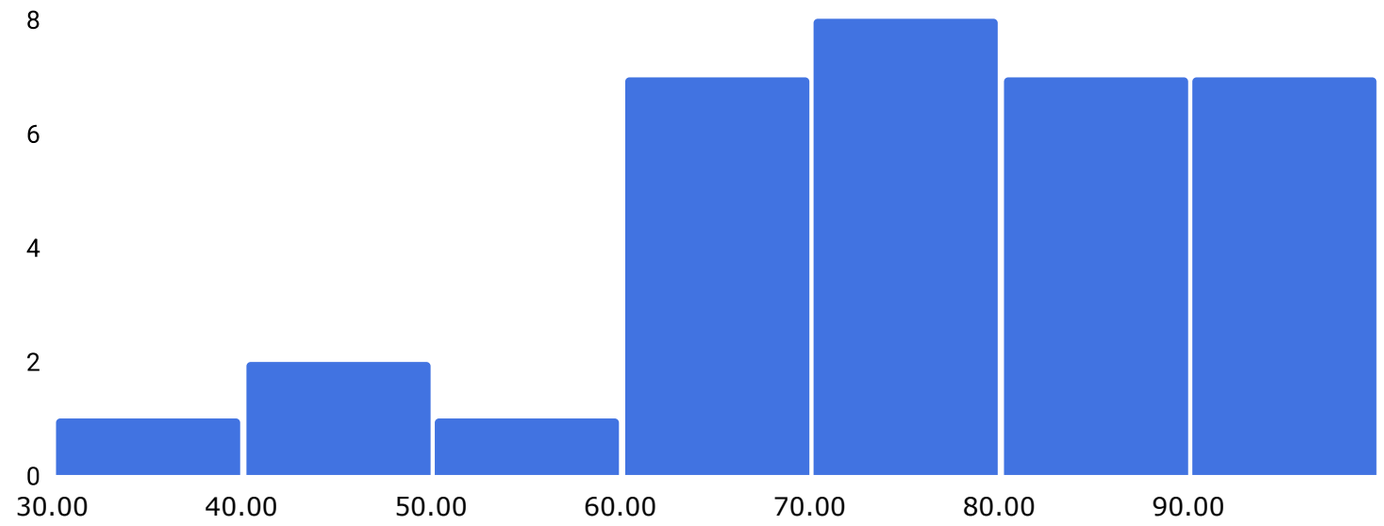
Carnegie Mellon University
Heinz College

94-775 Unstructured Data Analytics for Policy

Lecture 13: Image analysis with convolutional neural nets; (time permitting) intro to word embeddings

Slides by George H. Chen

Quiz 2



Median	Maximum	Mean	Std Dev
78.0	97.0	75.85	15.66

Quiz 2 turned out to be harder than we intended despite us trying *not* to make it difficult ☹️

Remember: letter grades are assigned based on a curve
(please do not panic! — it's still the case that no one is at risk of failing)

Solutions are in Canvas -> Files -> “Quiz 2 solutions.pdf”

Regrade requests (use Gradescope’s regrade request feature)
are due **Fri April 24, 11:59pm**
(for if you think there’s a genuine grading error)

Administrivia

- Quiz 2 & HW2 regrades due this Friday 4/24 by 11:59pm
- Piazza closes Friday 4/24, 11:59pm (if you're interested in Quiz 2 bonus)
- Final project presentations are next Monday 4/27 starting at 8:30am 😱
- Final project reports are due next Monday 11:59pm consist of:
  - Jupyter notebook (edited down to be clean, concise)
  - Slide deck for your final project presentation
- The final slide deck could be modified compared to what is presented in class beforehand (e.g., to incorporate feedback)
- I'm setting a hard limit of 12 minutes for your final project presentation
  - If you go over 12 minutes, you will not receive full credit for the "presentation" aspect of your final project score
- Presentation order: random! (You'll find Monday 4/27 at ~8:30am)

(Flashback) Final Project Rubric

- **Policy question (15%):** what public policy question are you addressing? Please be clear and concise.
- **Data analysis (30%):** clearly state what part of your data are unstructured (some but not all of the data you are analyzing must be unstructured), and carefully justify every step of your analysis with supporting visualizations/intermediate outputs as needed
- **Code (30%):** your code should actually run!
- **Conclusions (15%):** come up with insights that are based on your quantitative data analysis and that address your original policy question
- **Presentation (10%):** how polished is your final project presentation? — this is based on the live presentation your group makes (changes made to the slides after the presentation don't affect this score)

(Flashback) Final Project Rubric

- **Code (30%):** your code should actually run!

We recommend that you send us 2 notebooks, 1 with preprocessing (that saves a small, preprocessed version of the dataset) and another notebook that runs your analyses (using the preprocessed version of the data)

We will only run the notebook that does the analysis after loading in the preprocessed data (i.e., we won't recompute the preprocessed data)

If your dataset is overall small to begin with and you'd prefer just sending in a single notebook with the dataset, that's fine too

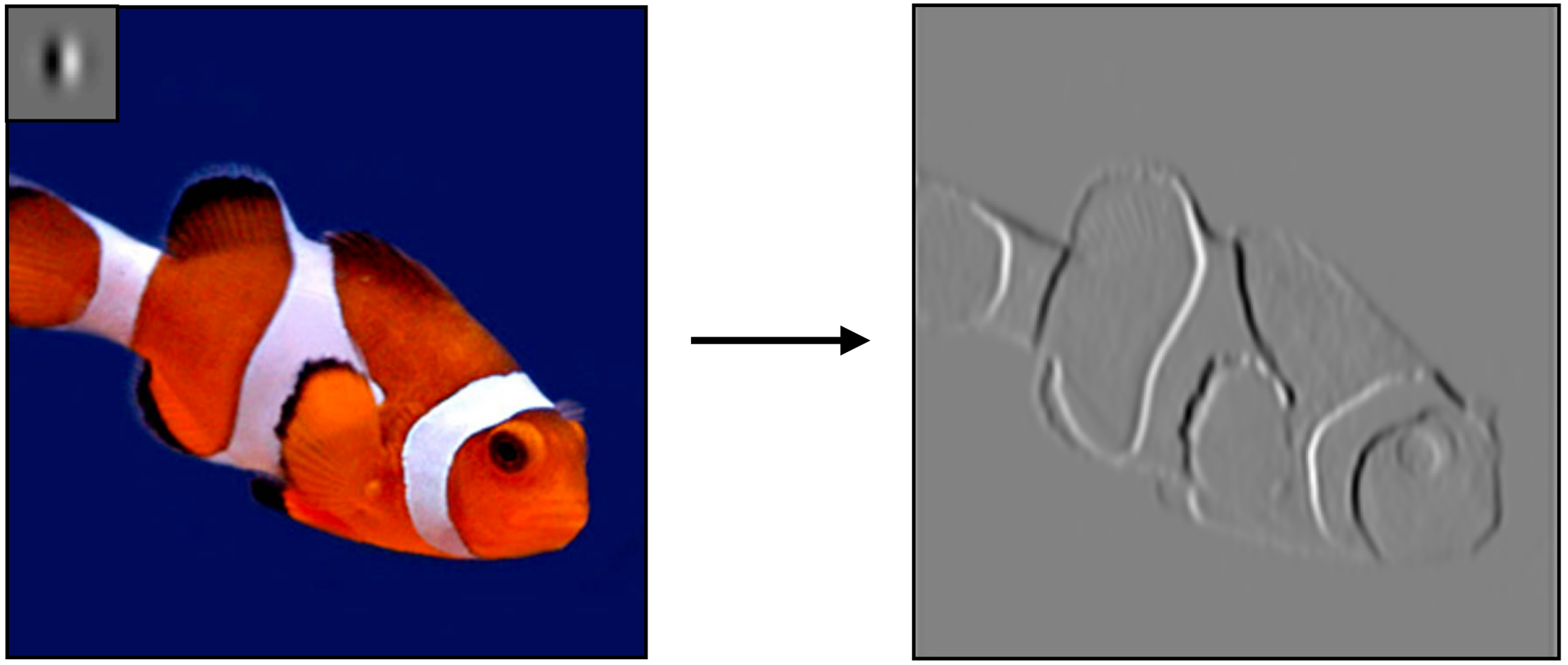
We don't have a strict definition of how small your preprocessed dataset is

Your code should ideally take us trivially less than 1 hour to run

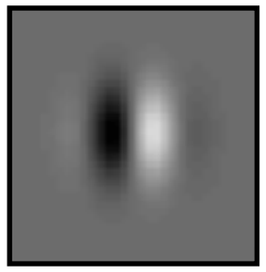
Please try to make your notebooks clear & concise
(in the real world, when you ask people to look at your code notebook, you should've cleaned it up so that it doesn't show exhaustively everything that you tried… and it should also be well-documented)

Accounting for image structure: convolutional neural nets (CNNs or convnets)

Convolution



filter



Slide by Phillip Isola

Convolution

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

-1	**-1**	**-1**
2	**2**	**2**
-1	**-1**	**-1**

Filter
(also called “kernel”)

Convolution

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

-1	**-1**	**-1**
2	**2**	**2**
-1	**-1**	**-1**

Filter
(also called “kernel”)

Convolution

Take dot product!

0(-1)+0(-1)+0(-1)+0(2)+0(2)+1(2)+0(-1)+1(-1)+1(-1)

0 -1	0 -1	0 -1	0	0	0	0
0 2	0 2	1 2	1	1	0	0
0 -1	1 -1	1 -1	1	1	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

0

Output image

Convolution

Take dot product!

0(-1)+0(-1)+0(-1)+0(2)+1(2)+1(2)+1(-1)+1(-1)+1(-1)

0	0 -1	0 -1	0 -1	0	0	0
0	0 2	1 2	1 2	1	0	0
0	1 -1	1 -1	1 -1	1	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

0	1

Output image

Convolution

Take dot product!

0	0	0 (-1)	0 (-1)	0 (-1)	0	0
0	0	1 (2)	1 (2)	1 (2)	0	0
0	1	1 (-1)	1 (-1)	1 (-1)	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

0	1	3

Output image

Convolution

Take dot product!

0	0	0	0 (-1)	0 (-1)	0 (-1)	0
0	0	1	1 (2)	1 (2)	0 (2)	0
0	1	1	1 (-1)	1 (-1)	1 (-1)	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

0	1	3	1

Output image

Convolution

Take dot product!

0	0	0	0	0 -1	0 -1	0 -1
0	0	1	1	1 2	0 2	0 2
0	1	1	1	1 -1	1 -1	0 -1
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

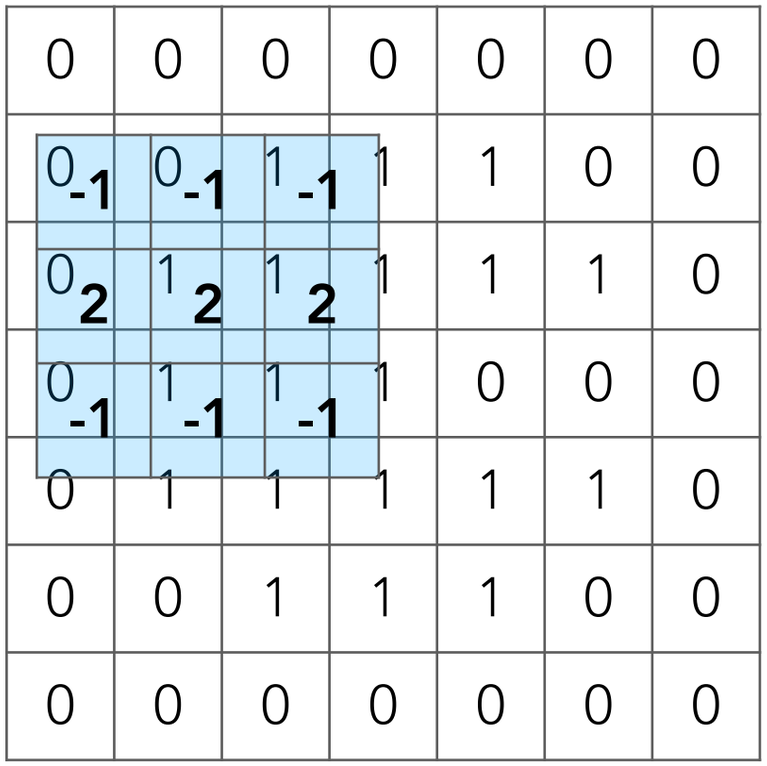
Input image

0	1	3	1	0

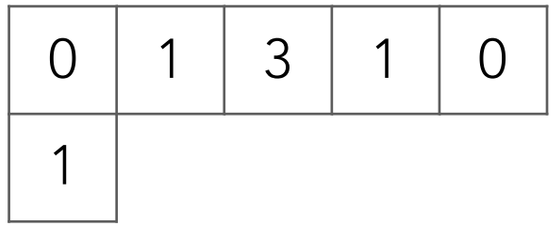
Output image

Convolution

Take dot product!



Input image



Output image

Convolution

Take dot product!

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

0	1	3	1	0
1	1			

Output image

Convolution

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

$*$

-1	**-1**	**-1**
2	**2**	**2**
-1	**-1**	**-1**

$=$

0	1	3	1	0
1	1	1	3	3
0	0	-2	-4	-4
1	1	1	3	3
0	1	3	1	0

Output image

Note: output image is smaller than input image in this case

Convolution

Very commonly used for:

- Blurring an image

*

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

=

- Finding edges

*

-1	-1	-1
2	2	2
-1	-1	-1

=

(this example finds horizontal edges)

Images from: http://aishack.in/tutorials/image-convolution-examples/

Convolution Layer

Activation layer (such as ReLU)

Conv2d layer

convolve with each filter

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

add bias → apply activation

-1	-1	-1
2	2	2
-1	-1	-1

add bias → apply activation

0	-1	0
-1	4	-1
0	-1	0

add bias → apply activation

filters & biases (1 bias number per filter) are unknown and are learned!

Images from: http://aishack.in/tutorials/image-convolution-examples/

Convolution Layer

Input

Conv2d
(3 filters,
each size 3x3),
ReLU activation

Output images

Images from: http://aishack.in/tutorials/image-convolution-examples/

Convolution Layer

Input

shape:
1 (# channels),
height,
width

Conv2d
(3 filters,
each size 3x3),
ReLU activation

Stack output images into a single "output feature map"

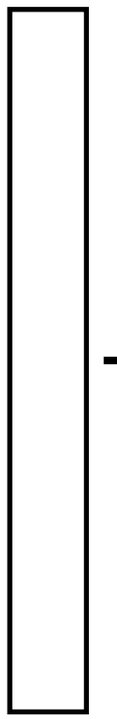
shape:
3,
height-2,
width-2

Images from: http://aishack.in/tutorials/image-convolution-examples/

Convolution Layer

Input

shape:
1 (# channels),
height,
width



Conv2d
(*m* filters,
each size 3x3),
ReLU activation

Stack output images into a single "output feature map"

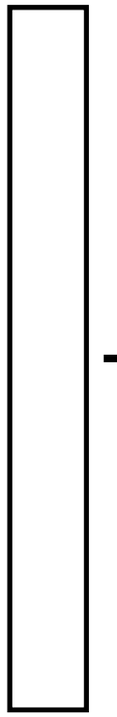
shape:
m,
height-2,
width-2

Images from: http://aishack.in/tutorials/image-convolution-examples/

Convolution Layer

Input

shape:
d (# channels)
height,
width



Conv2d
(*m* filters,
each size *d*x3x3),
ReLU activation

Stack output images into a single "output feature map"

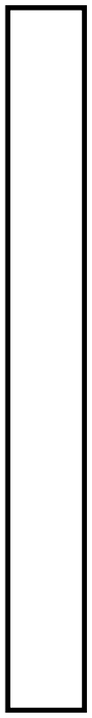
shape:
m,
height-2,
width-2

Images from: http://aishack.in/tutorials/image-convolution-examples/

Convolution Layer

Input

shape:
d (# channels)
height,
width

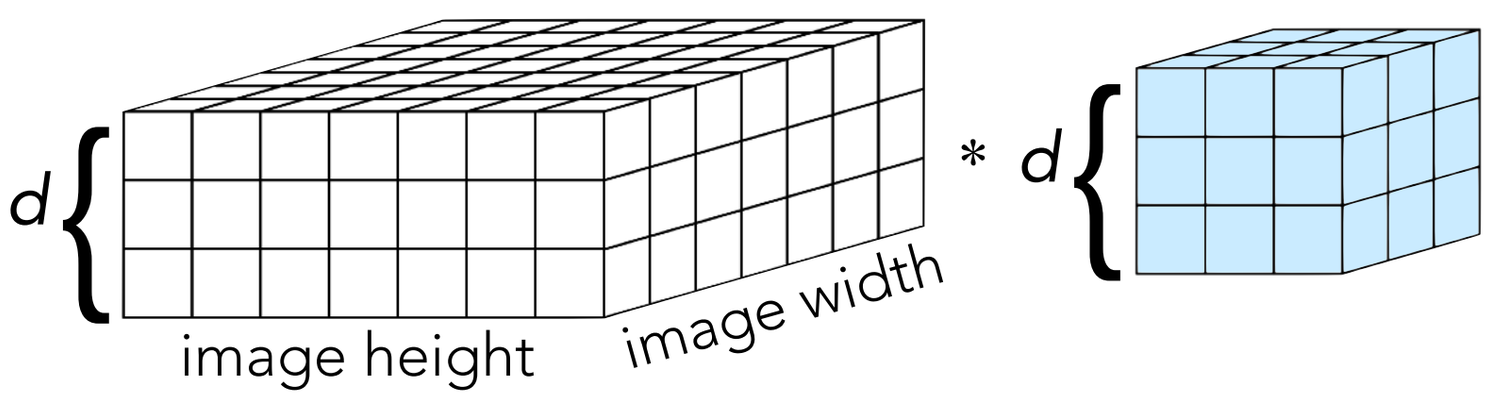


Conv2d
(*m* filters,
each size *d*x3x3),
ReLU activation

Stack output images into a single "output feature map"

shape:
m,
height-2,
width-2

Each filter:



Pooling

- Produces smaller image summarizing original larger image

- To produce this smaller image, need to aggregate or “pool” together information

Max Pooling

Called “2-by-2” max pooling since this green box is 2 rows by 2 columns

Take maximum value

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

Output image

3-by-4 max pooling would mean that the green box is 3 rows by 4 columns, etc

Max Pooling

Called “2-by-2” max pooling since this green box is 2 rows by 2 columns

Take maximum value

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

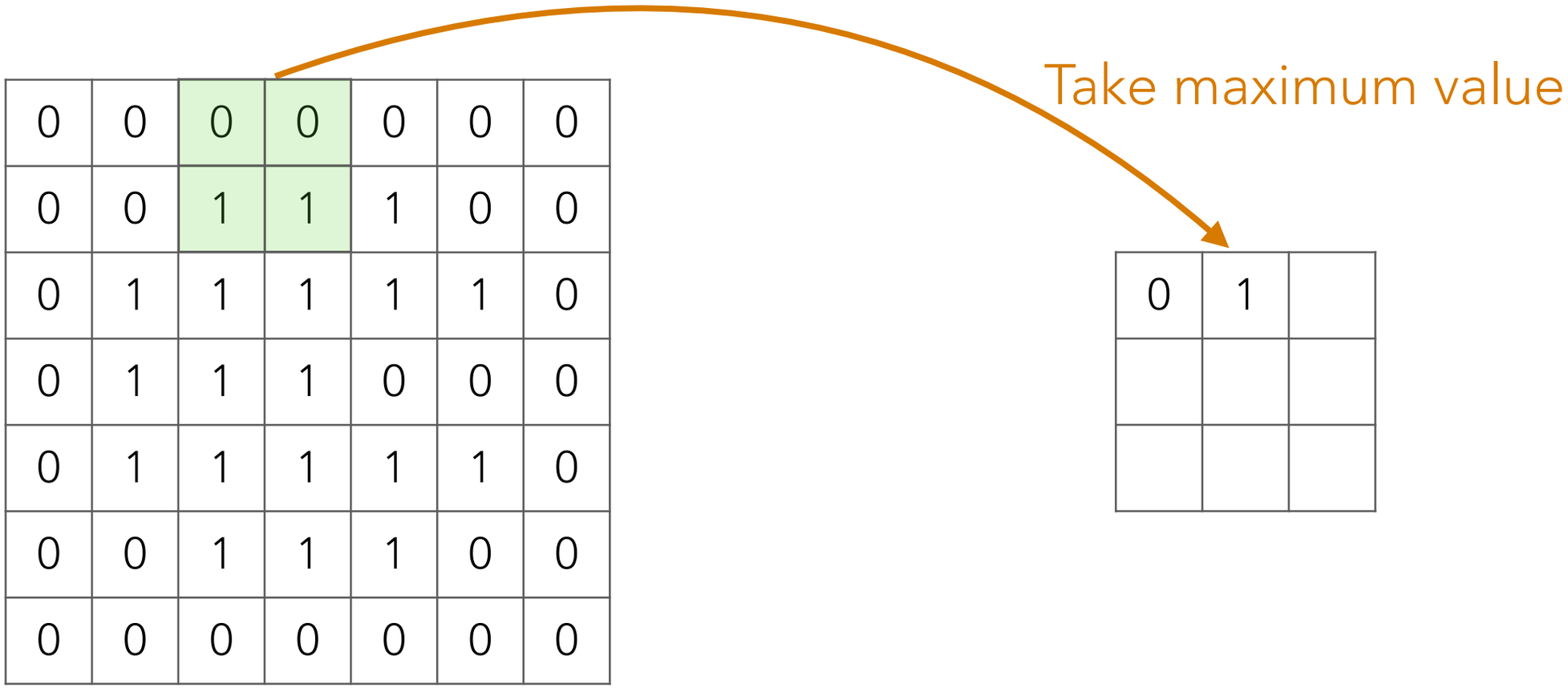
0		

Output image

3-by-4 max pooling would mean that the green box is 3 rows by 4 columns, etc

Max Pooling

Called “2-by-2” max pooling since this green box is 2 rows by 2 columns



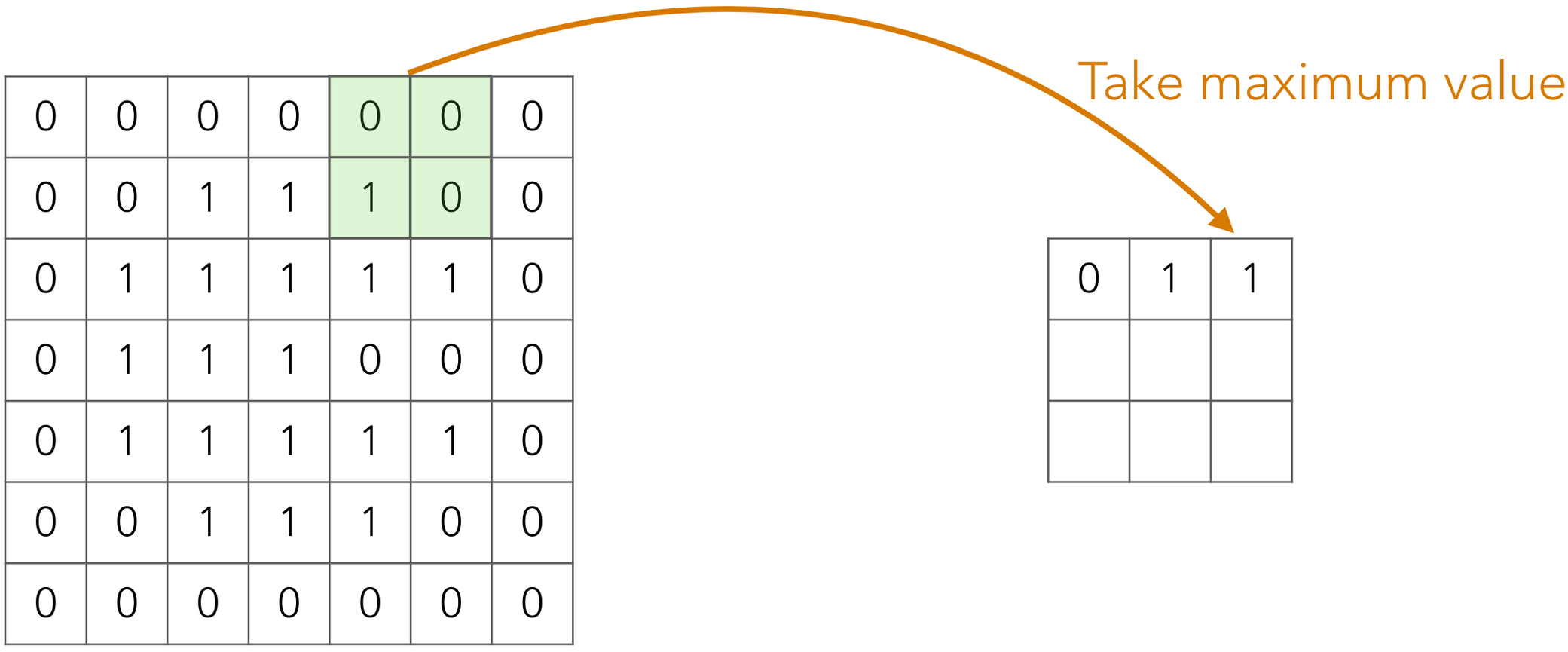
Input image

Output image

3-by-4 max pooling would mean that the green box is 3 rows by 4 columns, etc

Max Pooling

Called “2-by-2” max pooling since this green box is 2 rows by 2 columns



Input image

Output image

3-by-4 max pooling would mean that the green box is 3 rows by 4 columns, etc

Max Pooling

Called “2-by-2” max pooling since this green box is 2 rows by 2 columns

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input image

Take maximum value

0	1	1
1		

Output image

3-by-4 max pooling would mean that the green box is 3 rows by 4 columns, etc

Max Pooling

Called “2-by-2” max pooling since this green box is 2 rows by 2 columns

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

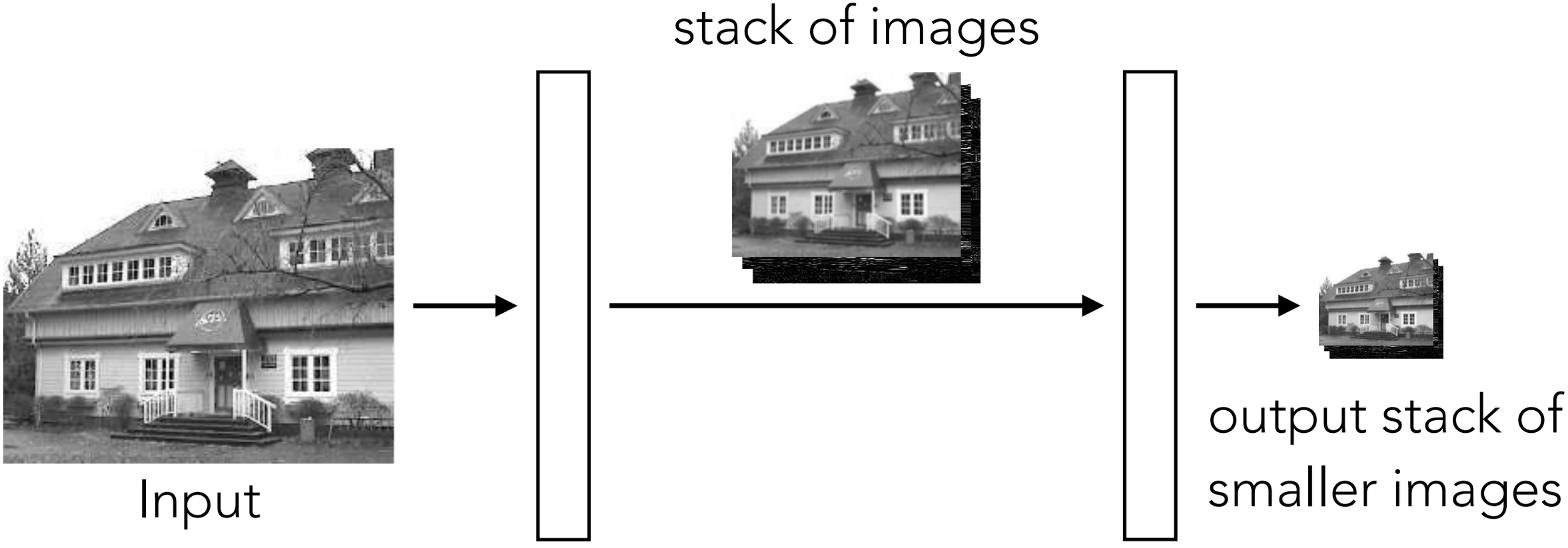
Input image

0	1	1
1	1	1
1	1	1

Output image

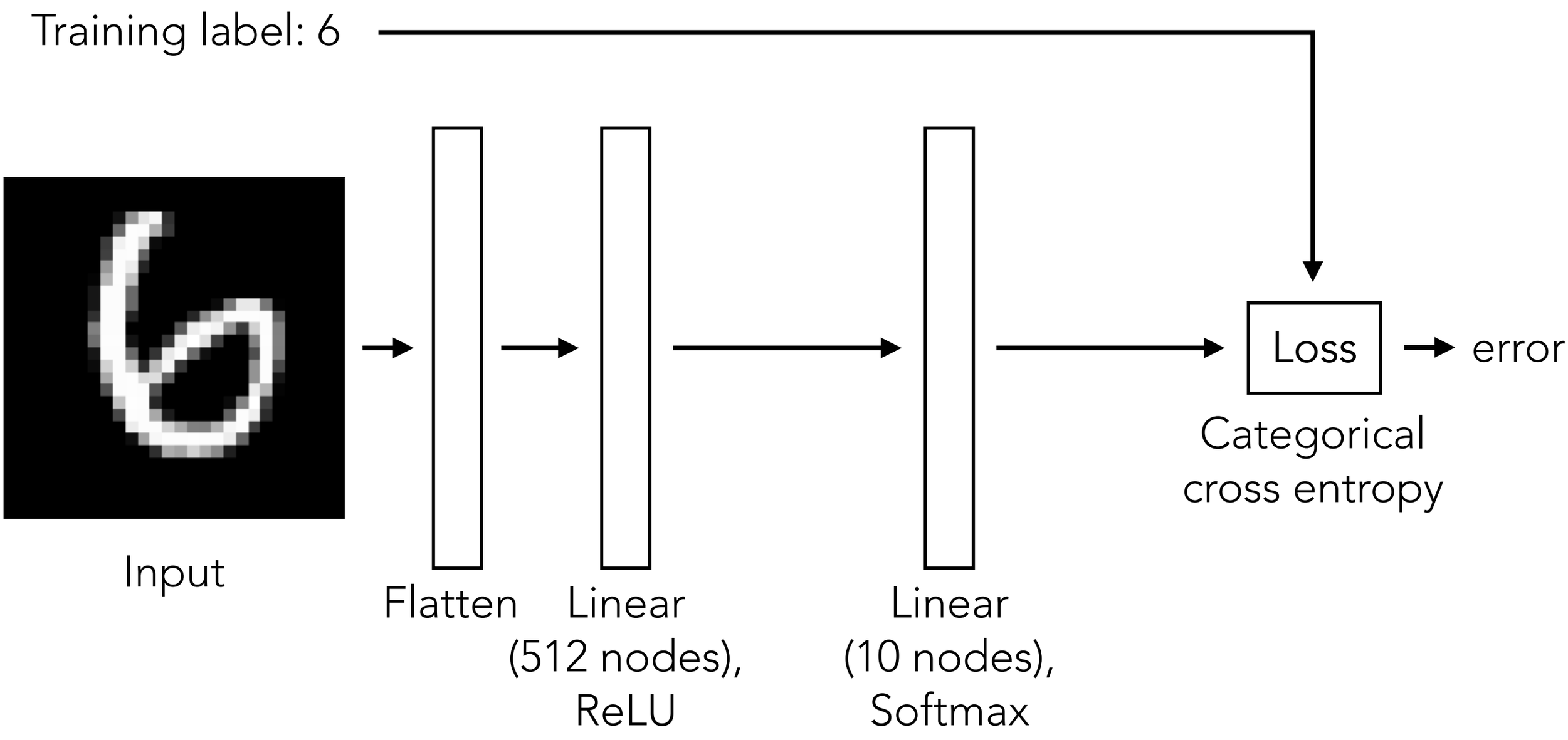
3-by-4 max pooling would mean that the green box is 3 rows by 4 columns, etc

Common Building Block of CNNs

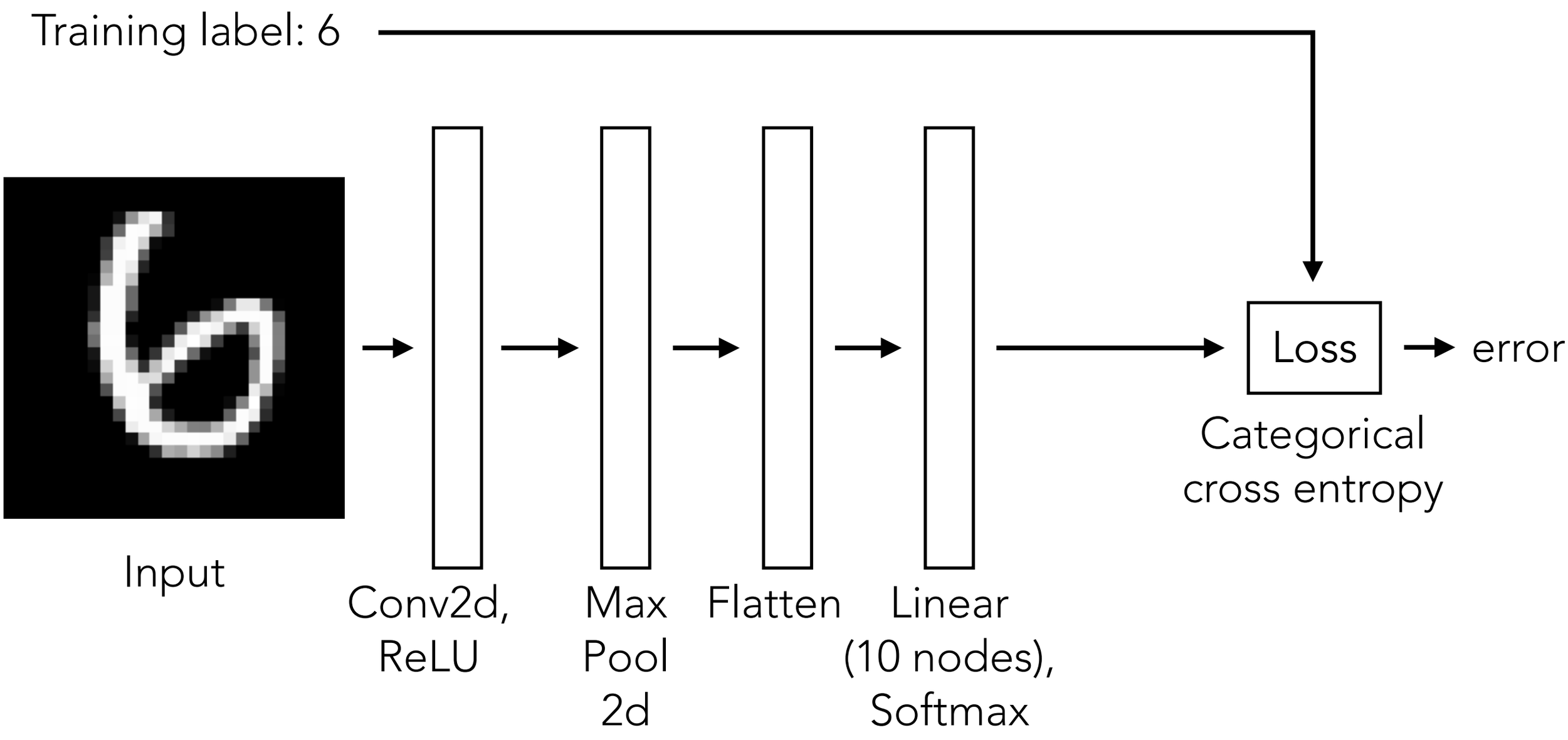


Images from: http://aishack.in/tutorials/image-convolution-examples/

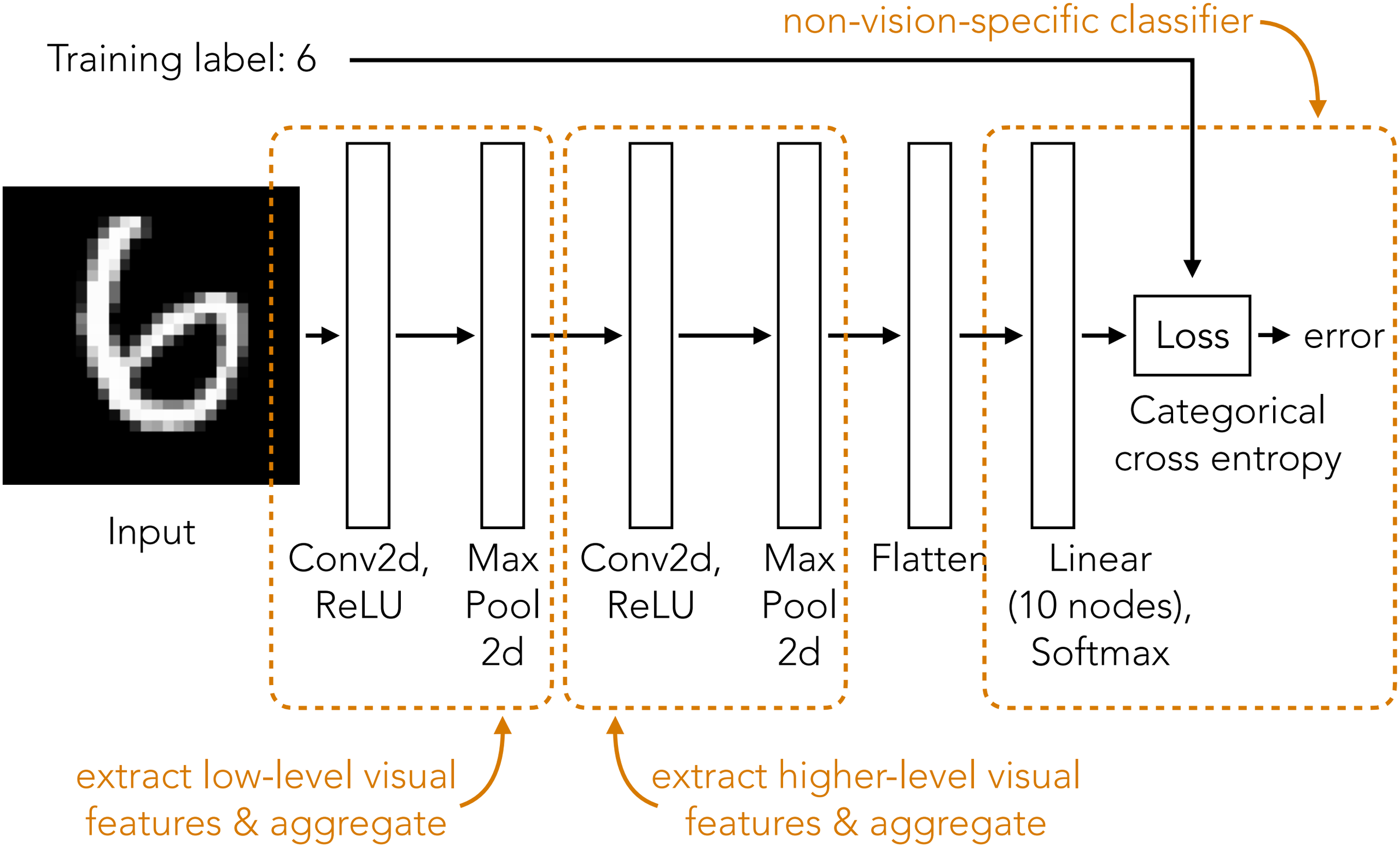
Handwritten Digit Recognition



Handwritten Digit Recognition



Handwritten Digit Recognition



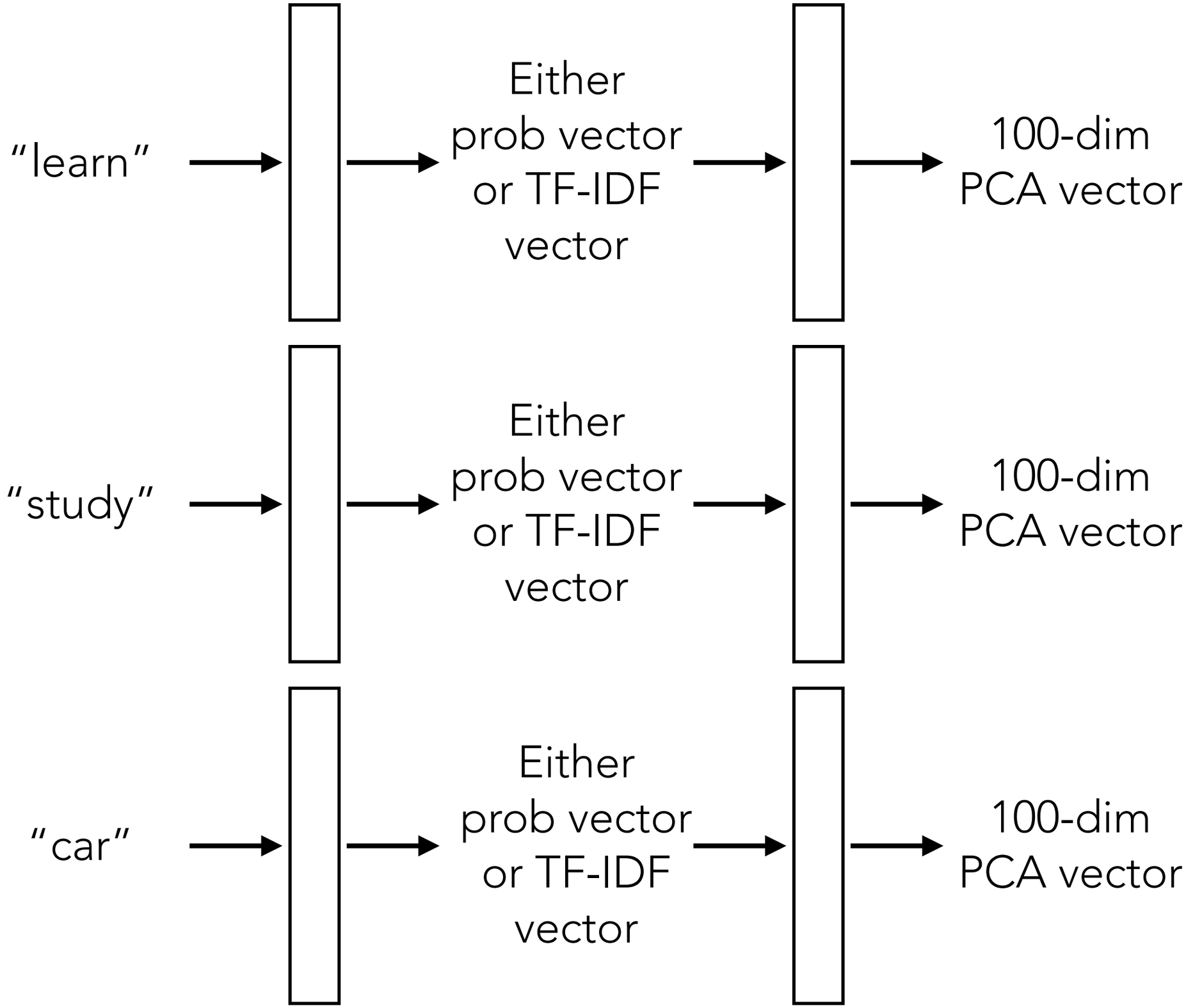
CNNs

Demo

A brief glimpse at word embeddings

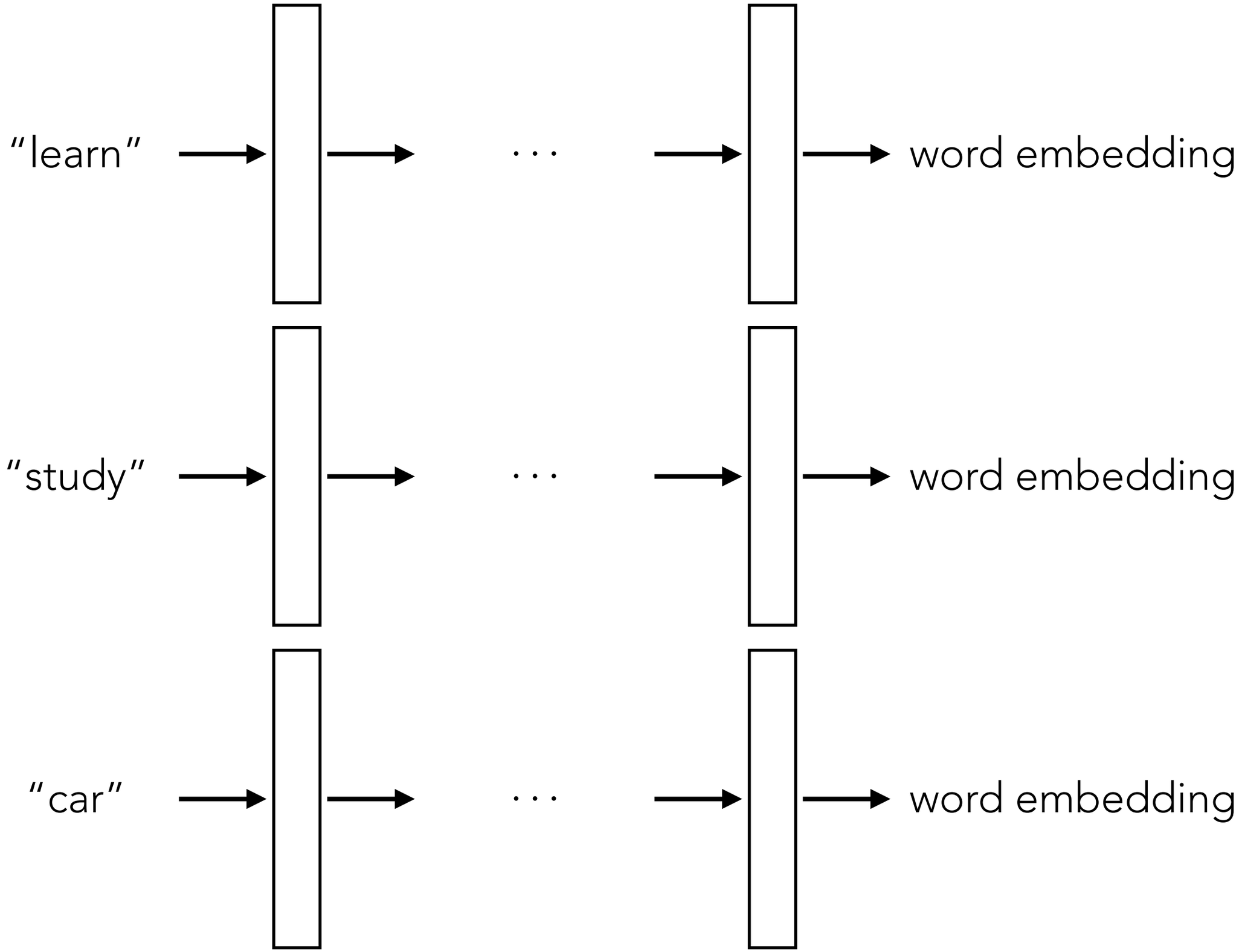
We came up with TF representation
and then normalized into prob vector
or converted to TF-IDF vector

PCA
(e.g., 100-dim)



Tokens/words

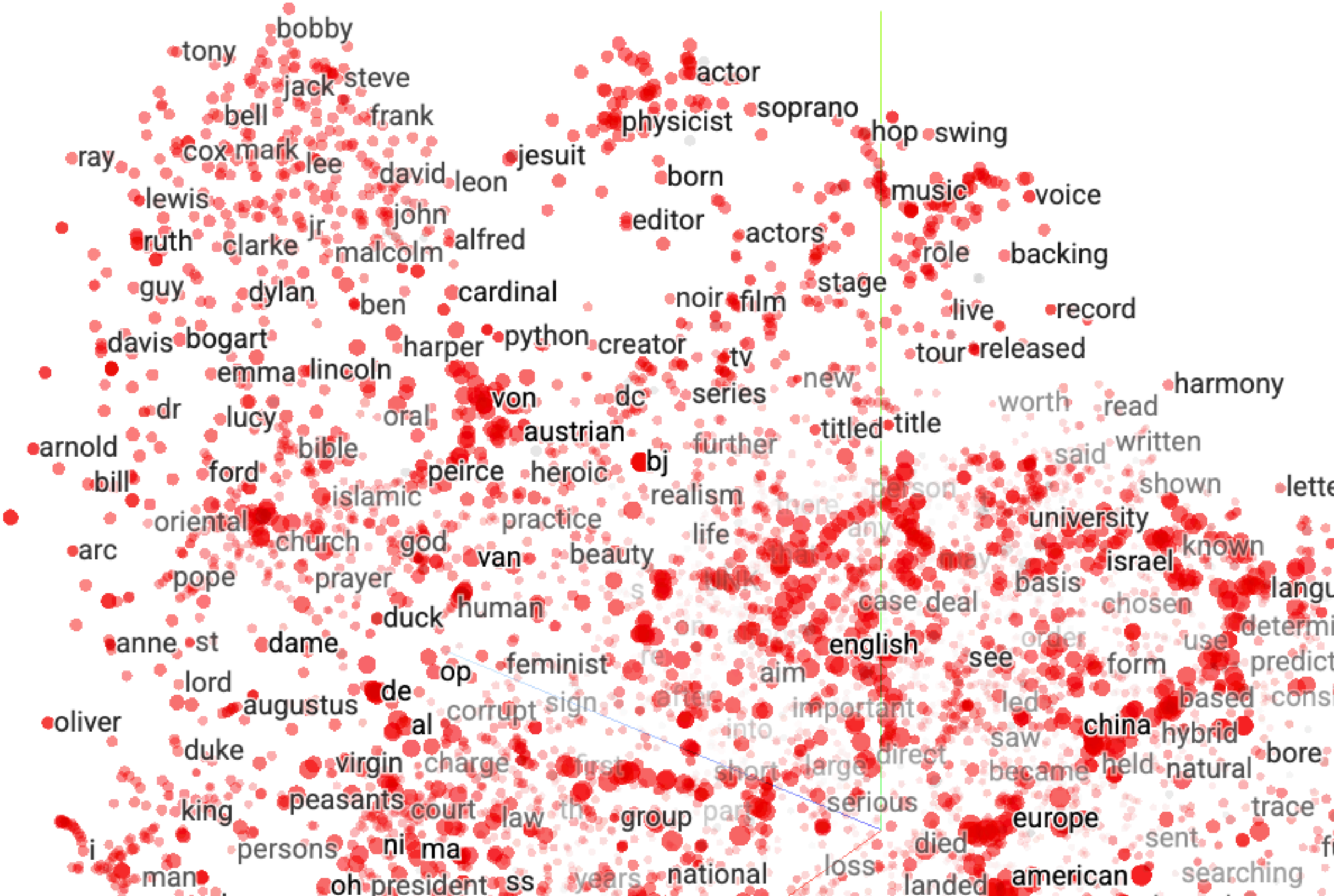
Neural net model



(Flashback) Do Data Actually Live on Manifolds?

Image source: projector.tensorflow.org

(Flashback) Do Data Actually Live on Manifolds?



Word Embeddings:
Even without labels, we can set up a prediction problem!

Hide part of training data and try to predict what you've hid!

Word Embeddings: word2vec (2013)

Can solve tasks like the following:

Man is to King as Woman is to ???

Word Embeddings: word2vec (2013)

Can solve tasks like the following:

Man is to King as Woman is to Queen

Word Embeddings: word2vec (2013)

Can solve tasks like the following:

Man is to King as Woman is to Queen

Which word doesn't belong?
blue, red, green, crimson, transparent

Word Embeddings: word2vec (2013)

Can solve tasks like the following:

Man is to King as Woman is to Queen

Which word doesn't belong?

blue, red, green, crimson, transparent

Word Embeddings: word2vec (2013)

The opioid epidemic or opioid crisis is the rapid increase in the use of prescription and non-prescription opioid drugs in the United States and Canada in the 2010s.

Predict context of each word!

Training data point: epidemic

“Training labels”: the, opioid, or, opioid

Word Embeddings: word2vec (2013)

Predict context of each word!

Training data point: or

"Training labels": opioid, epidemic, opioid, crisis

Word Embeddings: word2vec (2013)

Predict context of each word!

Training data point: opioid

"Training labels": epidemic, or, crisis, is

These are "positive" (correct) examples of what context words are for "opioid"

Also provide "negative" examples of words that are *not* likely to be context words (by randomly sampling words elsewhere in document)

Word Embeddings: word2vec (2013)

randomly sampled word

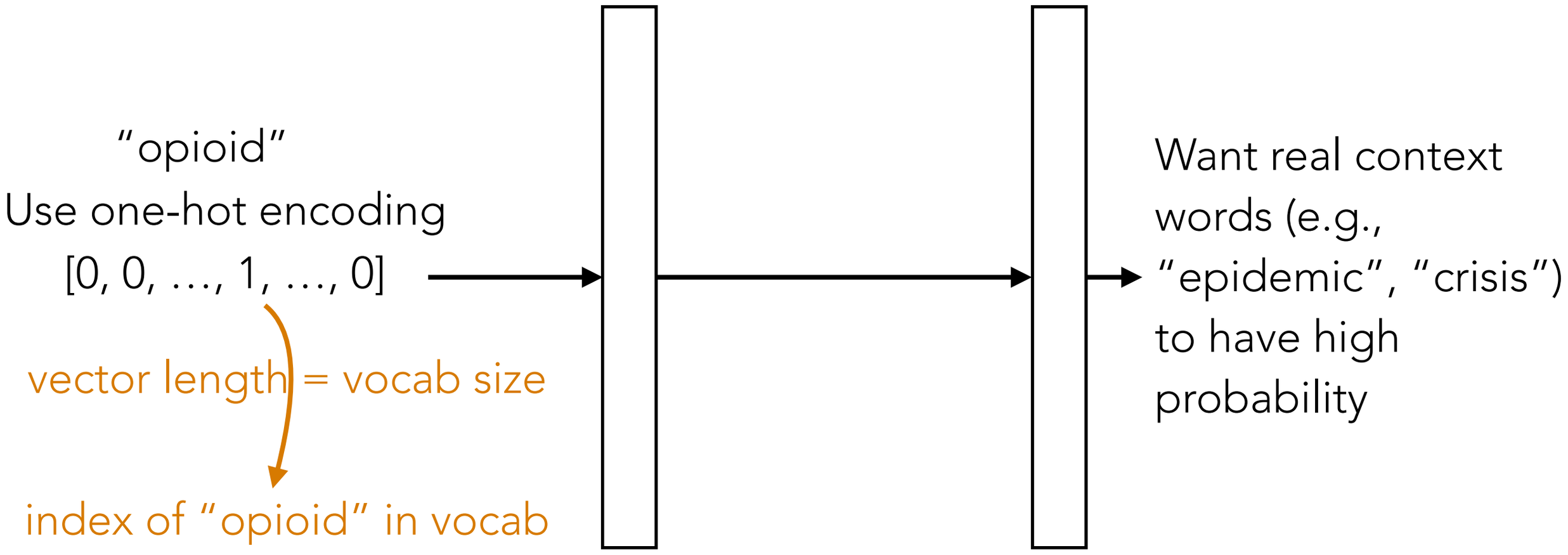
Predict context of each word!

Training data point: opioid

"Negative training label": 2010s

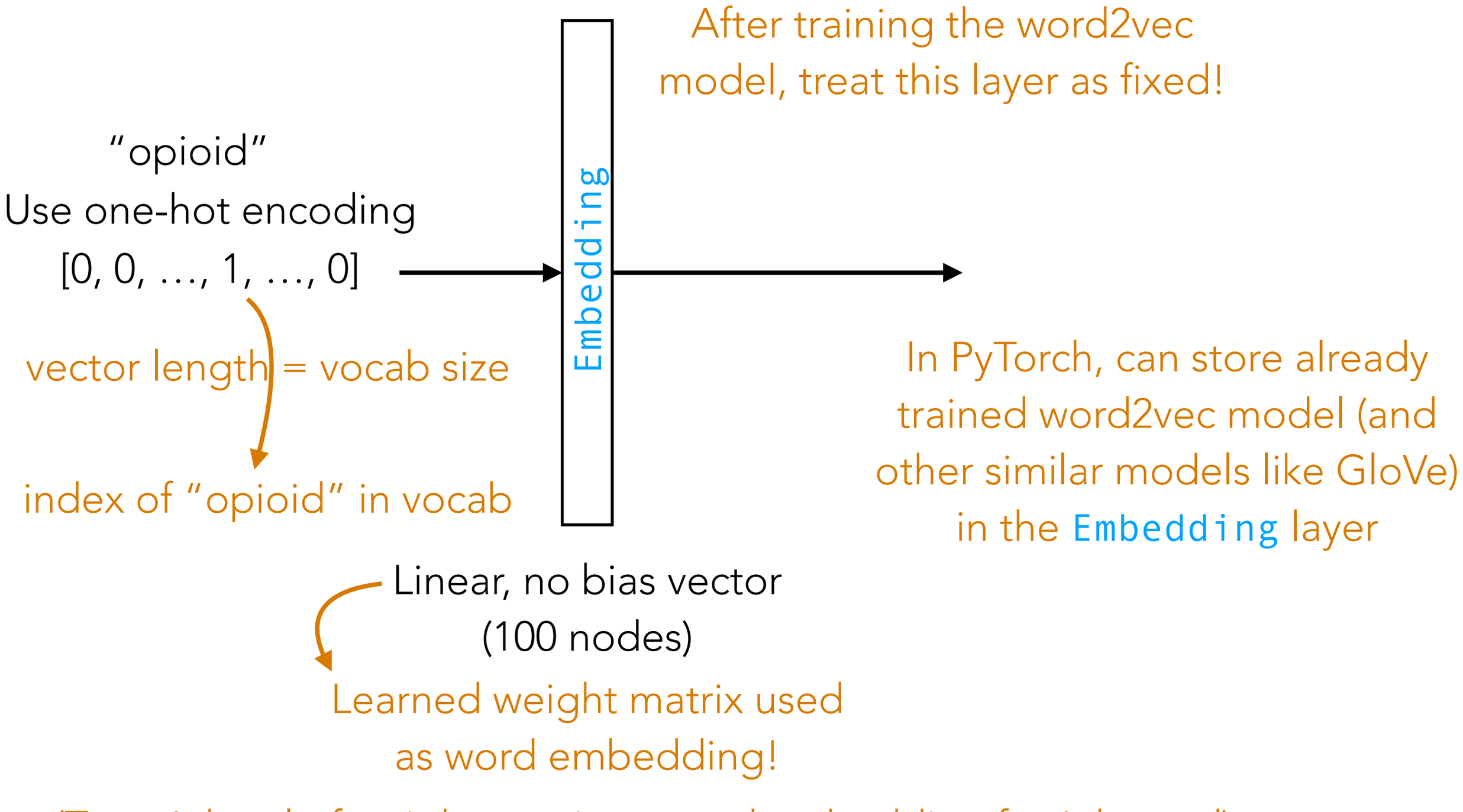
Also provide "negative" examples of words that are *not* likely to be context words (by randomly sampling words elsewhere in document)

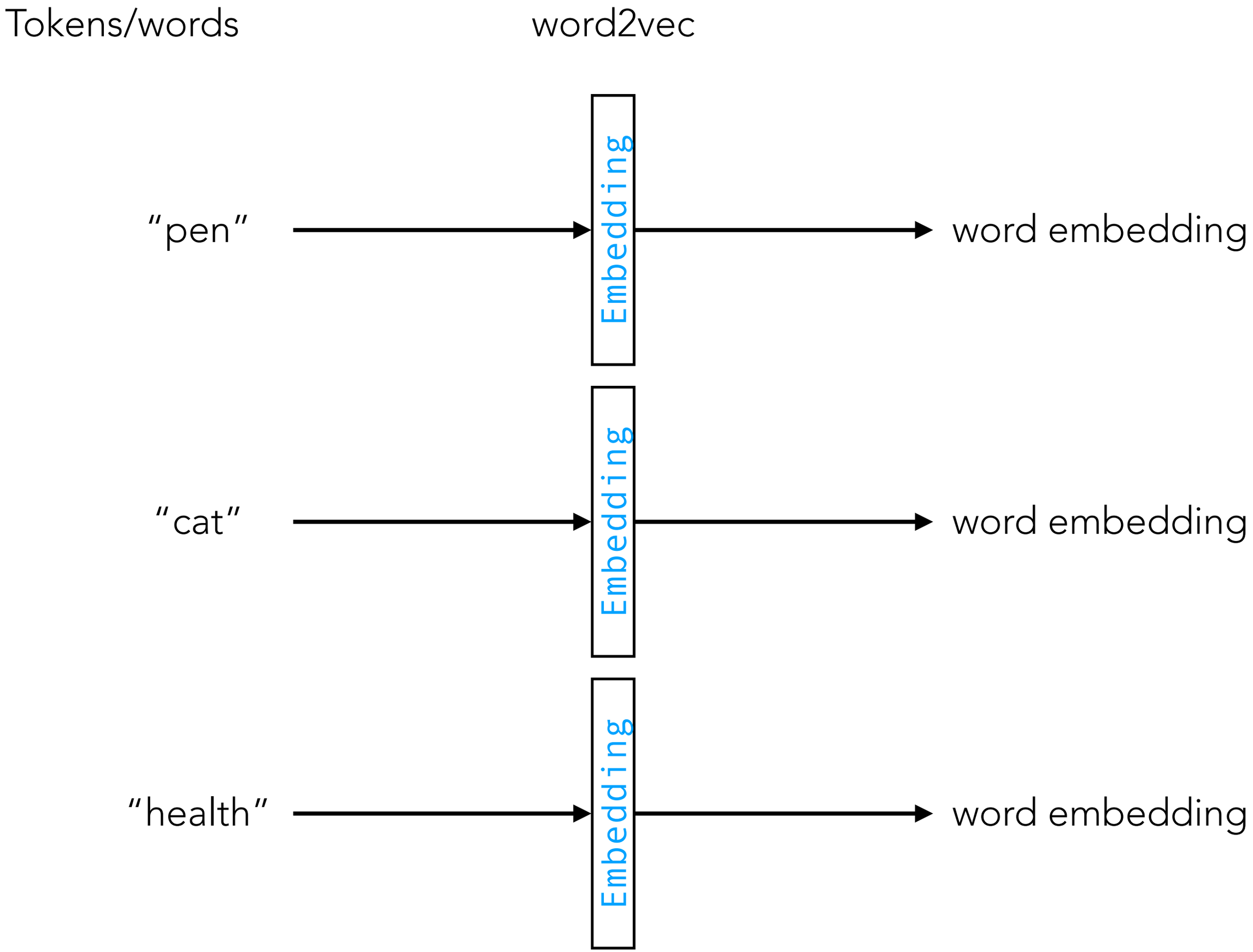
Word2vec Neural Net



(Treat *i*-th col of weight matrix as word embedding for *i*-th word)

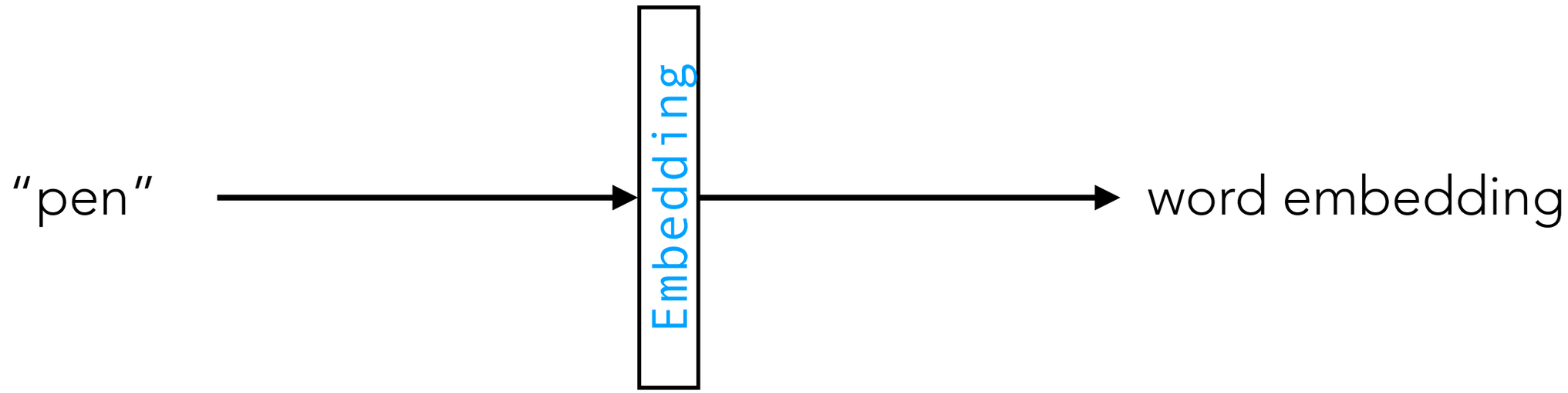
Word2vec Neural Net




Tokens/words
word2vec
“pen”
Embedding
word embedding
“cat”
Embedding
word embedding
“health”
Embedding
word embedding

Tokens/words

word2vec



Even though "pen" has multiple meanings
(e.g., what you write with vs a play pen),
word2vec would produce the same word embedding for "pen"

(Flashback)

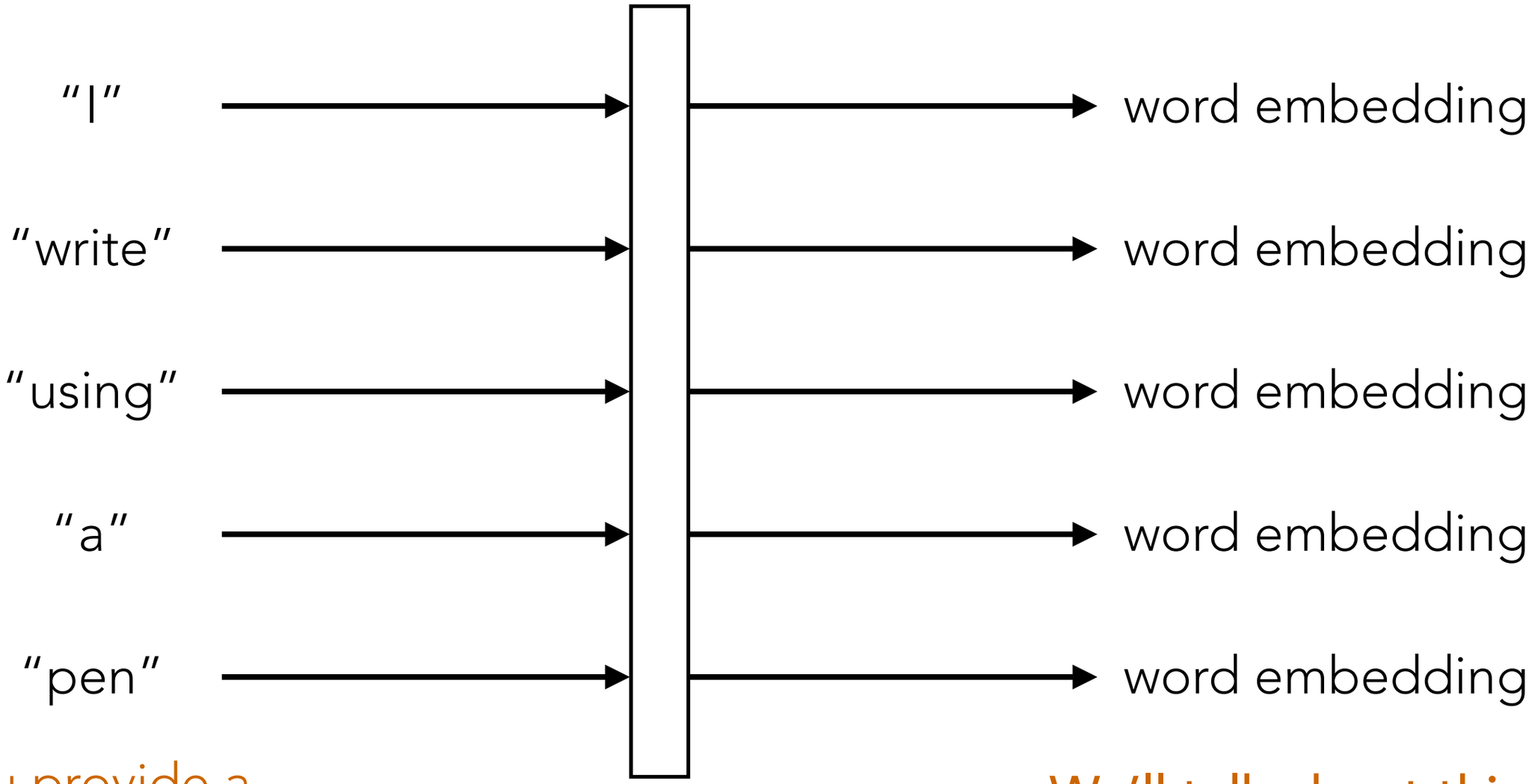
What about a word that has multiple meanings?

Challenging: try to split up word into multiple words depending on meaning (requires inferring meaning from context)

This problem is called word sense disambiguation (WSD)

Modern Word Embeddings Use Context

(such as BERT, which came out in 2018)



You provide a whole sentence (or a longer document)

More complicated neural net

We'll talk about this sort of idea more next week

This Friday's recitation goes over how to use this technology!