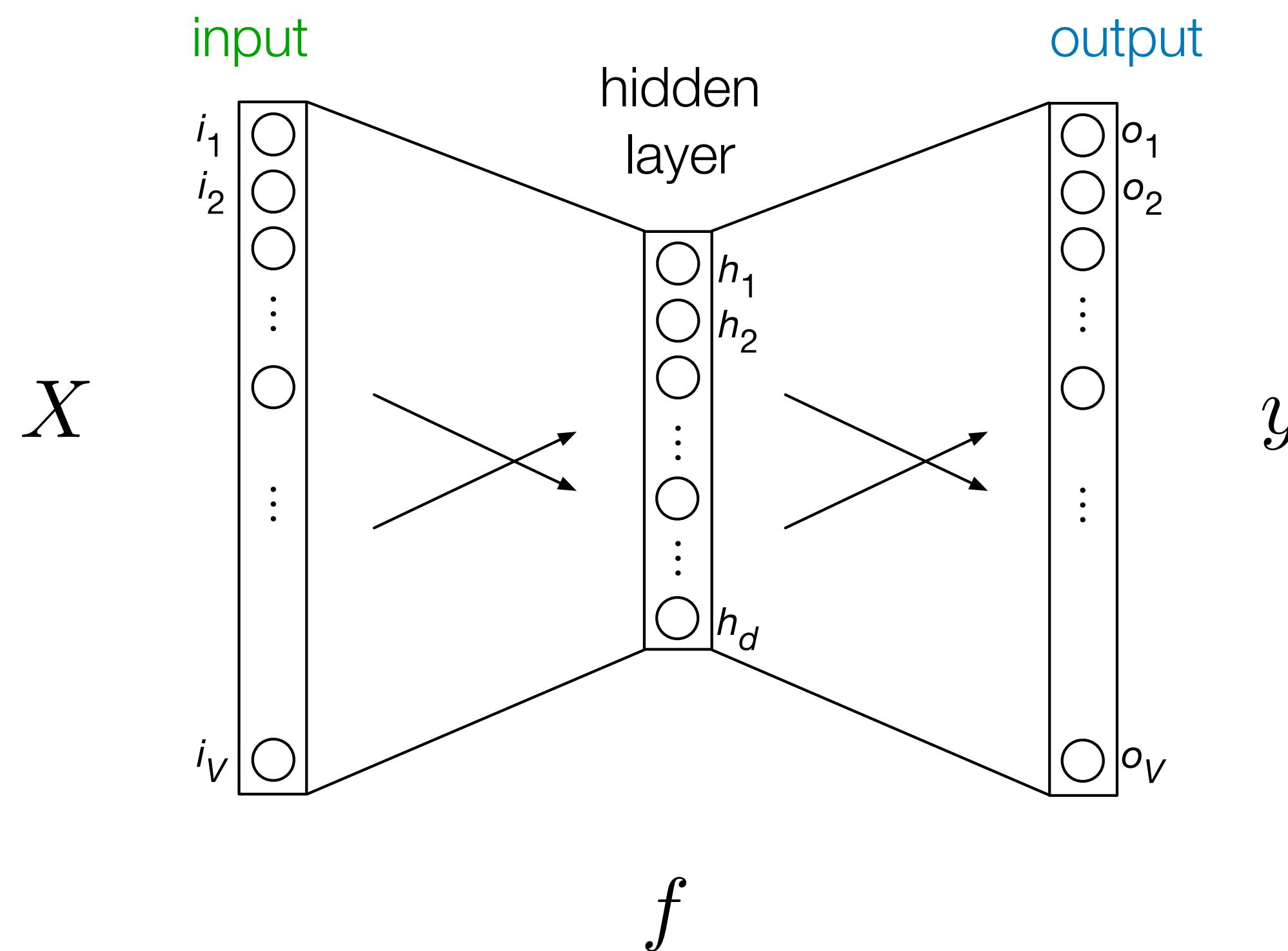


ML has become very good  
at **supervised learning**

$$y = f(X)$$

Many breakthroughs are **new mappings** of non-supervised  
problems into **supervised problems**

# Machine Learning takeaway: the power of **sourcing training data**



ML has become very good  
at **supervised learning**

$$y = f(X)$$

Many breakthroughs are **new**  
**mappings** of non-supervised  
problems into **supervised problems**

We just saw a technique for doing this with **text**: finding word-context pairs

# Summary - NLP tasks

## Related but Non-NLP tasks:

- OCR,
- Speech recognition
- Speech generation

## NLP Tasks

- Part-of-Speech (POS) tagging
- Named Entity Recognition
- Sentence parsing
- Question answering
- Summarization
- Text classification / clustering  
(topic models)
- Recognizing Textual Entailment
- Sentiment analysis
- Machine translation (sequence to sequence)

# Summary - NLP tasks

## Related but Non-NLP tasks:

- OCR,
- Speech recognition
- Speech generation

## Upstream task – computational inference of **semantics**

- Deal with Synonym, Polysemy

**Distributional hypothesis**



**Distributed representations → vector "embeddings"**

**(Current state-of-the-art)**

## NLP Tasks

- Part-of-Speech (POS) tagging
- Named Entity Recognition
- Sentence parsing
- Question answering
- Summarization
- Text classification / **clustering** (**topic models**)
- Recognizing Textual Entailment
- Sentiment analysis
- Machine translation (sequence to sequence)