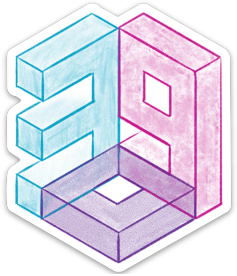


https://introml.mit.edu/

6.390 Intro to Machine Learning

Lecture 8: Representation Learning

Shen Shen

April 4, 2025

11am, Room 10-250

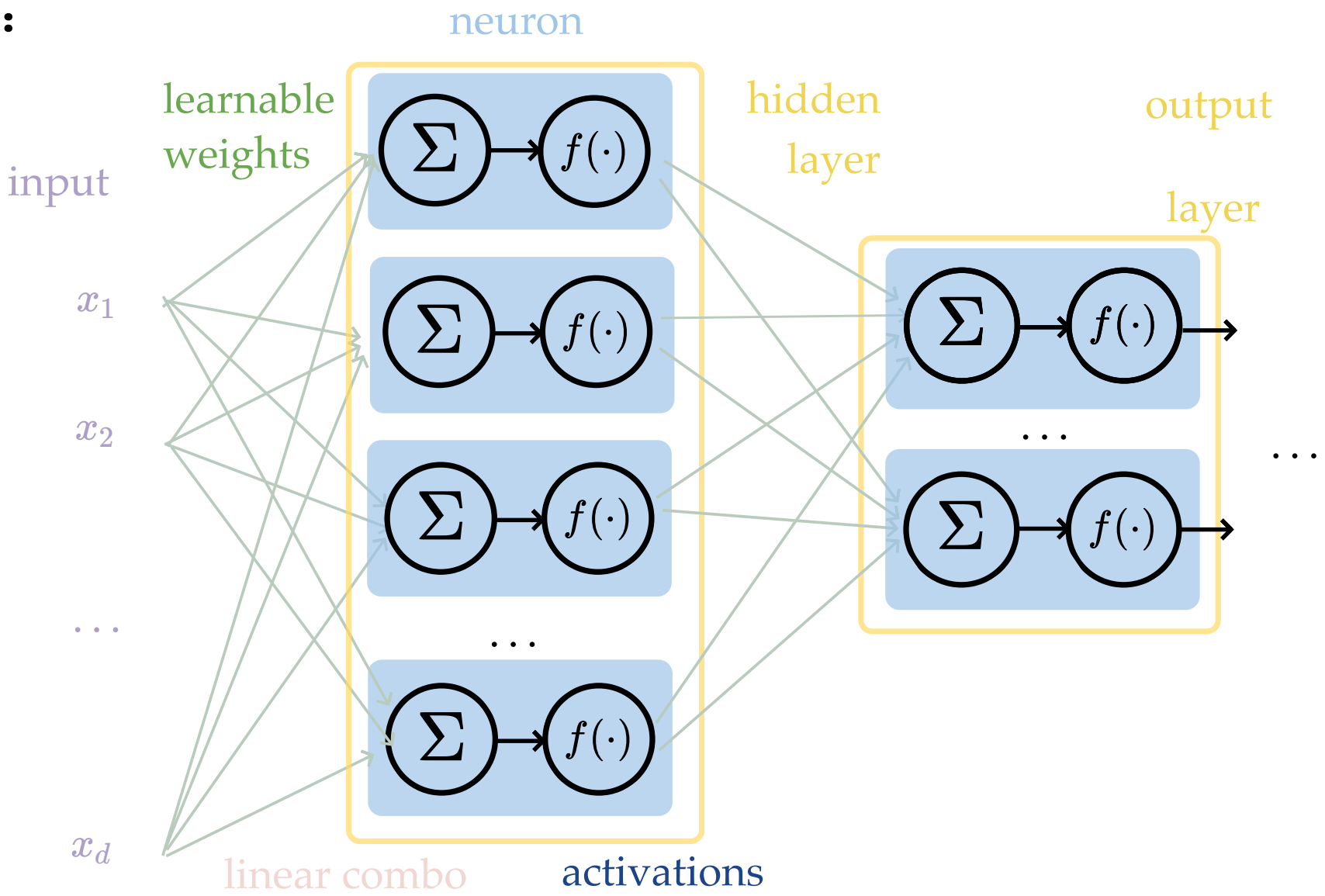
Outline

- Neural networks are *representation* learners
- Auto-encoders
- Unsupervised and self-supervised learning
- Word embeddings
- (Some recent representation learning ideas)

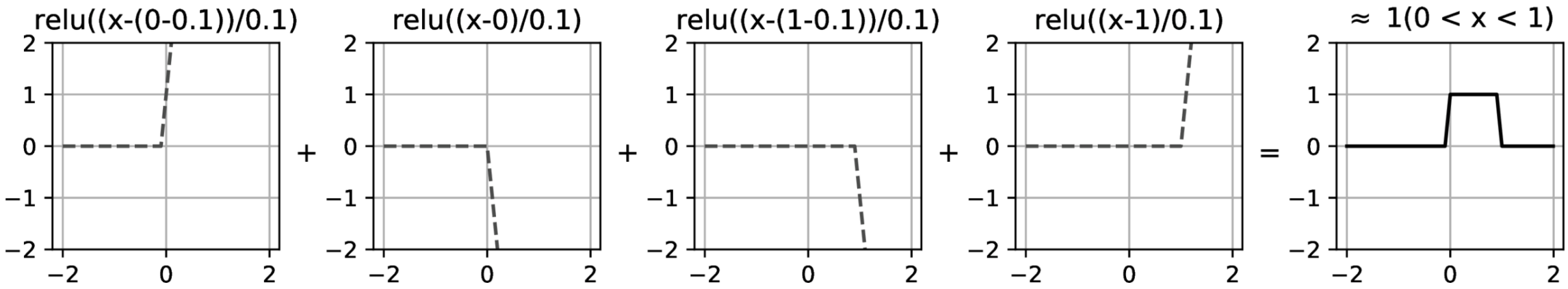
Outline

- Neural networks are *representation* learners
- Auto-encoders
- Unsupervised and self-supervised learning
- Word embeddings
- (Some recent representation learning ideas)

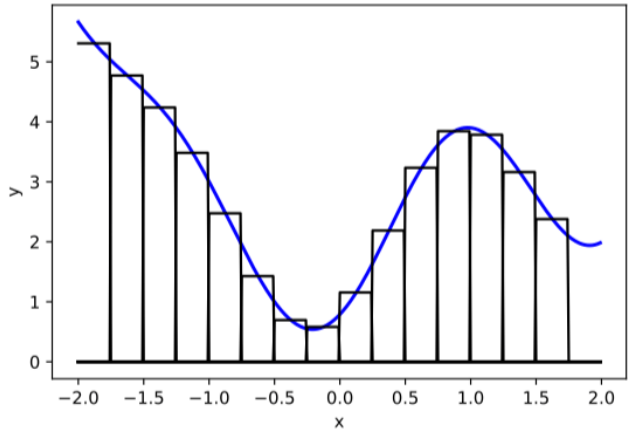
Recap:


neuron
learnable
weights
input
hidden
layer
output
layer
x_1
x_2
...
x_d
Σ
$f(\cdot)$
...
linear combo
activations

compositions of ReLU(s) can be quite expressive



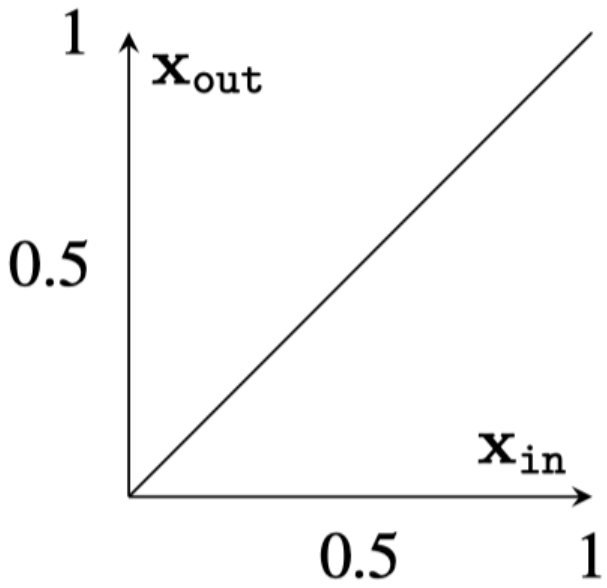
in fact, asymptotically, can approximate any function!



[image credit: Phillip Isola]

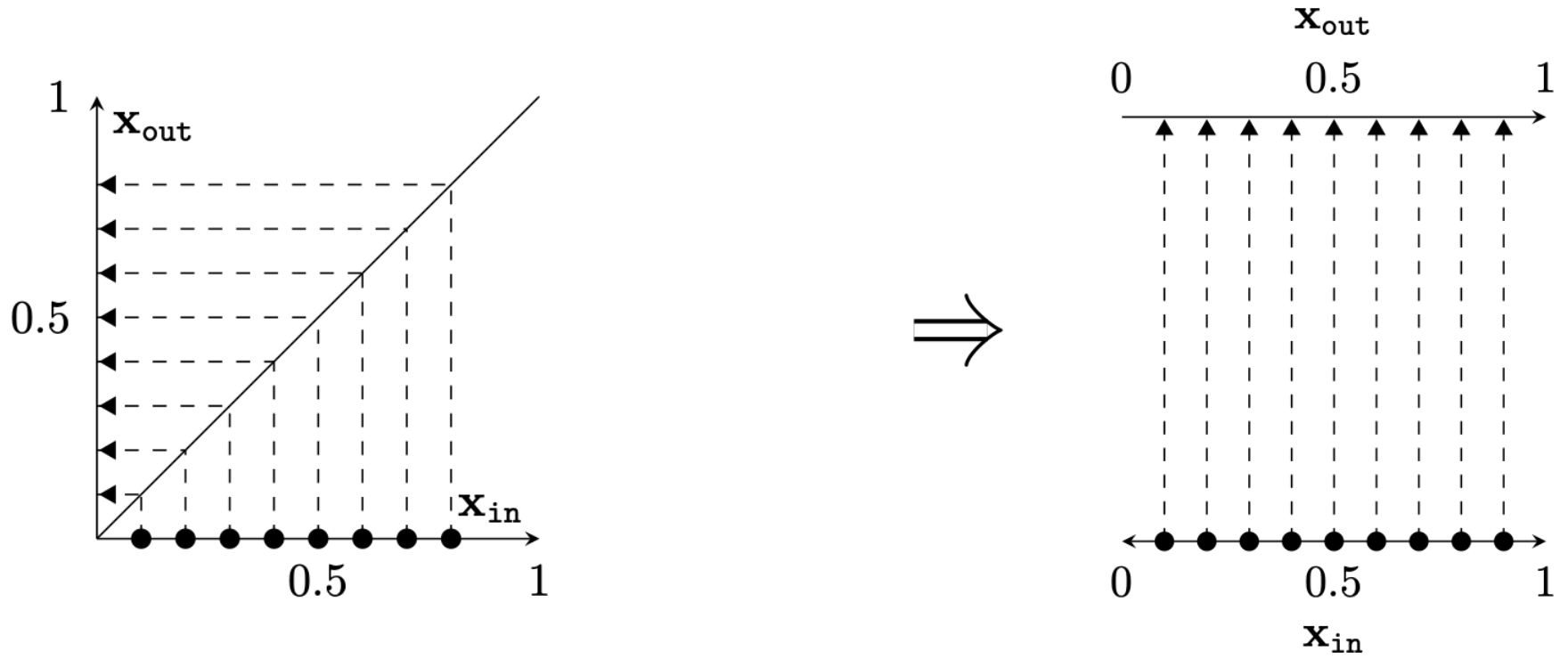
https://playground.tensorflow.org/

Two different ways to visualize a function



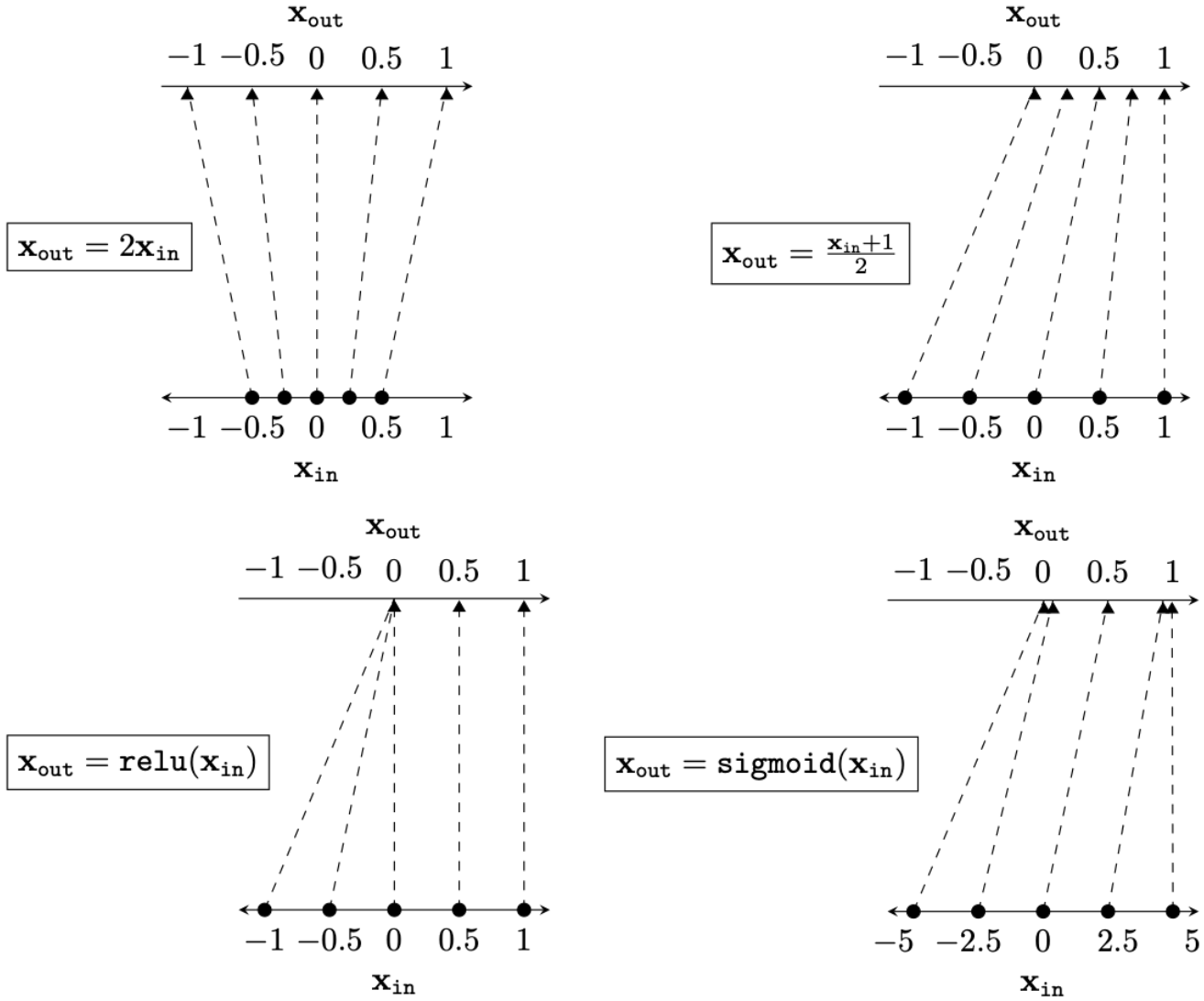
[images credit: visionbook.mit.edu]

Two different ways to visualize a function



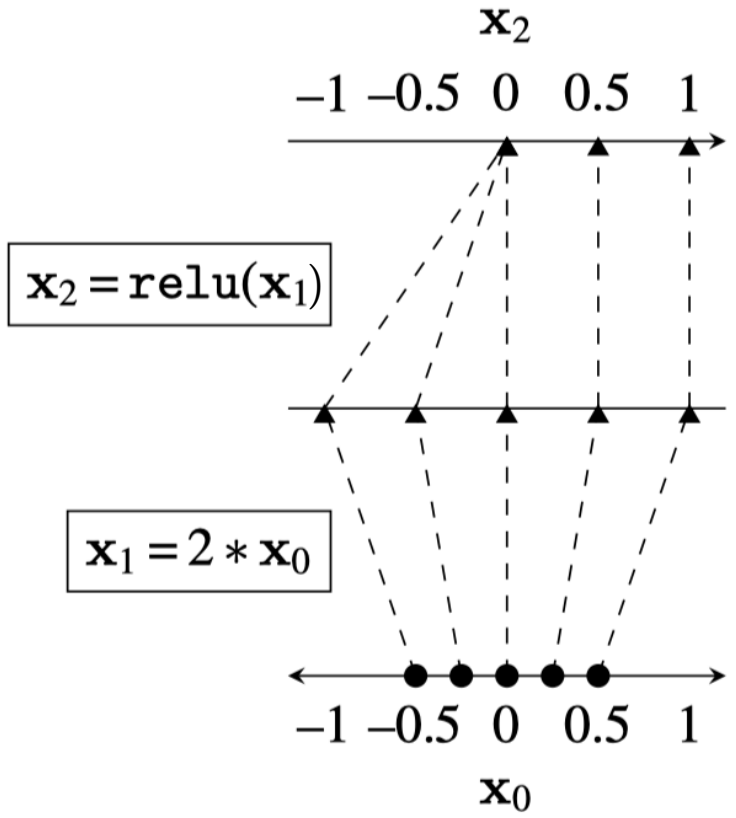
[images credit: visionbook.mit.edu]

Representation transformations for a variety of neural net operations



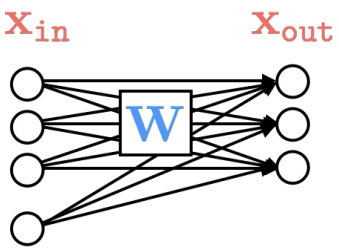
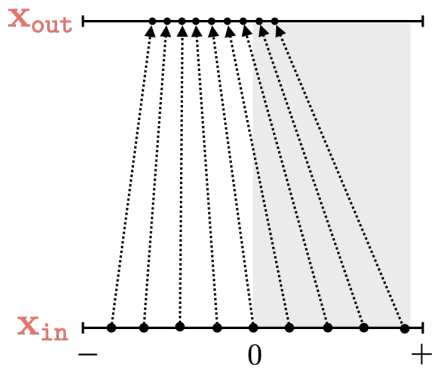
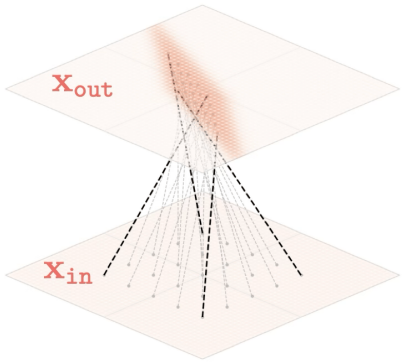
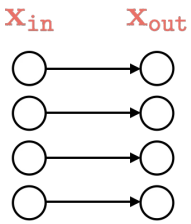
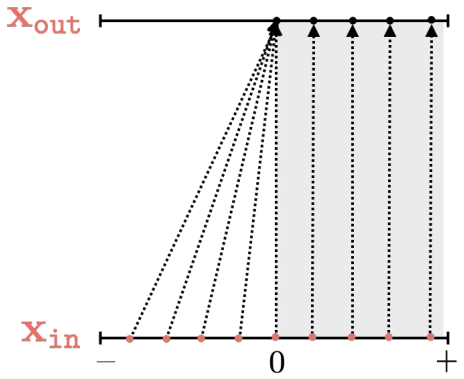
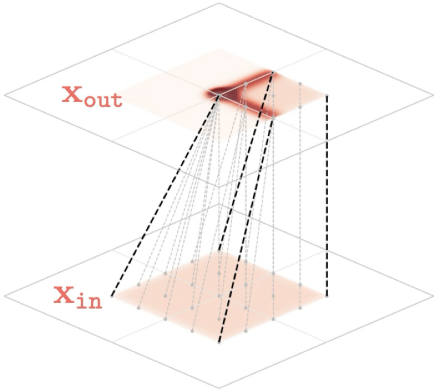
[images credit: visionbook.mit.edu]

and stack of neural net operations

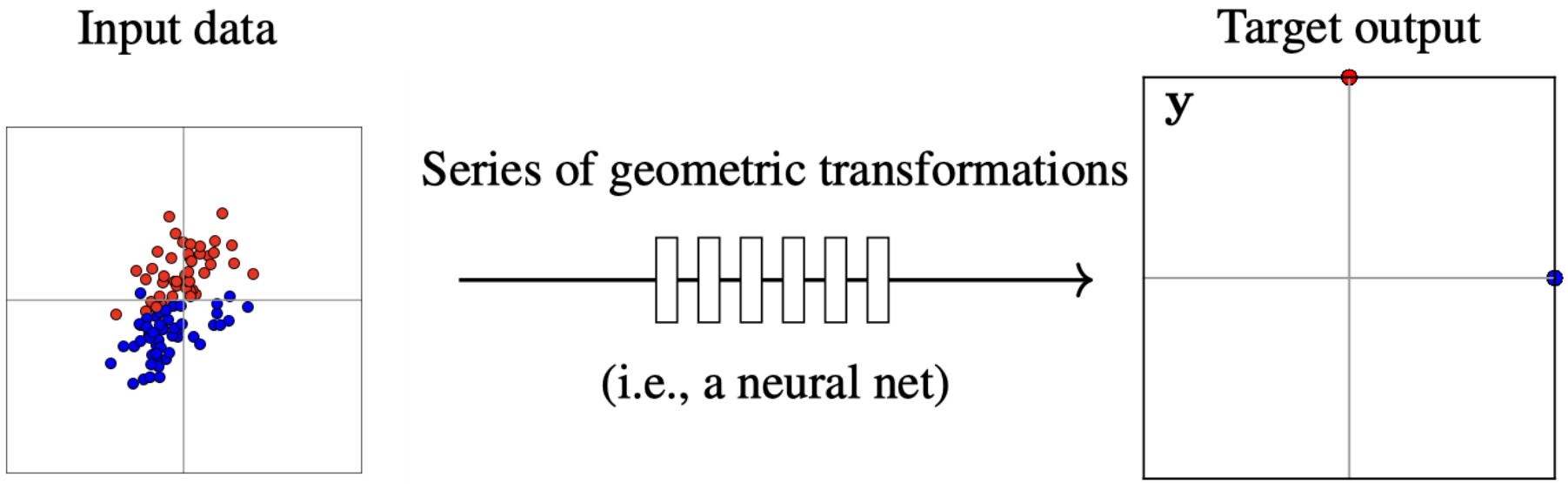


[images credit: visionbook.mit.edu]

Parameters

	wiring graph	equation	mapping 1D	mapping 2D
linear		$\mathbf{x}_{\text{out}} = \mathbf{W}\mathbf{x}_{\text{in}}$		
relu		$x_{\text{out}_i} = \max(x_{\text{in}_i}, 0)$		

[images credit: visionbook.mit.edu]



[images credit: visionbook.mit.edu]

$$g = \text{softmax}(z_2)$$

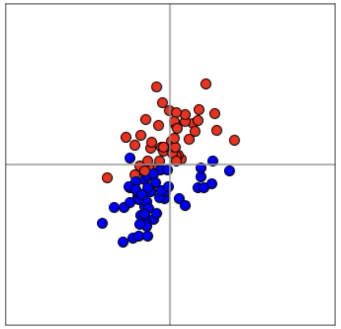
$$z_2 = \text{linear}\,(a_1)$$

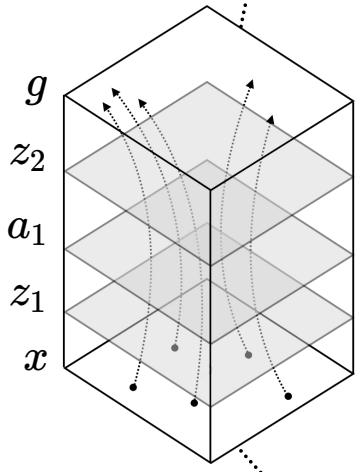
$$a_1 = \text{ReLU}(z_1)$$

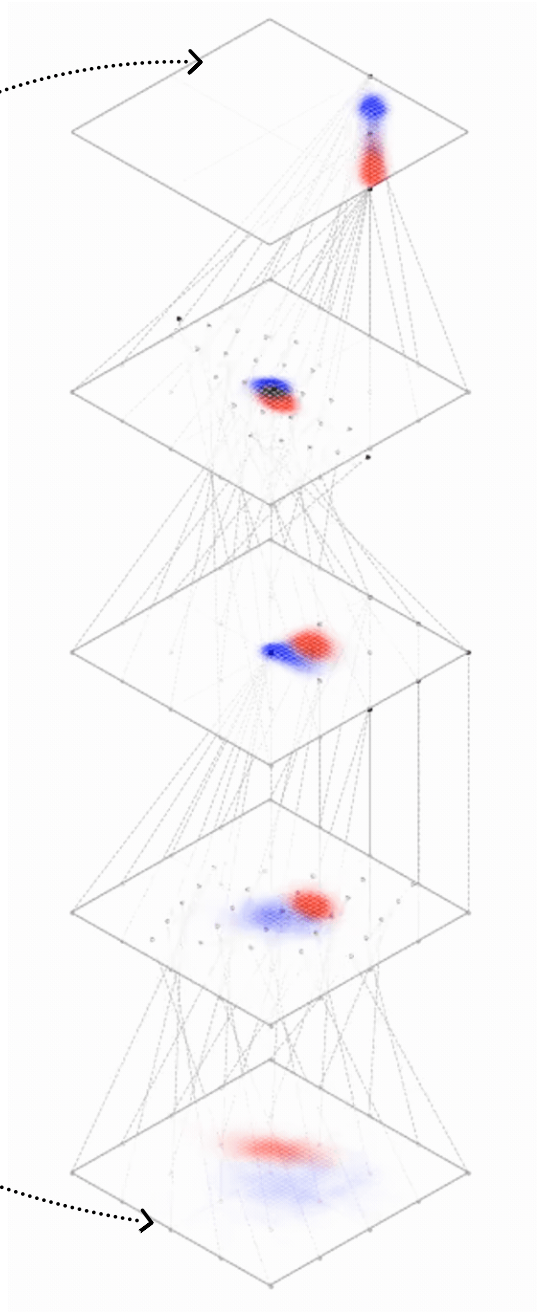
$$z_1 = \text{linear}\,(x)$$

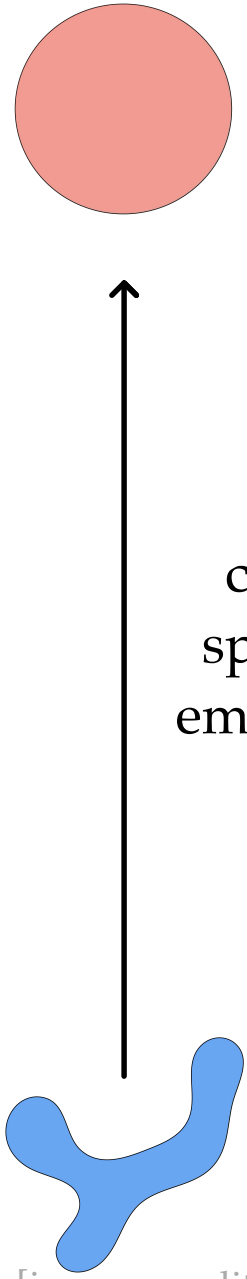
$$x \in \mathbb{R}^2$$

Training data



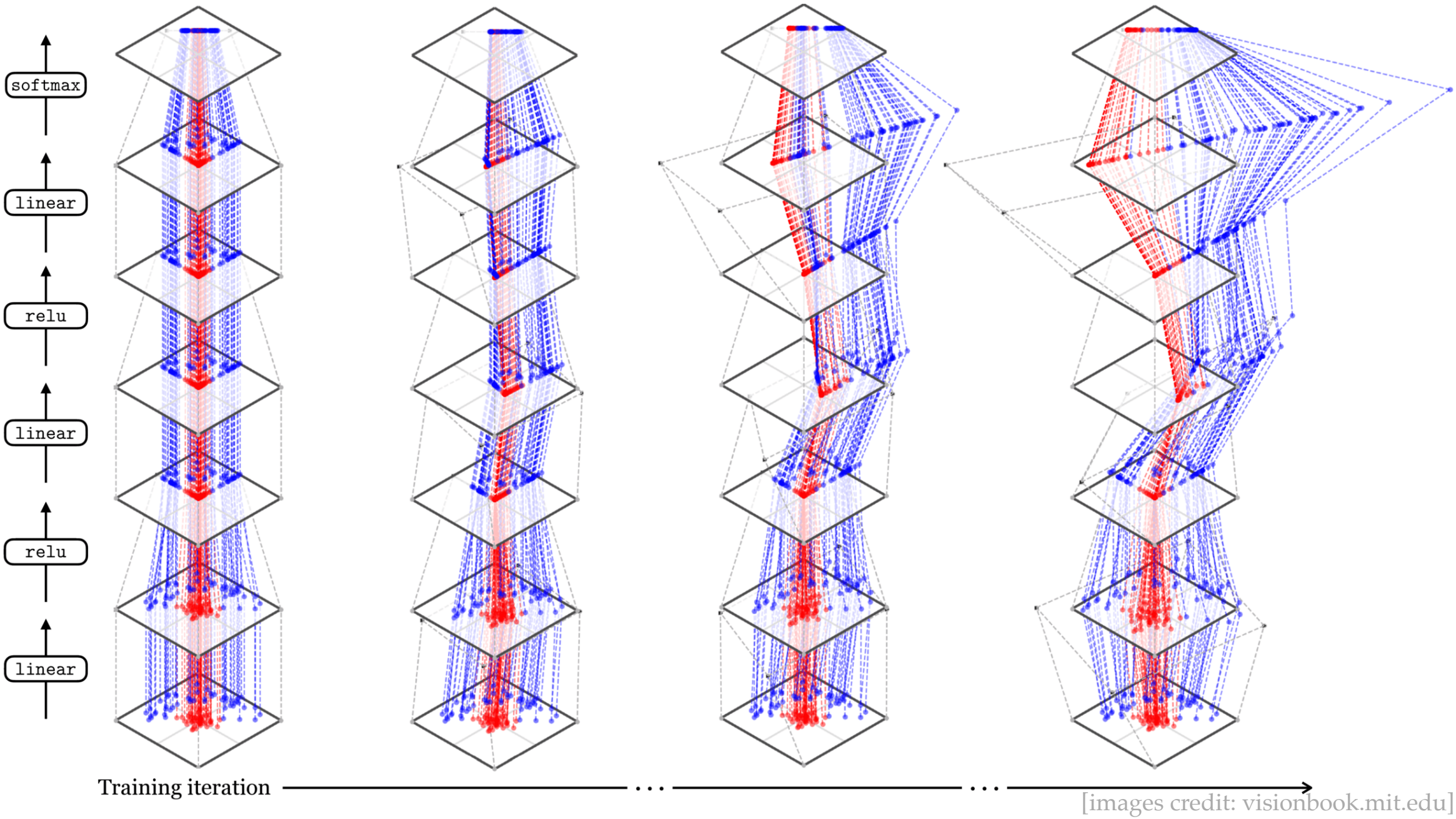






maps from complex data space to simple embedding space

[images credit: visionbook.mit.edu]

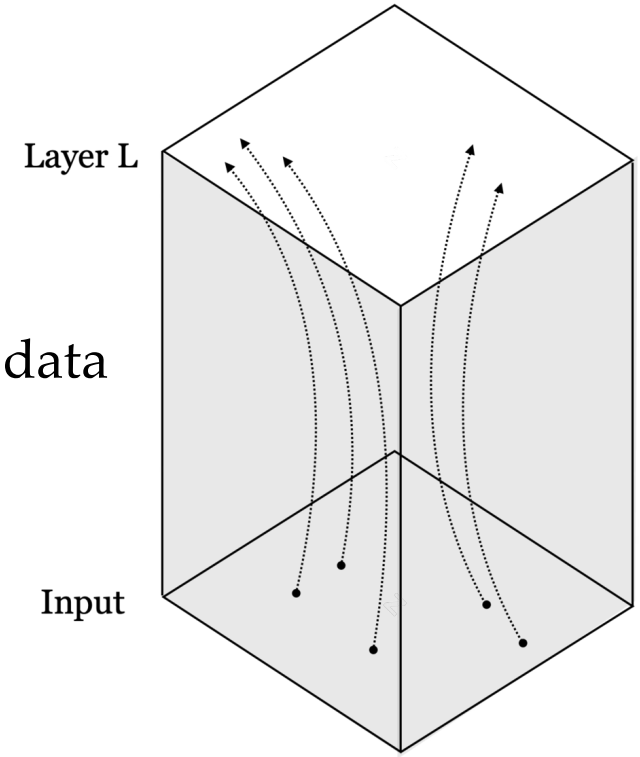


[images credit: visionbook.mit.edu]

Neural networks are representation learners

Deep nets transform datapoints, layer by layer

Each layer gives a different *representation* (aka *embedding*) of the data



[images credit: visionbook.mit.edu]

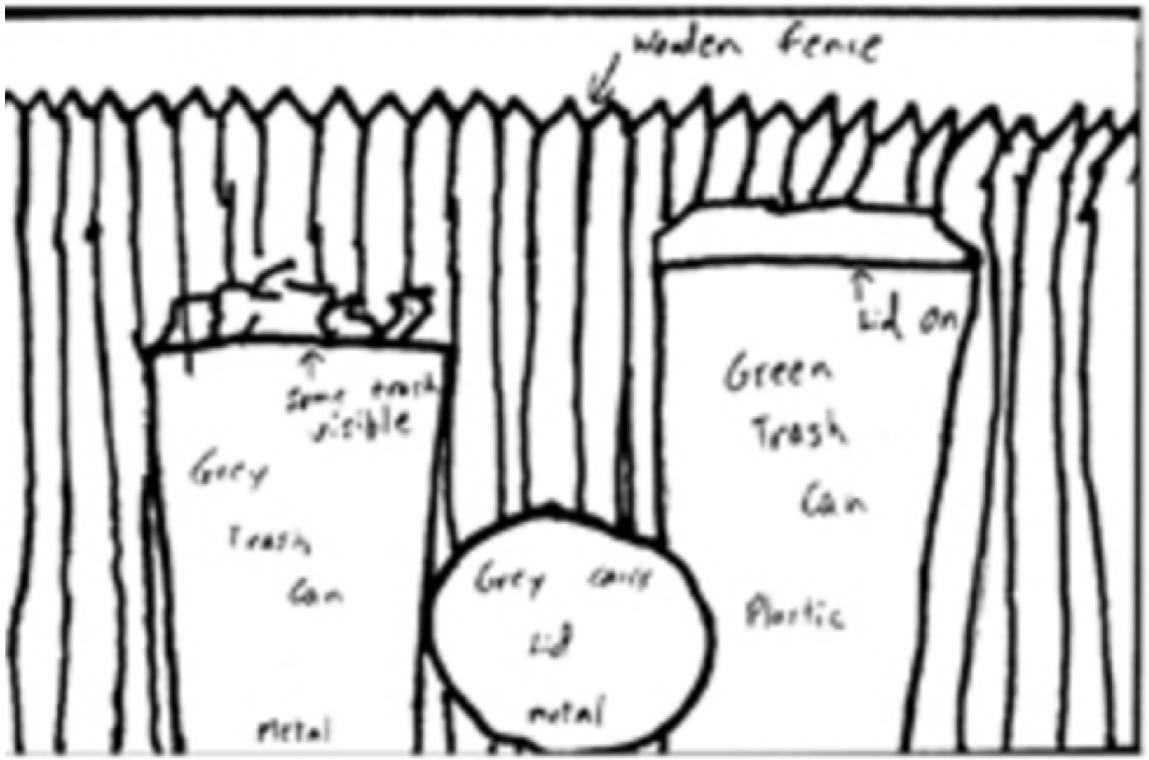
https://distill.pub/2017/feature-visualization/

Outline

- Neural networks are *representation* learners
- Auto-encoders
- Unsupervised and self-supervised learning
- Word embeddings
- (Some recent representation learning ideas)

Observed image

Drawn from memory



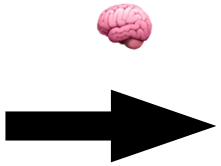
[Bartlett, 1932]
[Intraub & Richardson, 1989]

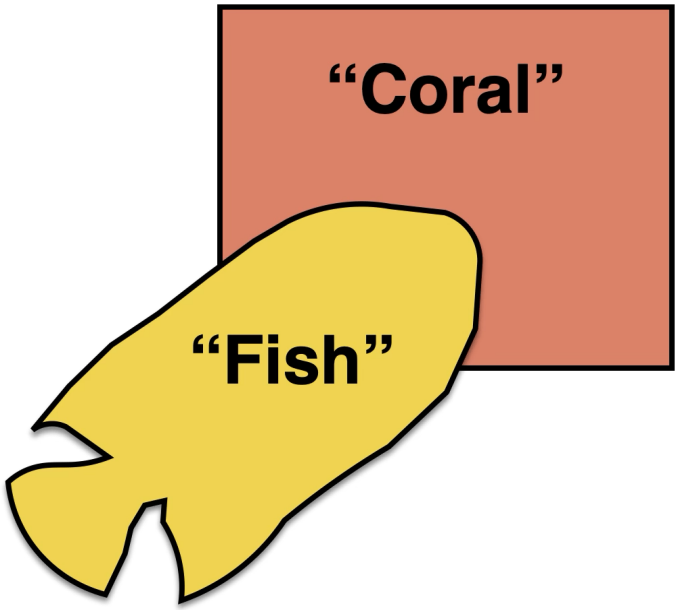
[https://www.behance.net/gallery/35437979/Velocipedia]

"I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have "327"? No. I have sky, house, and trees.”

— Max Wertheimer, 1923

humans also learn representations

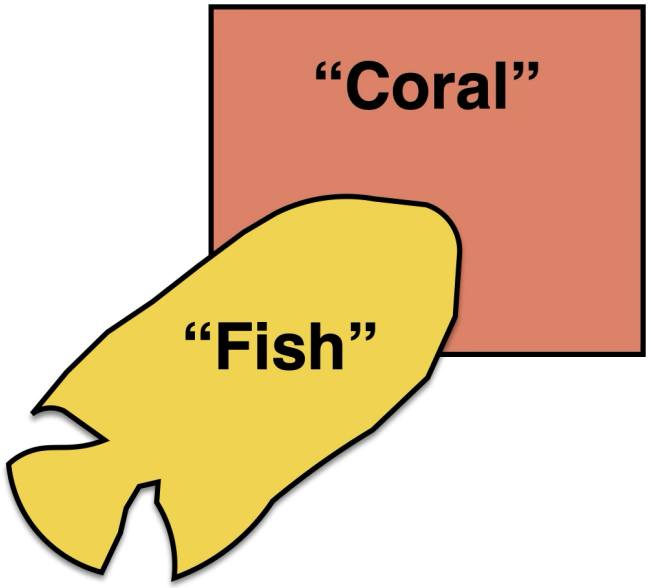




[images credit: visionbook.mit.edu]

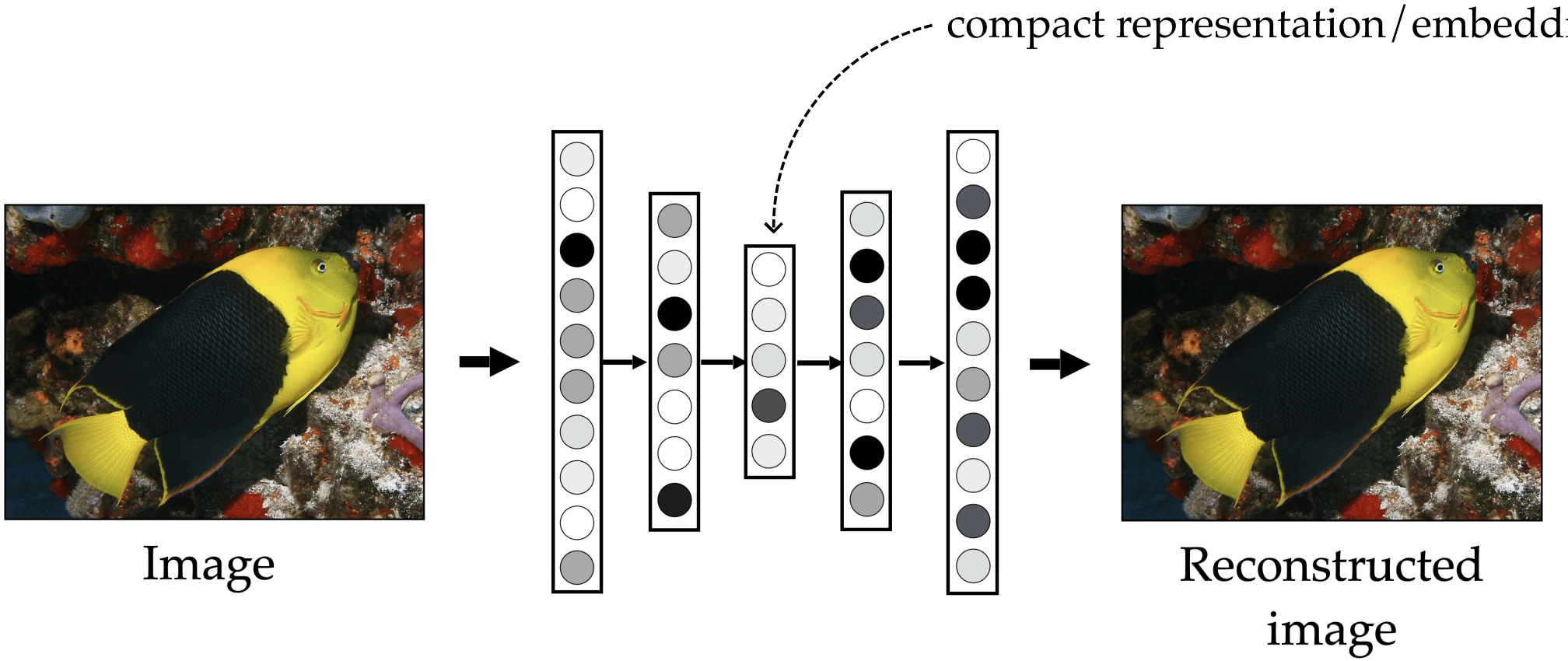
Good representations are:

- Compact (*minimal*)
- Explanatory (*roughly sufficient*)



[See “Representation Learning”, Bengio 2013, for more commentary]

Auto-encoder



"What I cannot create, I do not understand." Feynman

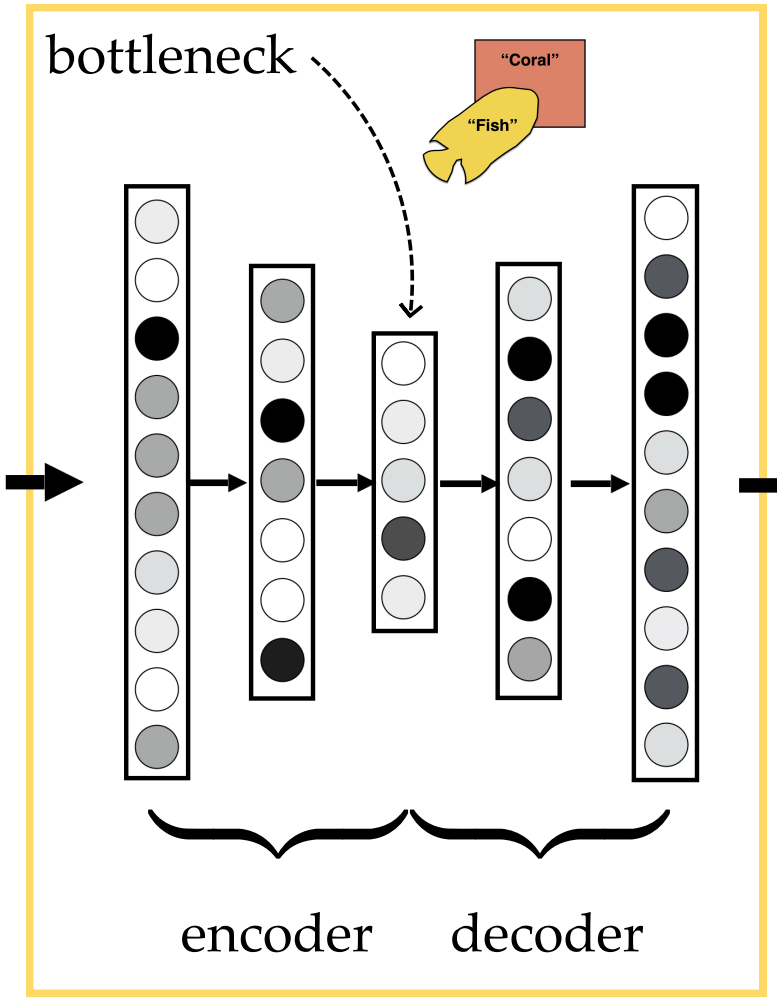
[images credit: visionbook.mit.edu]

Auto-encoder

$$\min_W ||x - \tilde{x}||^2$$

x

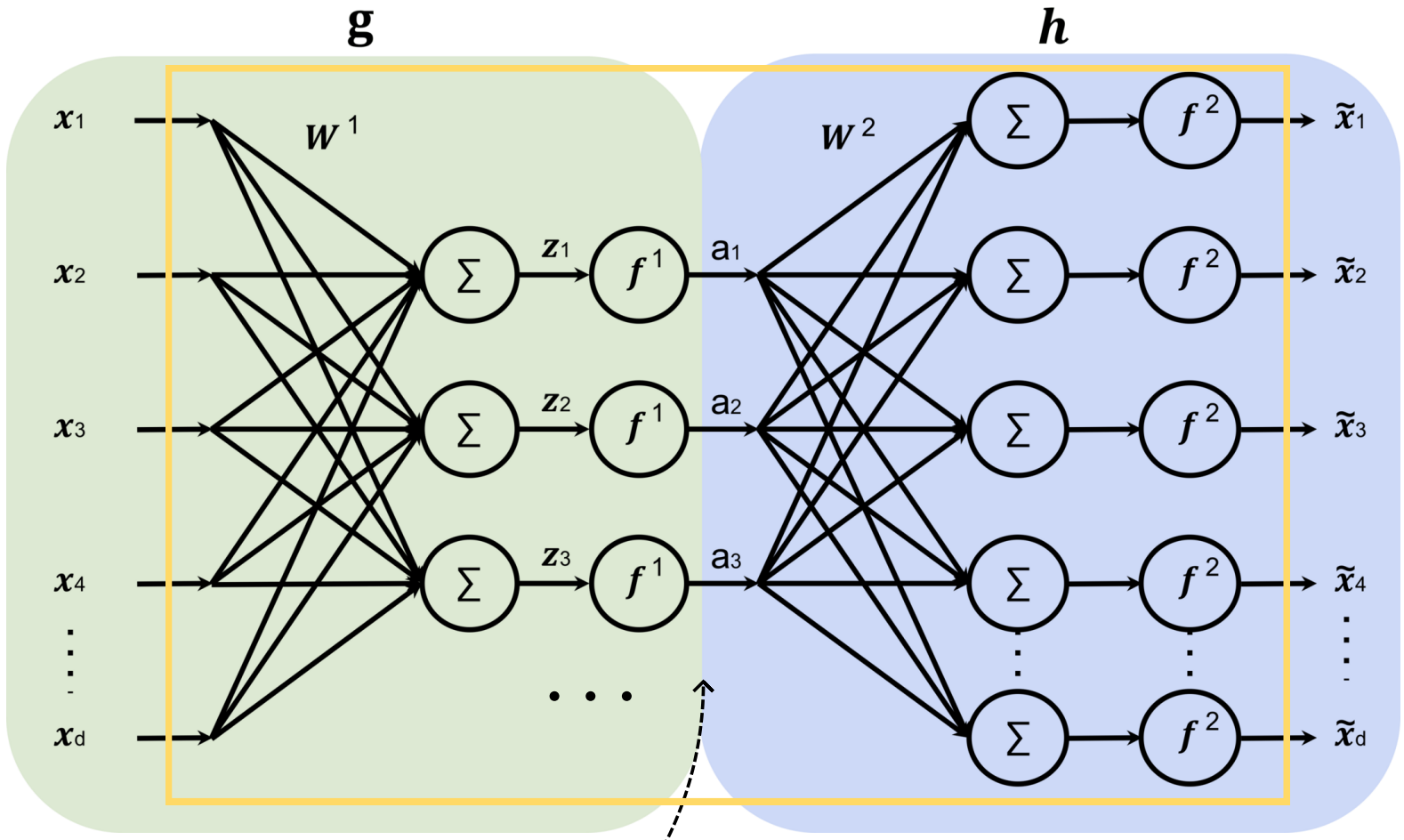
Image



$\tilde{x} = \text{NN}(x; W)$

Reconstructed image

input
$x \in \mathbb{R}^d$



output
$\tilde{x} \in \mathbb{R}^d$

bottleneck

typically, has lower dimension than d

Training Data

$$\left\{ x^{(i)} \right\}_{i=1}^{n}$$

Auto-encoder

loss / objective

$$\mathcal{L}(F(\mathbf{x}), \mathbf{x}) = \|F(\mathbf{x}) - \mathbf{x}\|^2$$

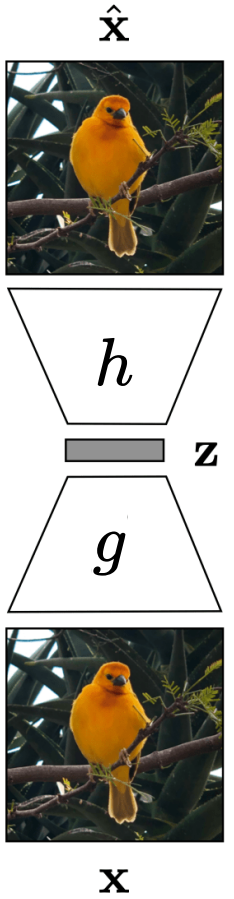
hypothesis class

$$F = h \circ g : \mathbb{R}^d \to \mathbb{R}^m \to \mathbb{R}^d$$

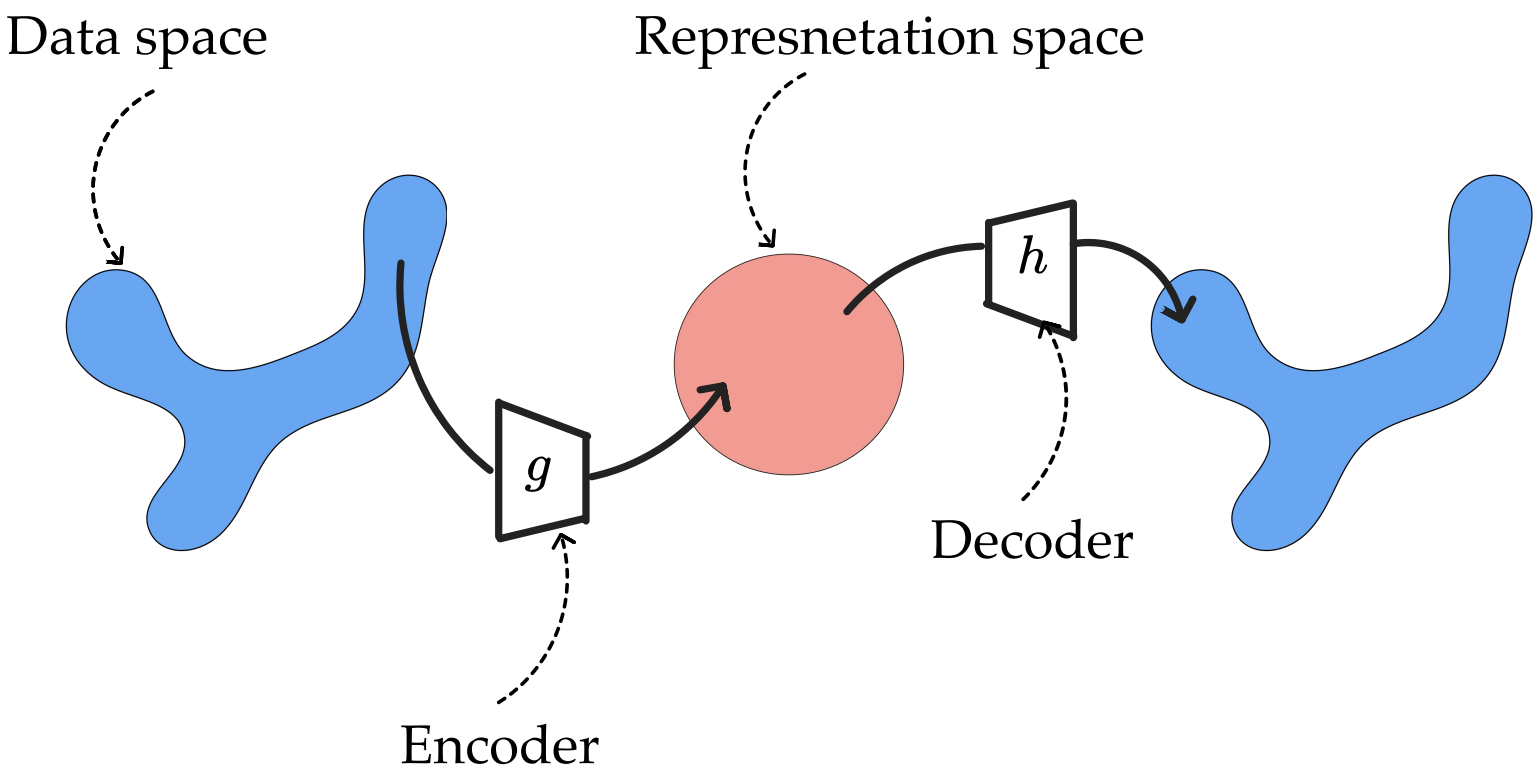
$$m < d$$

A model

f



[images credit: visionbook.mit.edu]



[Often, encoders can be kept to get "good representations"
whereas decoders can serve as "generative models"]

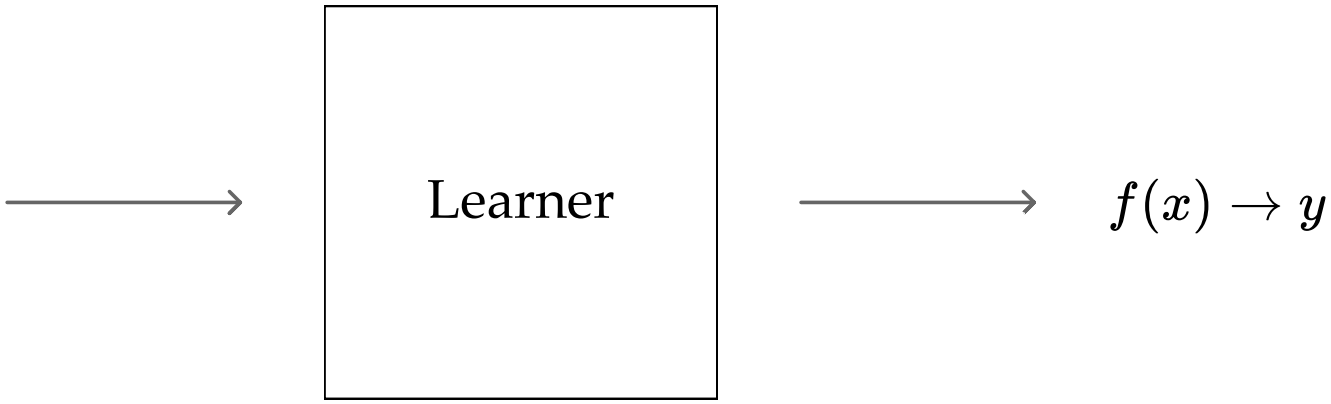
Outline

- Neural networks are *representation* learners
- Auto-encoders
- Unsupervised and self-supervised learning
- Word embeddings
- (Some recent representation learning ideas)

Supervised Learning

Training data

$\{x^{(1)}, y^{(1)}\}$
$\{x^{(2)}, y^{(2)}\}$
$\{x^{(3)}, y^{(3)}\}$
$\ldots$



Undergrads were time-consuming, algorithms were flawed, and the team didn't have money—Li said the project failed to win any of the federal grants she applied for, receiving comments on proposals that it was shameful Princeton would research this topic, and that the only strength of proposal was that Li was a woman.

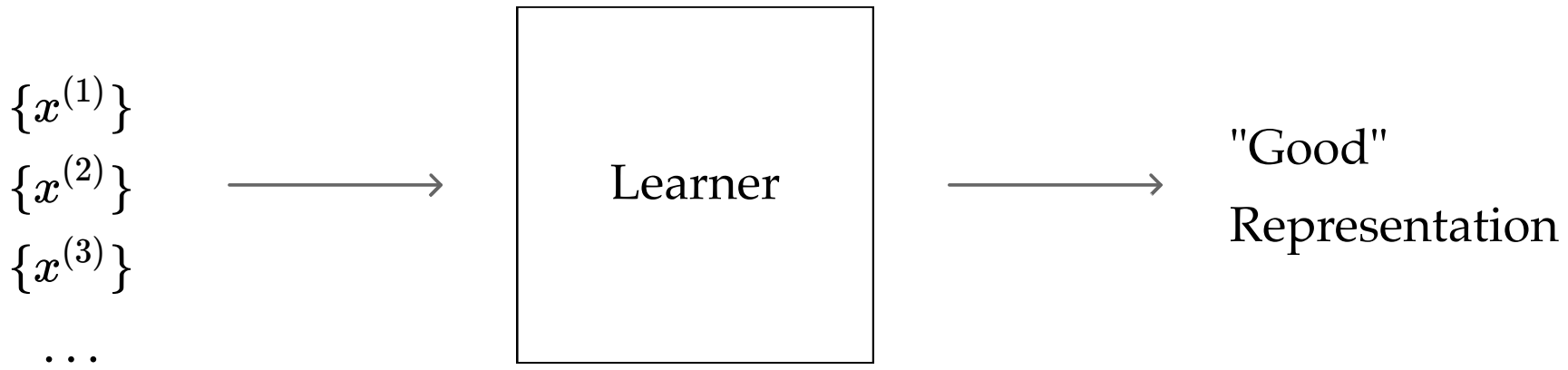
A solution finally surface
a graduate student who a
Amazon Mechanical Turk
sitting at computers arou
online tasks for pennies.

The Amazon Mechanical Turk backend for classifying images. Image: ImageNet

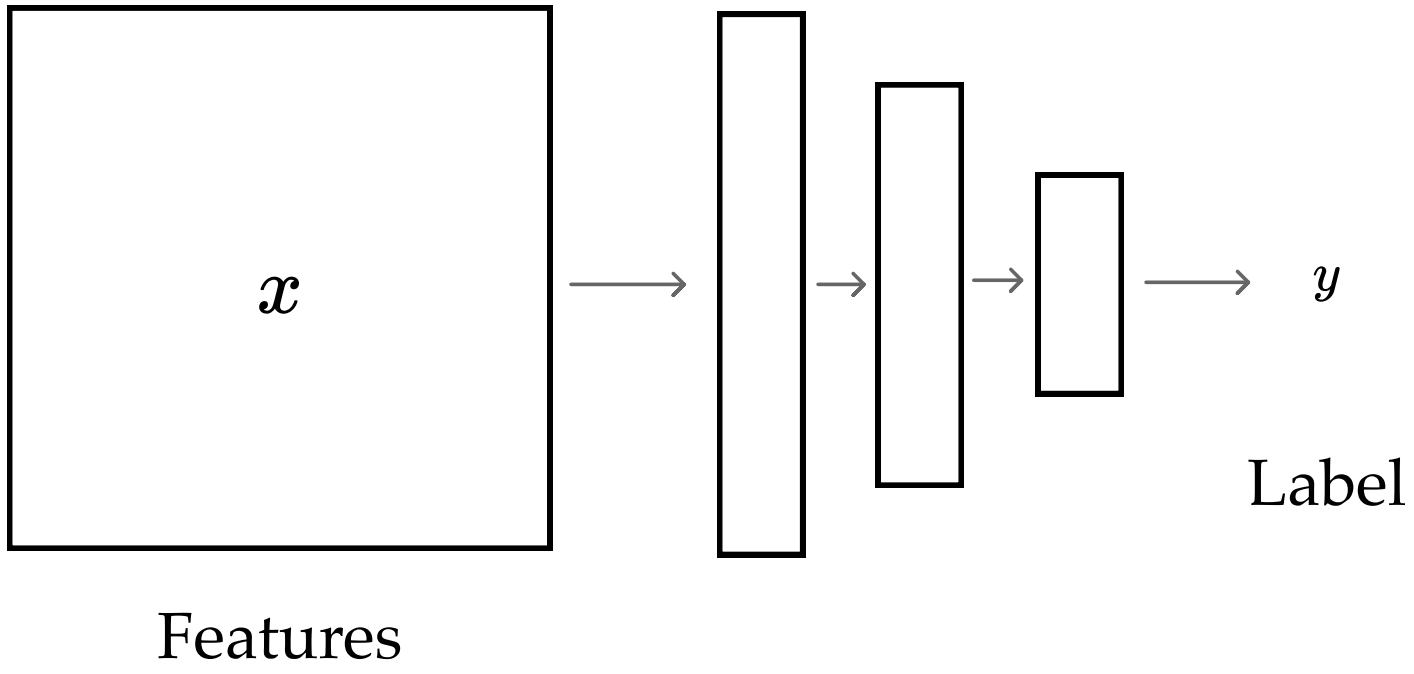
"He showed me the webs
knew the ImageNet project was going to happen," she said. "Suddenly we found a tool that could scale, that we could not possibly dream of by hiring Princeton undergrads."

Unsupervised Learning

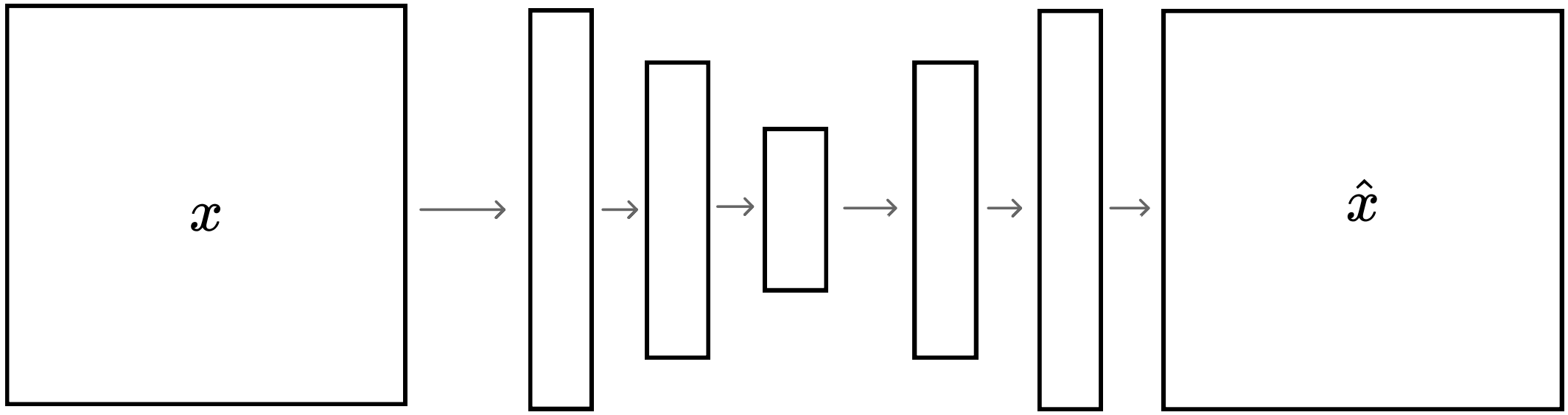
Training Data



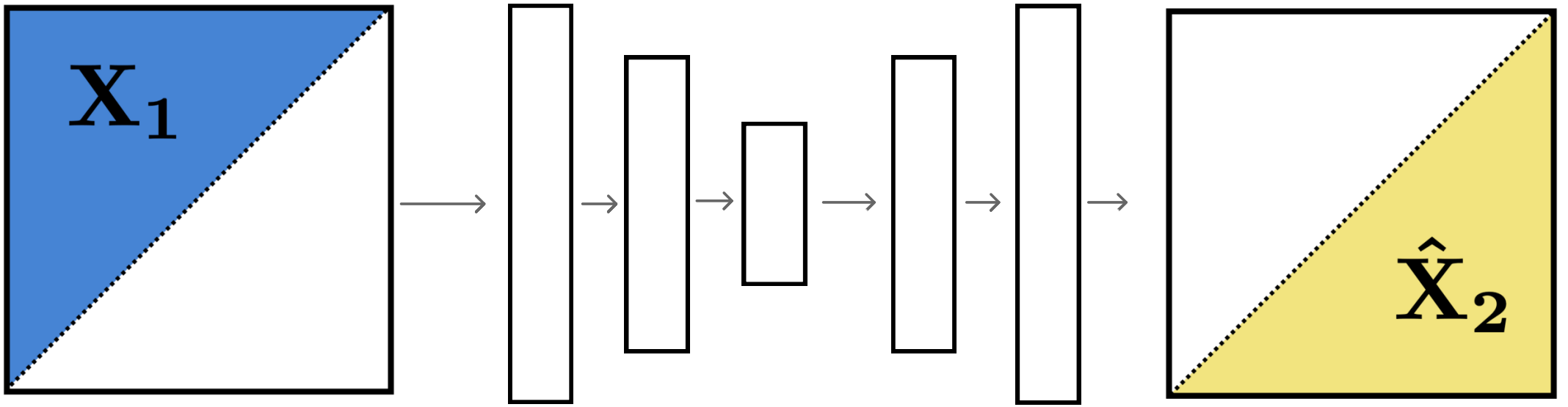
Label prediction (supervised learning)



Feature reconstruction (unsupervised learning)



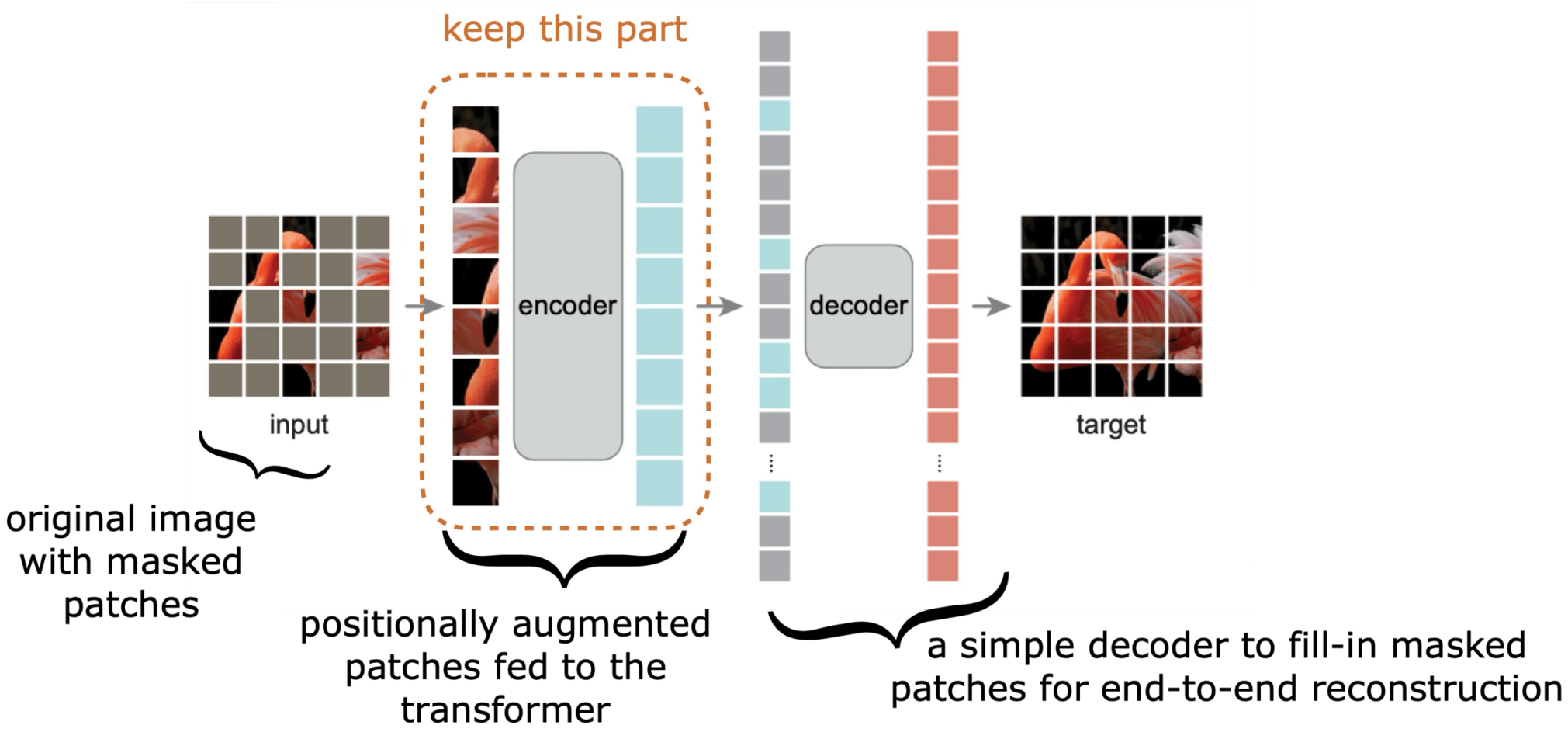
Self-supervised learning



Partial features

Other partial features

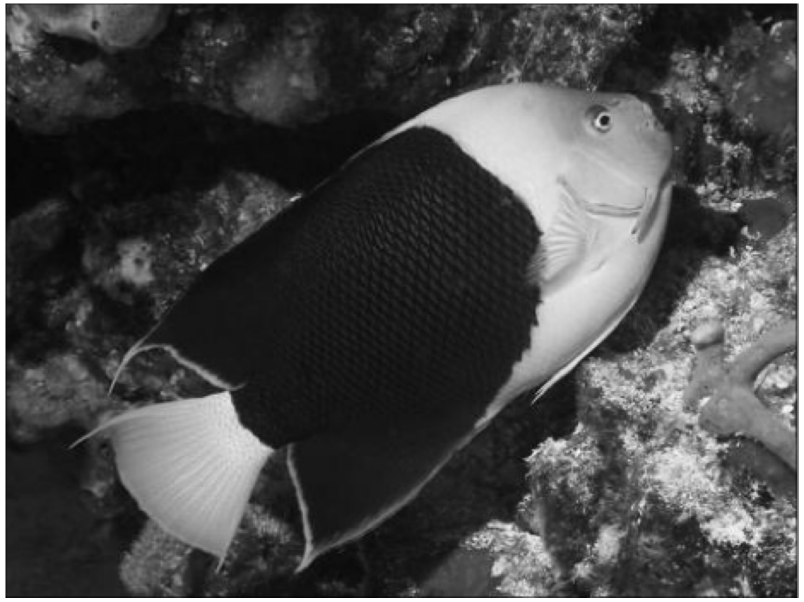
Masked Auto-encoder



[He, Chen, Xie, et al. MAE, 2021]

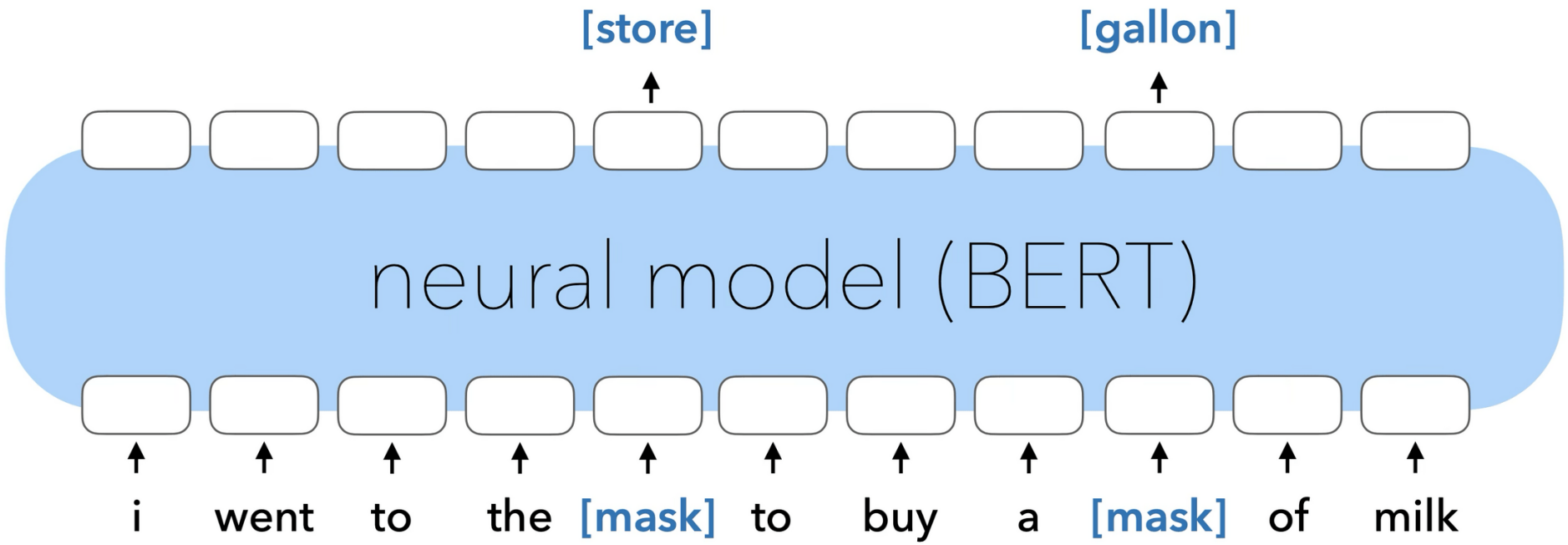
[Zhang, Isola, Efros, ECCV 2016]

predict color from gray-scale



[Zhang, Isola, Efros, ECCV 2016]

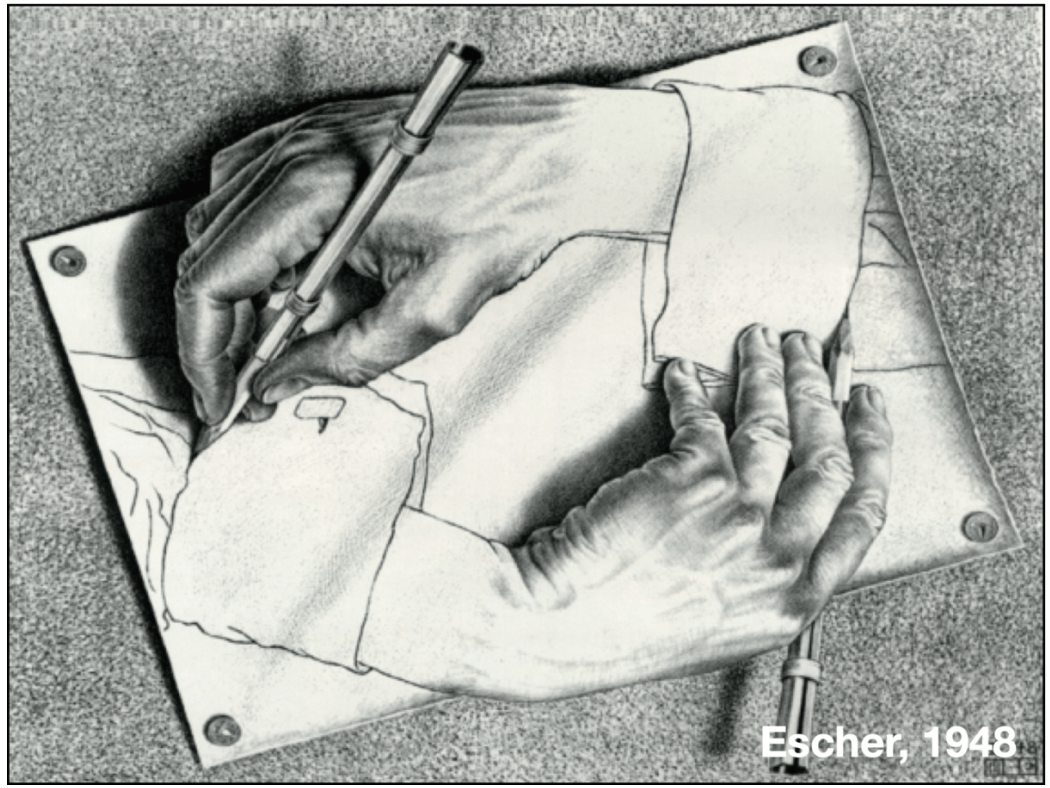
Masked Auto-encoder



Devlin et al., BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019

[Devlin, Chang, Lee, et al. 2019]

Self-supervised learning



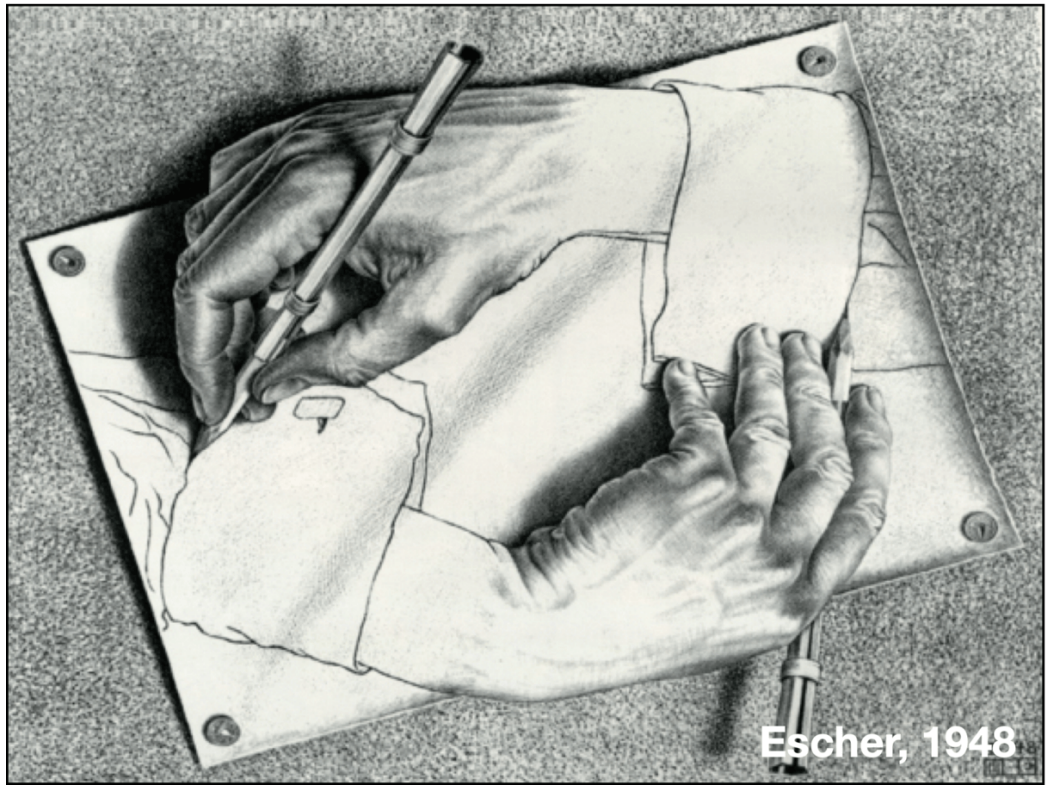
Common trick:

- Convert “unsupervised” problem into “supervised” setup
- Do so by cooking up “labels” (prediction targets) from the raw data itself — called *pretext* task

Outline

- Neural networks are *representation* learners
- Auto-encoders
- Unsupervised and self-supervised learning
- Word embeddings
- (Some recent representation learning ideas)

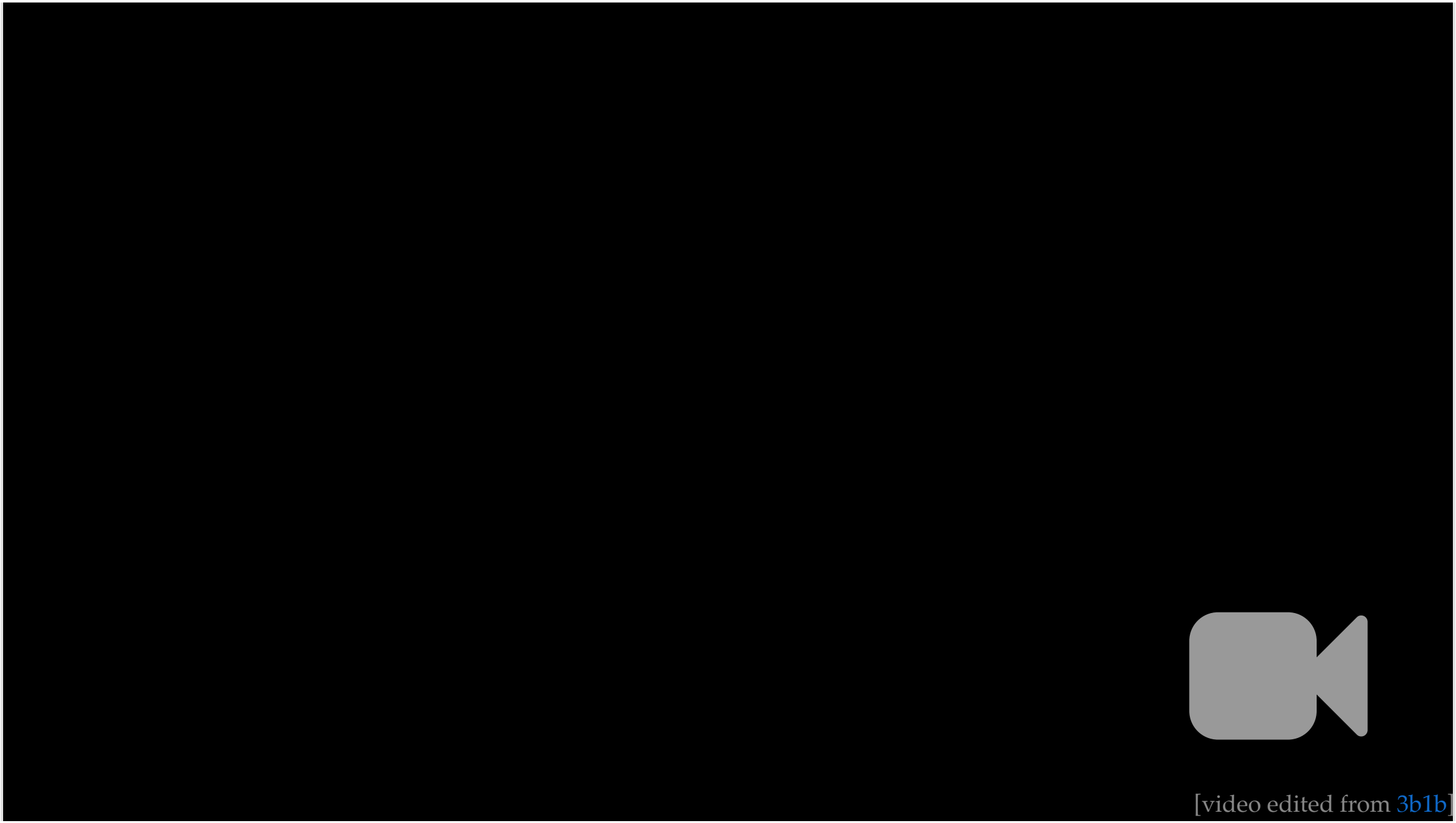
Large Language Models (LLMs) are trained in a self-supervised way



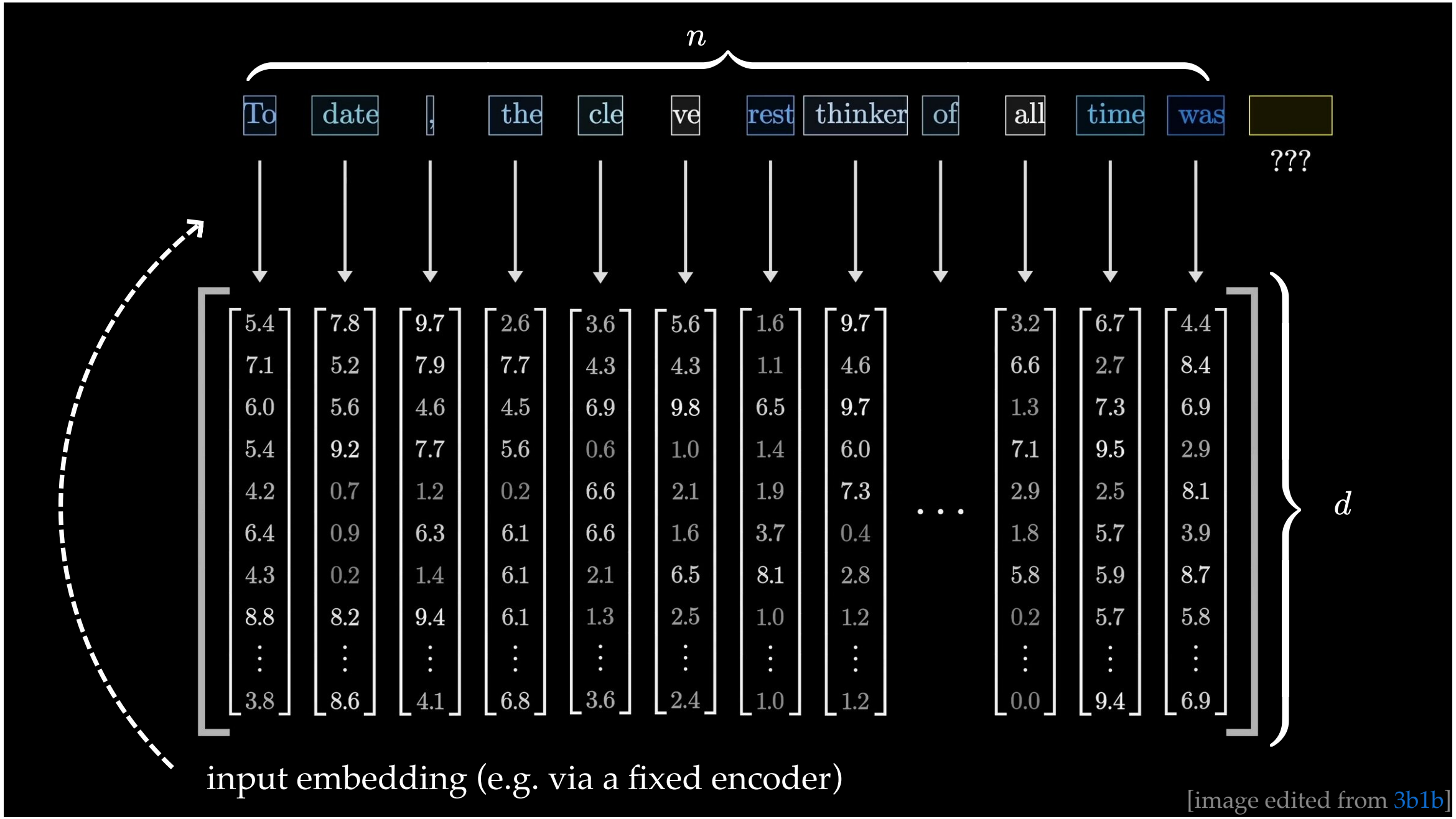
- Scrape the internet for unlabeled plain texts.
- Cook up “labels” (prediction targets) from the unlabeled texts.
- Convert “unsupervised” problem into “supervised” setup.

"To date, the cleverest thinker of all time was Issac. "

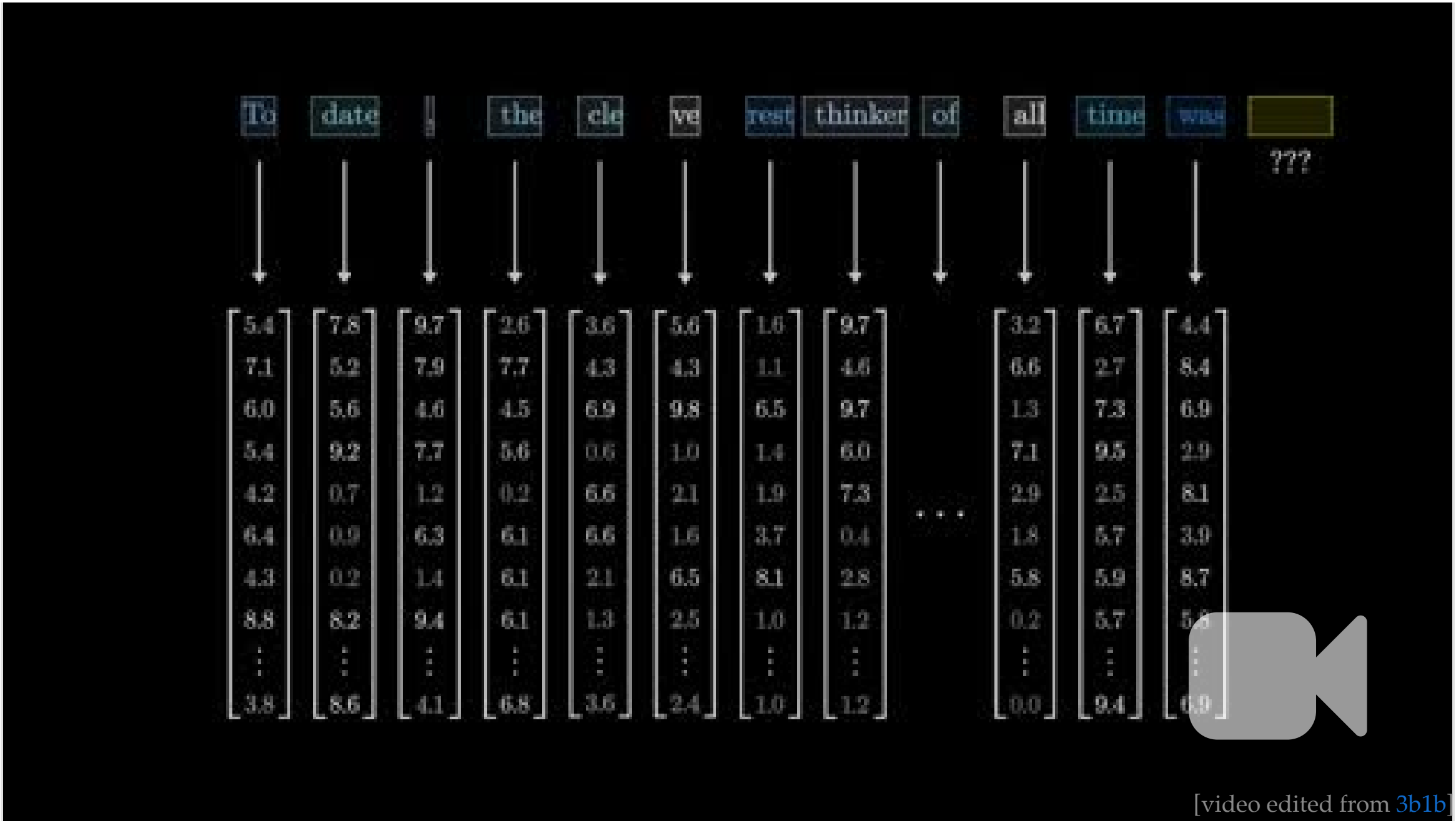
feature	label
⋮	⋮
To date, the	cleverest
To date, the cleverest	thinker
To date, the cleverest thinker	was
⋮	⋮
To date, the cleverest thinker of all time was	Issac



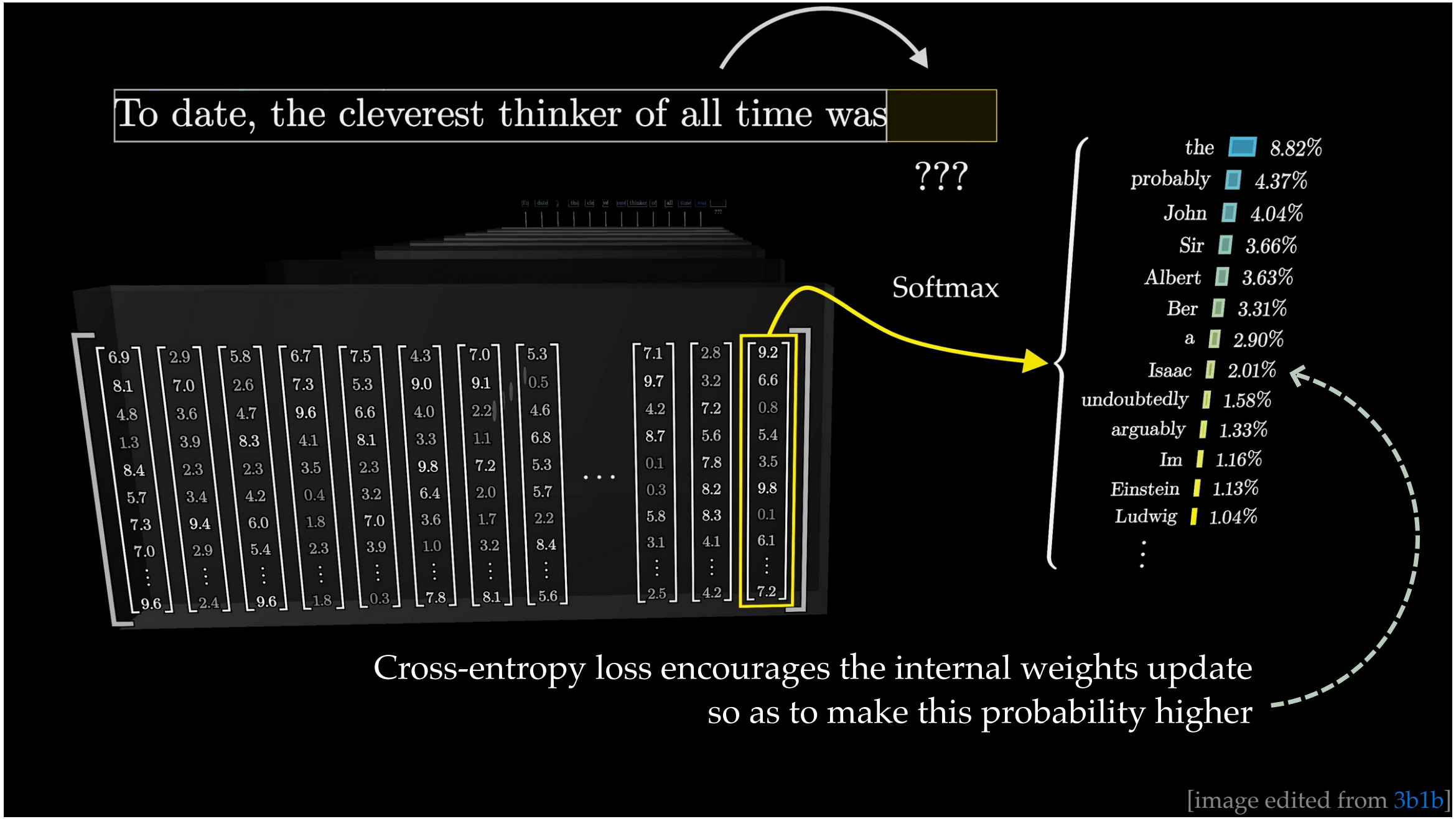
[video edited from 3b1b]



[image edited from 3b1b]

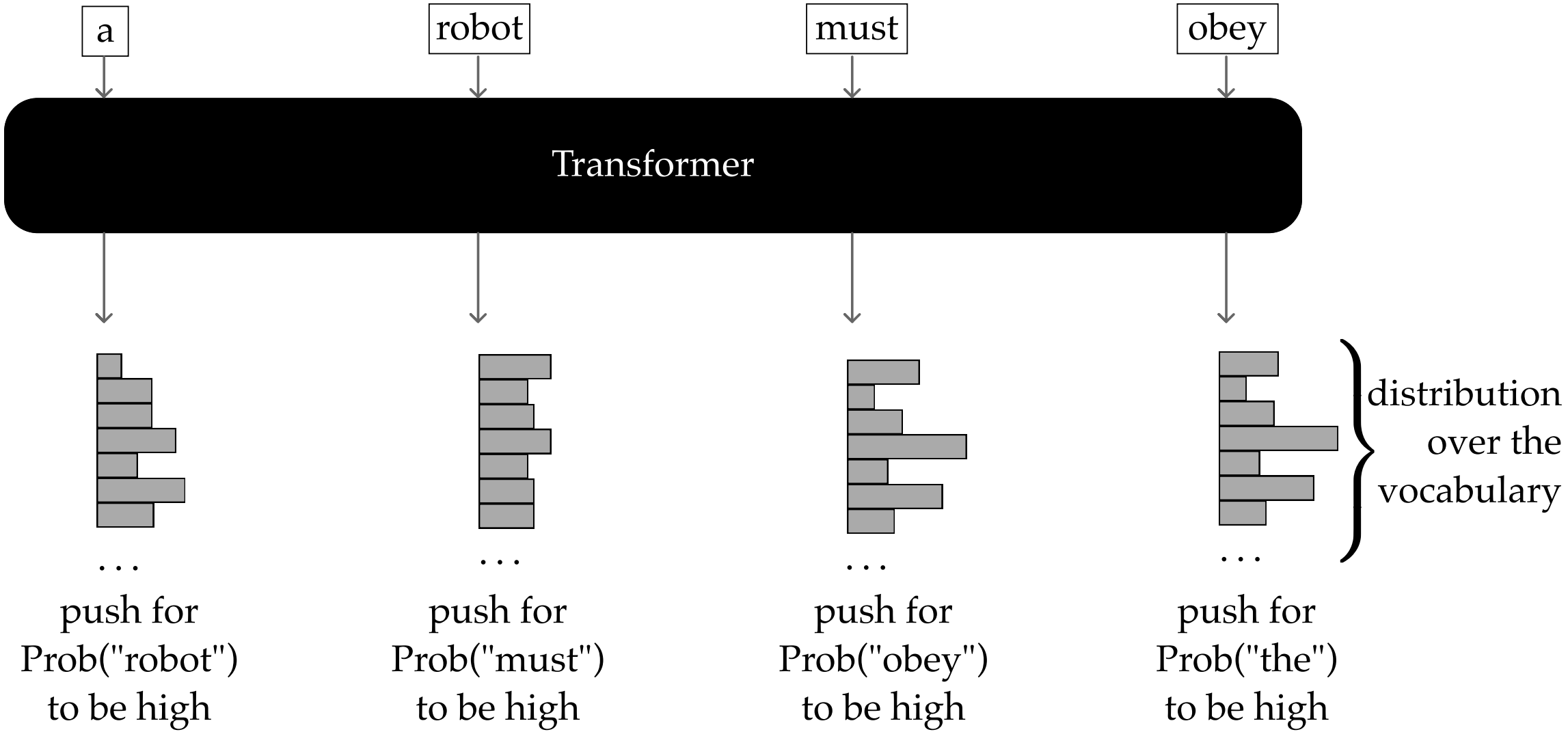


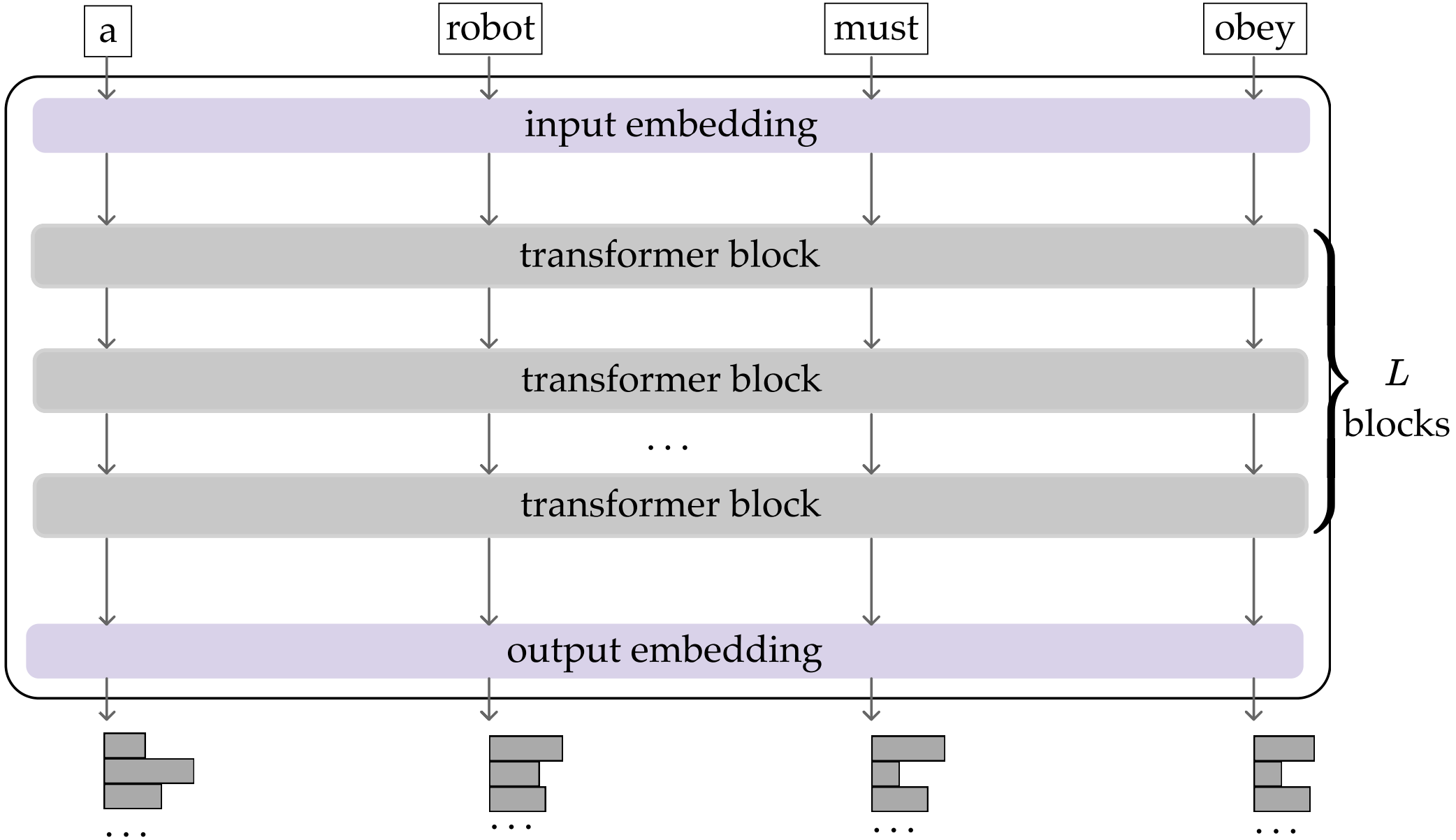
[video edited from 3b1b]



[image edited from 3b1b]

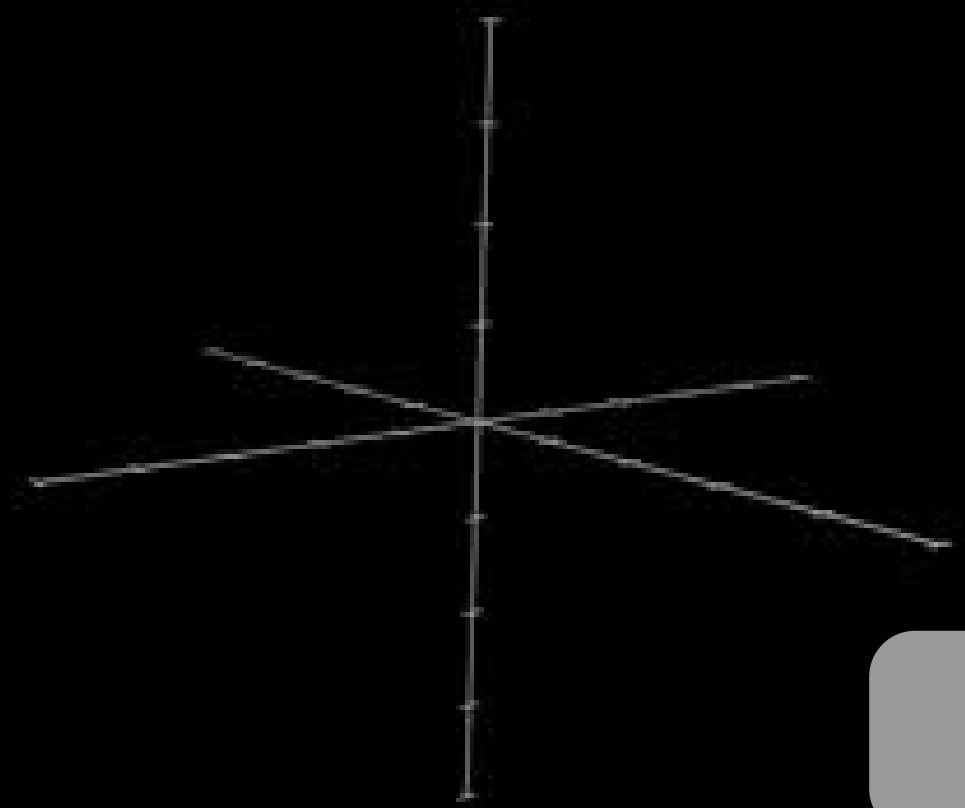
"A robot must obey the orders given it by human beings ..."




a
robot
must
obey
input embedding
transformer block
transformer block
...
transformer block
output embedding
L
blocks
...
...
...
...

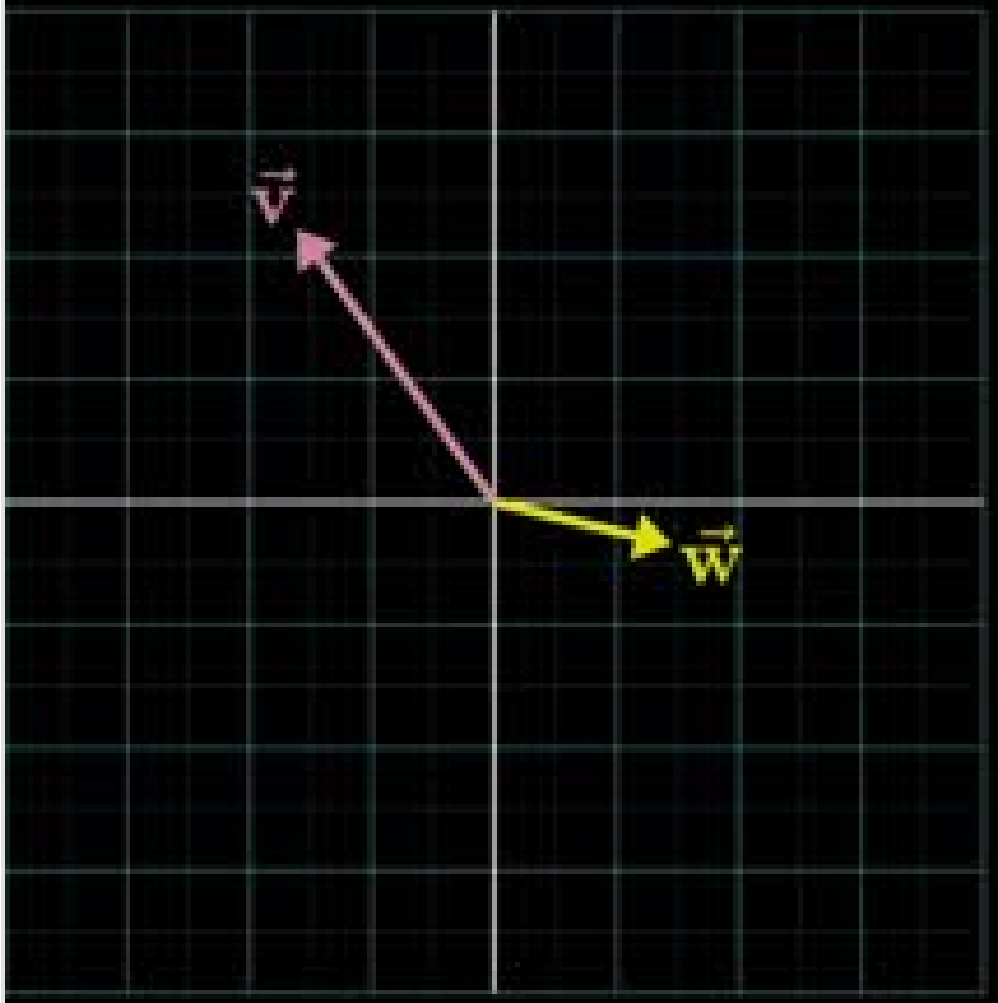
Word embedding

Words → Vectors



[video edited from 3b1b]

dot-product similarity



$$\underbrace{\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_n \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix}}_{\text{Dot product}} = \begin{matrix} v_1 w_1 \\ + \\ v_2 w_2 \\ + \\ v_3 w_3 \\ + \\ \vdots \\ + \\ v_n w_n \\ \| \\ -3.11 \end{matrix}$$

good embedding representation => sensible dot-product similarity

=> enables effective *attention* in transformers next week.

For now, let's think about dictionary look-up:

```
dict_en2fr = {
  "apple" : "pomme",
  "banana": "banane",
  "lemon" : "citron"}
```

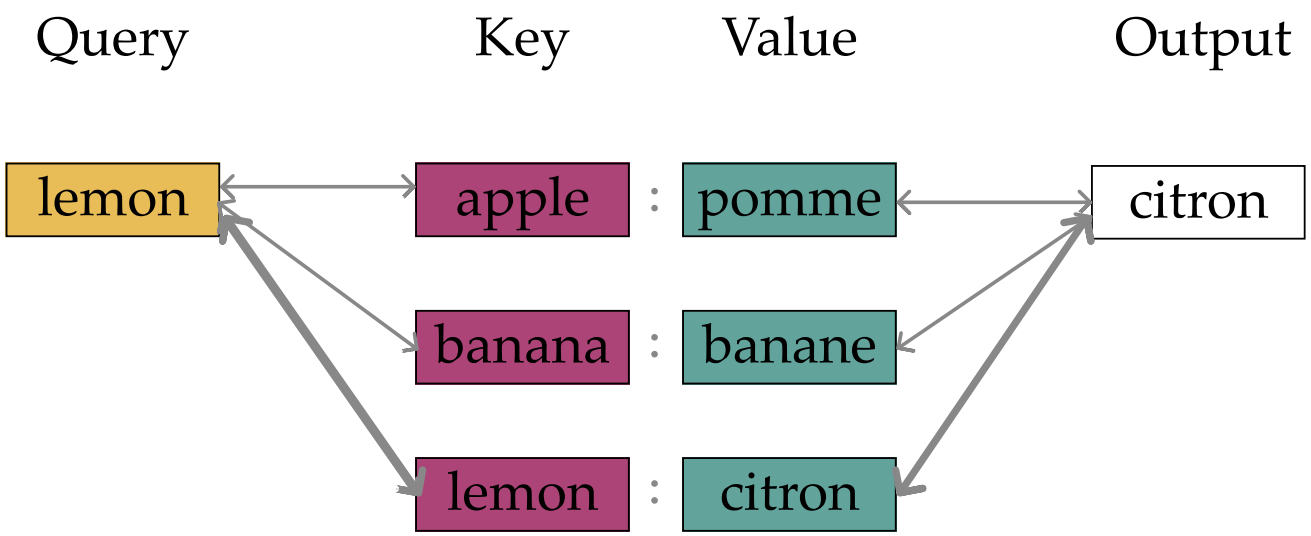
Key Value

apple : pomme

banana : banane

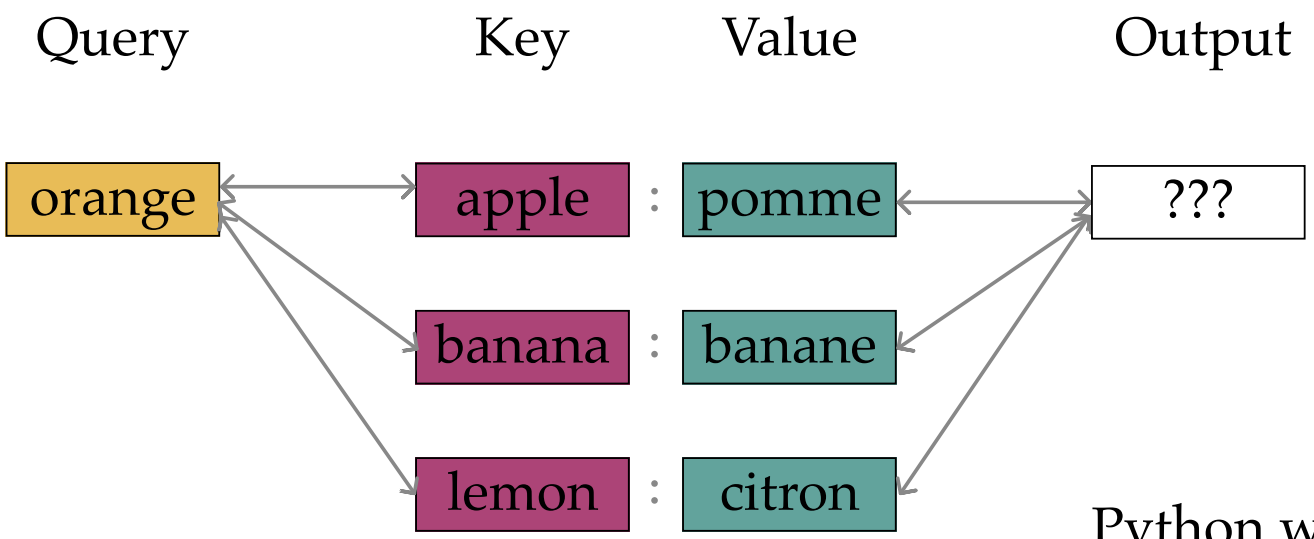
lemon : citron

```python
dict_en2fr = {
  "apple" : "pomme",
  "banana": "banane",
  "lemon" : "citron"}

query = "lemon"
output = dict_en2fr[query]
```



What if we run

```
dict_en2fr = {
  "apple" : "pomme",
  "banana": "banane",
  "lemon" : "citron"}

query = "orange"
output = dict_en2fr[query]
```

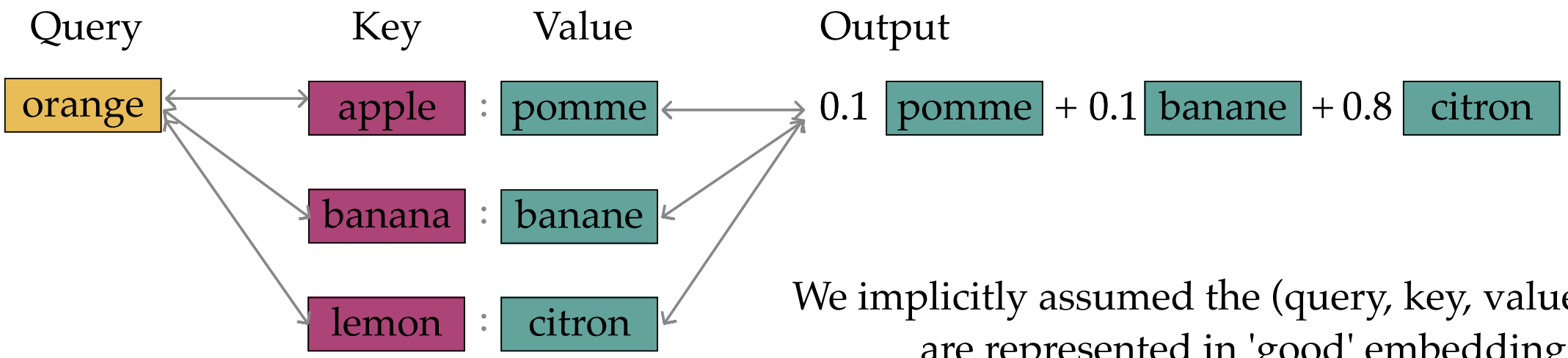


Python would complain. 🤯

What if we run

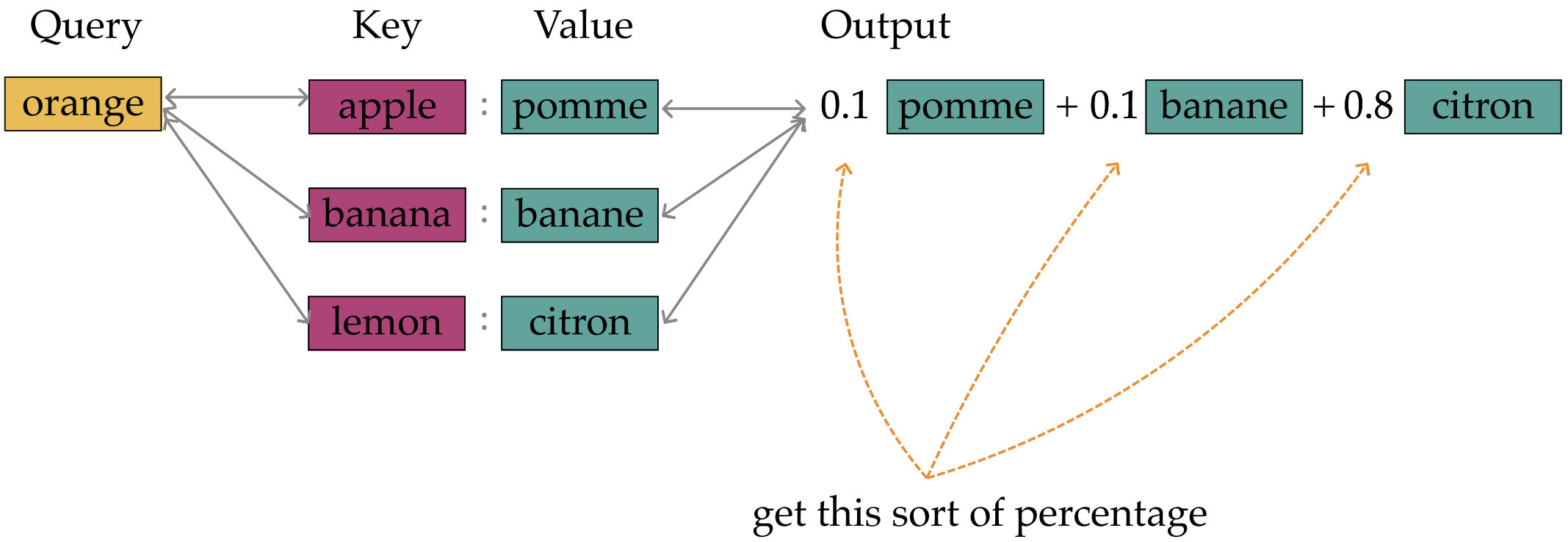
```
dict_en2fr = {
  "apple" : "pomme",
  "banana": "banane",
  "lemon" : "citron"}

query = "orange"
output = dict_en2fr[query]
```

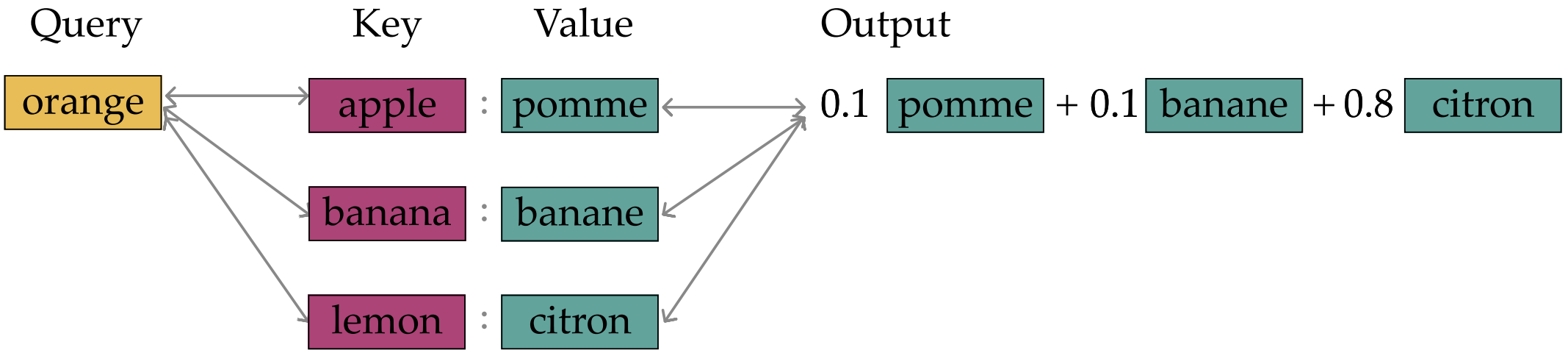
But we can probably see the rationale behind this:



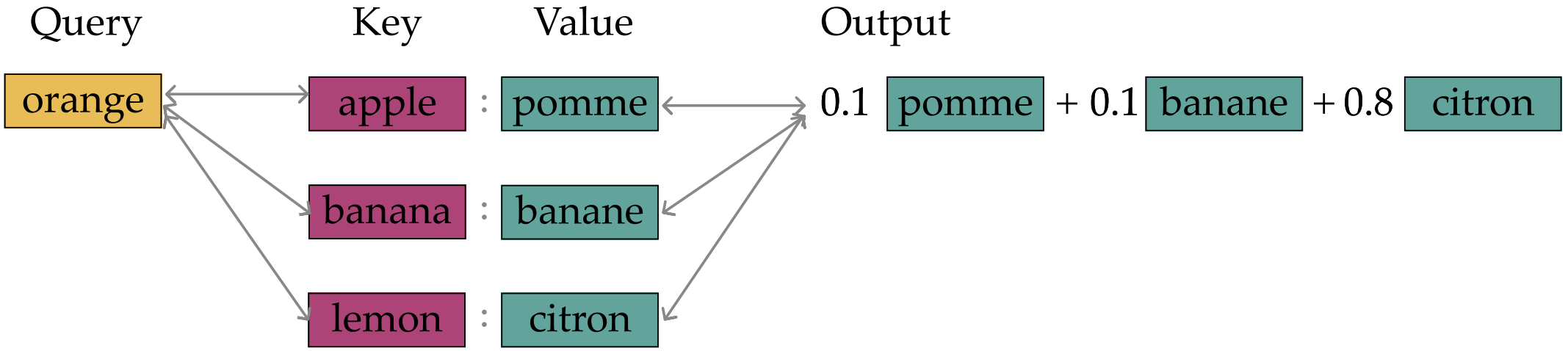
We implicitly assumed the (query, key, value) are represented in 'good' embeddings.

If we are to generalize this idea, we need to:

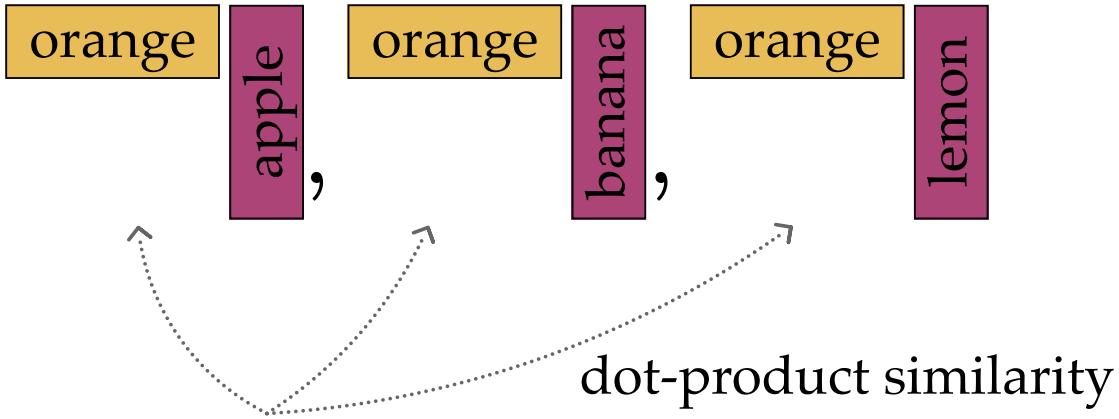


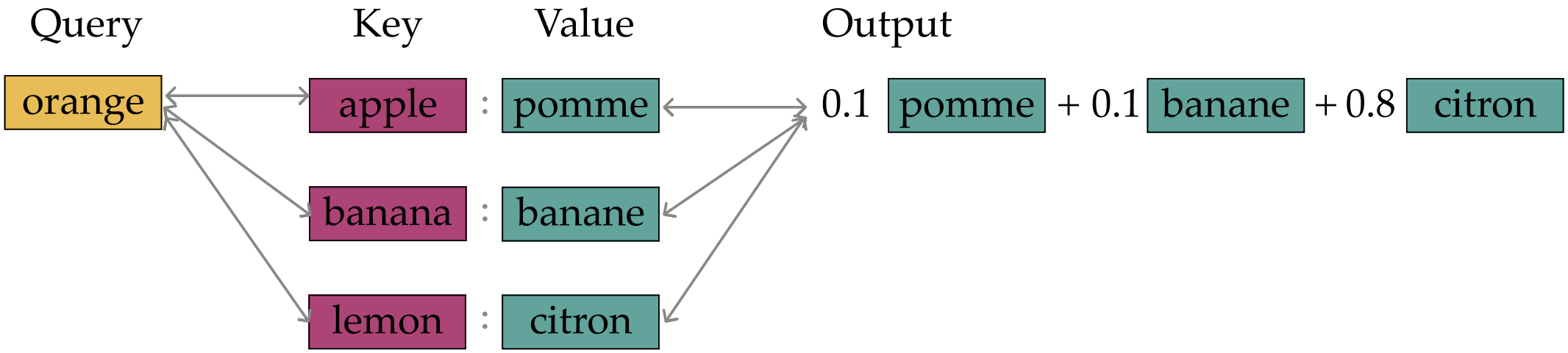


very roughly, the attention mechanism does exactly this kind of "soft" look-up:



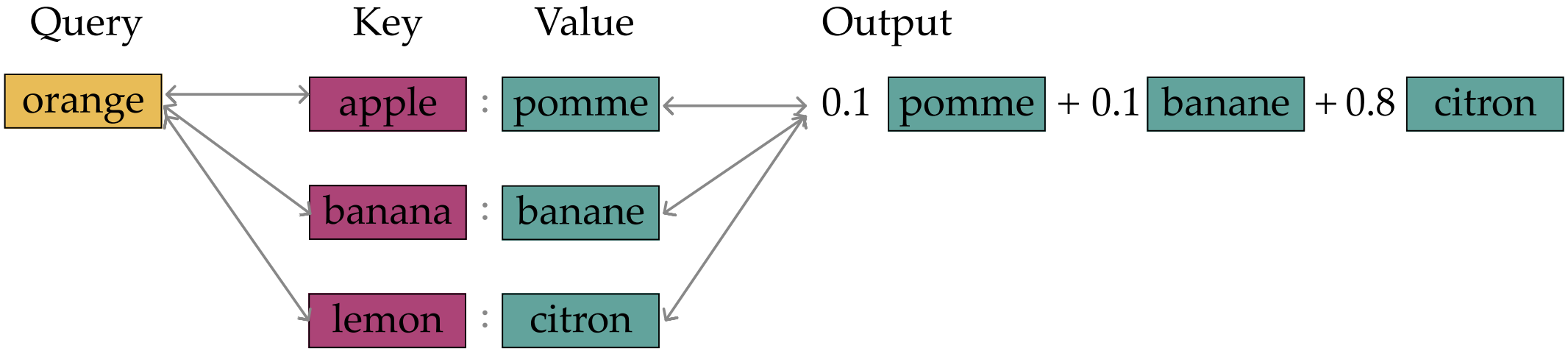
very roughly, the attention mechanism does exactly this kind of "soft" look-up:





very roughly, the attention mechanism does exactly this kind of "soft" look-up:

$$\text{softmax}\left(\text{orange}\,\text{apple}, \text{orange}\,\text{banana}, \text{orange}\,\text{lemon}\right) = \begin{bmatrix} 0.1 & 0.1 & 0.8 \end{bmatrix}$$



very roughly, the attention mechanism does exactly this kind of "soft" look-up:

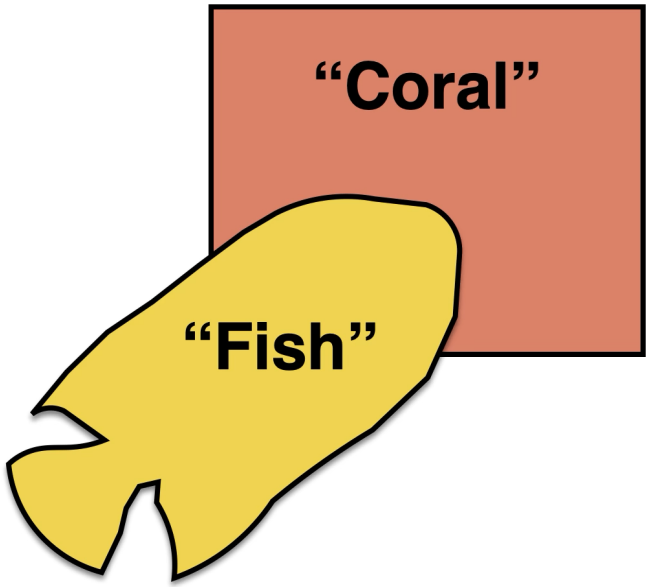
$$\text{softmax}\left(\text{orange}\,\text{apple}, \text{orange}\,\text{banana}, \text{orange}\,\text{lemon}\right) = \begin{bmatrix} 0.1 & 0.1 & 0.8 \end{bmatrix}$$

and output 0.1 pomme $+ 0.1$ banane $+ 0.8$ citron — weighted average over values

Outline

- Neural networks are *representation* learners
- Auto-encoders
- Unsupervised and self-supervised learning
- Word embeddings
- (Some recent representation learning ideas)

Good representations are:



Auto-encoders try to achieve these

- Compact (*minimal*)
- Explanatory (*roughly sufficient*)

these may just emerge as well

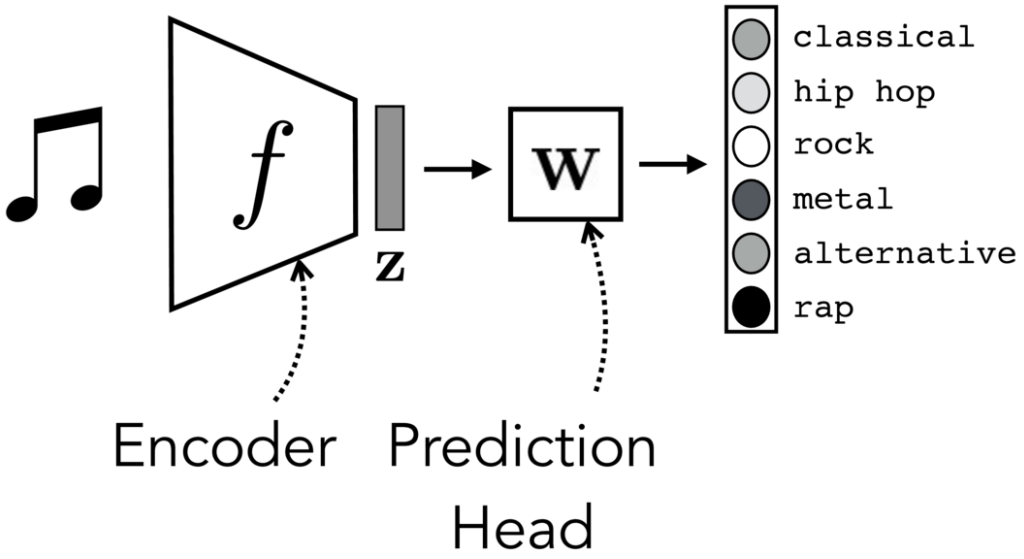
- Disentangled (*independent* factors)
- Interpretable
- Make *subsequent problem solving easy*

[See “Representation Learning”, Bengio 2013, for more commentary]

- pre-training
- contrastive
- multi-modality

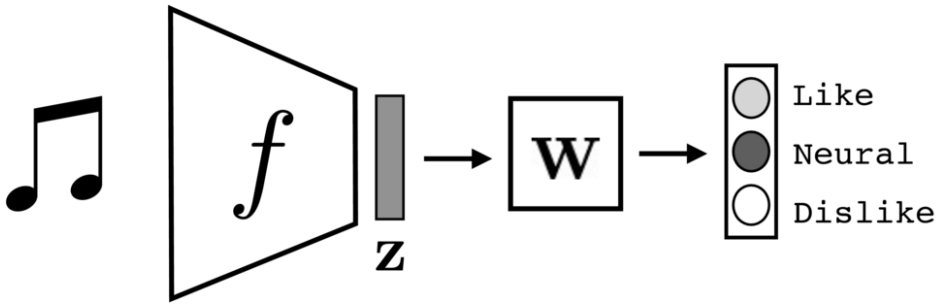
Training

Genre recognition



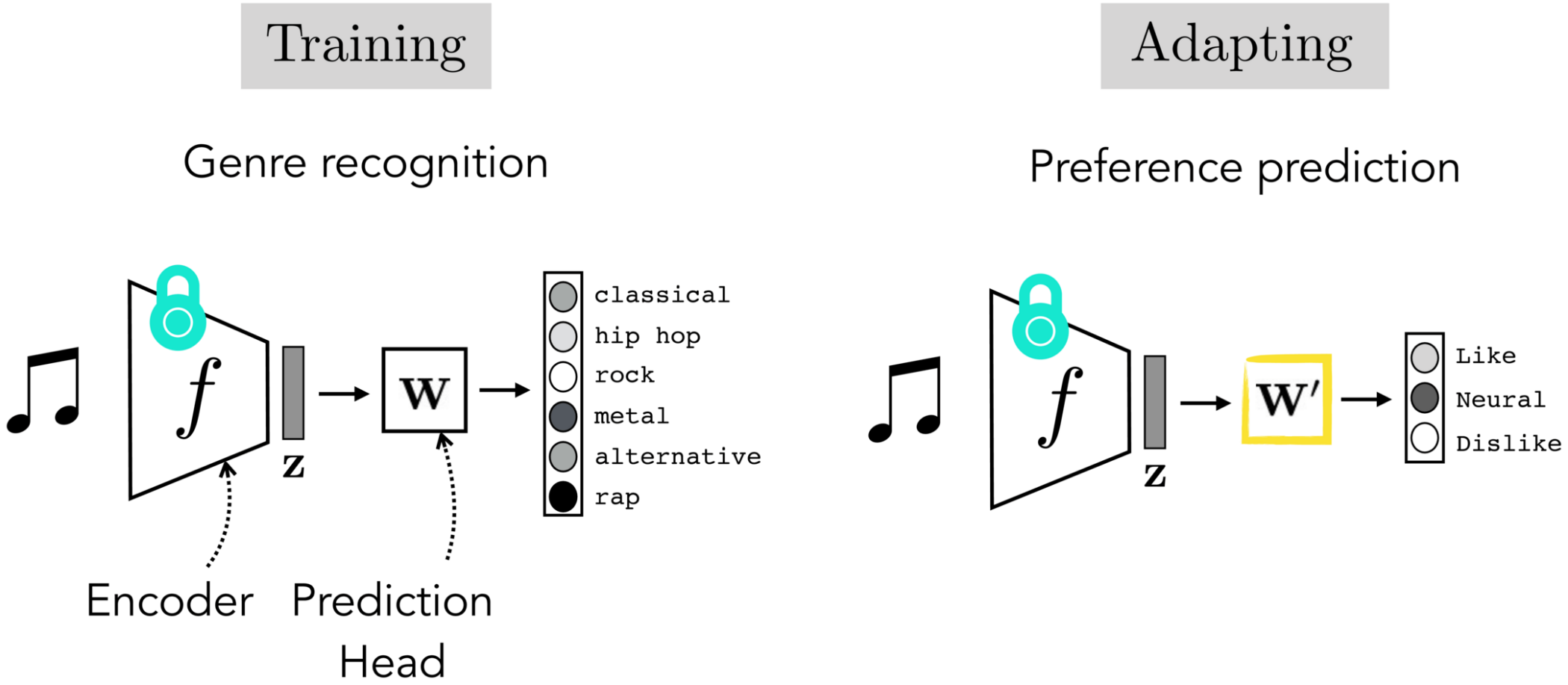
Testing

Preference prediction



Often, what we will be "tested" on is not what we were trained on.

[images credit: visionbook.mit.edu]

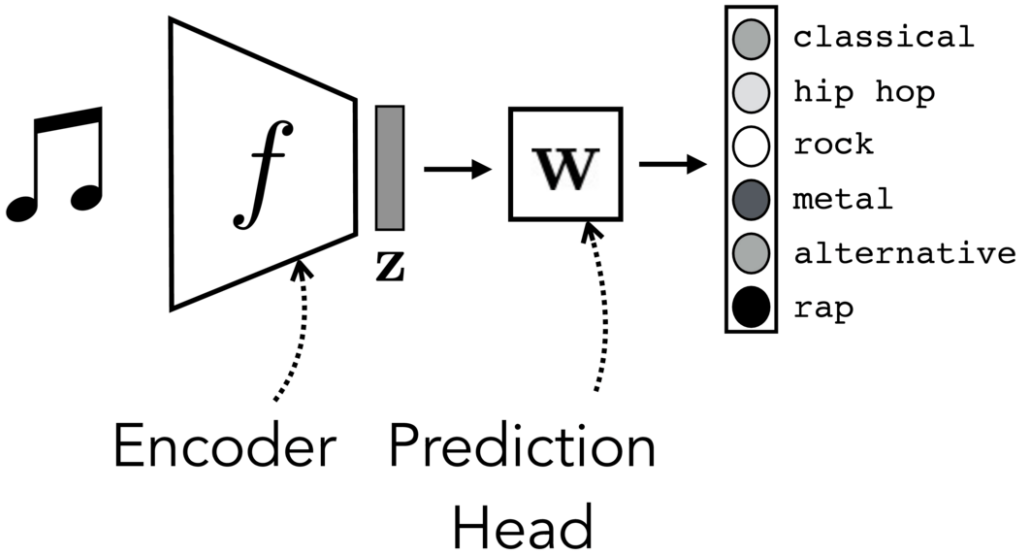


Final-layer adaptation: freeze f, train a new final layer to new target data

[images credit: visionbook.mit.edu]

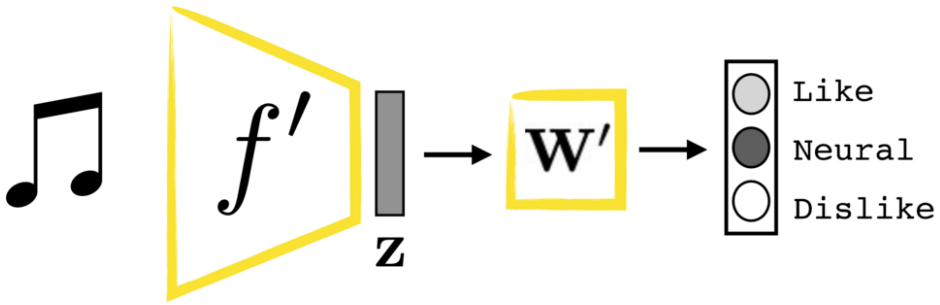
Training

Genre recognition



Adapting

Preference prediction

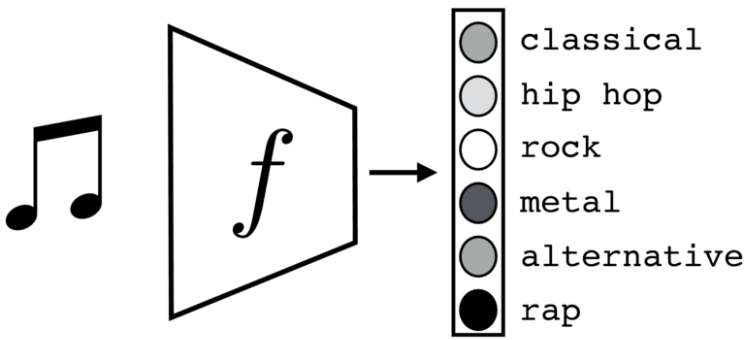


Finetuning: initialize f' as f, then continue training for f' as well, on new target data

[images credit: visionbook.mit.edu]

Pretraining

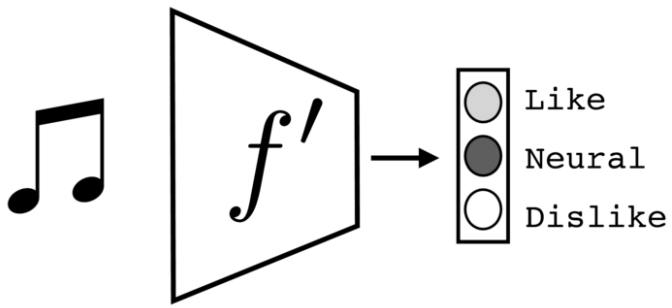
Genre recognition



A lot of data

Adapting

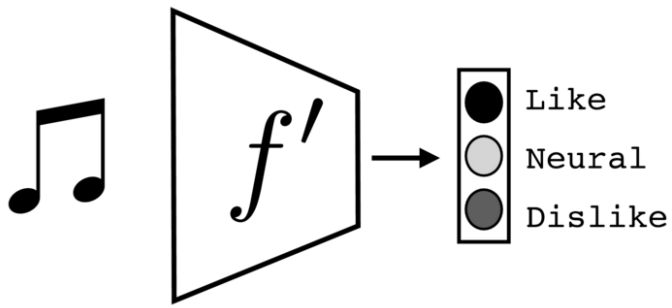
Preference prediction



A little data

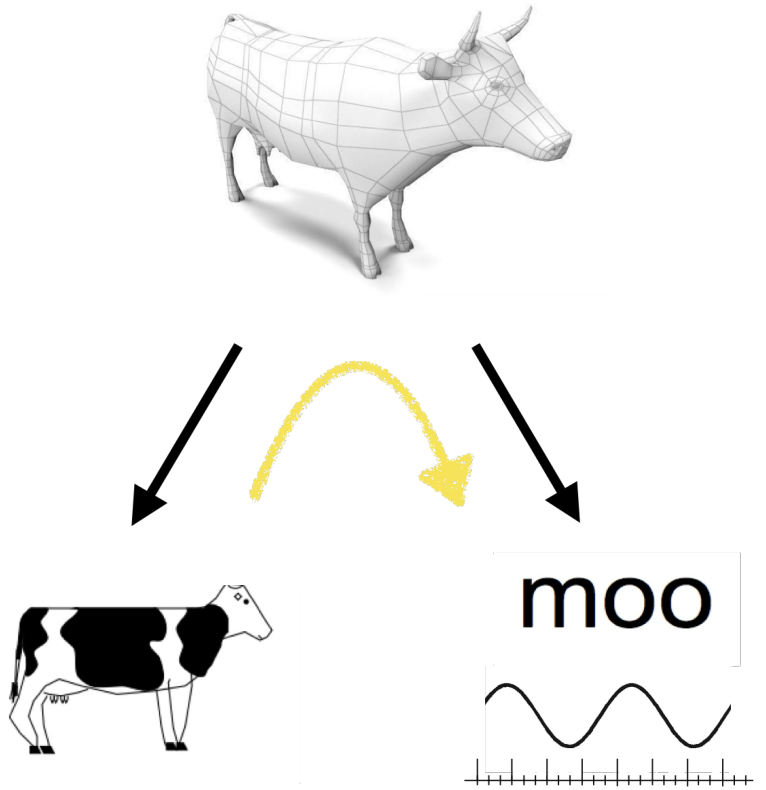
Testing

Preference prediction



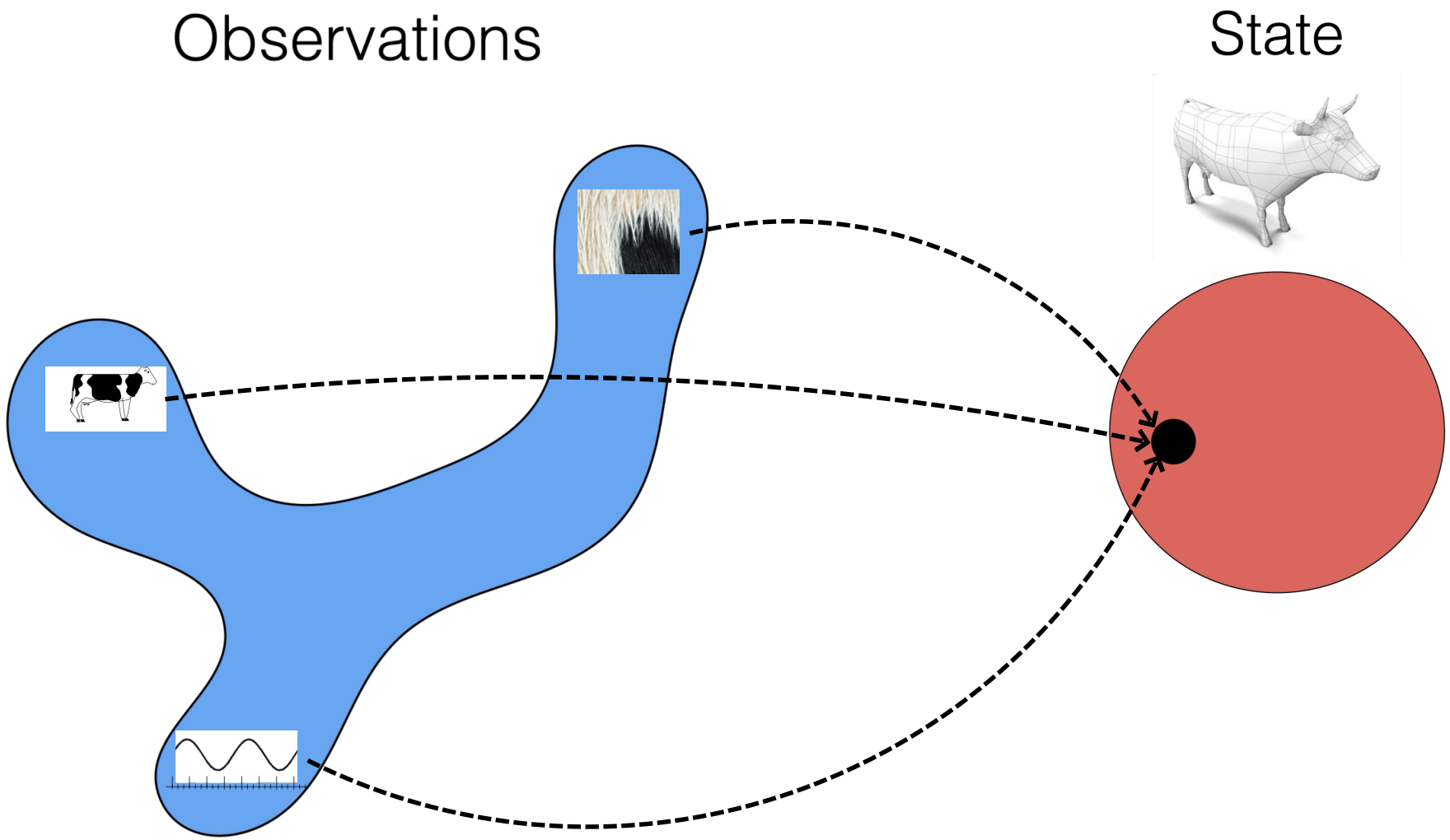
[images credit: visionbook.mit.edu]

The allegory of the cave



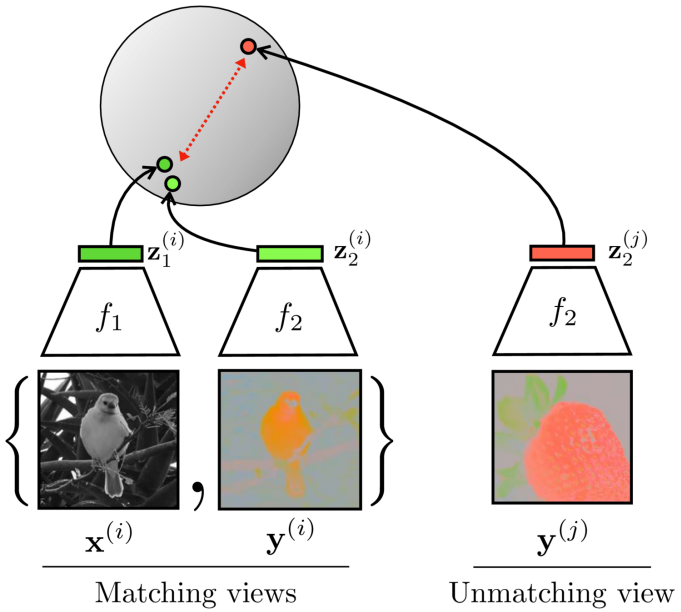
[images credit: visionbook.mit.edu]

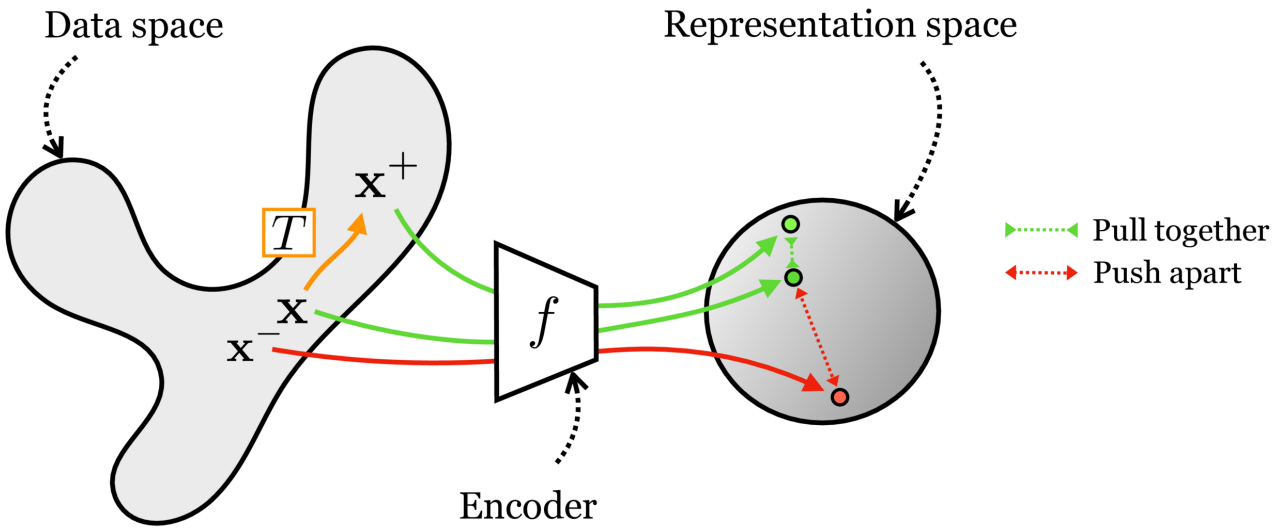
Contrastive learning



[images credit: visionbook.mit.edu]

Contrastive learning





[images credit: visionbook.mit.edu]

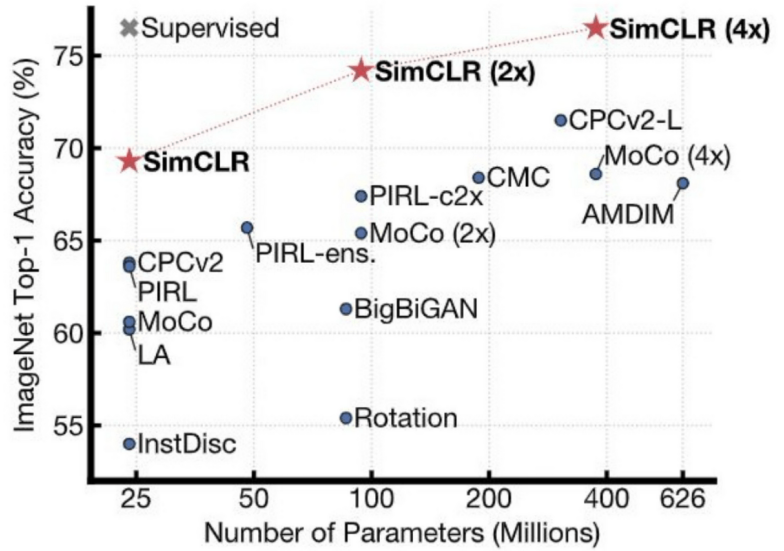
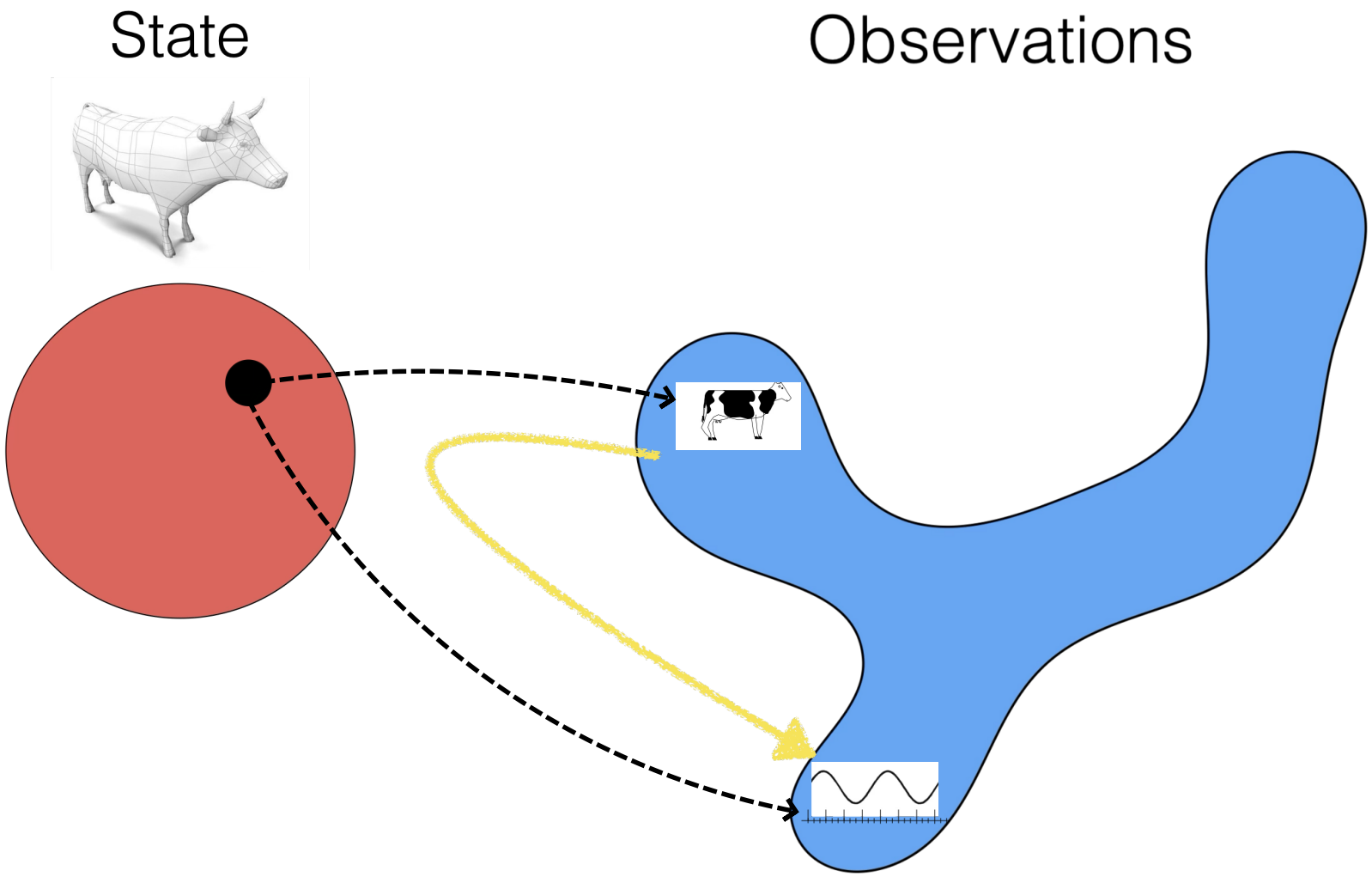


Figure 1. ImageNet top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Our method, SimCLR, is shown in bold.

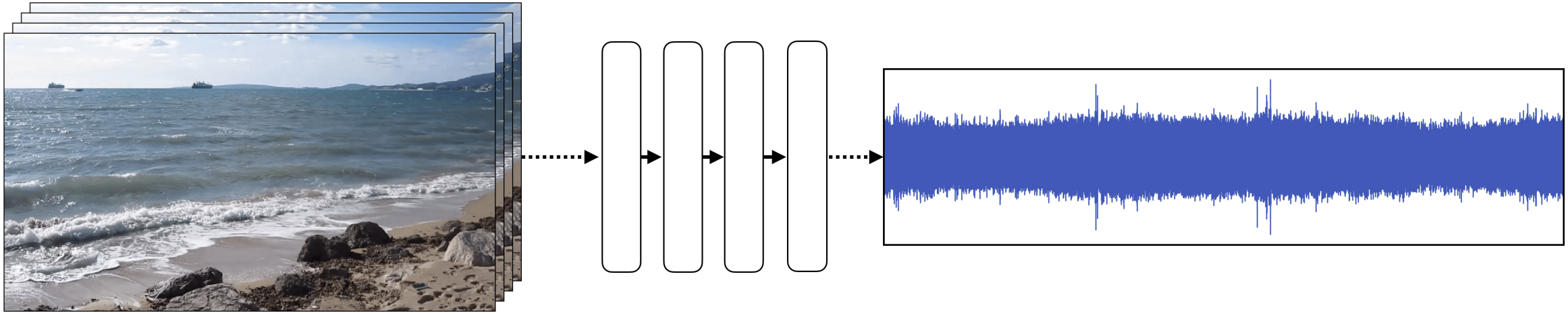
[Chen, Kornblith, Norouzi, Hinton, ICML 2020]

Multi-modality



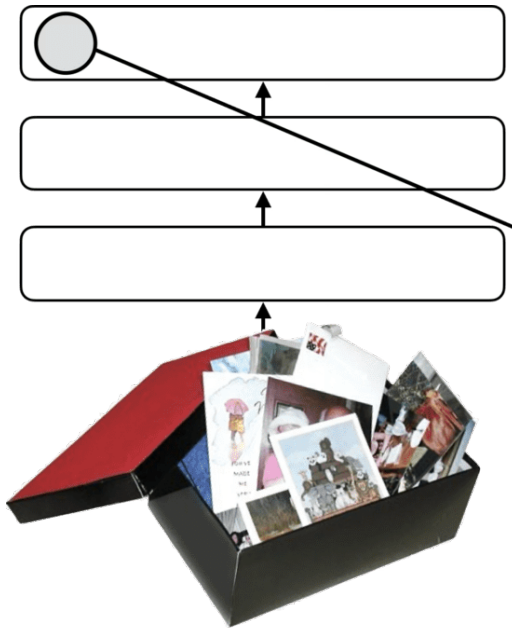
[images credit: visionbook.mit.edu]

Video - audio



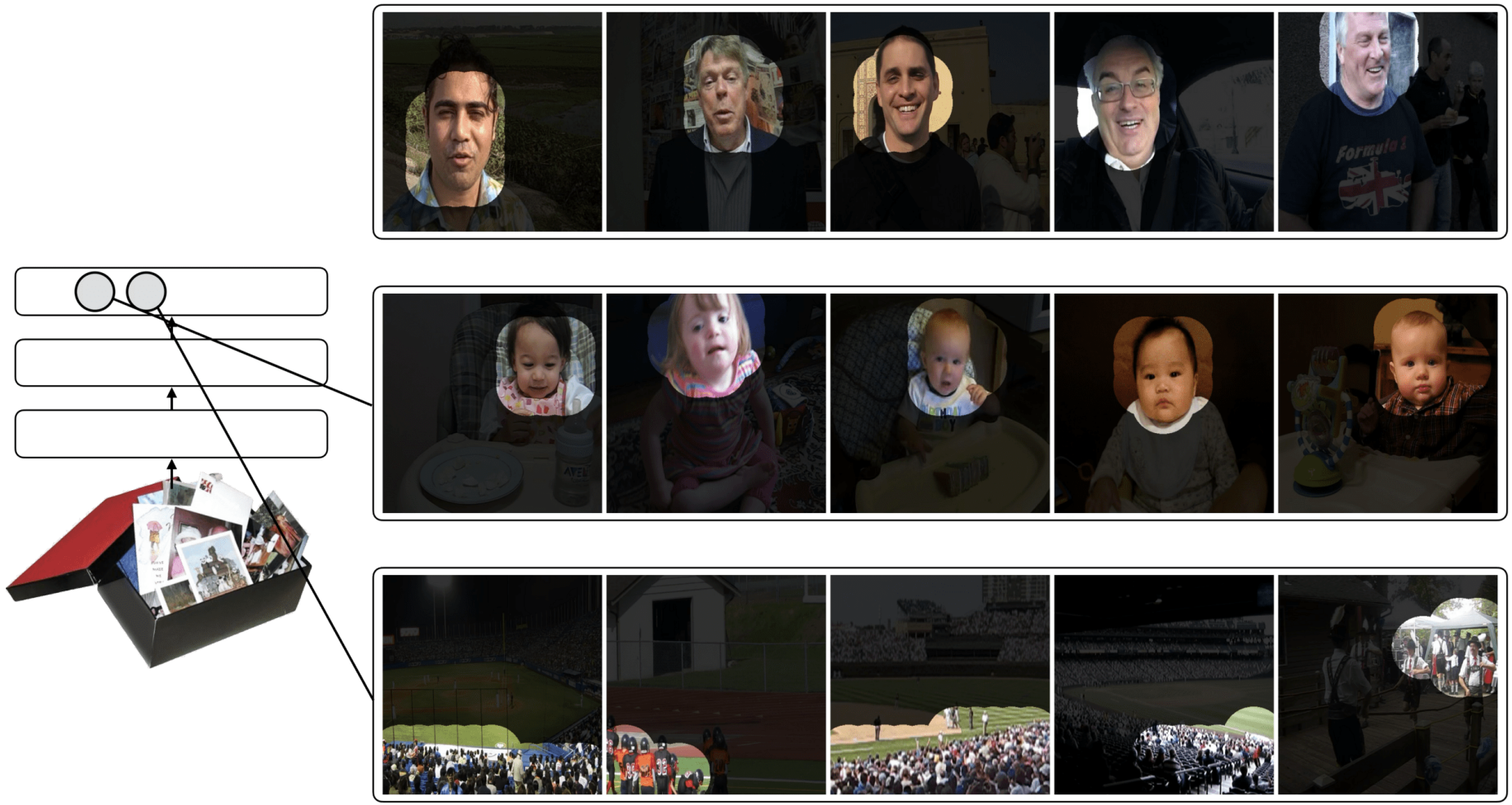
[Owens et al, Ambient Sound Provides Supervision for Visual Learning, ECCV 2016]
[Slide credit: Andrew Owens]

What did the model learn?



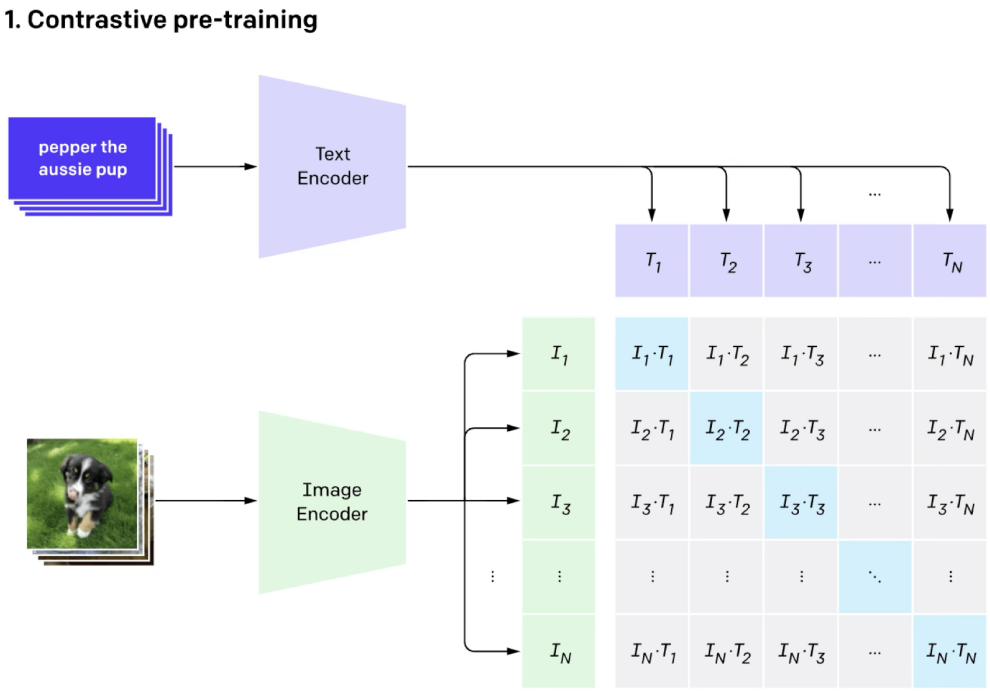
Strongest responses in dataset

[Owens et al, Ambient Sound Provides Supervision for Visual Learning, ECCV 2016]
[Slide credit: Andrew Owens]



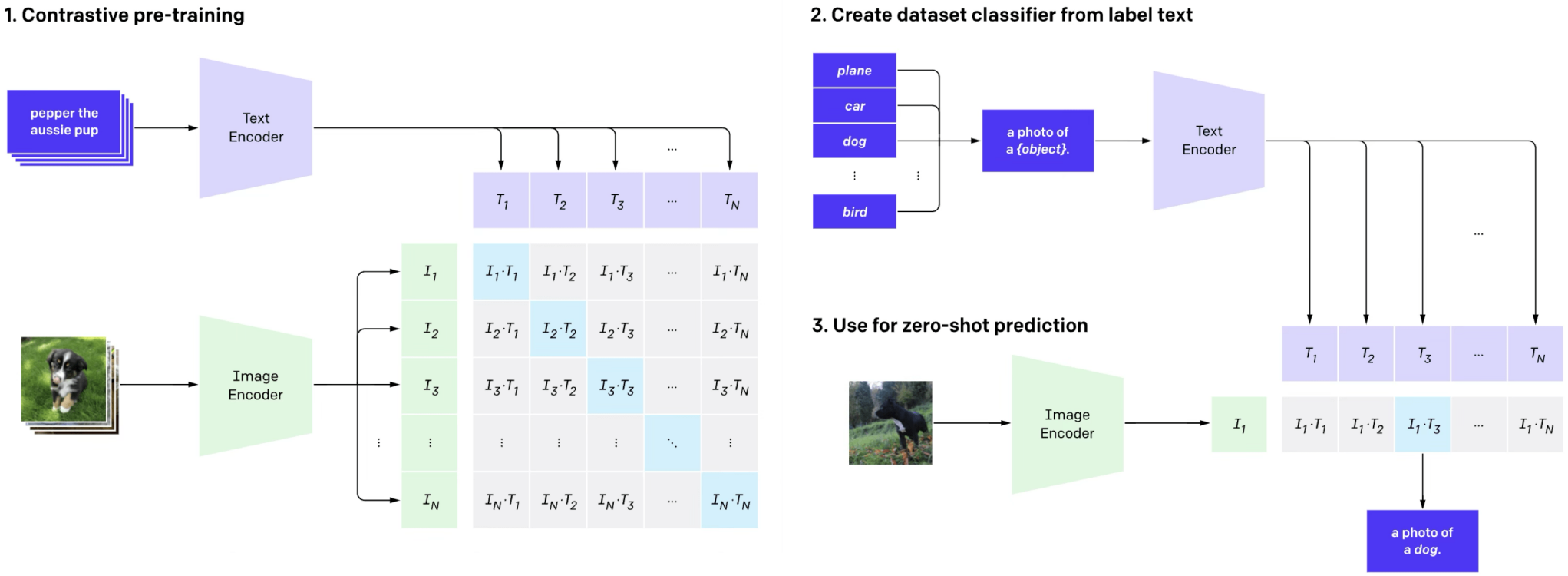
[Owens et al, Ambient Sound Provides Supervision for Visual Learning, ECCV 2016]
[Slide credit: Andrew Owens]

Image classification (done in the contrastive way)




```
# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

Image classification (done in the contrastive way)



Dall-E: text-image generation

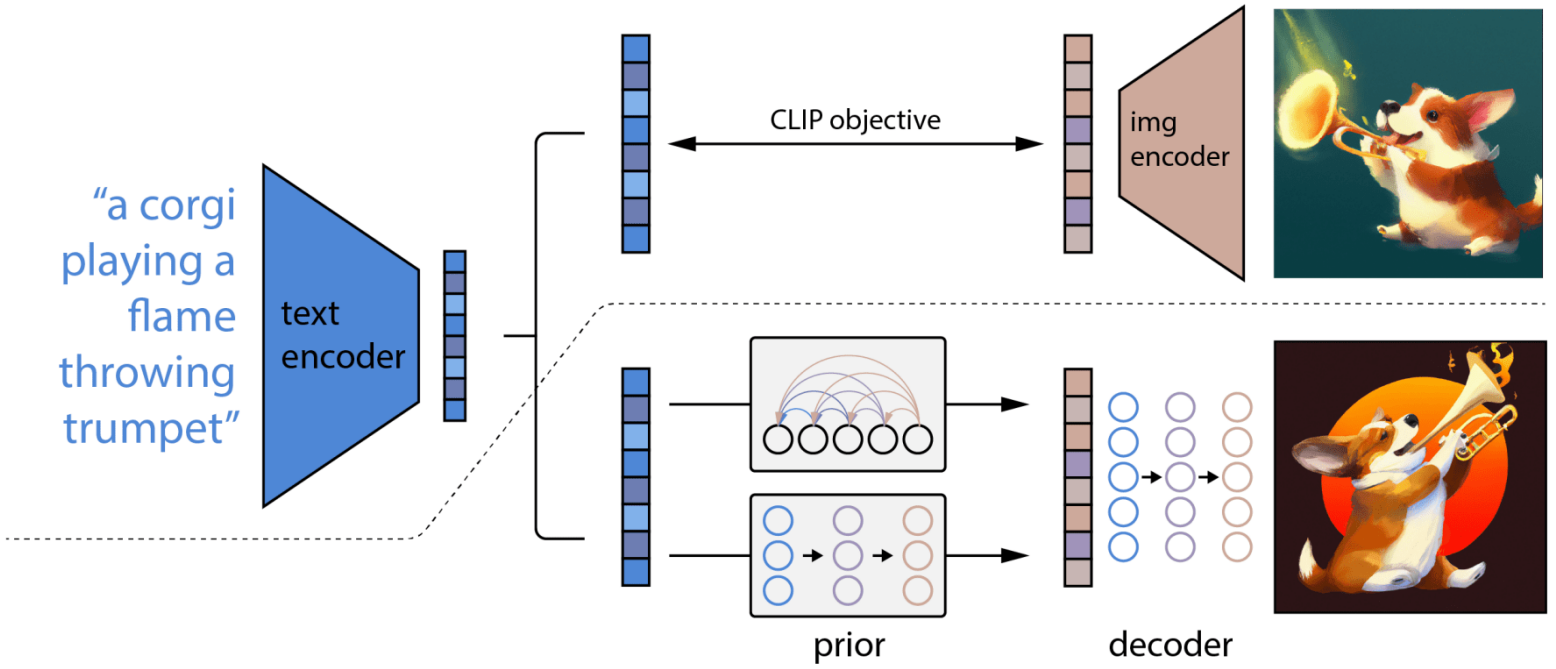


Figure 2: A high-level overview of unCLIP. Above the dotted line, we depict the CLIP training process, through which we learn a joint representation space for text and images. Below the dotted line, we depict our text-to-image generation process: a CLIP text embedding is first fed to an autoregressive or diffusion prior to produce an image embedding, and then this embedding is used to condition a diffusion decoder which produces a final image. Note that the CLIP model is frozen during training of the prior and decoder.

[https://arxiv.org/pdf/2204.06125.pdf]

How Much Information is the Machine Given during Learning?

Y. LeCun

- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Self-Supervised Learning (cake génoise)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

[Slide Credit: Yann LeCun]

Summary

- We looked at the mechanics of neural net. Today we see deep nets learn representations, just like our brains do.
- This is useful because representations transfer — they act as prior knowledge that enables quick learning on new tasks.
- Representations can also be learned without labels, e.g. as we do in unsupervised, or self-supervised learning. This is great since labels are expensive and limiting.
- Without labels there are many ways to learn representations. We saw today:
  - representations as compressed codes, auto-encoder with bottleneck
  - (representations that are predictive of their context)
  - (representations that are shared across sensory modalities)

https://forms.gle/GDq3R5Cdb6iRCN62A

We'd love to hear
your thoughts.

Thanks!