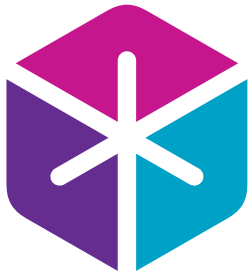
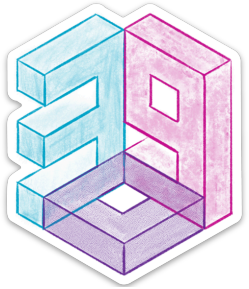




https://introml.mit.edu/

6.390 Intro to Machine Learning

Lecture 8: Representation Learning

Shen Shen

Mar 30, 2026

3pm, Room 10-250

Slides and Lecture Recording

Outline

Outline

- Neural networks are *representation* learners

Outline

- Neural networks are *representation* learners
- Self-supervised learning

Outline

- Neural networks are *representation* learners
- Self-supervised learning
- Auto-encoders

Outline

- Neural networks are *representation* learners
- Self-supervised learning
- Auto-encoders
- Word embeddings

Outline

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

Outline

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

Recap

Recap

input

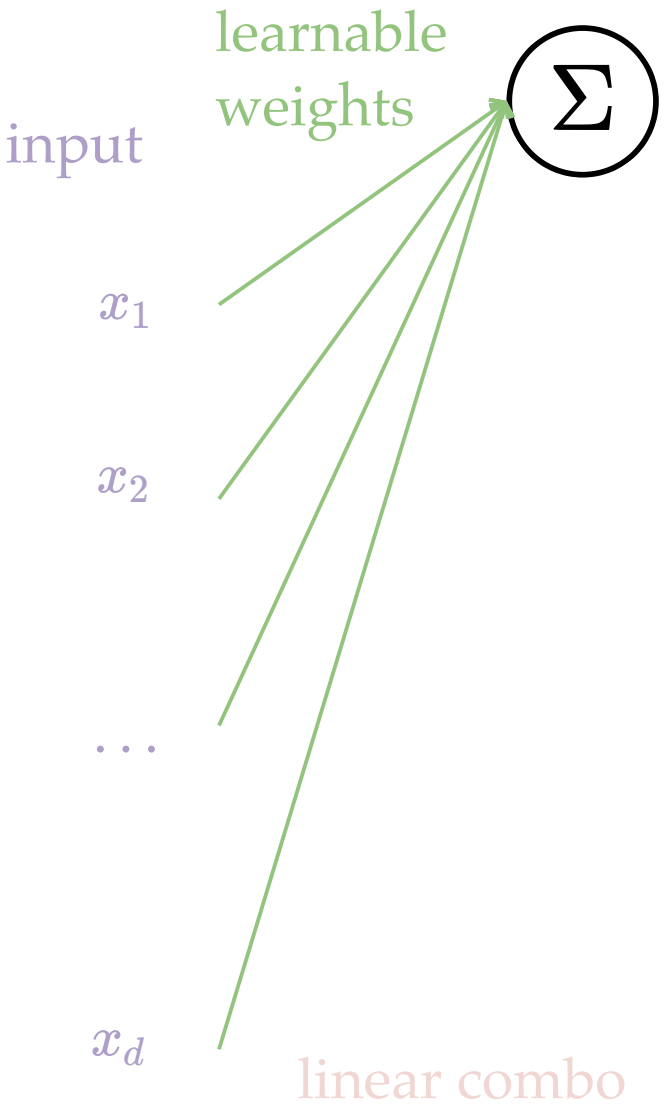
x_1

x_2

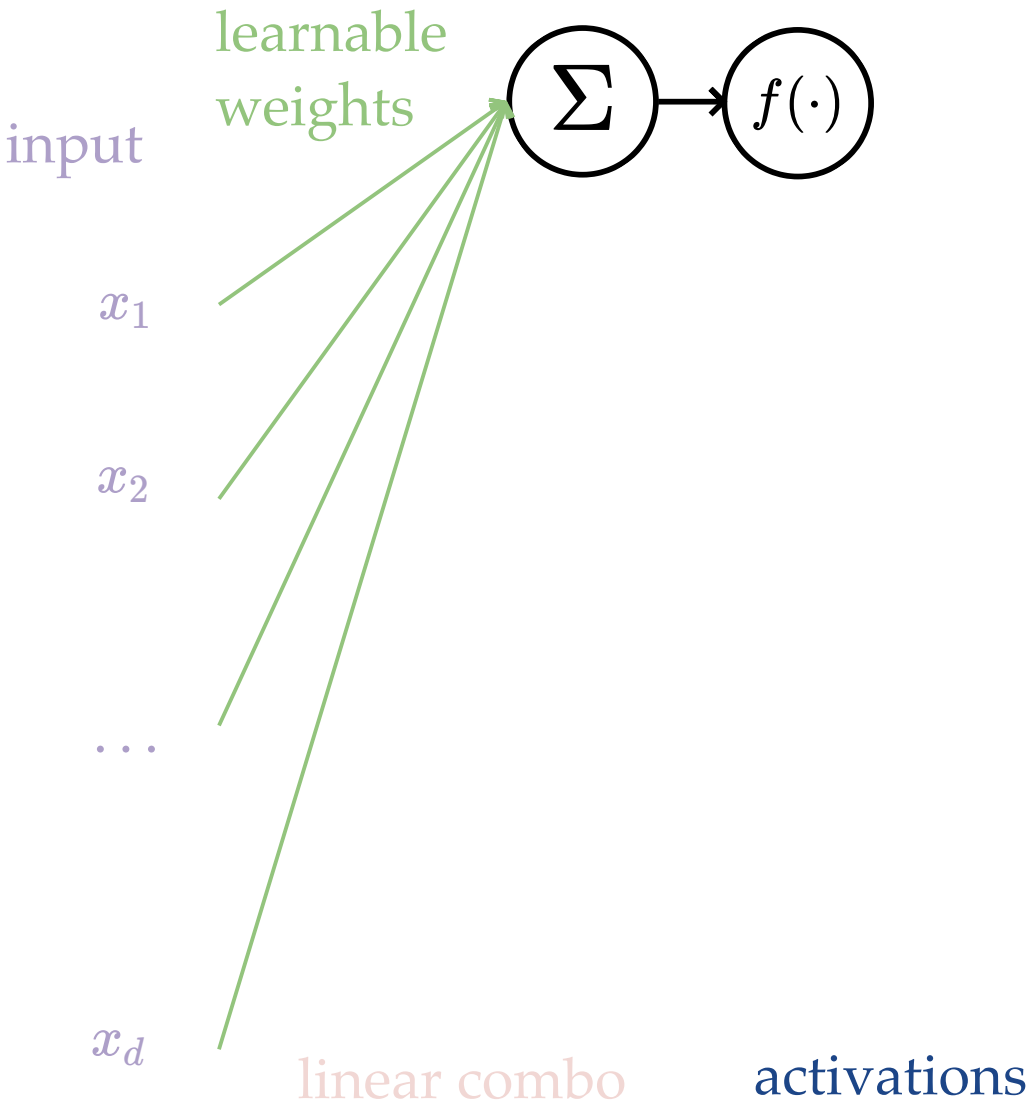
...

x_d

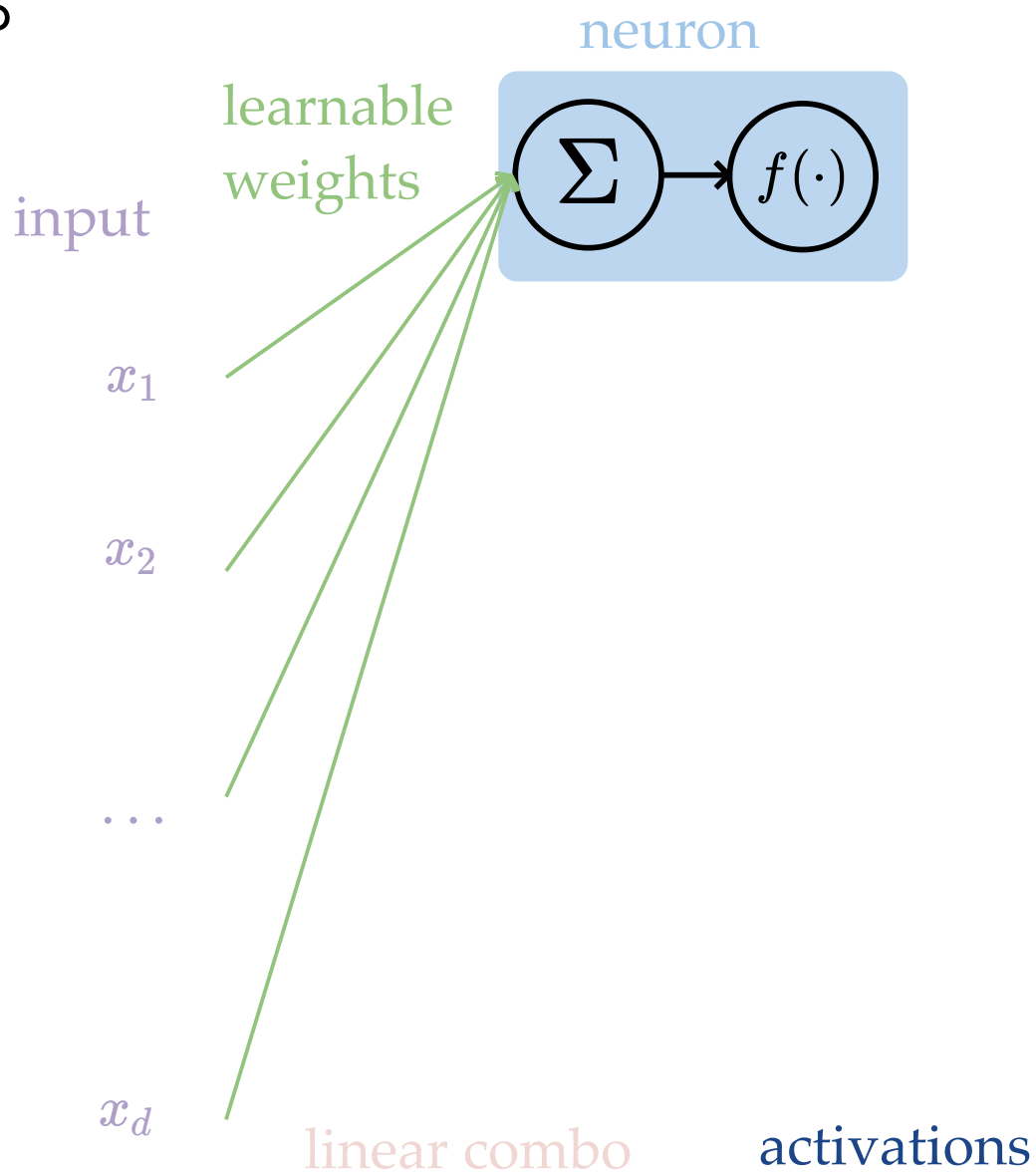
Recap



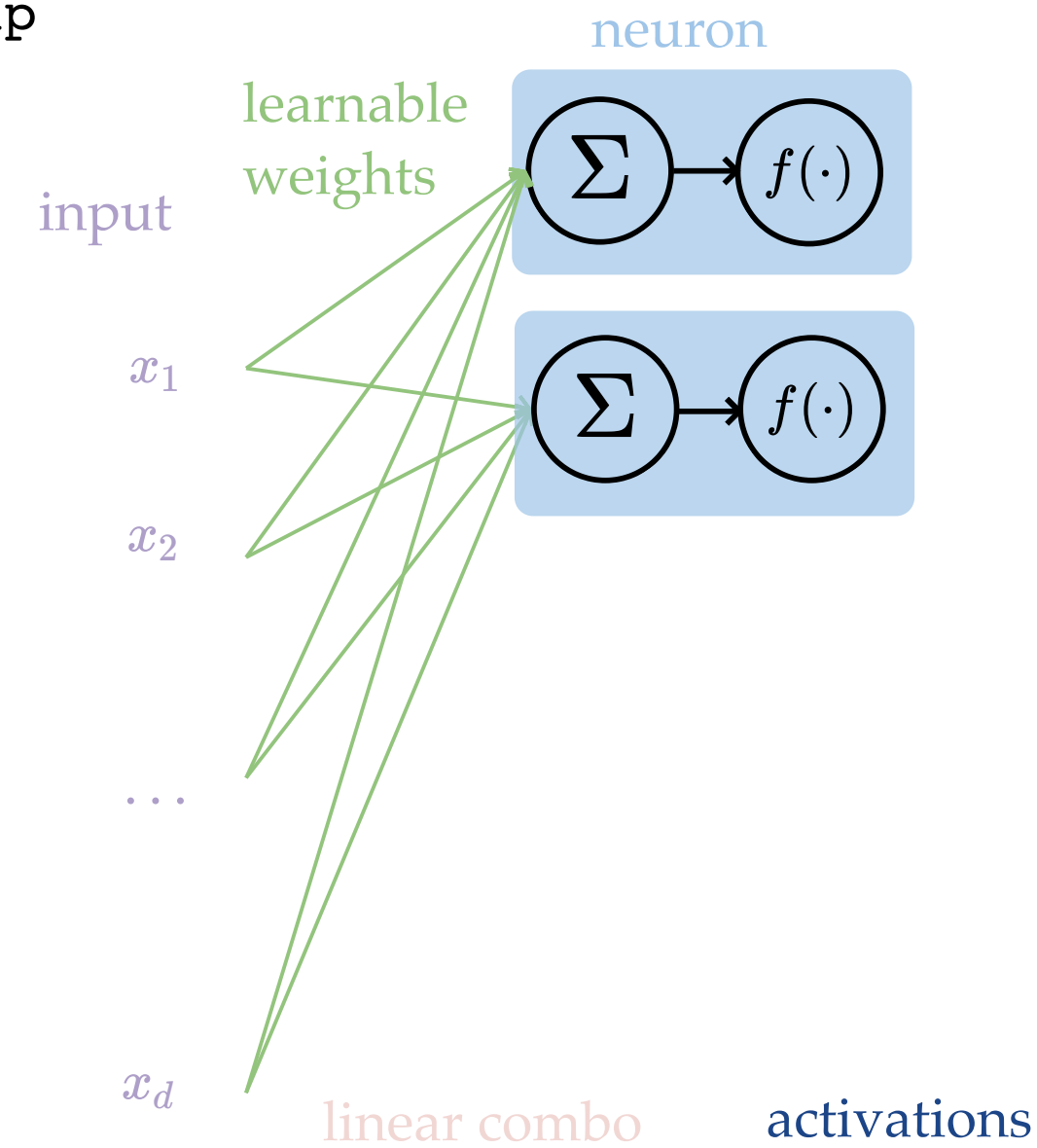
Recap


learnable
weights
input
Σ
$f(\cdot)$
x_1
x_2
$\dots$
x_d
linear combo
activations

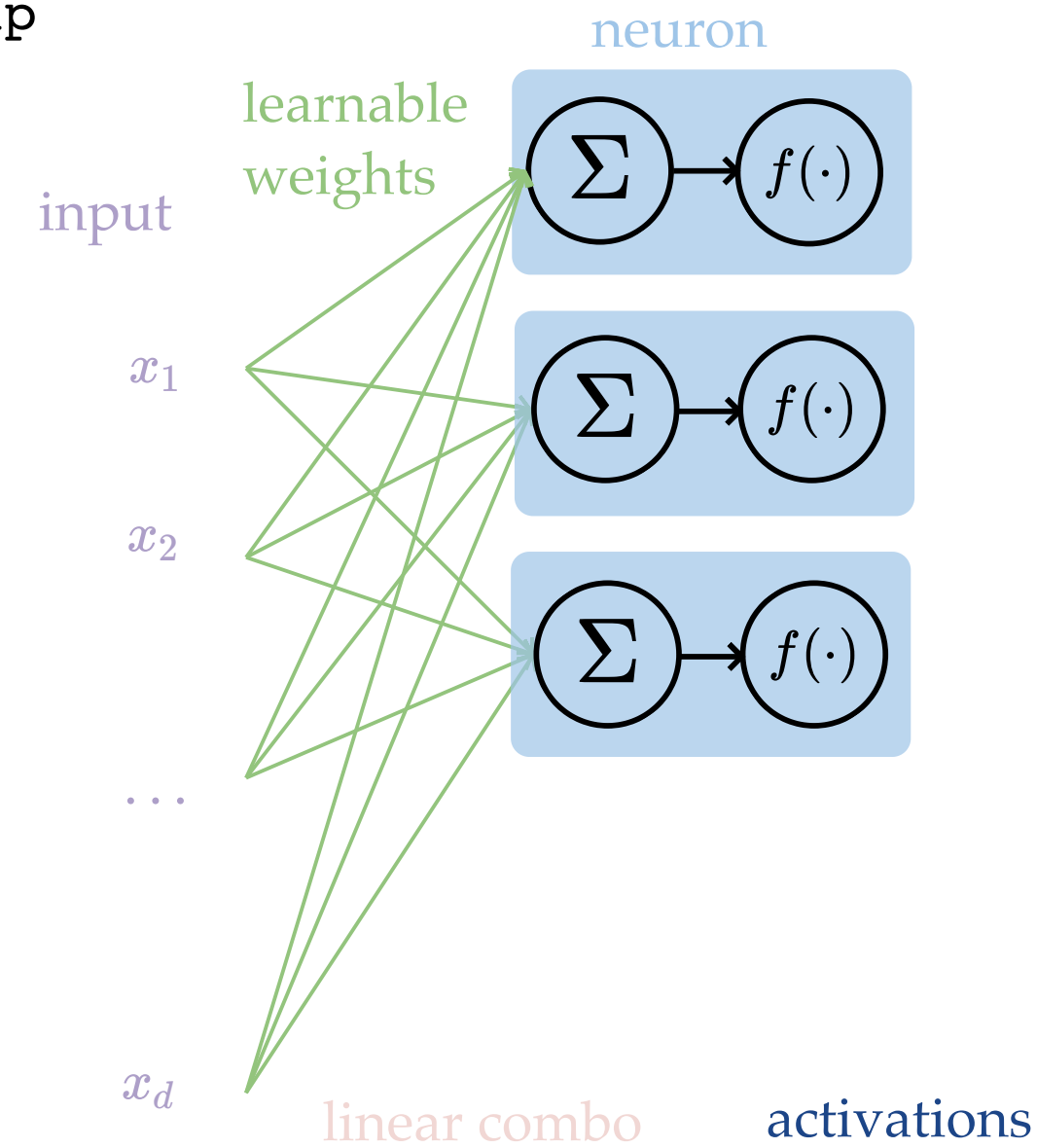
Recap


neuron
learnable
weights
input
Σ
$f(\cdot)$
x_1
x_2
$\cdots$
x_d
linear combo
activations

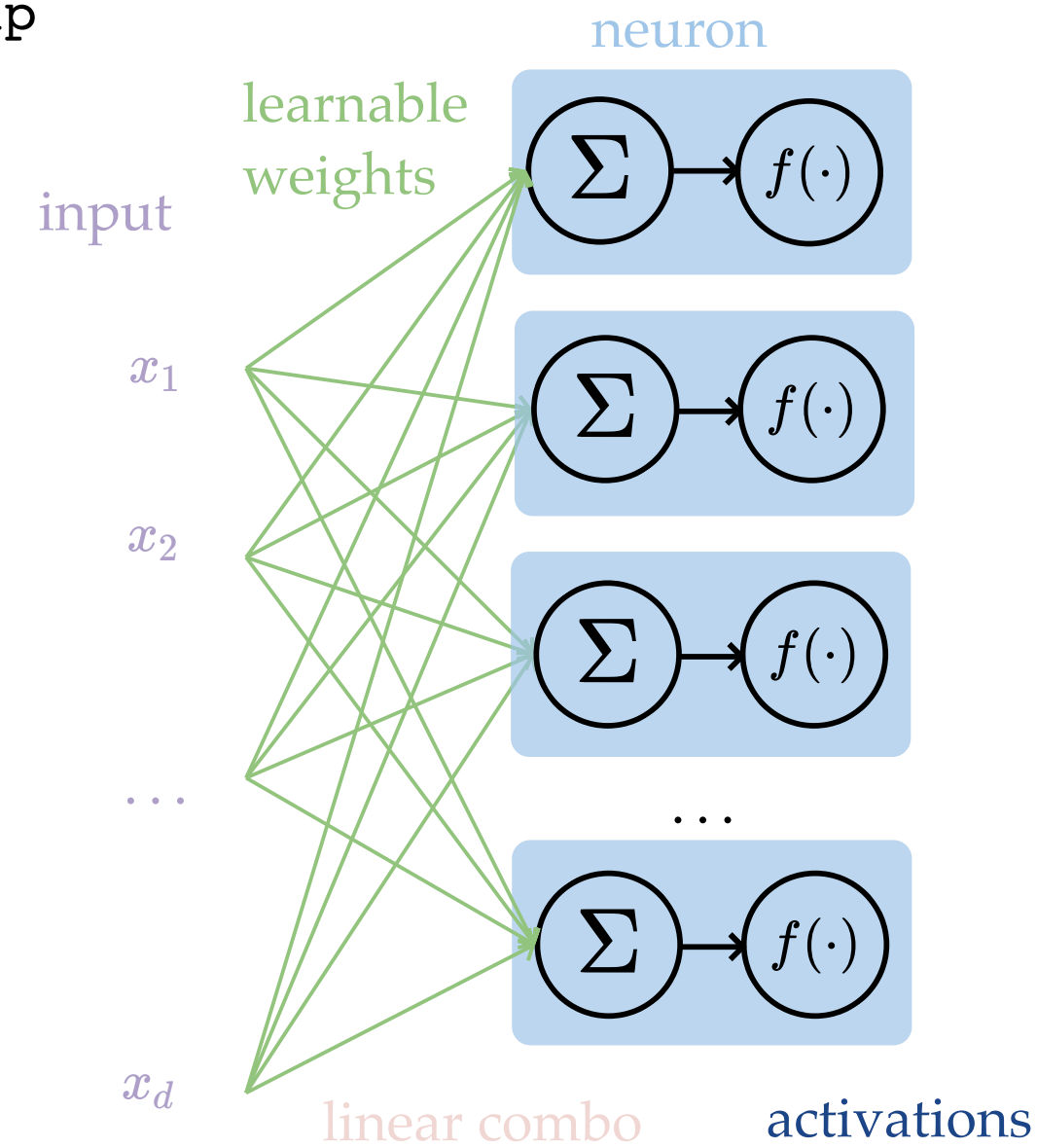
Recap


neuron
learnable
weights
input
x_1
x_2
$\cdots$
x_d
Σ
$f(\cdot)$
Σ
$f(\cdot)$
linear combo
activations

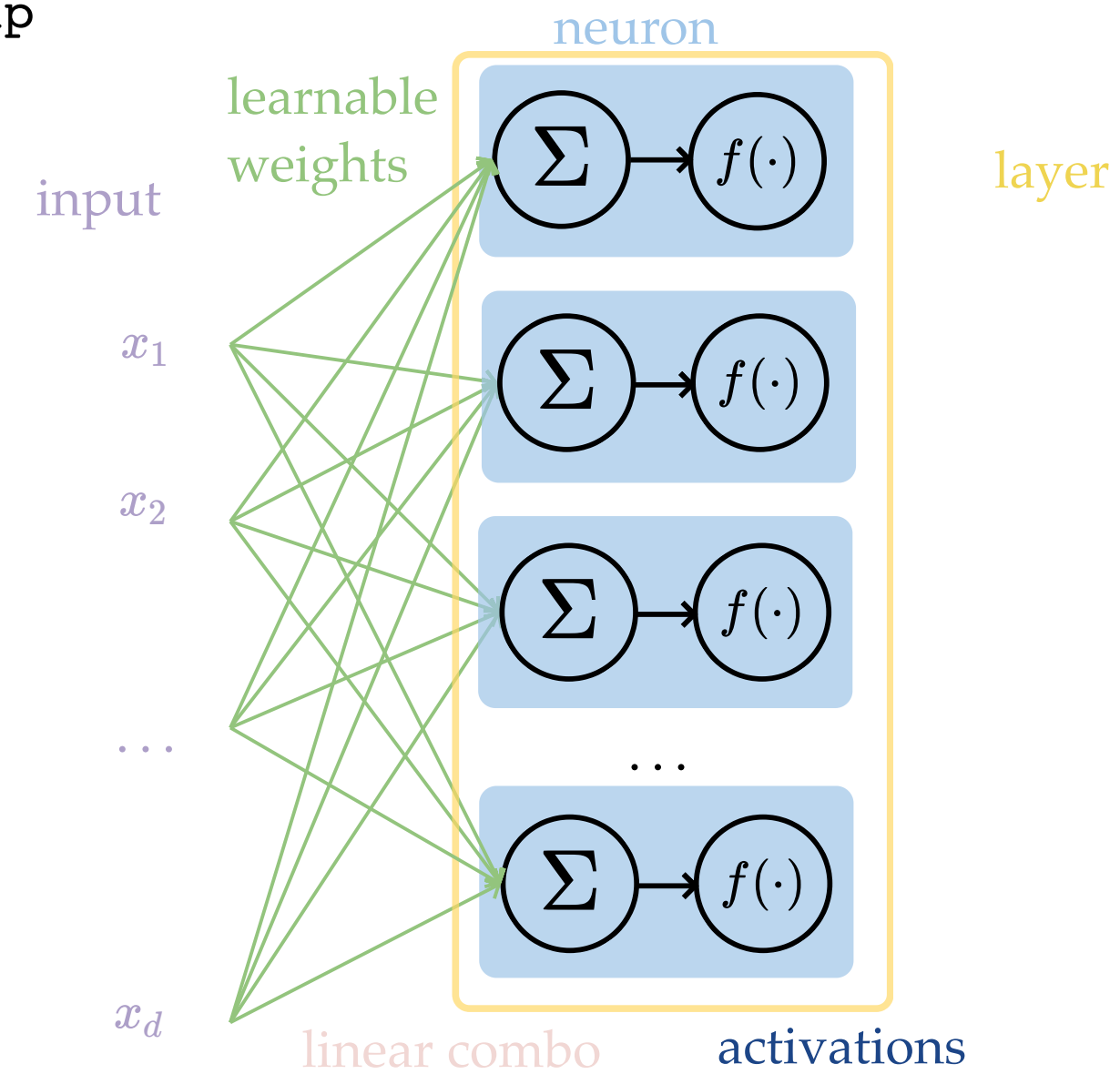
Recap


neuron
learnable
weights
input
x_1
x_2
$\cdots$
x_d
Σ
$f(\cdot)$
Σ
$f(\cdot)$
Σ
$f(\cdot)$
linear combo
activations

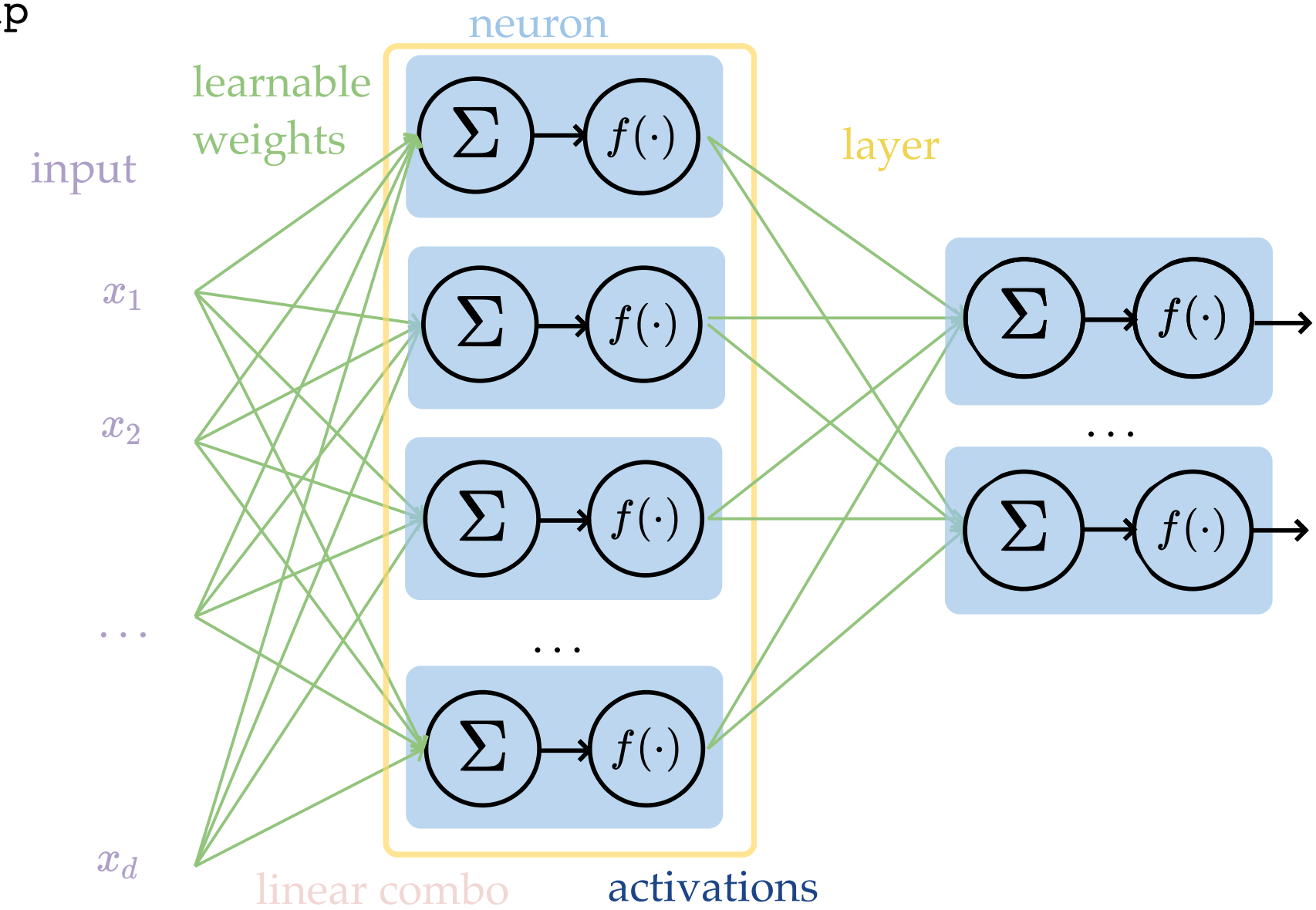
Recap


neuron
learnable
weights
input
x_1
x_2
$\cdots$
x_d
Σ
$f(\cdot)$
$\cdots$
linear combo
activations

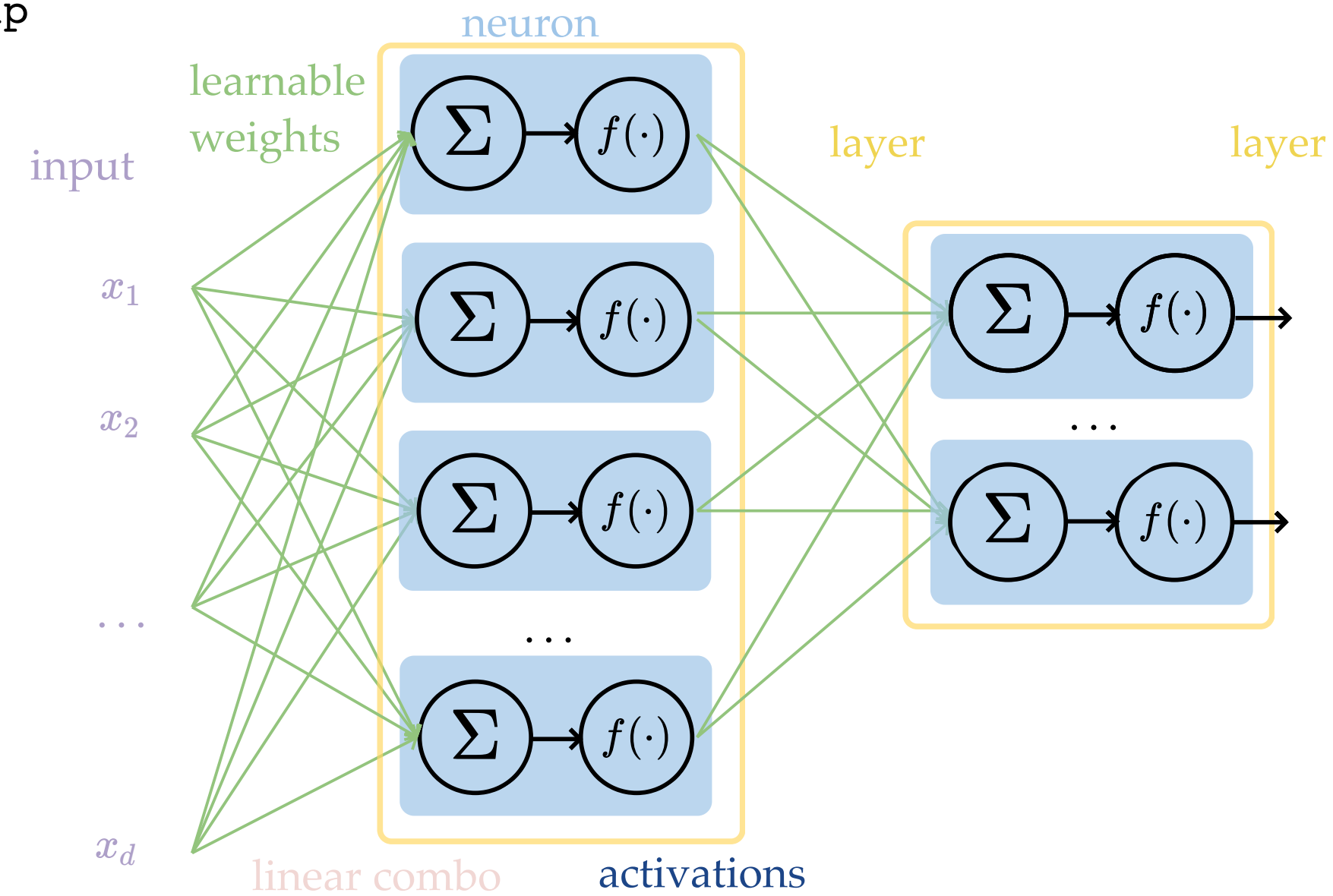
Recap


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

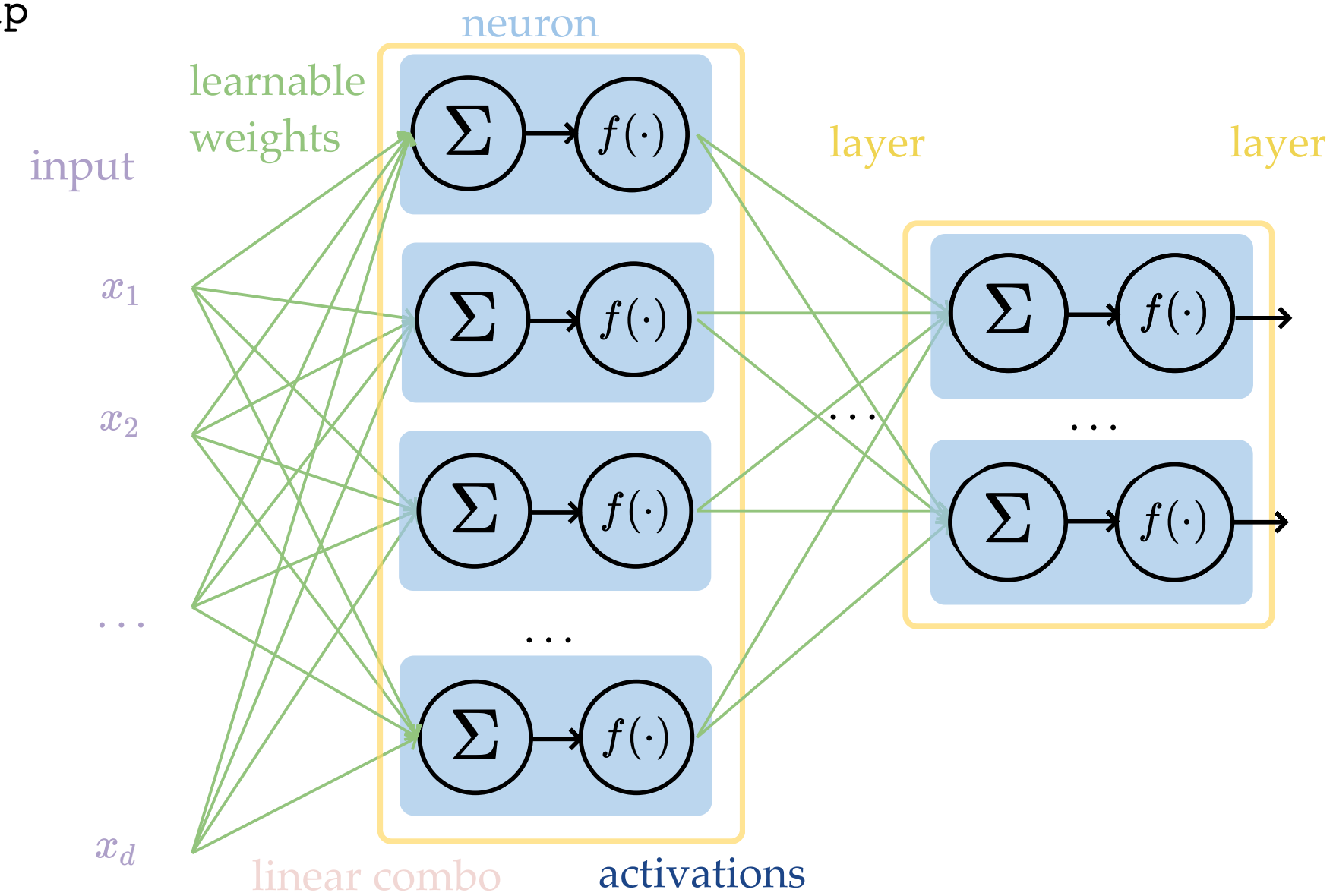
Recap


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

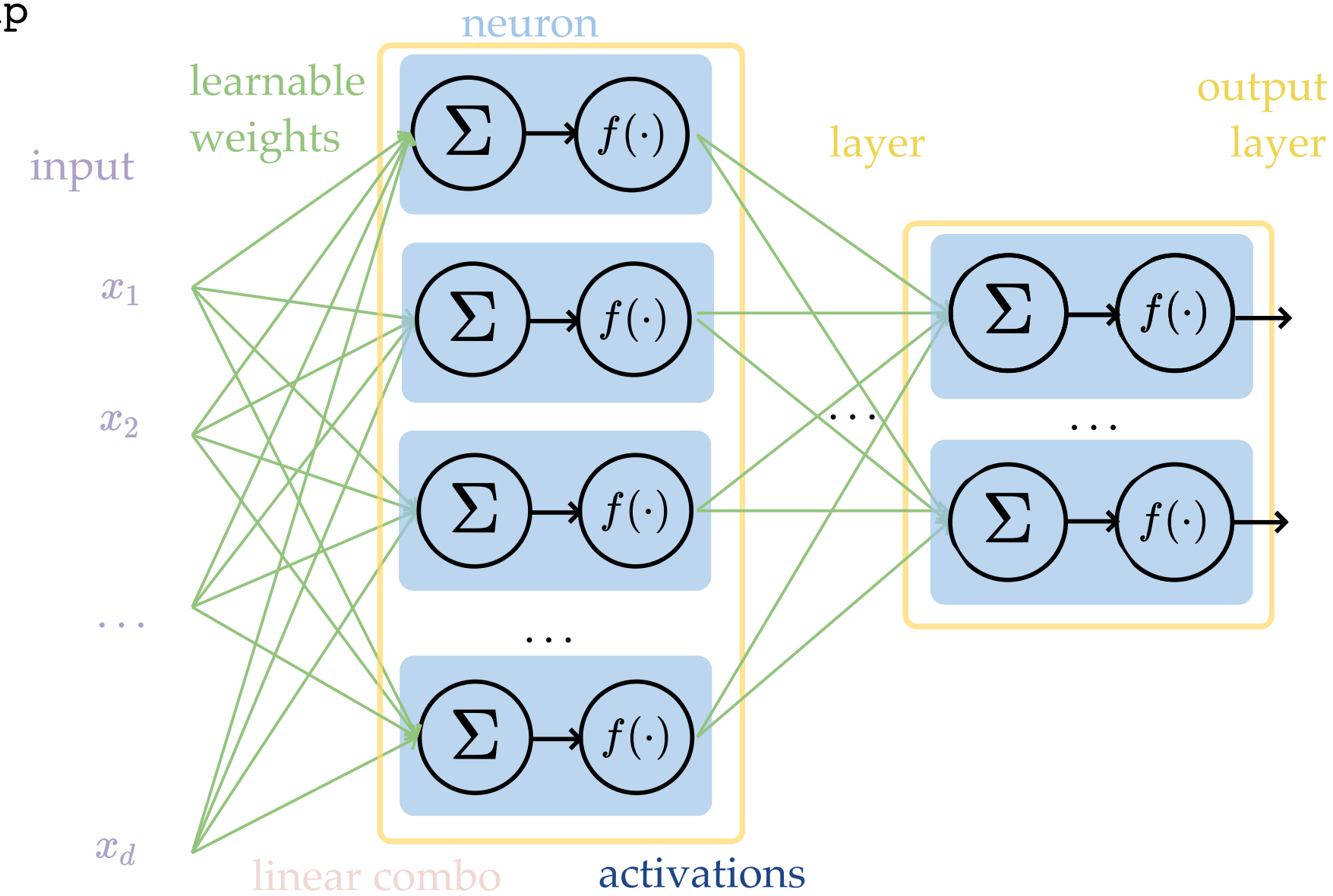
Recap


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

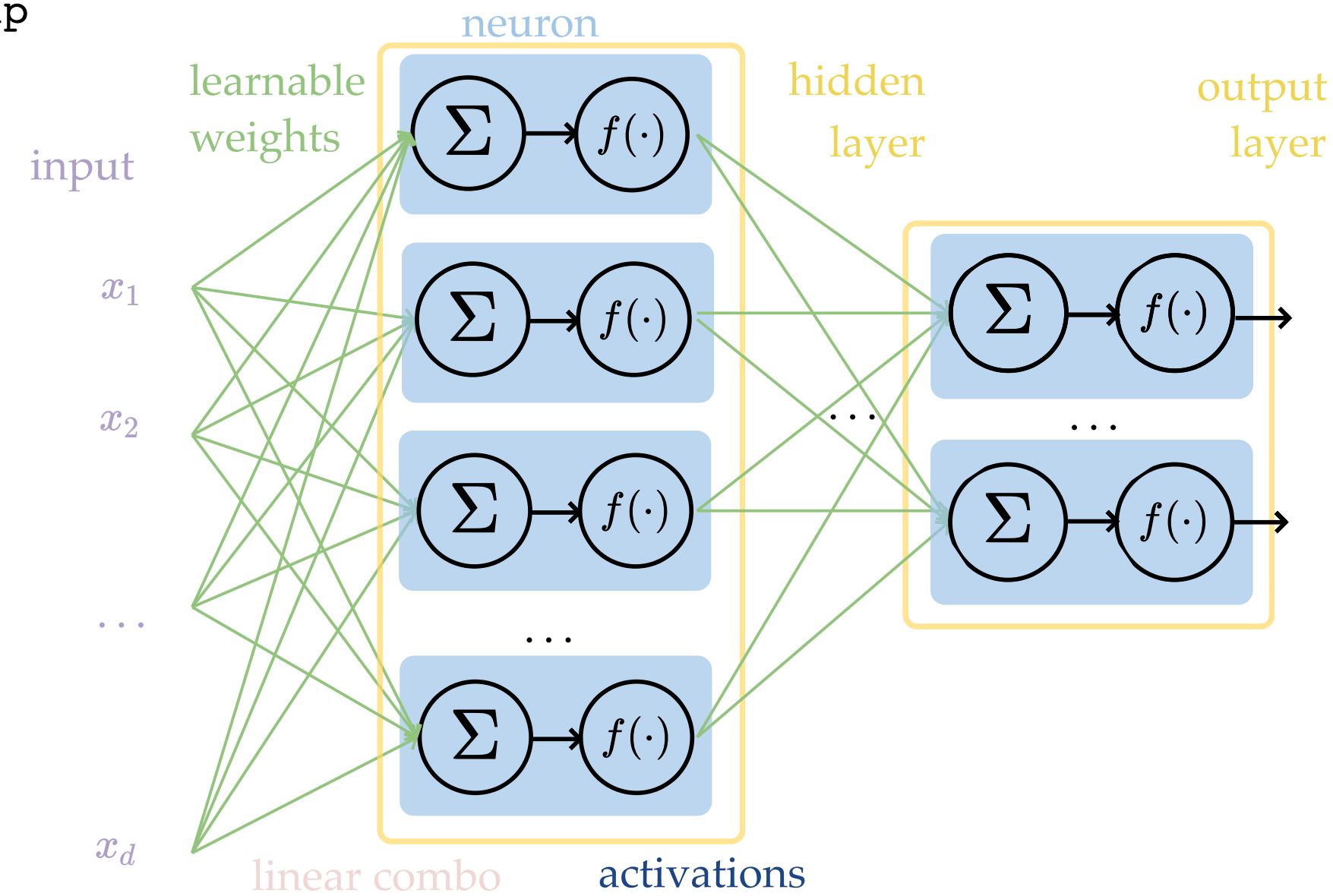
Recap

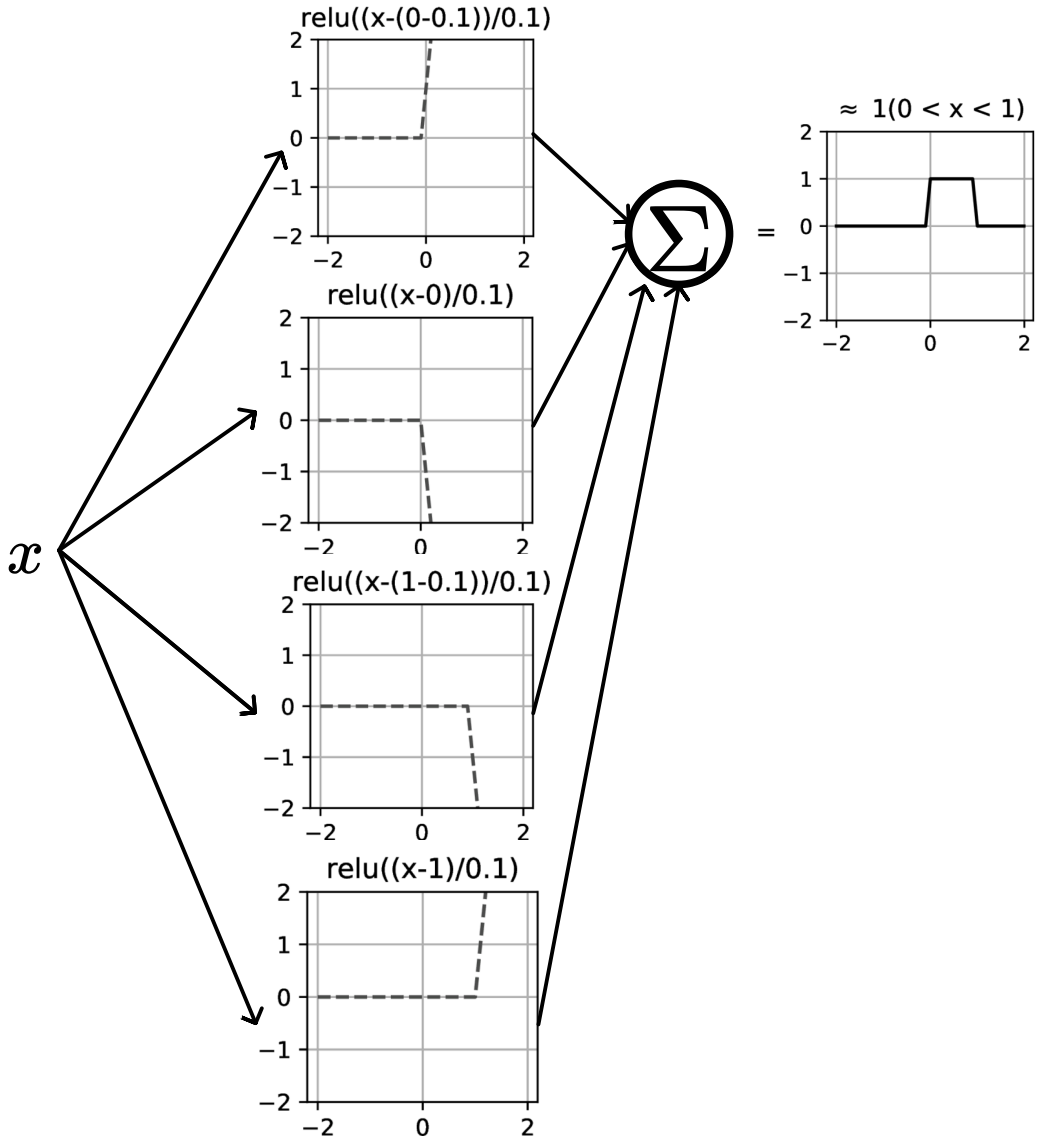

neuron
learnable
weights
input
layer
layer
x_1
x_2
$\cdots$
x_d
Σ
$f(\cdot)$
linear combo
activations

Recap

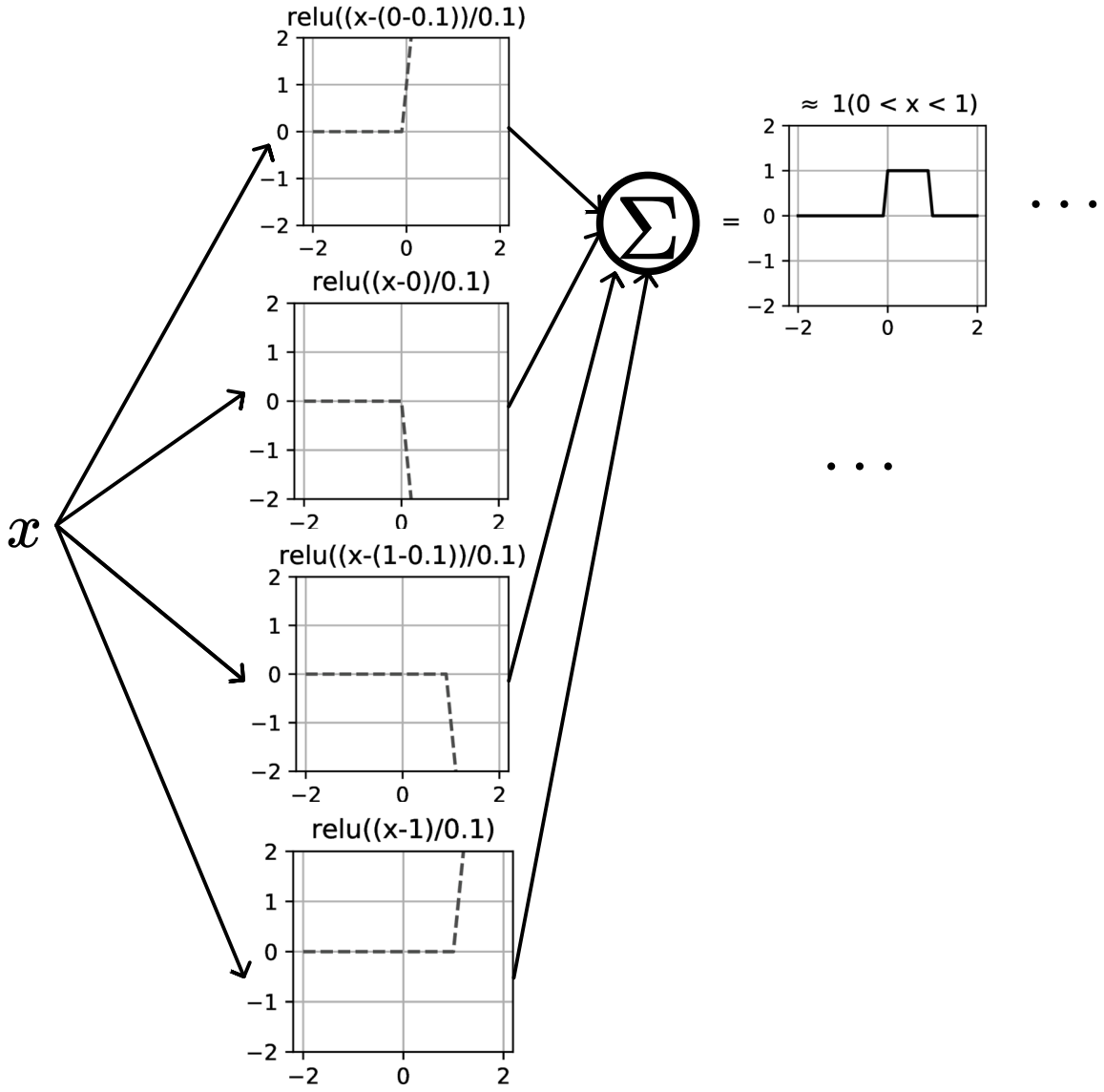

neuron
learnable
weights
input
layer
output
layer
x_1
x_2
$\cdots$
x_d
Σ
$f(\cdot)$
linear combo
activations

Recap

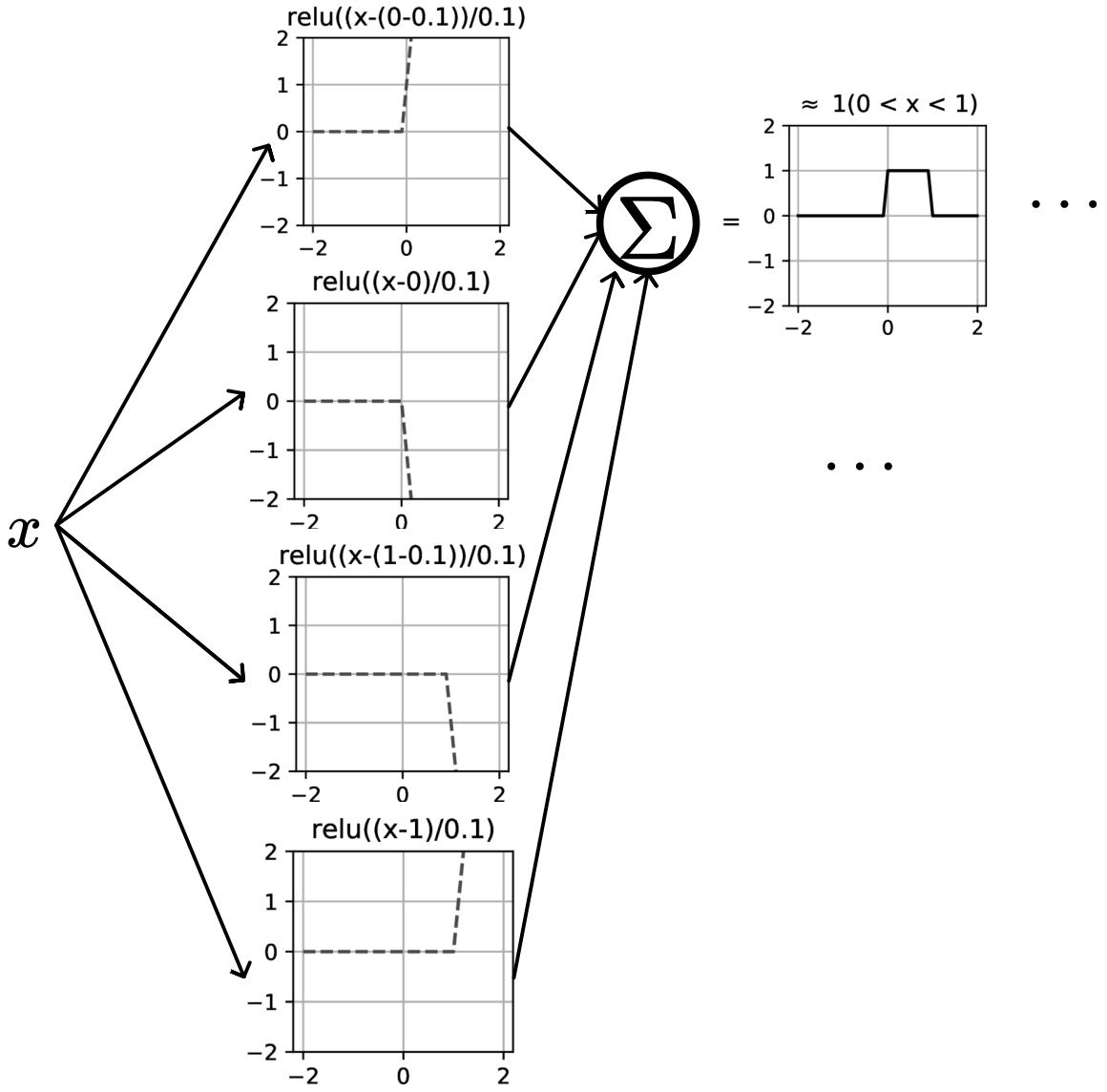

neuron
learnable
weights
hidden
layer
output
layer
input
x_1
x_2
$\cdots$
x_d
Σ
$f(\cdot)$
linear combo
activations



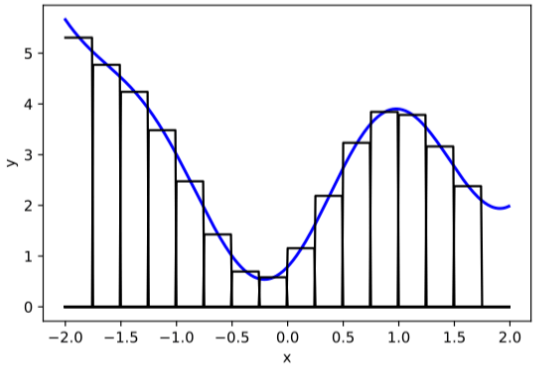
[images credit: visionbook.mit.edu]



[images credit: visionbook.mit.edu]



asymptotically, can approximate any function!



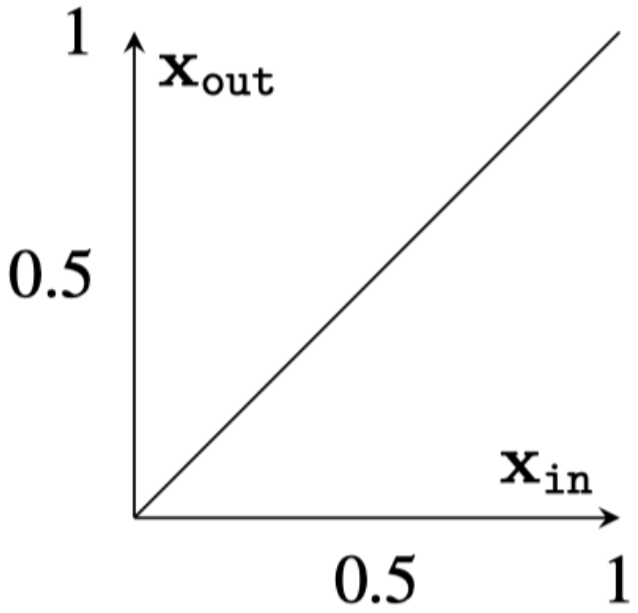
[images credit: visionbook.mit.edu]

Two different ways to visualize a function

[images credit: visionbook.mit.edu]

Two different ways to visualize a function

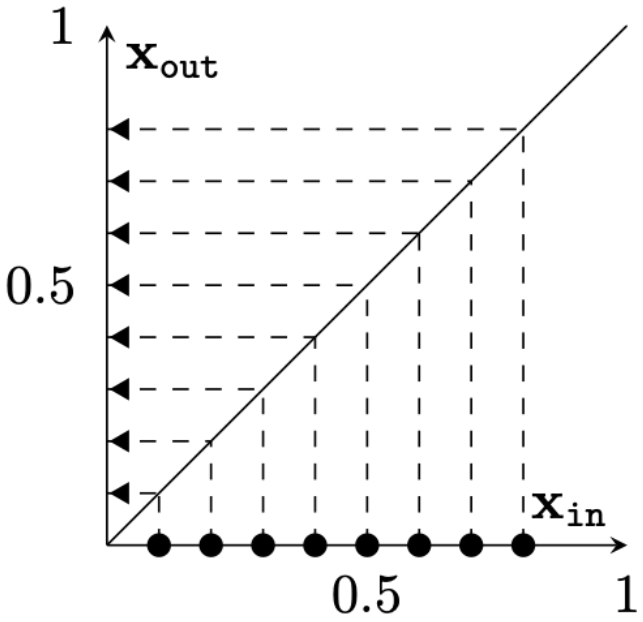
e.g. the identity function



[images credit: visionbook.mit.edu]

Two different ways to visualize a function

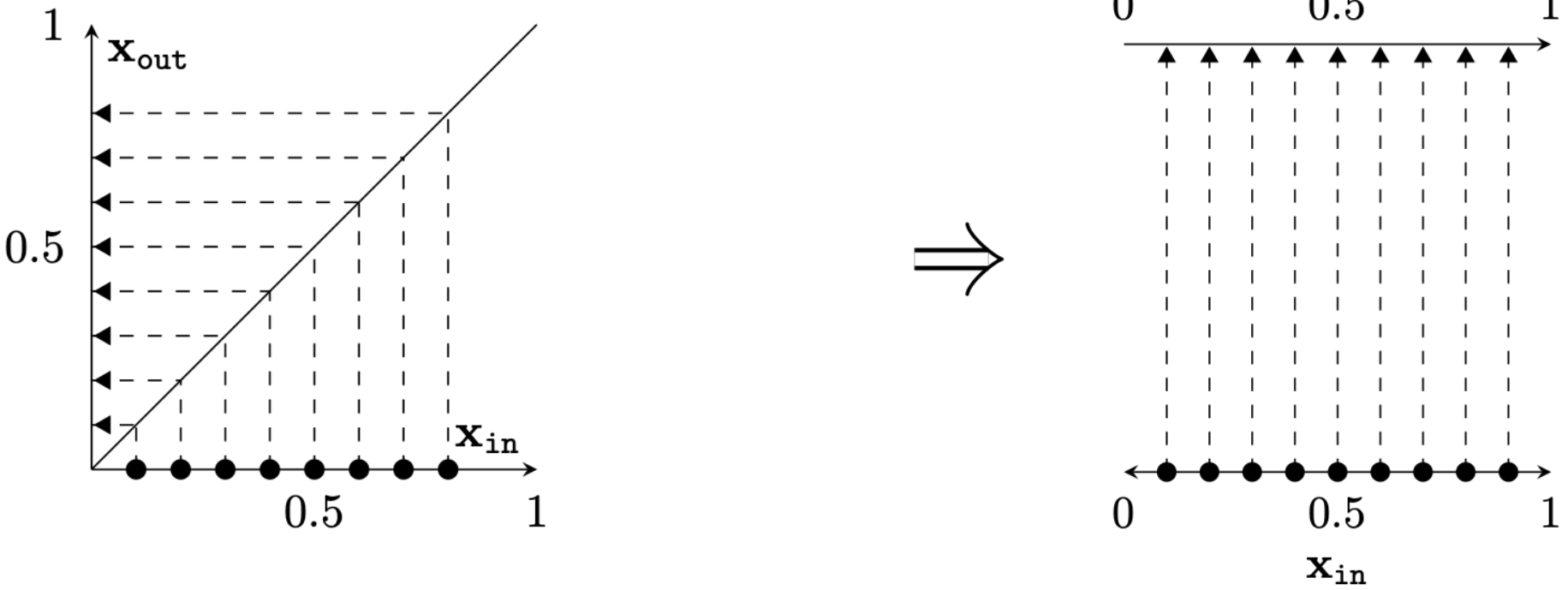
e.g. the identity function



[images credit: visionbook.mit.edu]

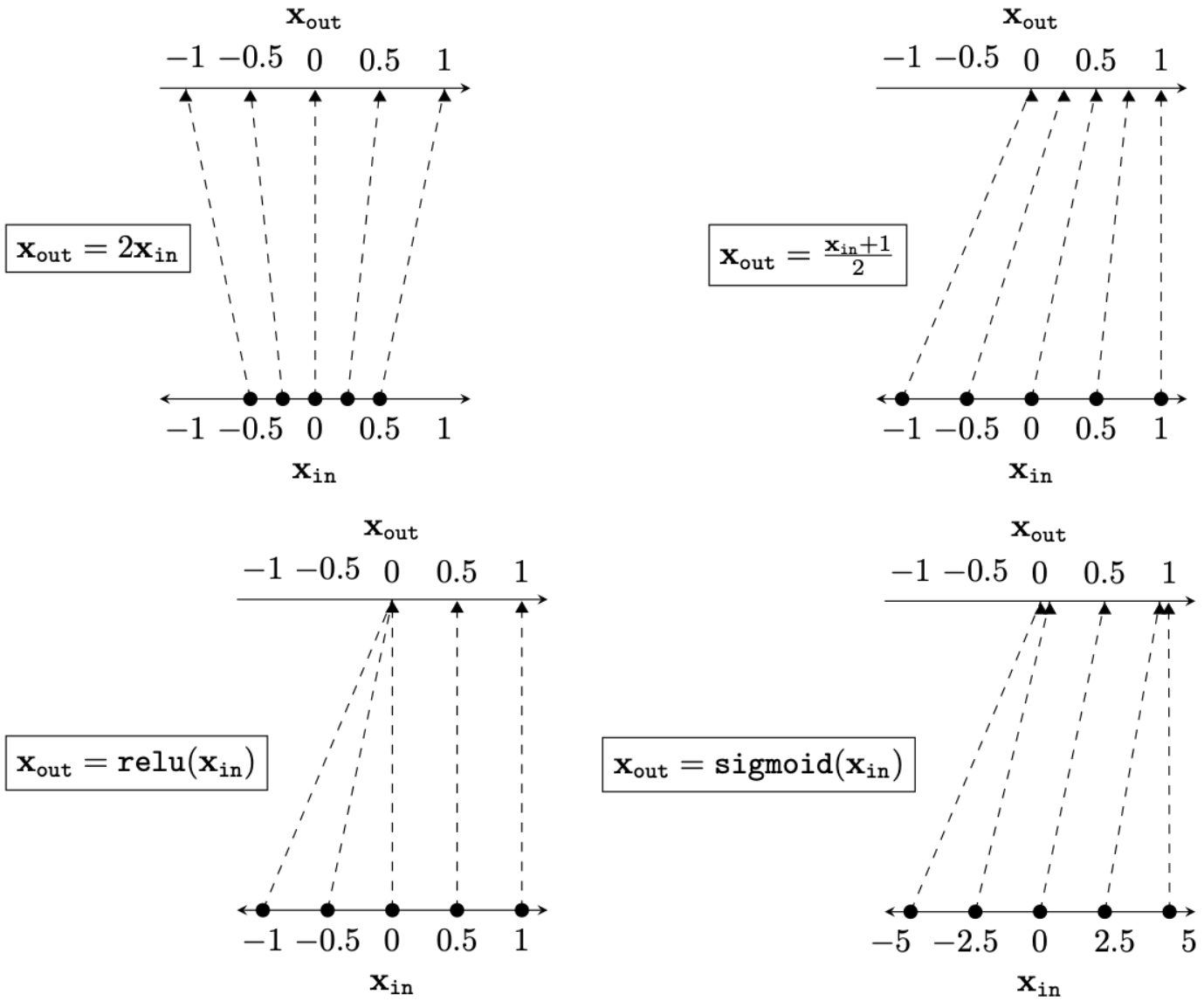
Two different ways to visualize a function

e.g. the identity function



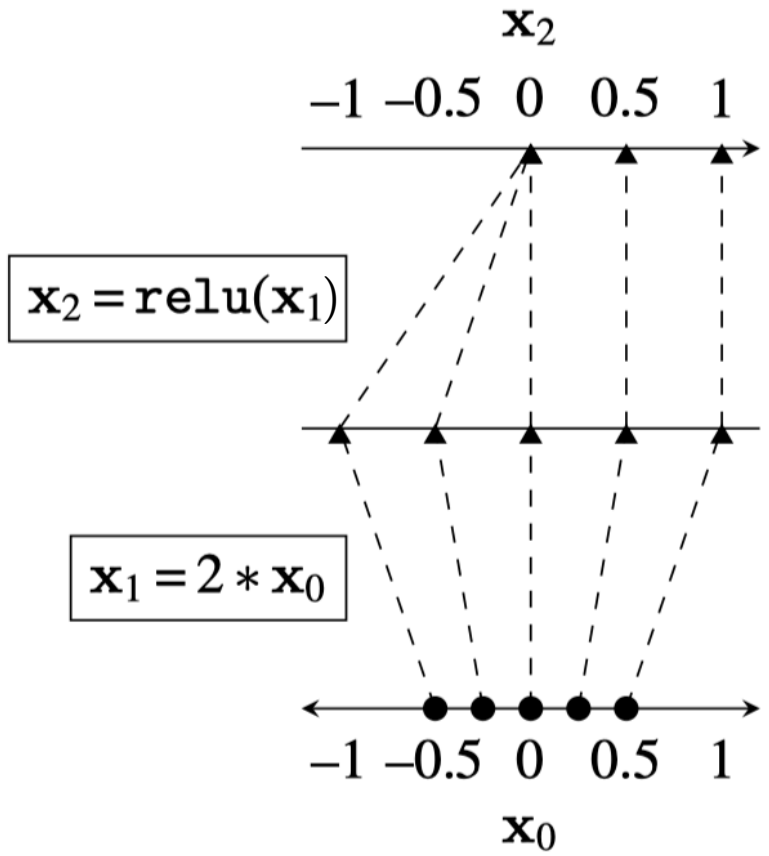
[images credit: visionbook.mit.edu]

Representation transformations for a variety of neural net operations



[images credit: visionbook.mit.edu]

and stack of neural net operations

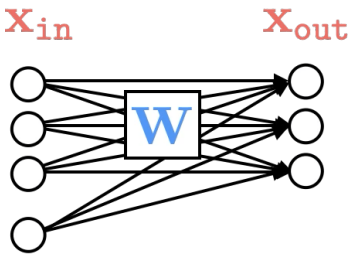
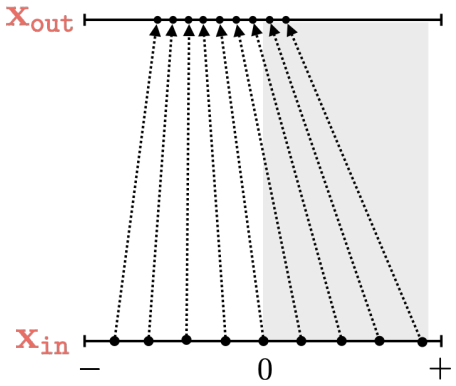
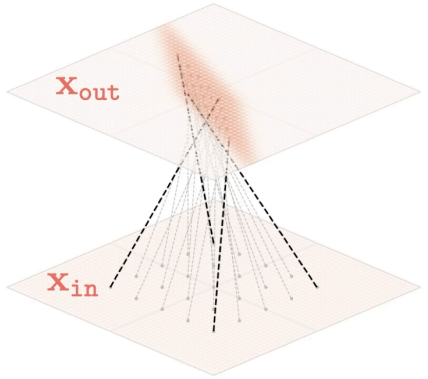


[images credit: visionbook.mit.edu]

Parameters

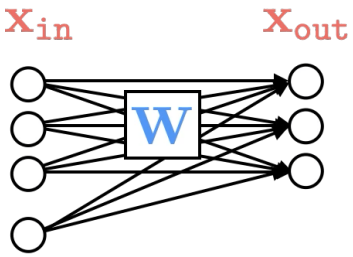
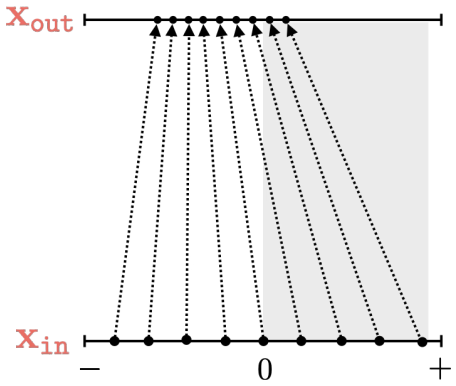
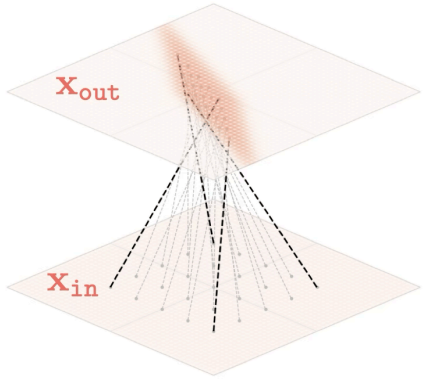
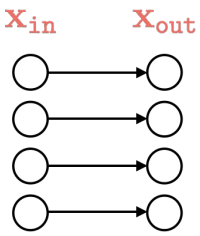
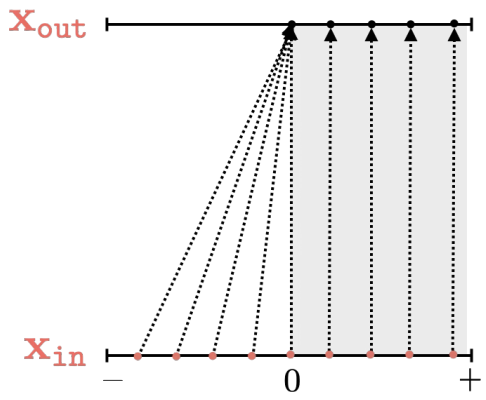
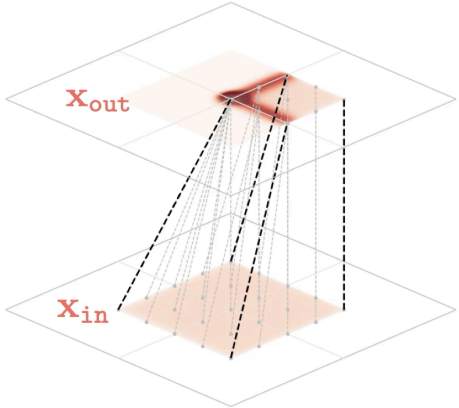
[images credit: visionbook.mit.edu]

Parameters

	wiring graph	equation	mapping 1D	mapping 2D
linear		$\mathbf{x}_{out} = \mathbf{W}\mathbf{x}_{in}$		

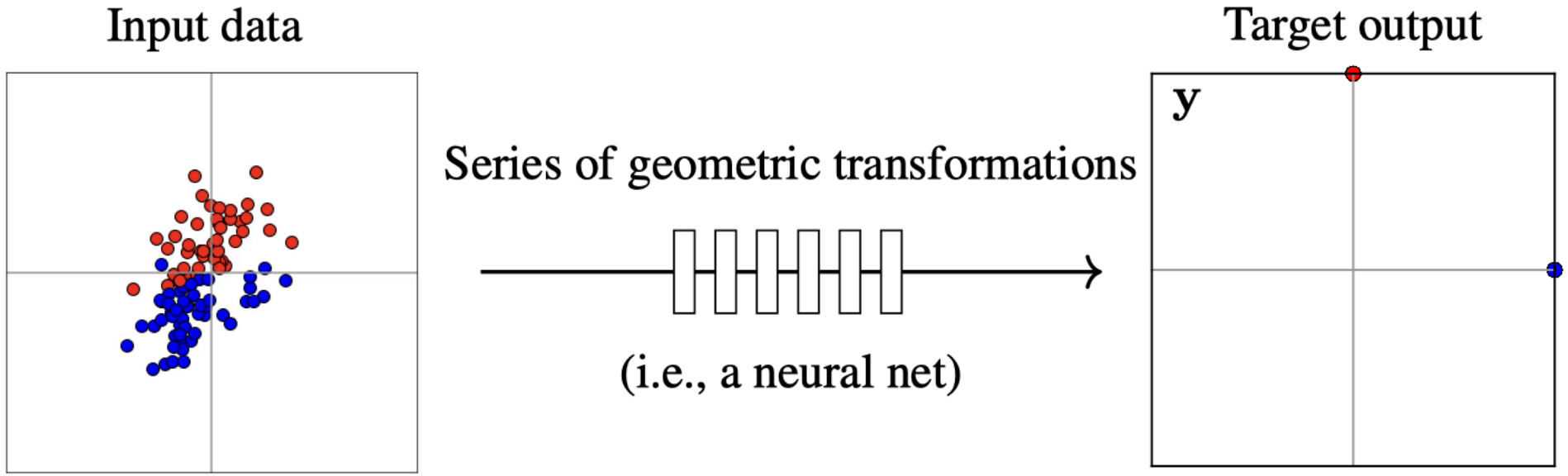
[images credit: visionbook.mit.edu]

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]

What does training a neural net classifier look like?



[images credit: visionbook.mit.edu]

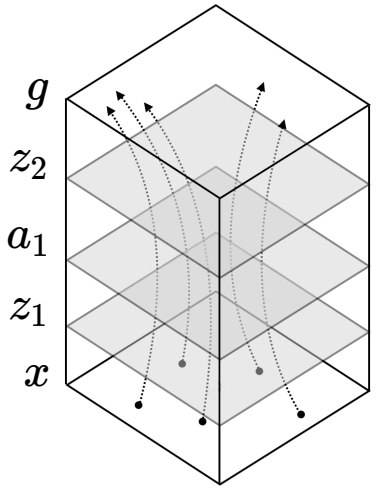
$$g = \text{softmax}(z_2)$$

$$z_2 = \text{linear}\,(a_1) \in \mathbb{R}^2$$

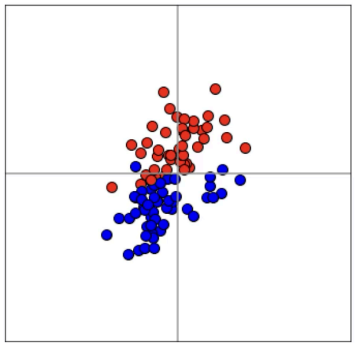
$$a_1 = \text{ReLU}(z_1)$$

$$z_1 = \text{linear}\,(x) \in \mathbb{R}^2$$

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



Training data



[images credit: visionbook.mit.edu]

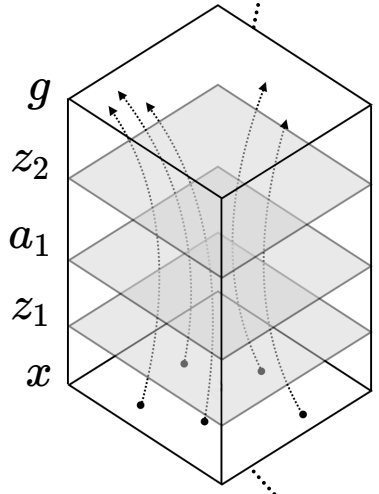
$$g = \text{softmax}(z_2)$$

$$z_2 = \text{linear}\,(a_1) \in \mathbb{R}^2$$

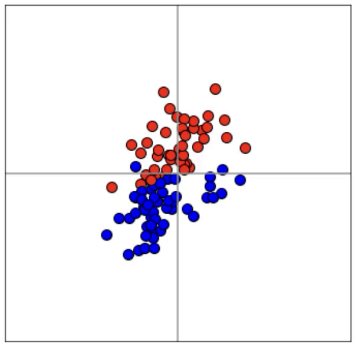
$$a_1 = \text{ReLU}(z_1)$$

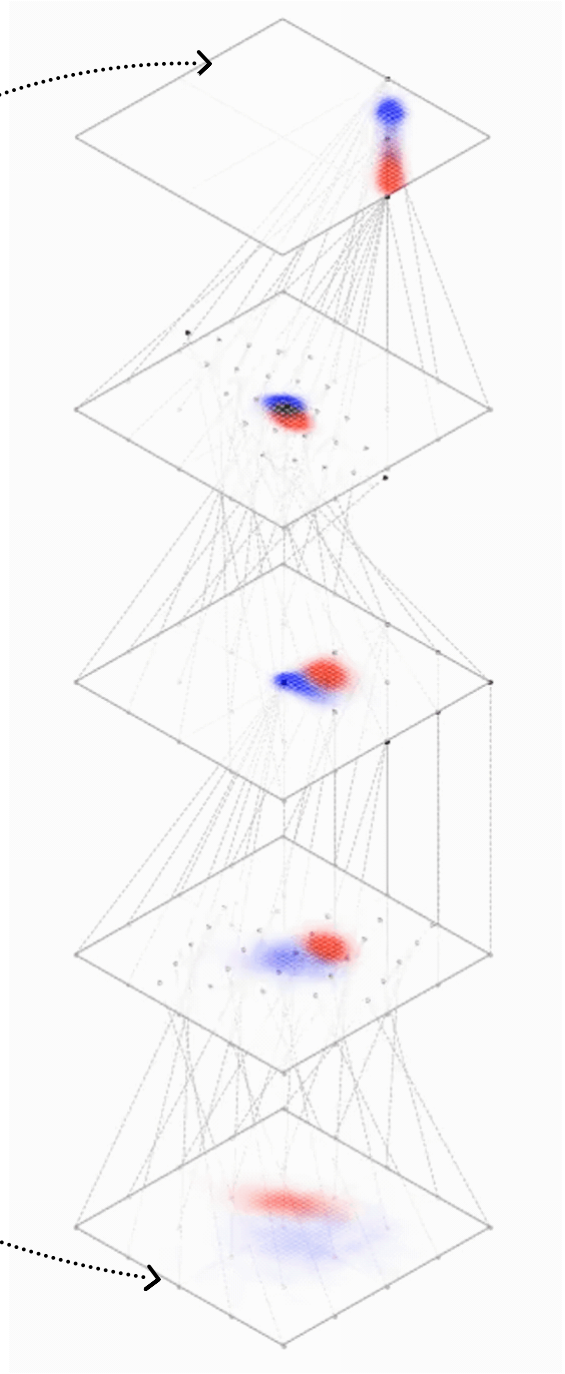
$$z_1 = \text{linear}\,(x) \in \mathbb{R}^2$$

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



Training data





[images credit: visionbook.mit.edu]

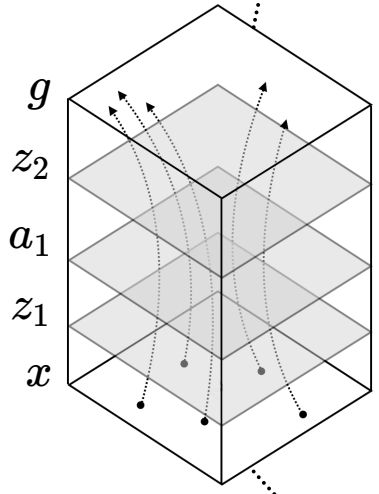
$$g = \text{softmax}(z_2)$$

$$z_2 = \text{linear}\,(a_1) \in \mathbb{R}^2$$

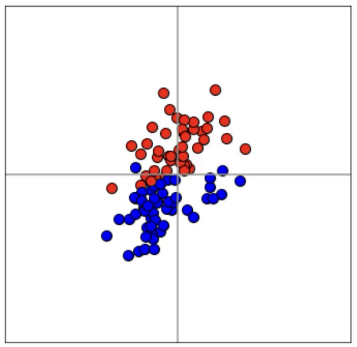
$$a_1 = \text{ReLU}(z_1)$$

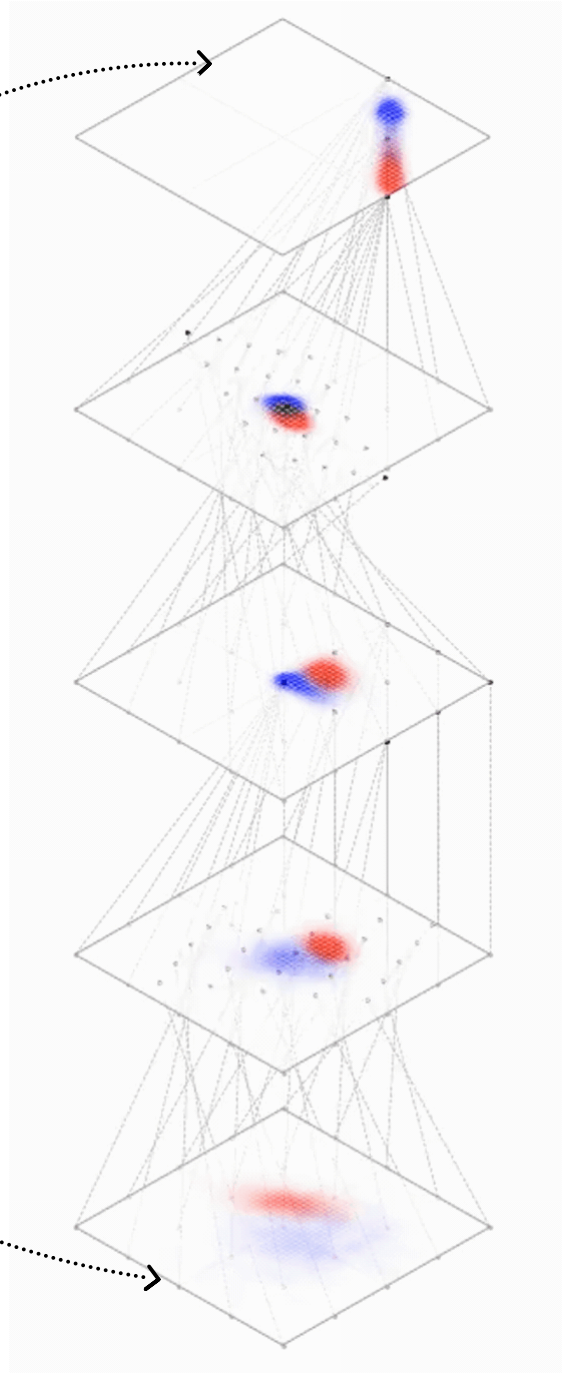
$$z_1 = \text{linear}\,(x) \in \mathbb{R}^2$$

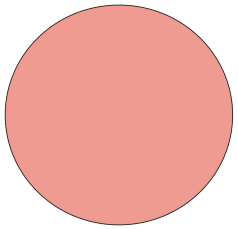
$$x \in \mathbb{R}^2$$

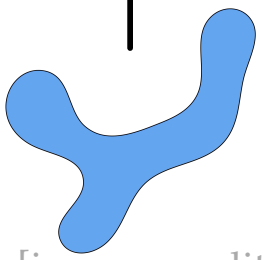


Training data





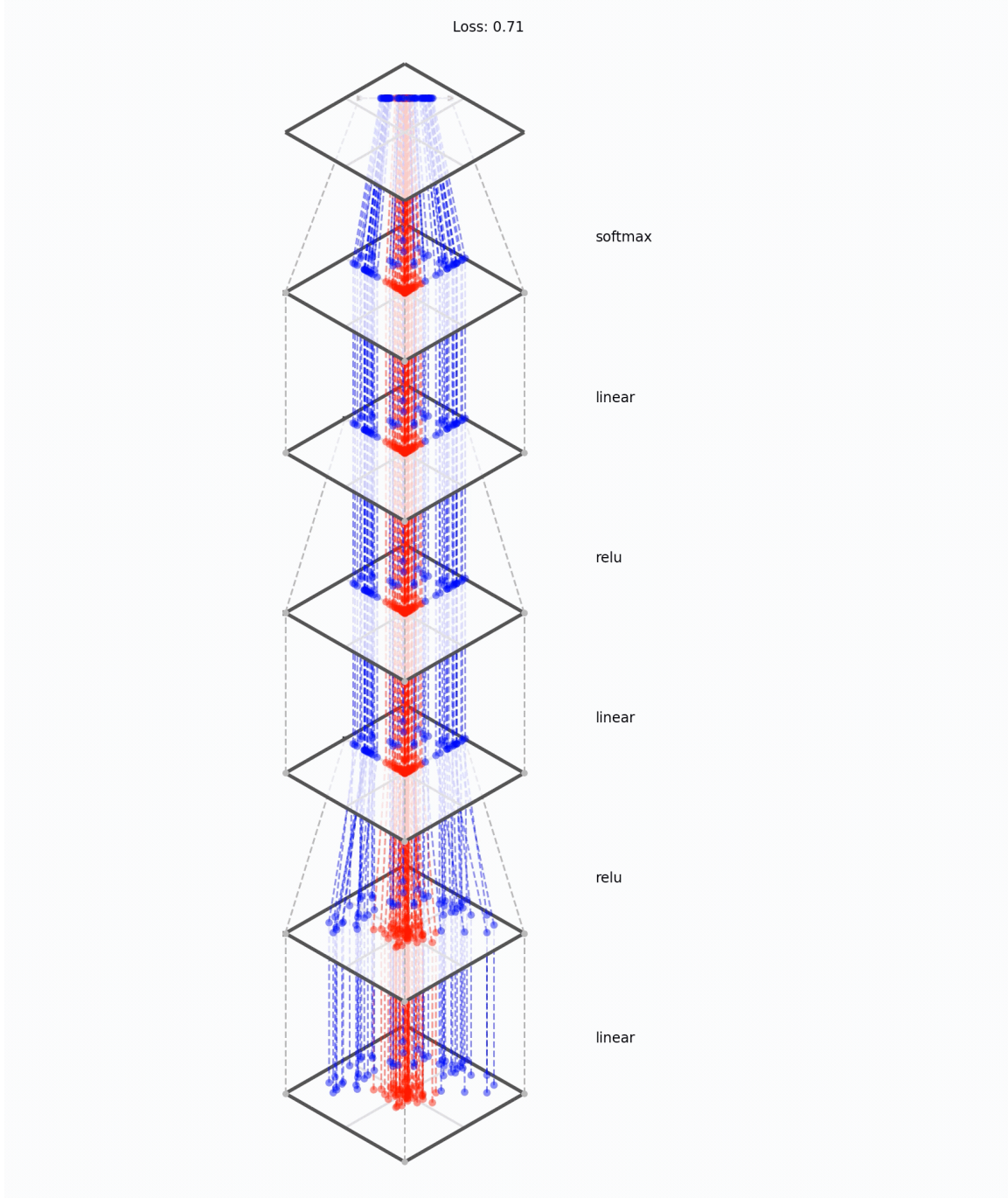




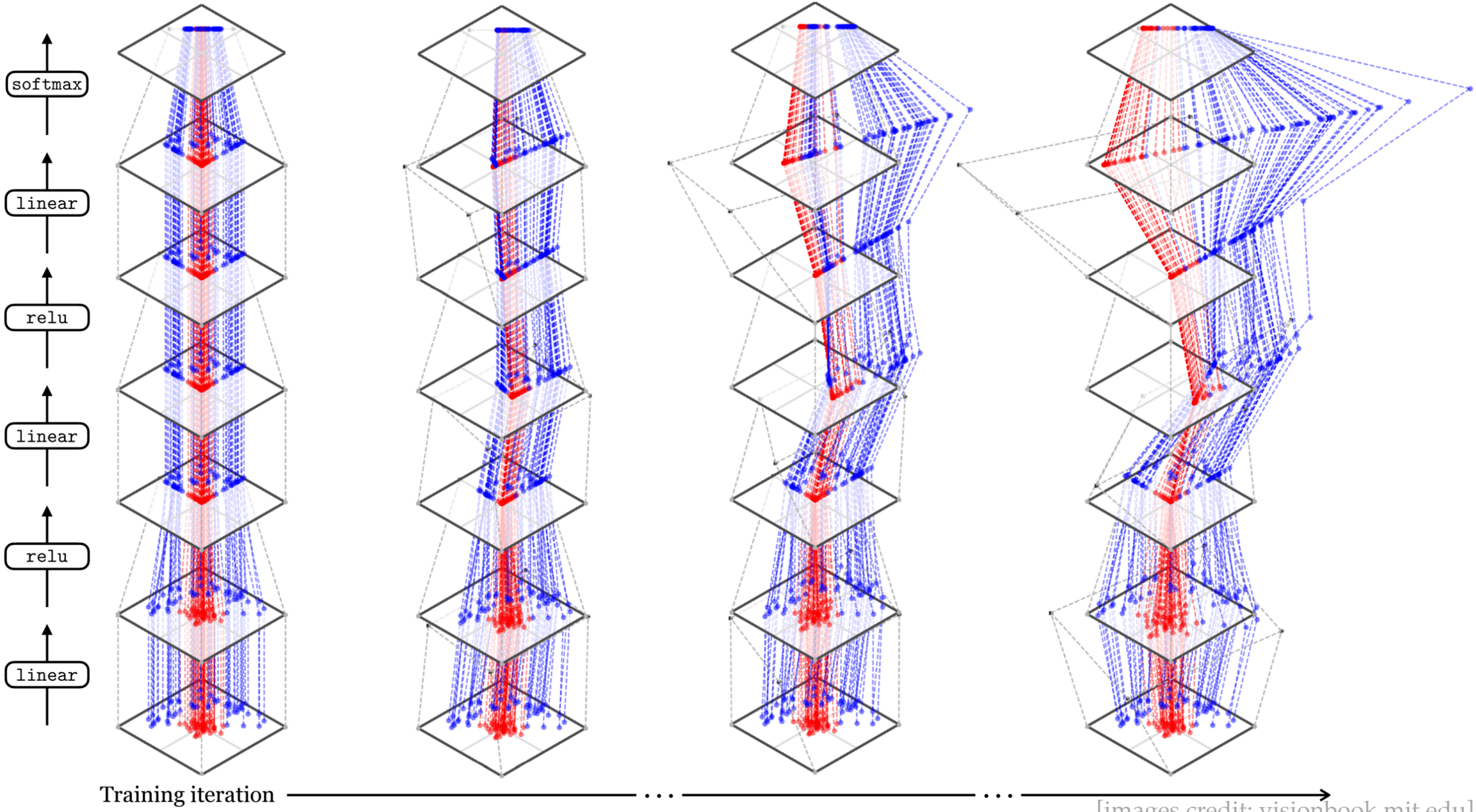
maps from complex data space to desired target space

[images credit: visionbook.mit.edu]

[images credit: visionbook.mit.edu]



[images credit: visionbook.mit.edu]



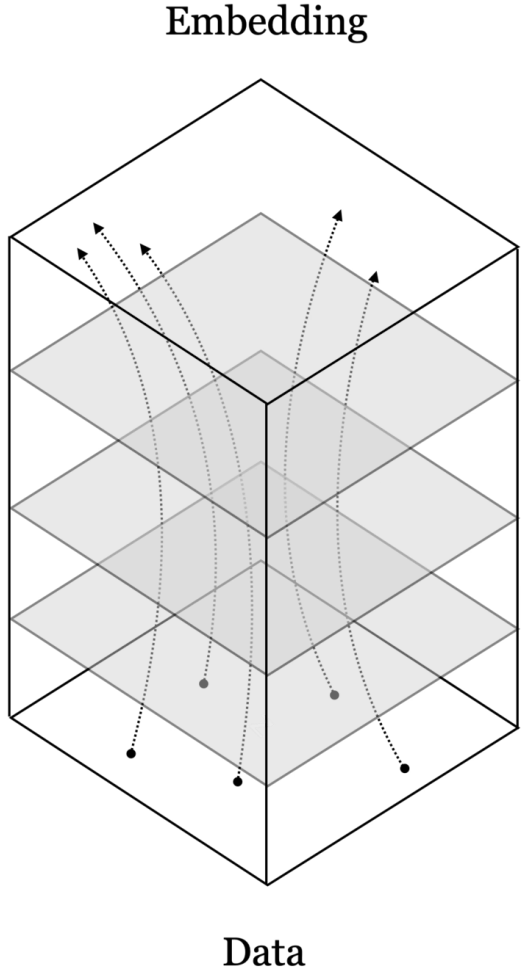
[images credit: visionbook.mit.edu]

Deep neural nets transform datapoints, layer by layer

[images credit: visionbook.mit.edu]

Deep neural nets transform datapoints, layer by layer

Each layer is a different *representation*, aka *embedding*, of the data

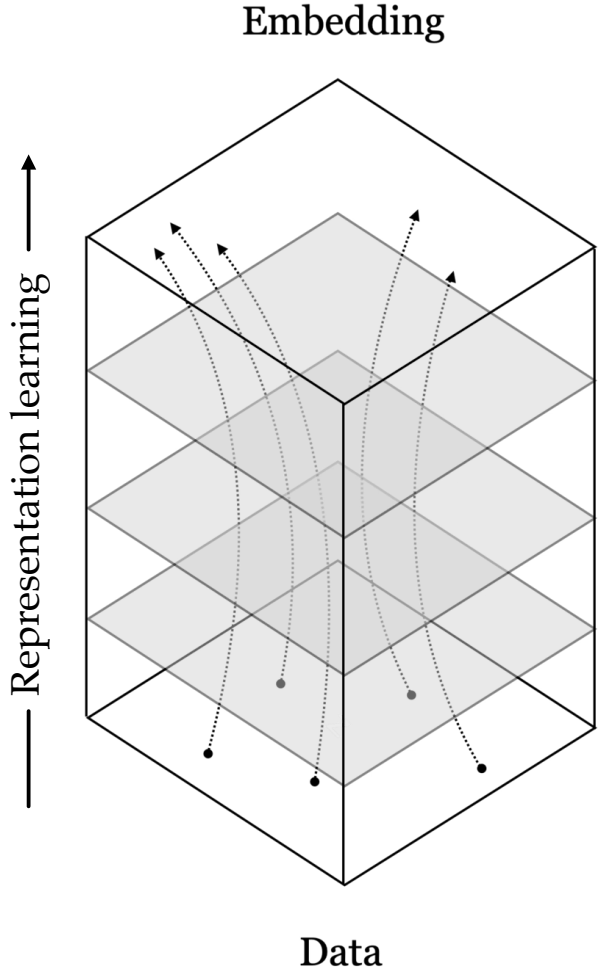


[images credit: visionbook.mit.edu]

Deep neural nets transform datapoints, layer by layer

Each layer is a different *representation,* aka *embedding,* of the data

From data to latent embeddings: representation learning



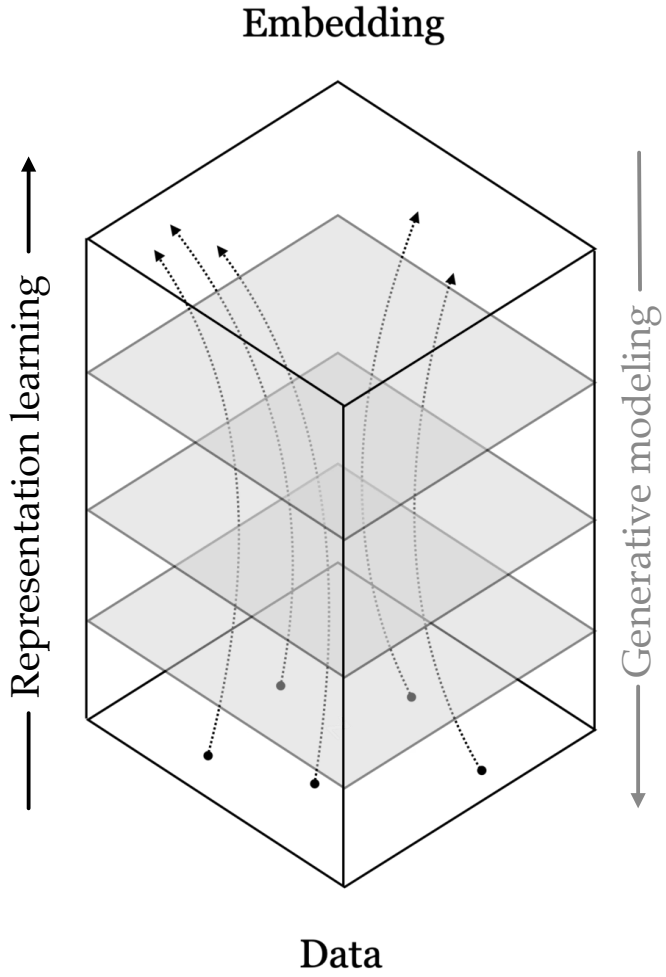
[images credit: visionbook.mit.edu]

Deep neural nets transform datapoints, layer by layer

Each layer is a different *representation*, aka *embedding*, of the data

From data to latent embeddings: representation learning

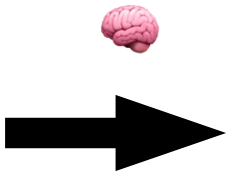
(From latent embeddings to data: generative modeling)

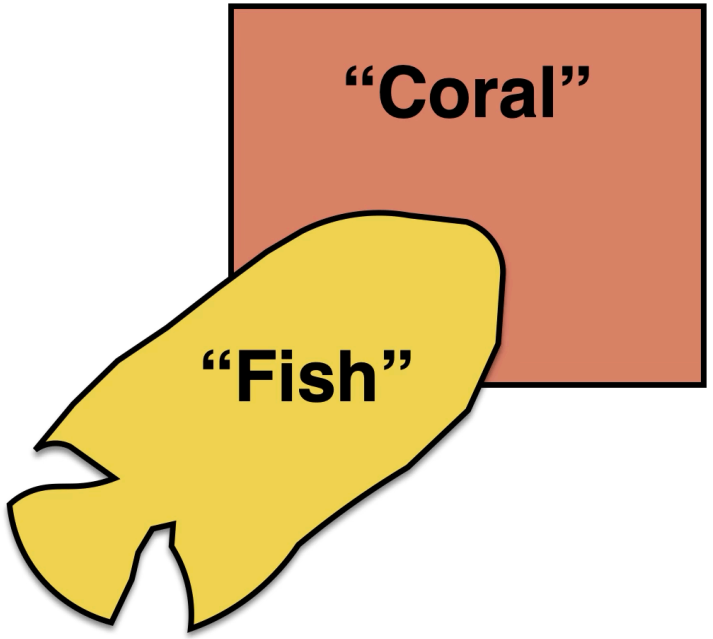


[images credit: visionbook.mit.edu]

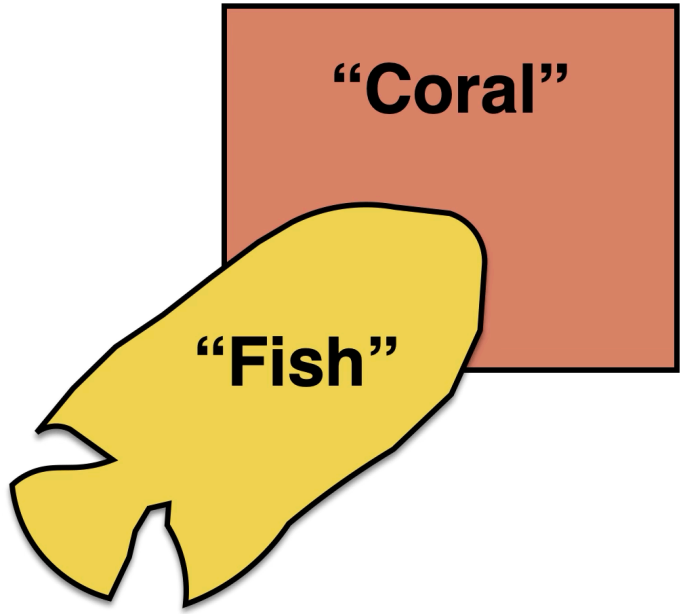
humans also learn representations

humans also learn representations





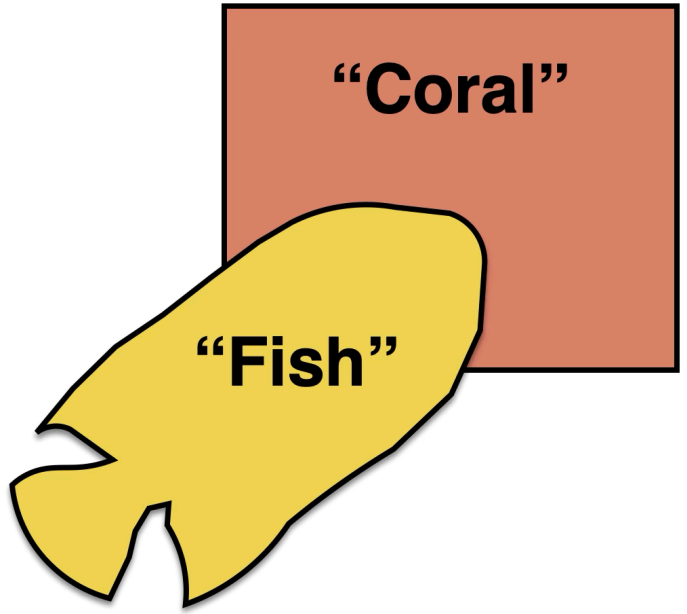
Good representations are:



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

Good representations are:

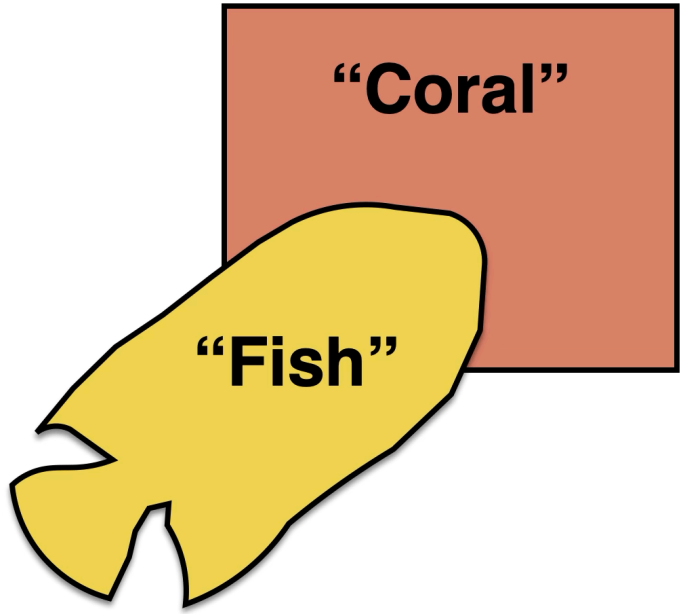
- Compact (*minimal*)



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

Good representations are:

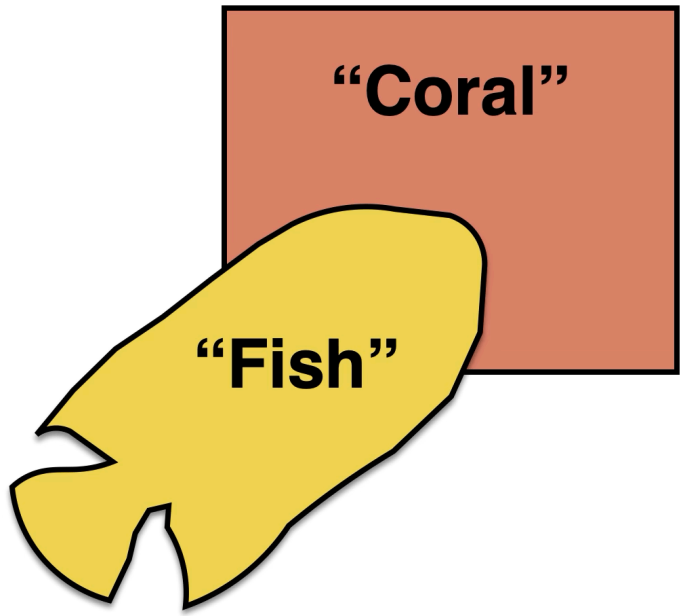
- Compact (*minimal*)
- Explanatory (*roughly sufficient*)



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

Good representations are:

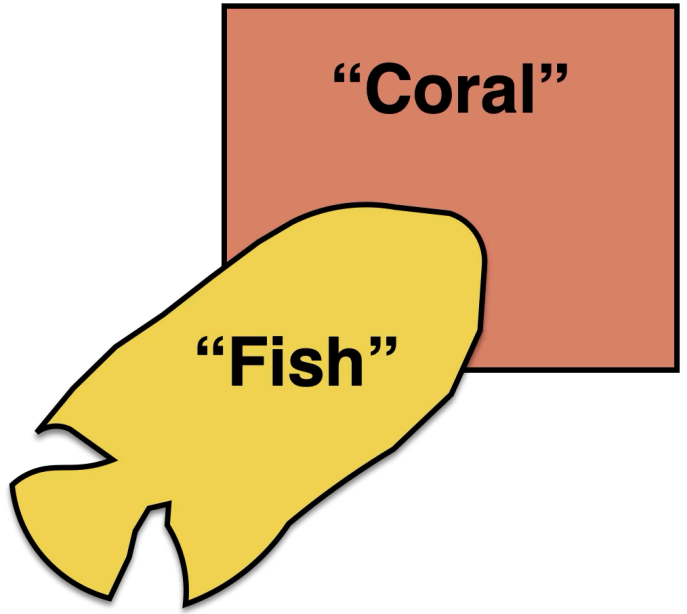
- Compact (*minimal*)
- Explanatory (*roughly sufficient*)
- Disentangled (*independent* factors)



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

Good representations are:

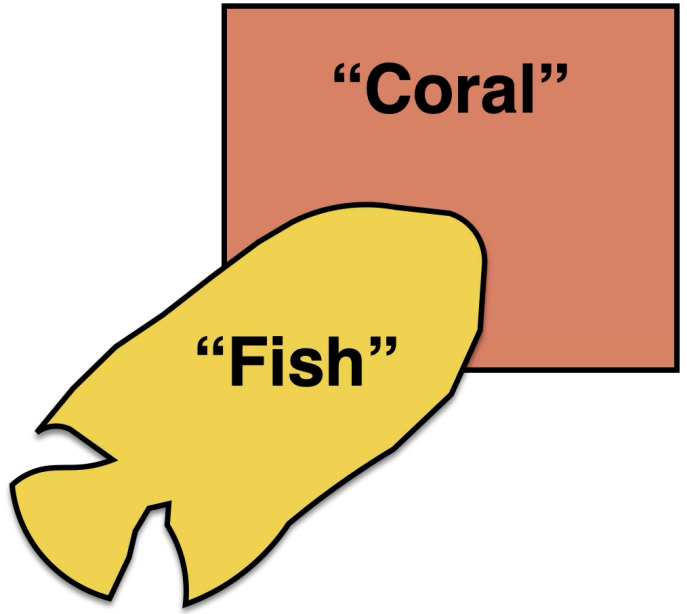
- Compact (*minimal*)
- Explanatory (*roughly sufficient*)
- Disentangled (*independent* factors)
- Interpretable (understandable by humans)



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

Good representations are:

- Compact (*minimal*)
- Explanatory (*roughly sufficient*)
- Disentangled (*independent* factors)
- Interpretable (understandable by humans)
- Make *subsequent problem solving easy*



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

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

ImageNet also taught us that labeling 14 million images by hand is brutal.

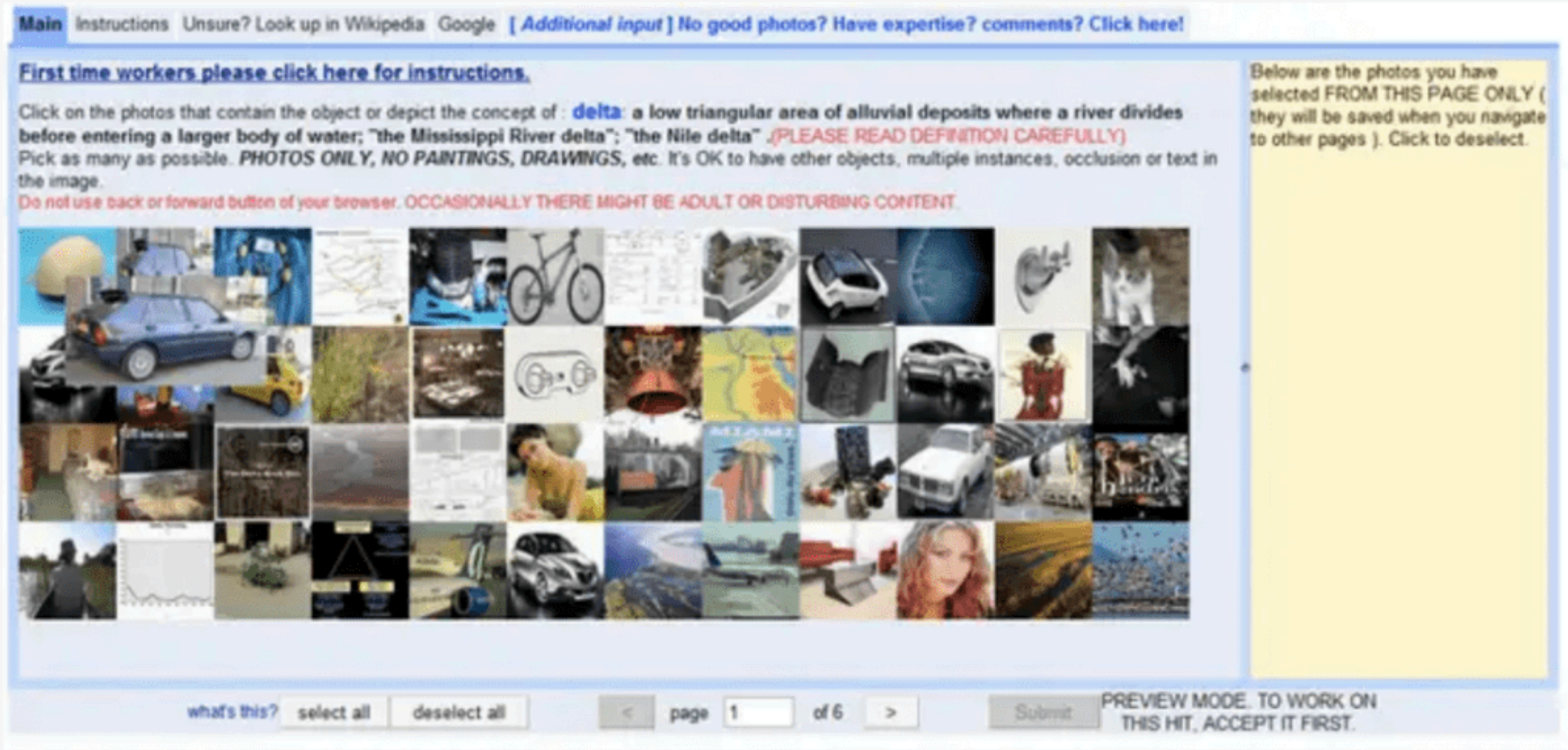
ImageNet also taught us that labeling 14 million images by hand is brutal.

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 surfaced in a chance hallway conversation with a graduate student who asked Li whether she had heard of Amazon Mechanical Turk, a service where hordes of humans sitting at computers around the world would complete small online tasks for pennies.

"He showed me the website, and I can tell you literally that day I 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."

ImageNet also taught us that labeling 14 million images by hand is brutal.

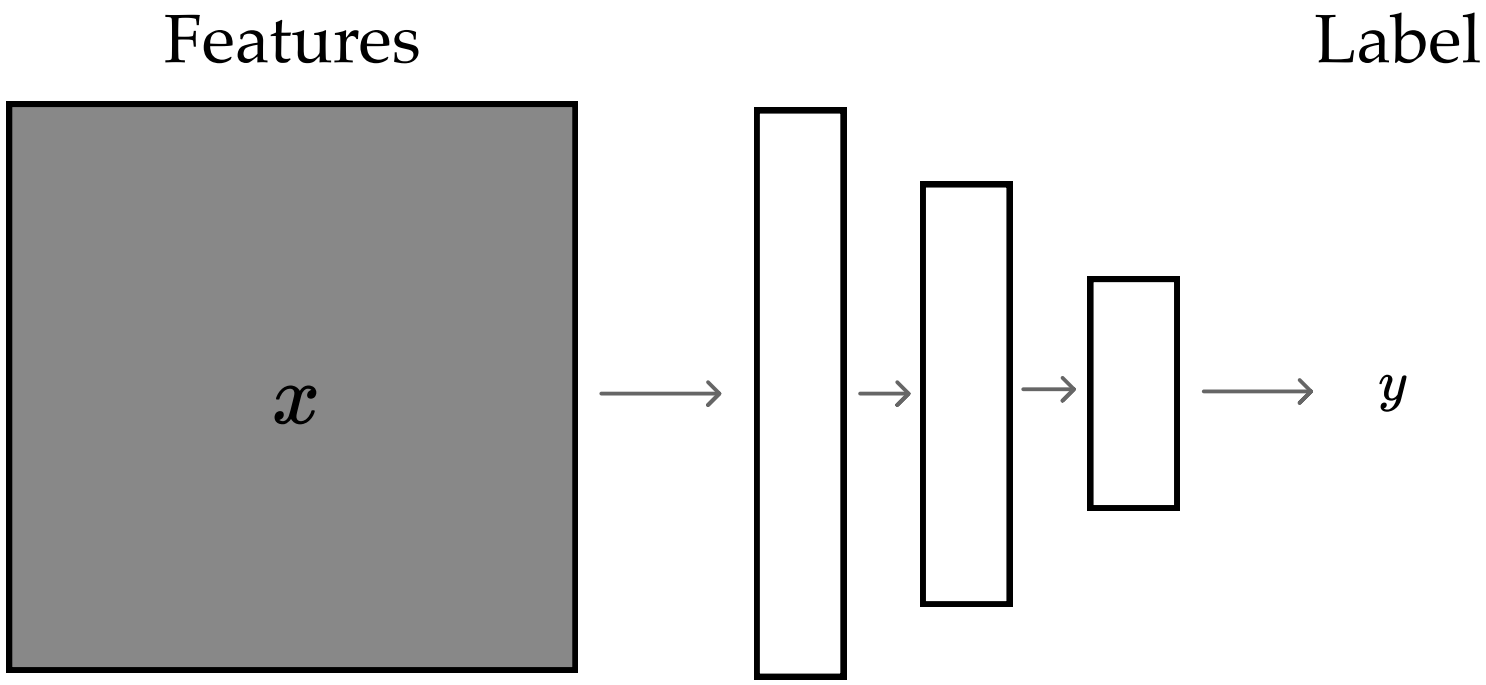


The Amazon Mechanical Turk backend for classifying images. Image: ImageNet

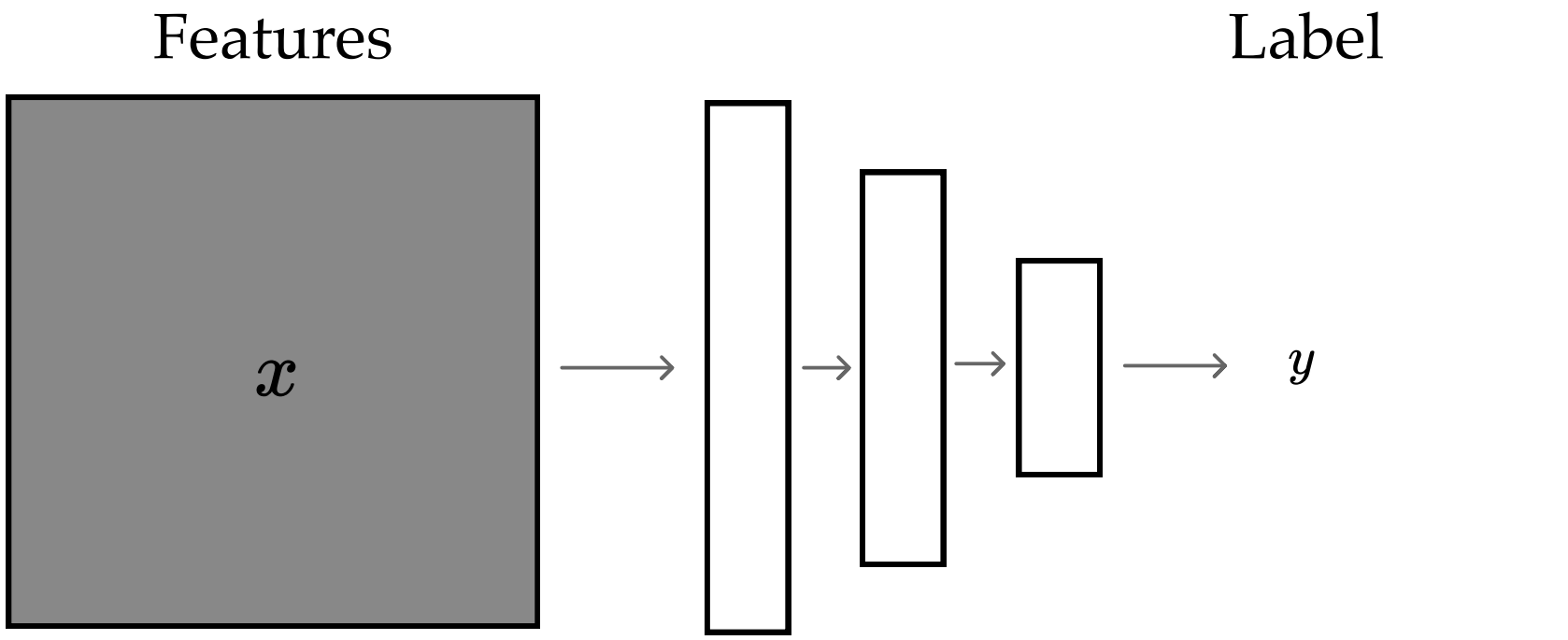
Outline

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

Label prediction (supervised learning)

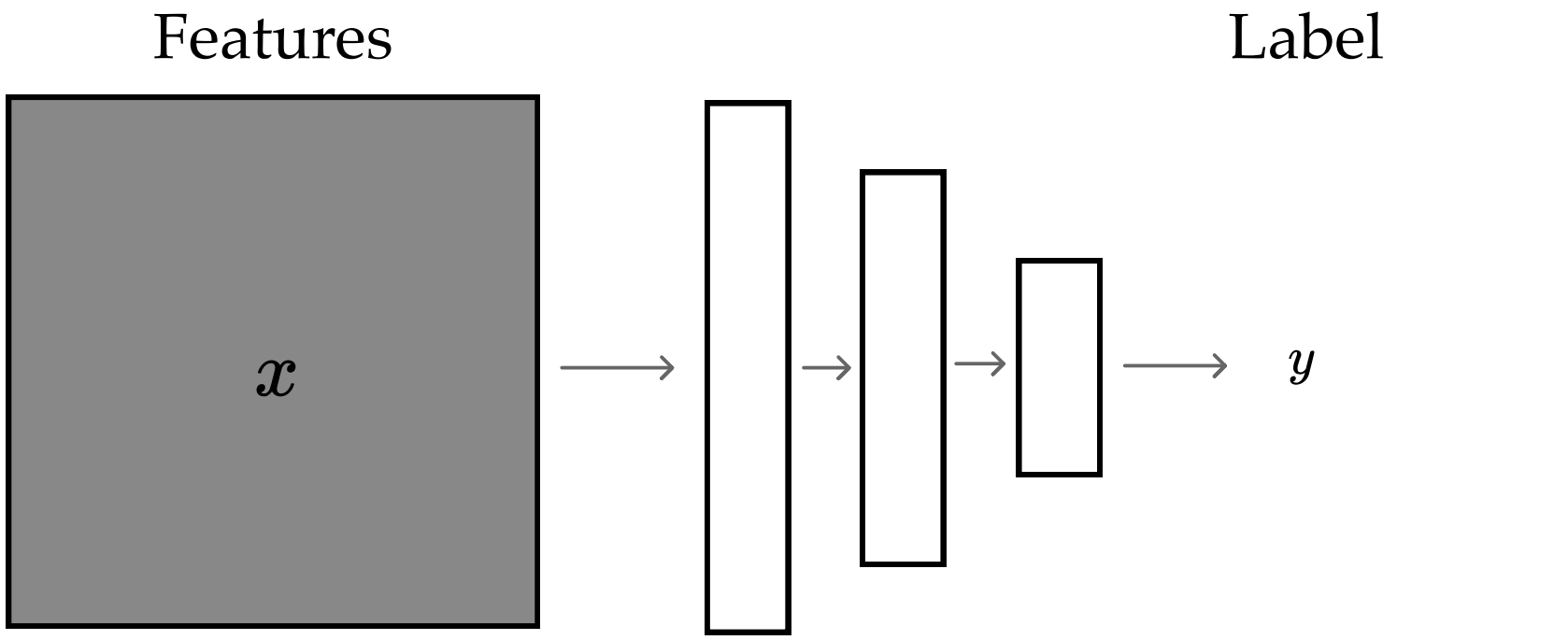


Label prediction (supervised learning)



Labels y are expensive…

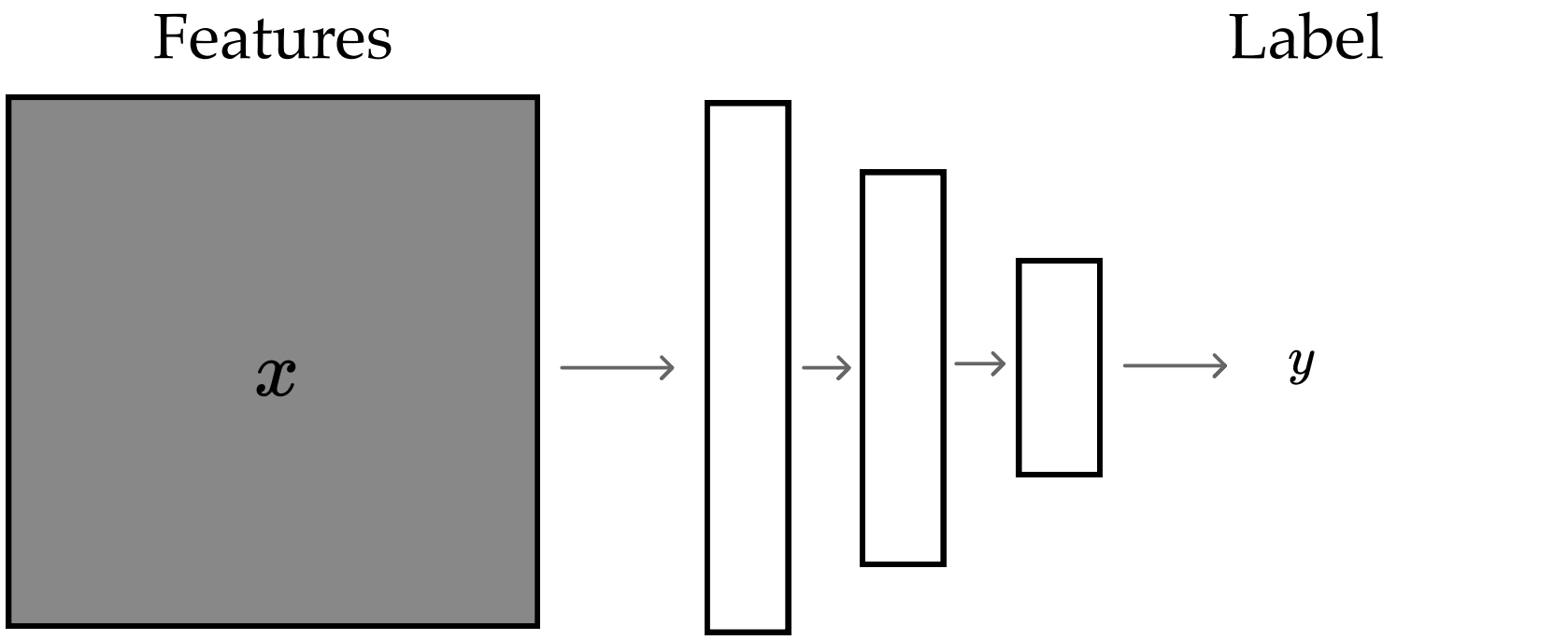
Label prediction (supervised learning)



Unlabeled features x are abundant

Labels y are expensive…

Label prediction (supervised learning)

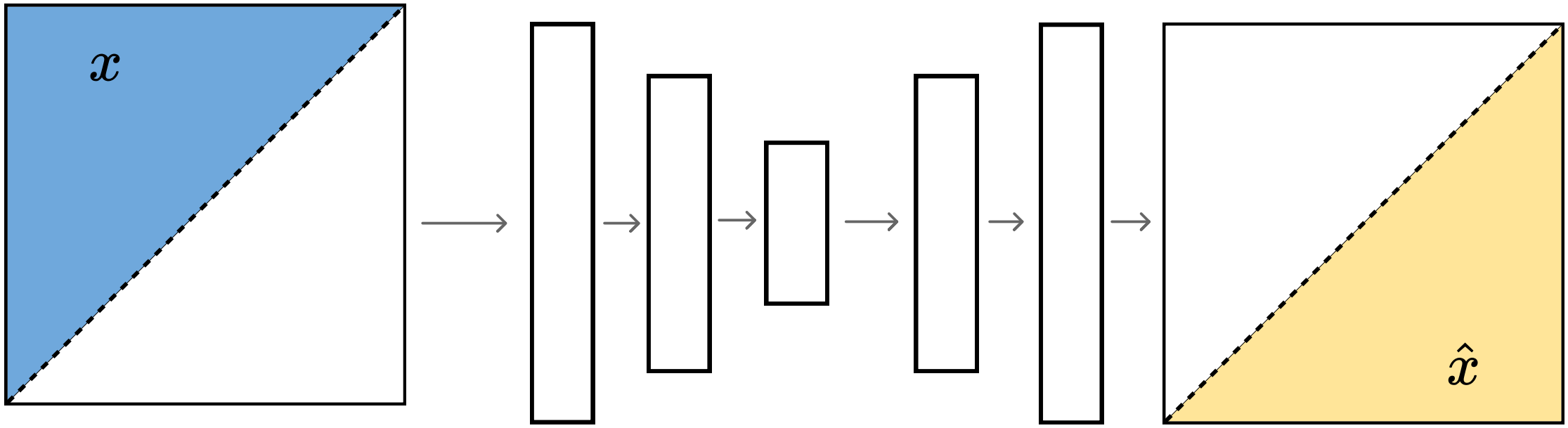


Unlabeled features x are abundant

Labels y are expensive…

Can we learn useful things with just x?

Self-supervised Learning (partial feature reconstruction)



Partial Features

Other Partial Features

Hard reconstruction, forces understanding.

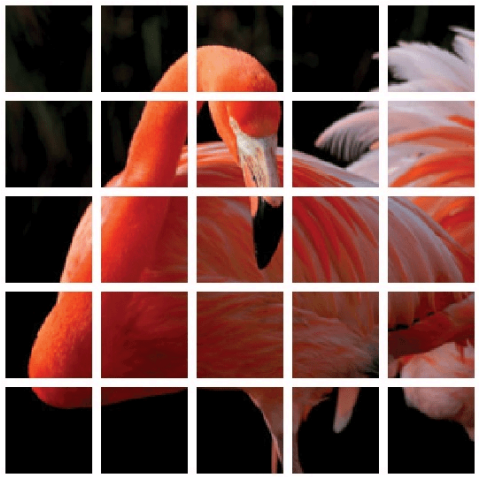
Masked Auto-Encoder (Vision)

[He, et al. Masked Autoencoders Are Scalable Vision Learners, 2021]

Masked Auto-Encoder (Vision)

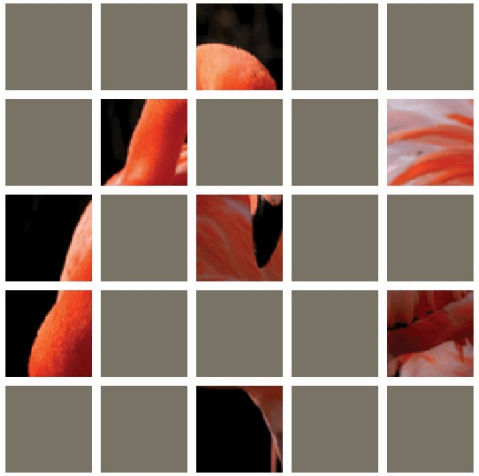
[He, et al. Masked Autoencoders Are Scalable Vision Learners, 2021]

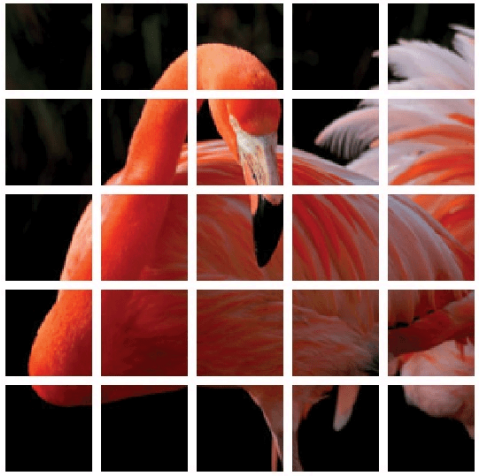
Masked Auto-Encoder (Vision)



[He, et al. Masked Autoencoders Are Scalable Vision Learners, 2021]

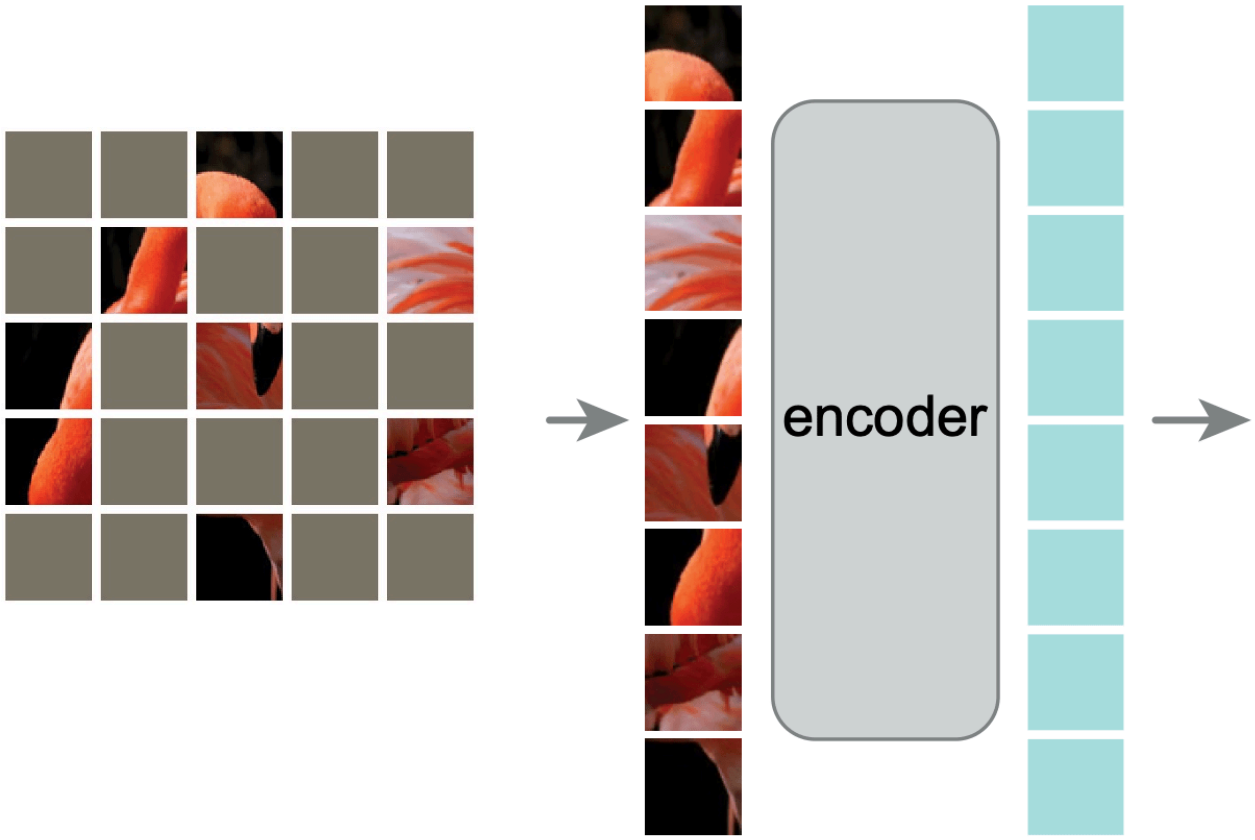
Masked Auto-Encoder (Vision)

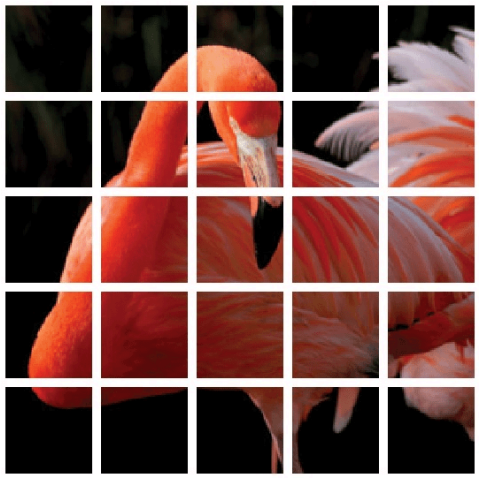




[He, et al. Masked Autoencoders Are Scalable Vision Learners, 2021]

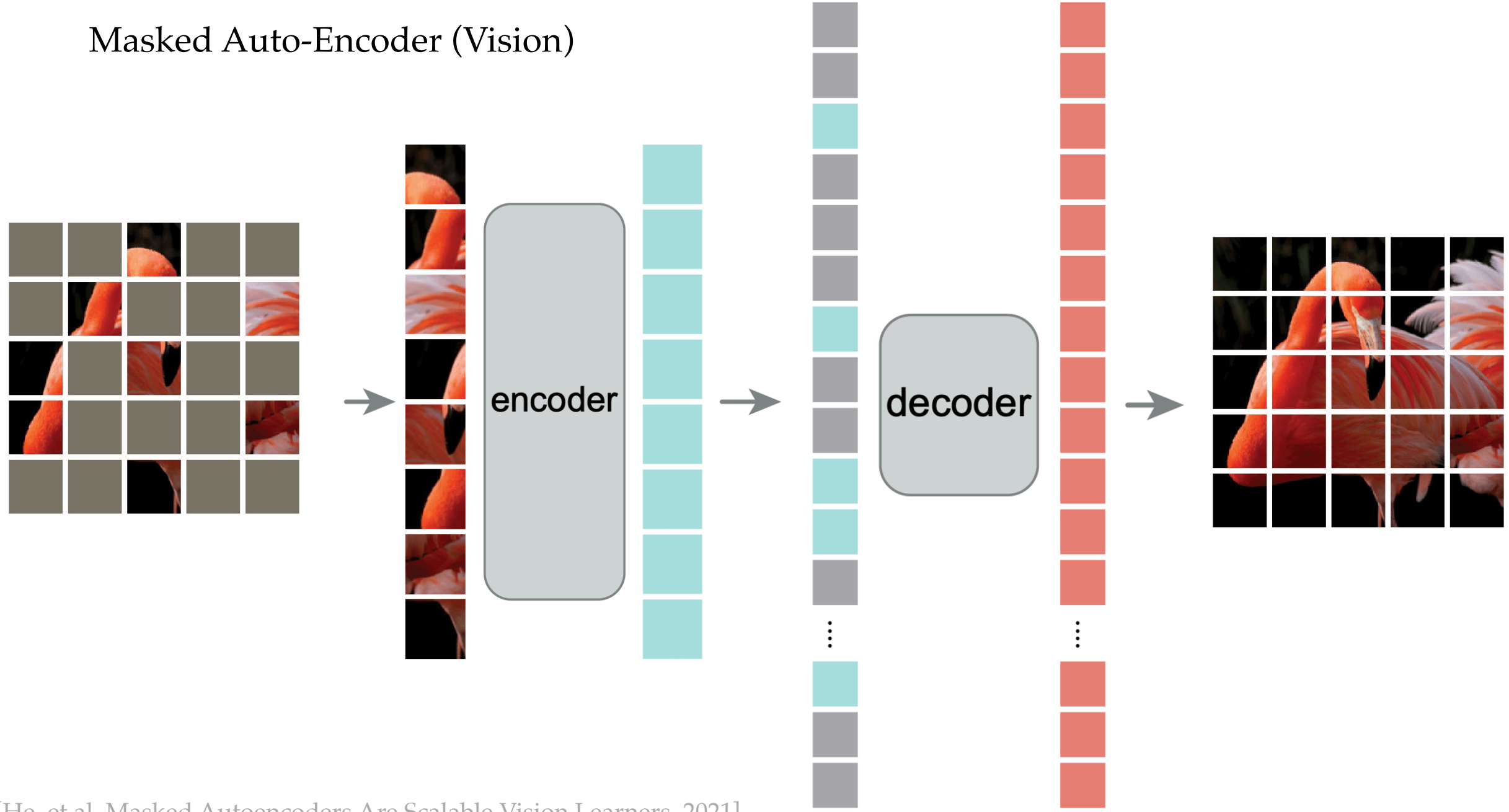
Masked Auto-Encoder (Vision)





[He, et al. Masked Autoencoders Are Scalable Vision Learners, 2021]

Masked Auto-Encoder (Vision)



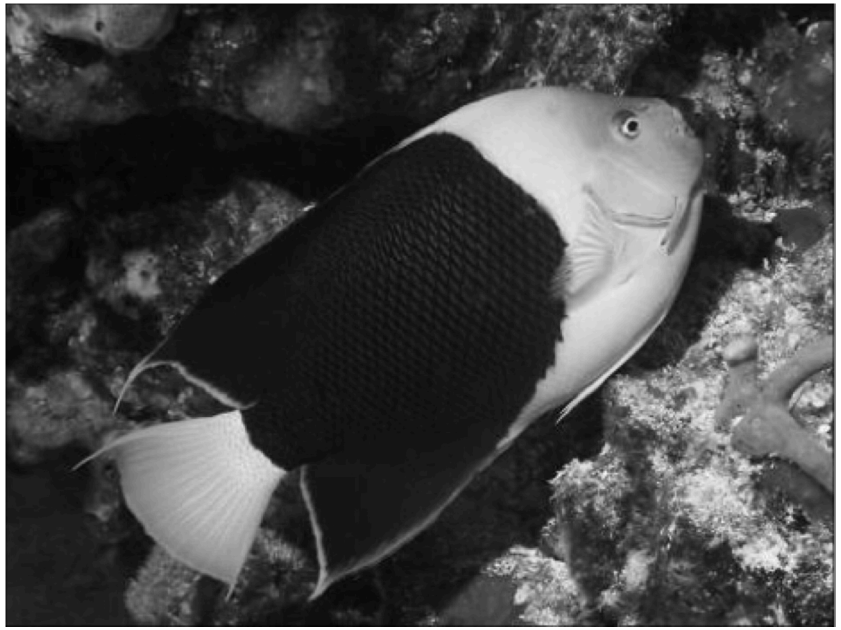
[He, et al. Masked Autoencoders Are Scalable Vision Learners, 2021]

Masking channels

e.g. masking the color

[Zhang, Isola, Efros, ECCV 2016]

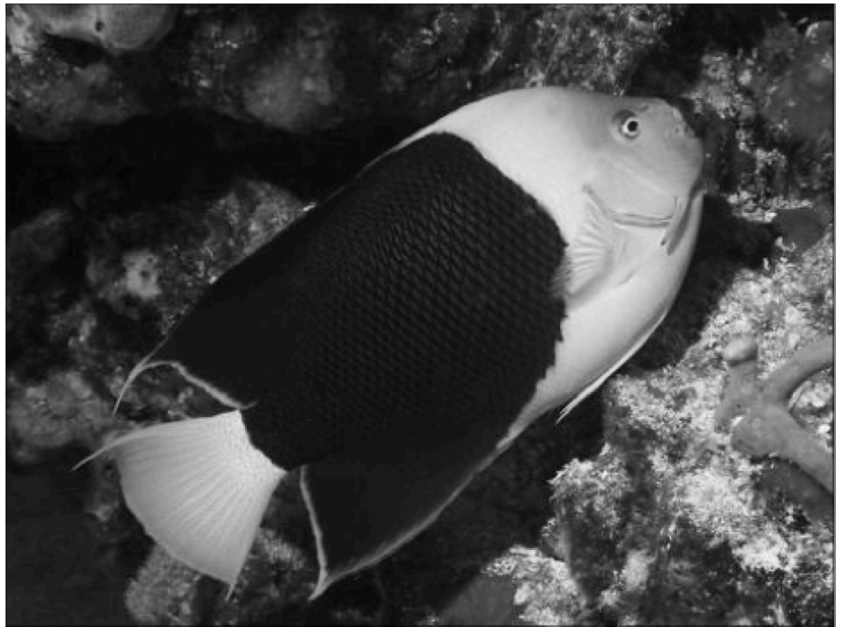
Masking channels

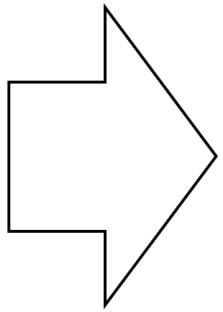


[Zhang, Isola, Efros, ECCV 2016]

Masking channels

predict color from gray-scale



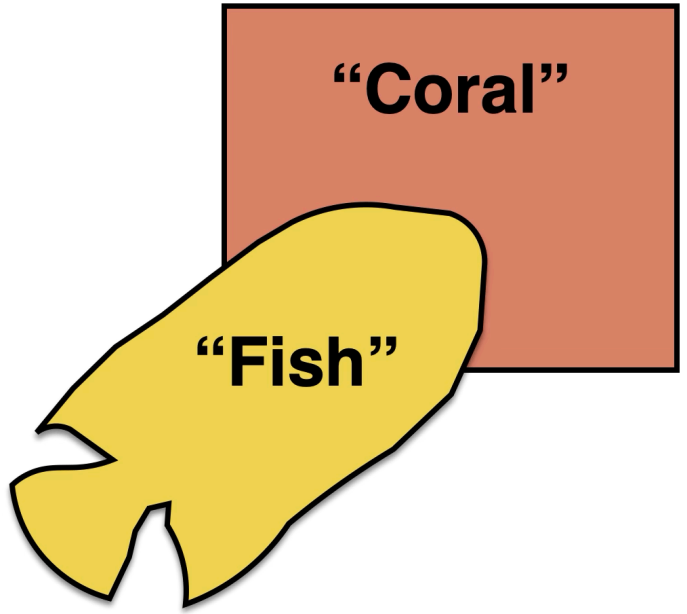


[Zhang, Isola, Efros, ECCV 2016]

Outline

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

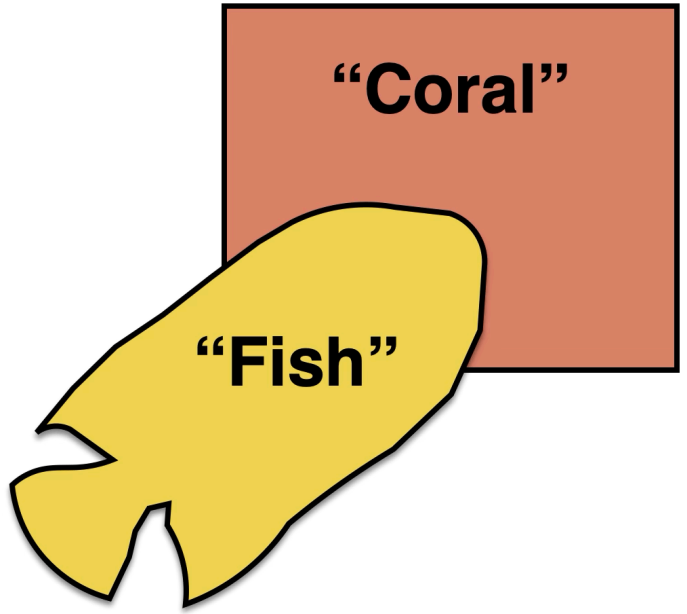
Good representations are:



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

Good representations are:

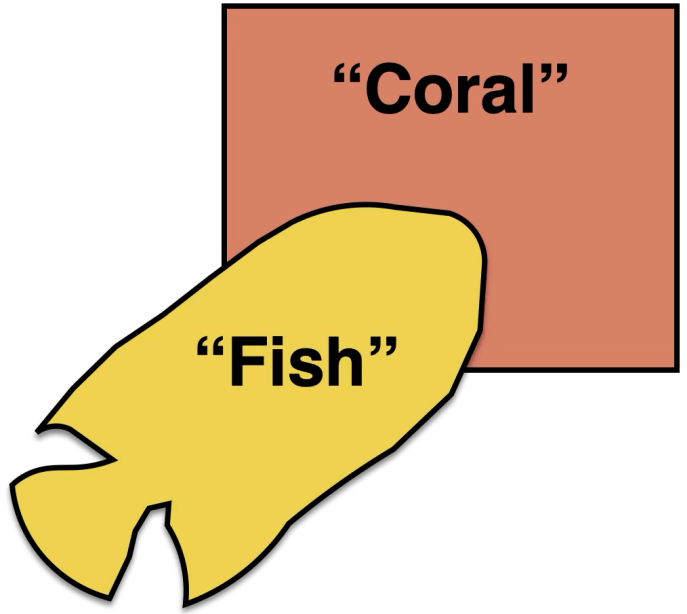
- Compact (*minimal*)



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

Good representations are:

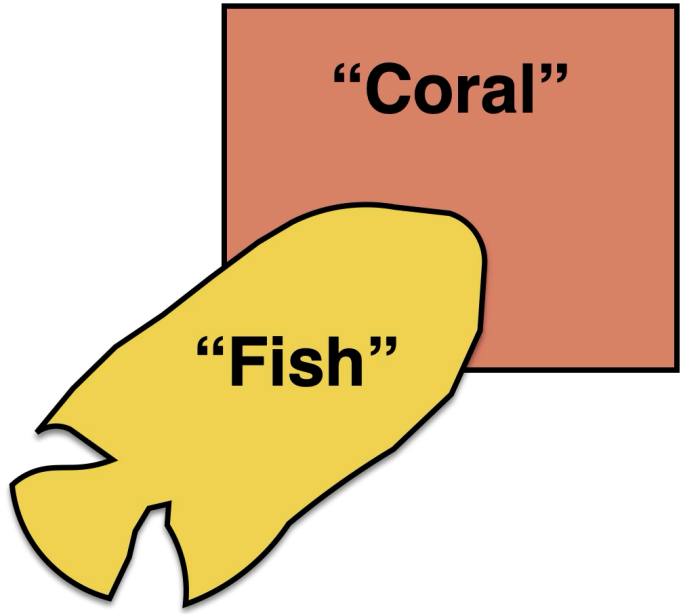
- Compact (*minimal*)
- Explanatory (*roughly sufficient*)



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

Good representations are:

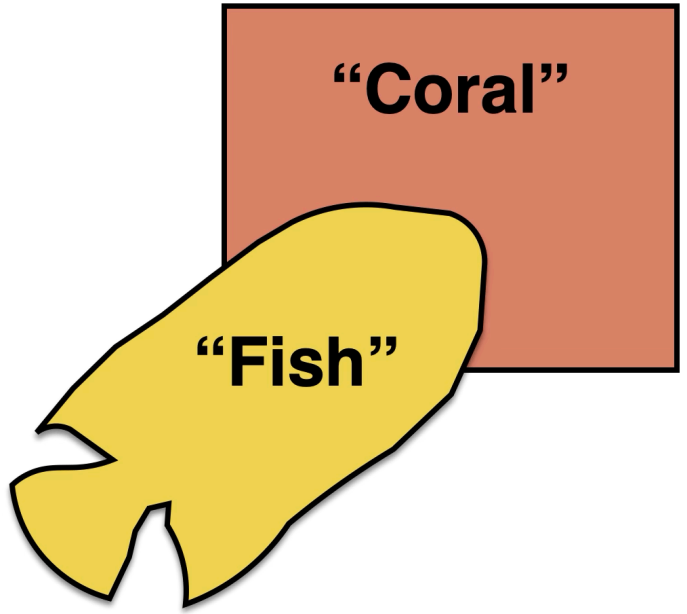
- Compact (*minimal*)
- Explanatory (*roughly sufficient*)
- Disentangled (*independent* factors)



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

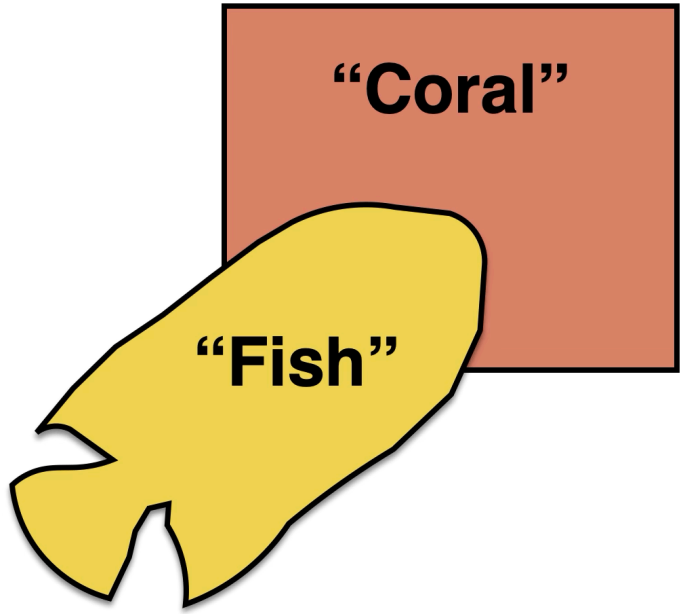
Good representations are:

- Compact (*minimal*)
- Explanatory (*roughly sufficient*)
- Disentangled (*independent* factors)
- Interpretable (understandable by humans)



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

Good representations are:



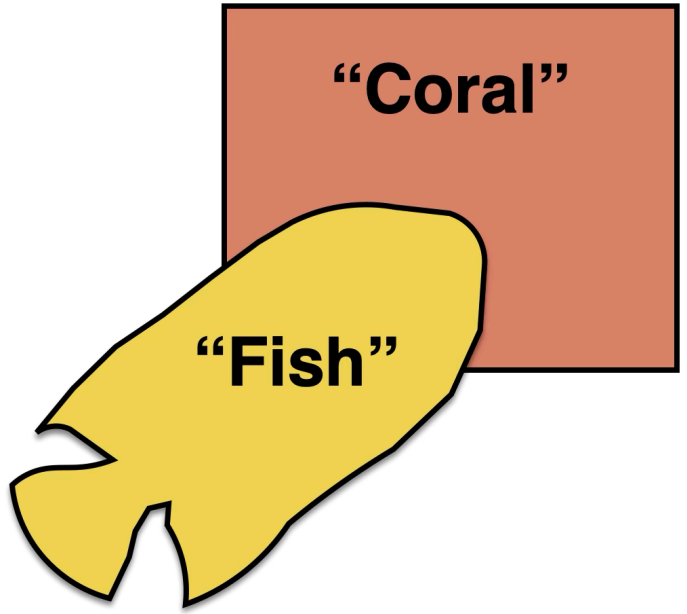
- Compact (*minimal*)
- Explanatory (*roughly sufficient*)
- Disentangled (*independent* factors)
- Interpretable (understandable by humans)
- Make *subsequent problem solving easy*

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

Good representations are:

Auto-encoders explicitly aims

- Compact (*minimal*)
- Explanatory (*roughly sufficient*)
- Disentangled (*independent* factors)
- Interpretable (understandable by humans)
- Make *subsequent problem solving easy*



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

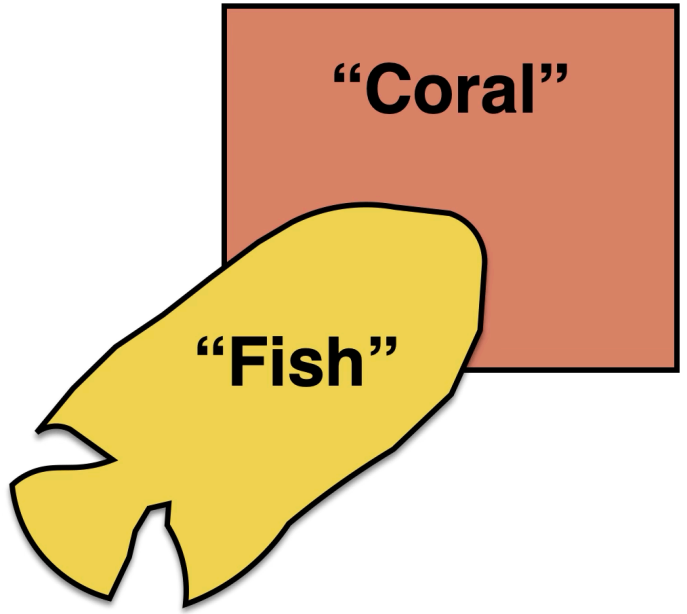
Good representations are:

Auto-encoders explicitly aims

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

these may just emerge as well

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



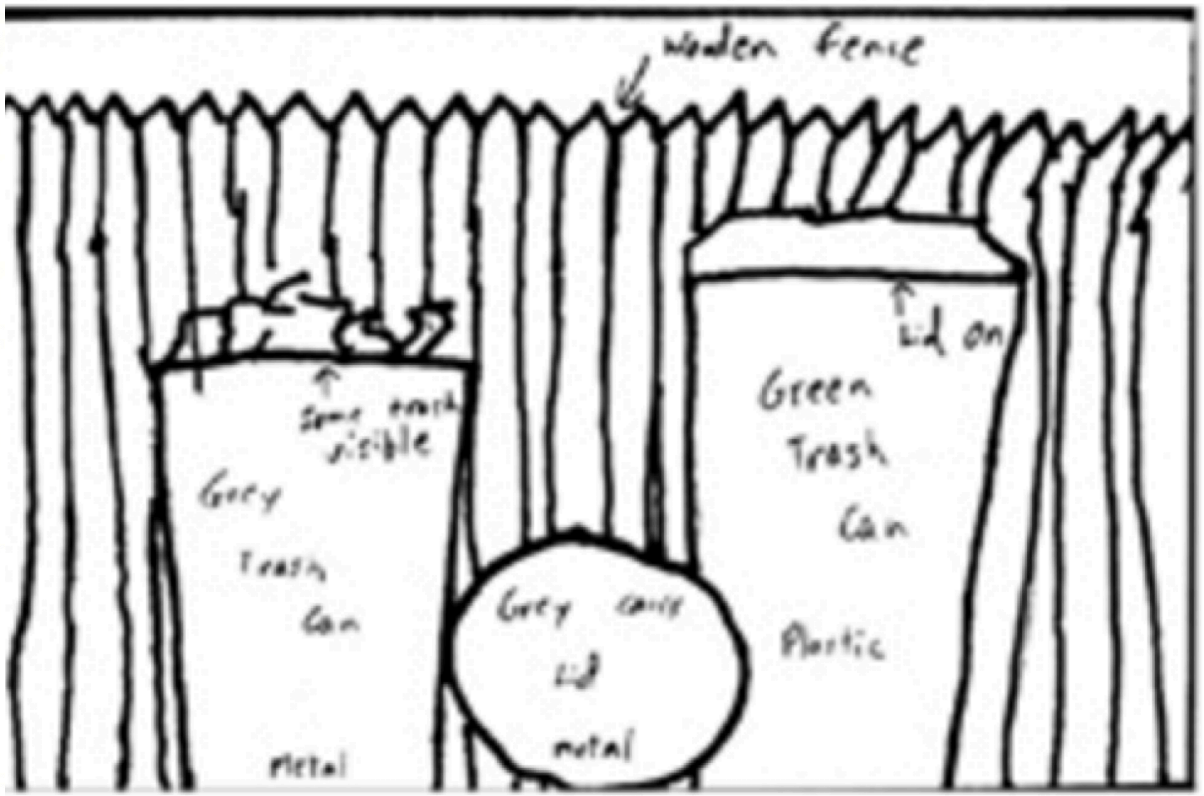
[See “Representation Learning”, Bengio 2013, for more commentary]

Observed image

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

Observed image

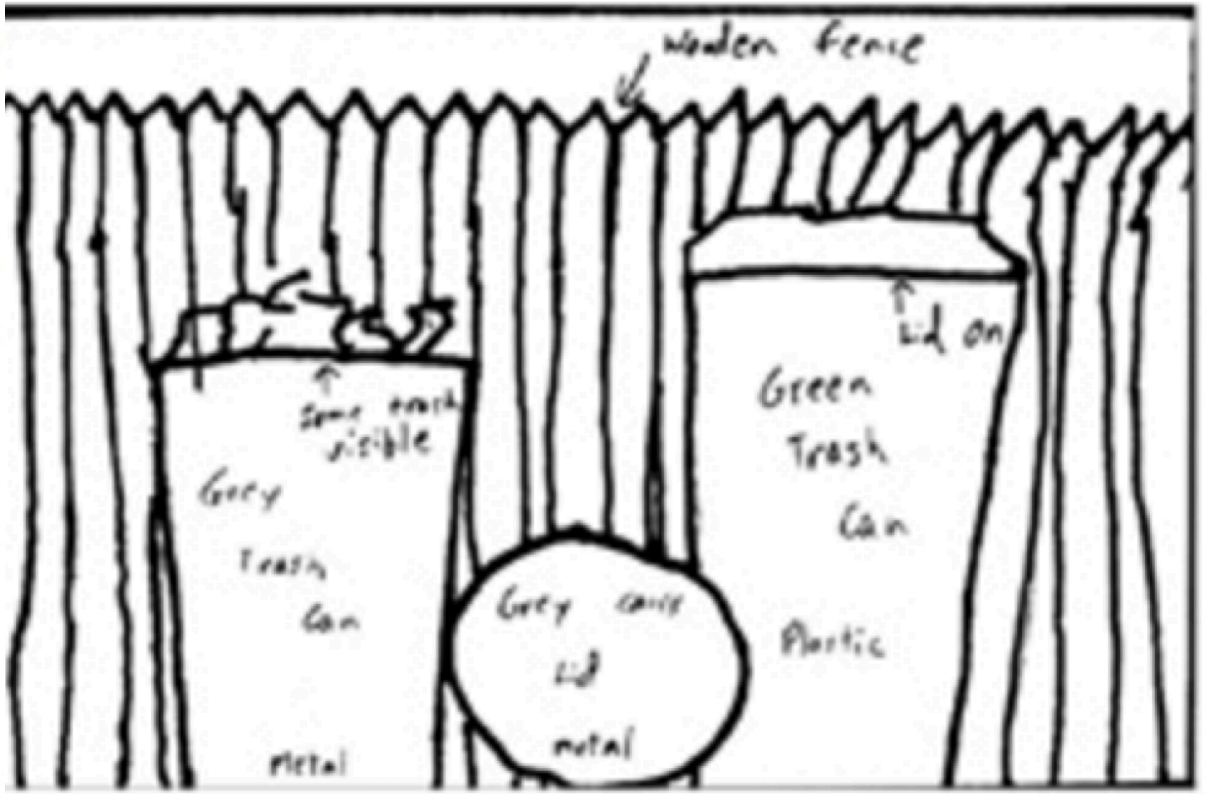
Drawn from memory



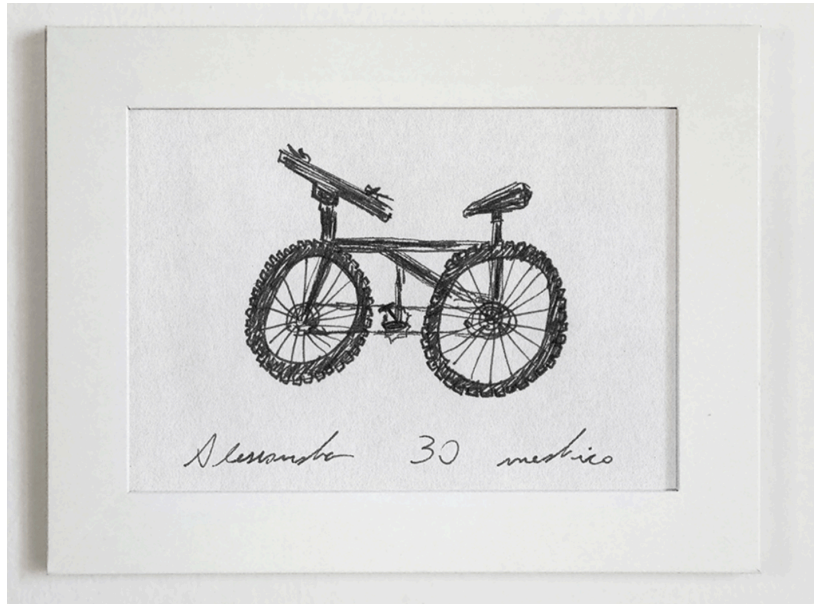
[Bartlett, 1932]
[Intraub & Richardson, 1989]

Observed image

Drawn from memory

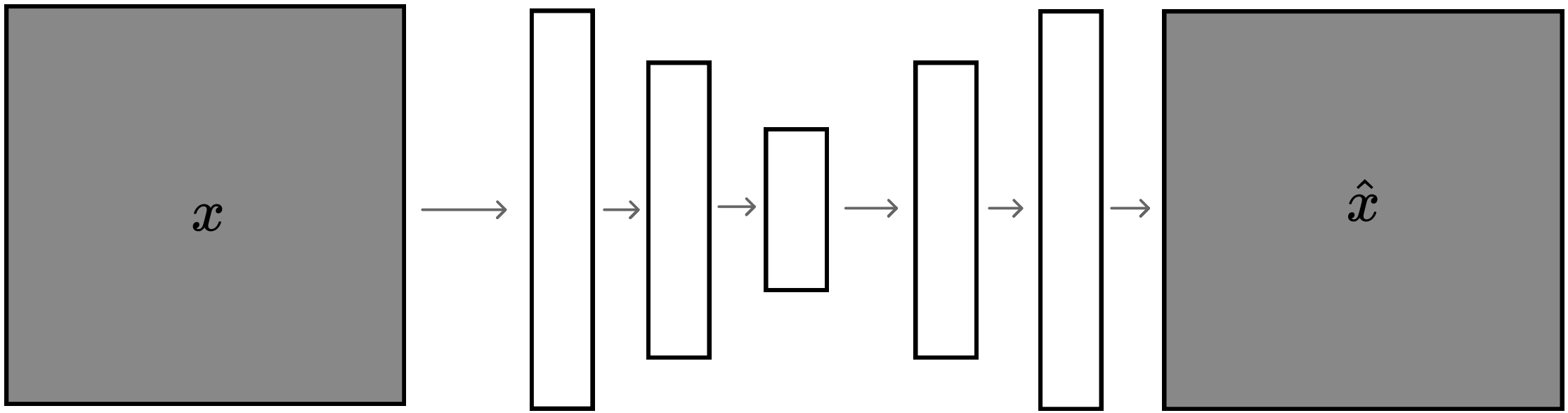


[Bartlett, 1932]
[Intraub & Richardson, 1989]



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

Unsupervised Learning (feature reconstruction)



Auto-encoder

Auto-encoder

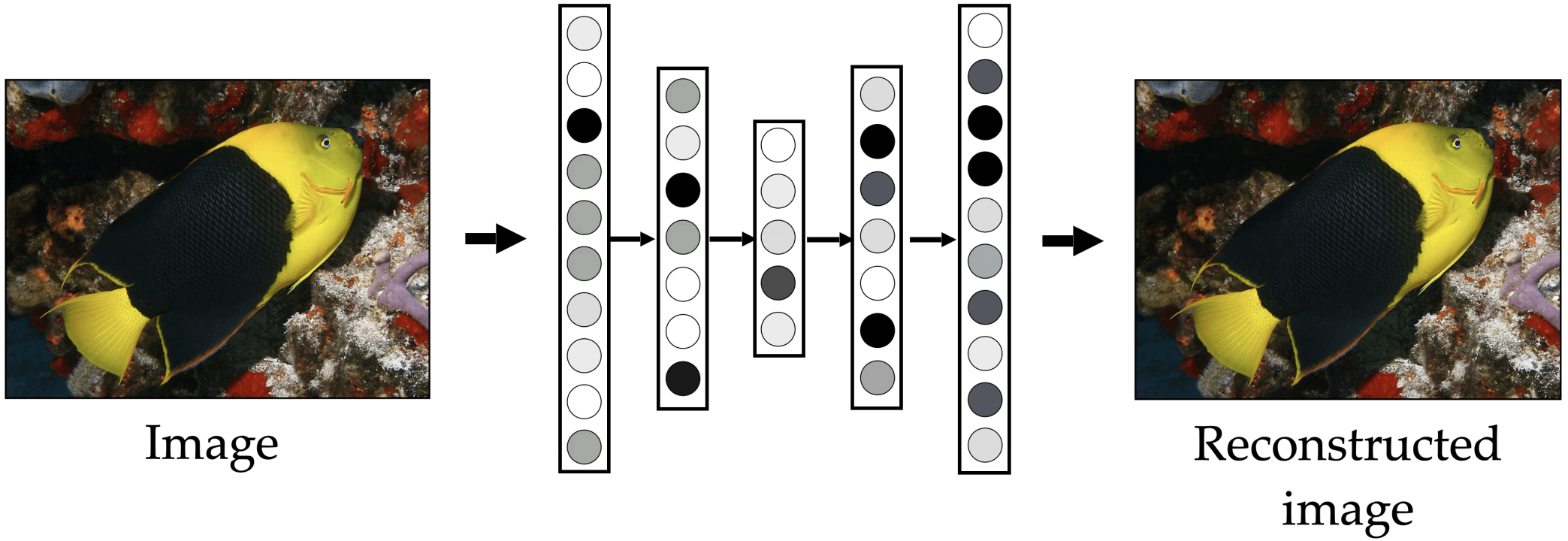
Image

Auto-encoder

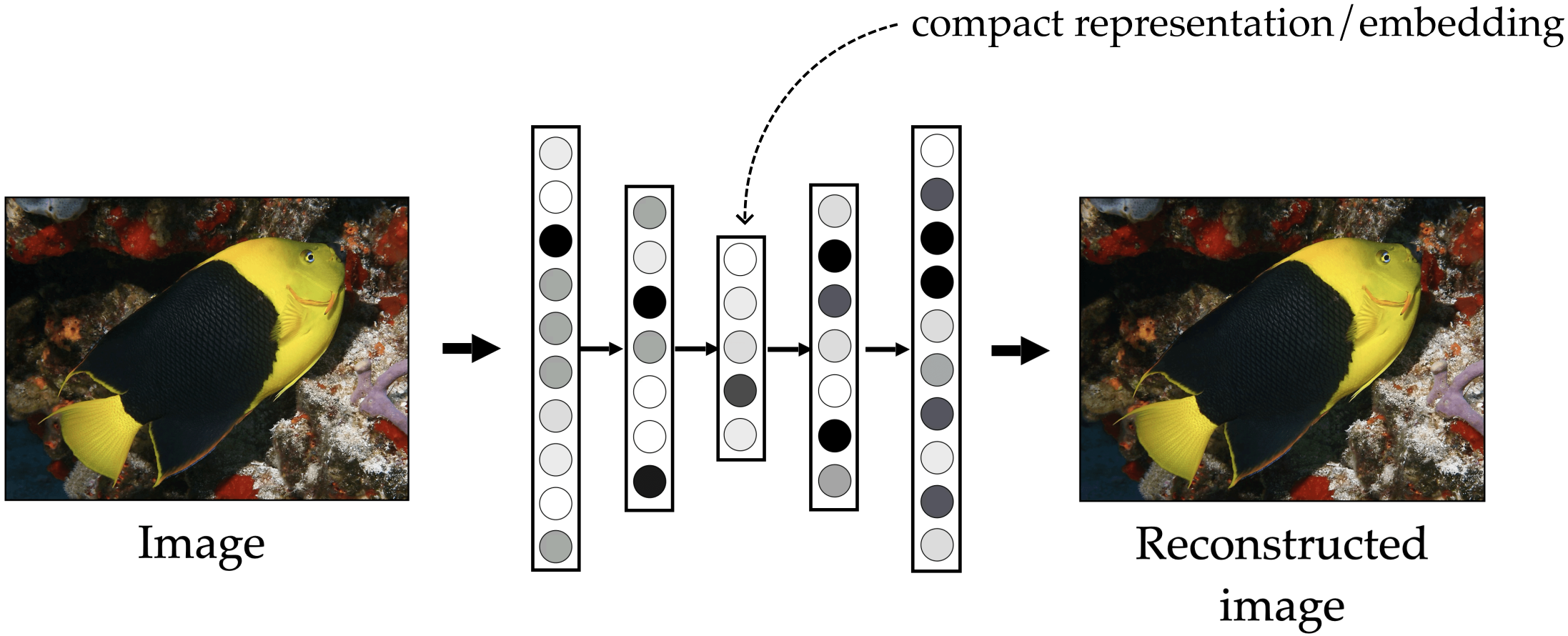
Image

Reconstructed image

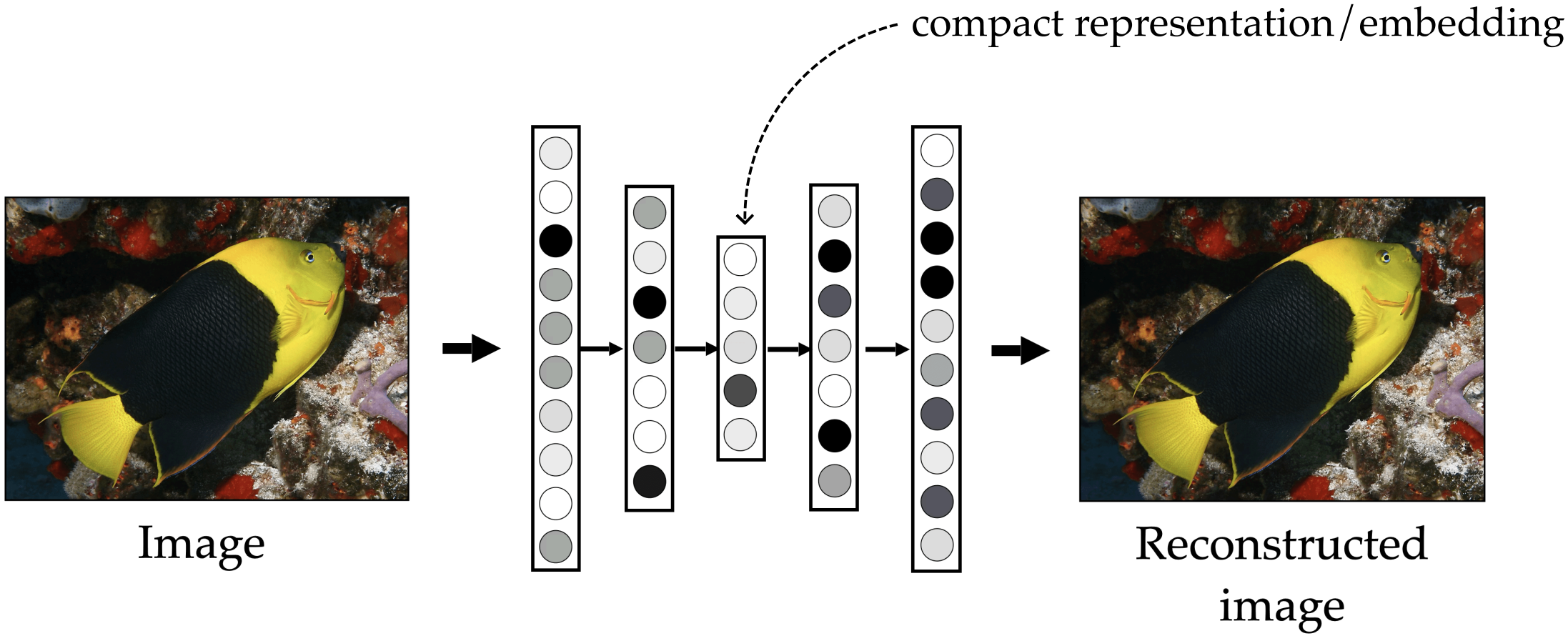
Auto-encoder



Auto-encoder



Auto-encoder

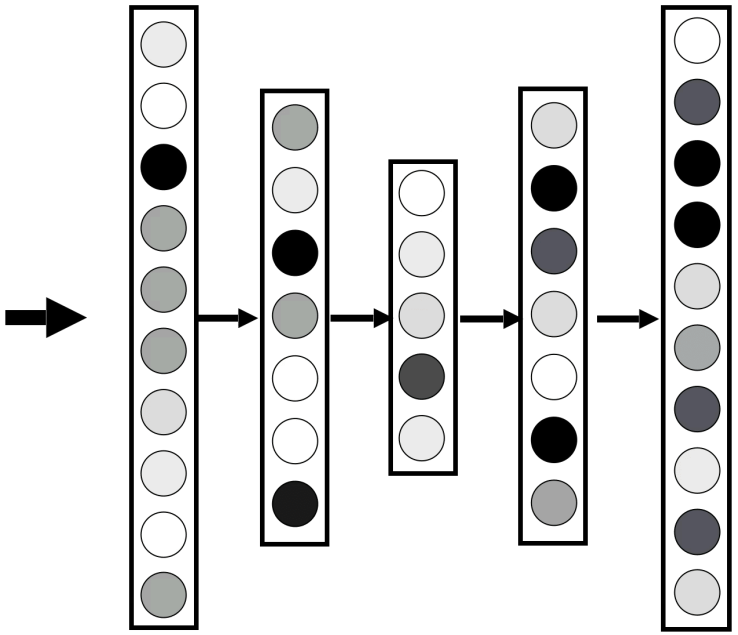


"*What I cannot create, I do not understand*." Feynman

Auto-encoder

x

Image

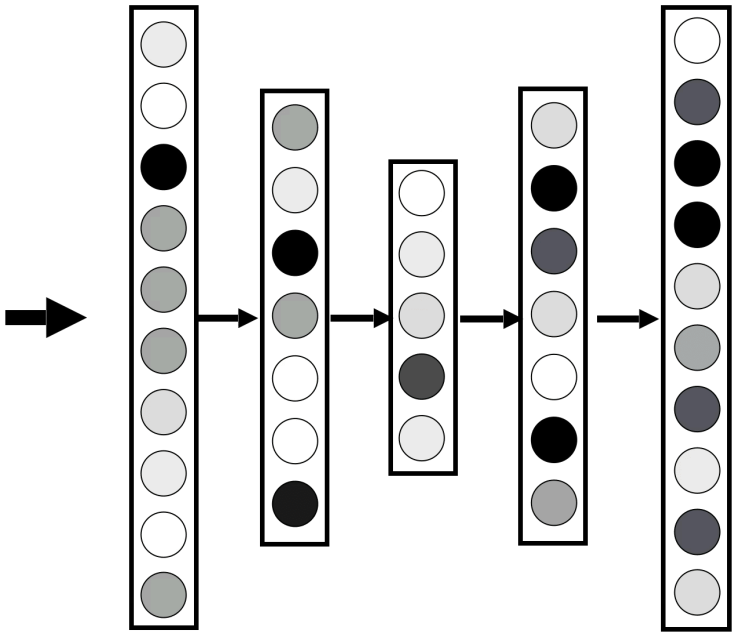


Reconstructed image

Auto-encoder

x

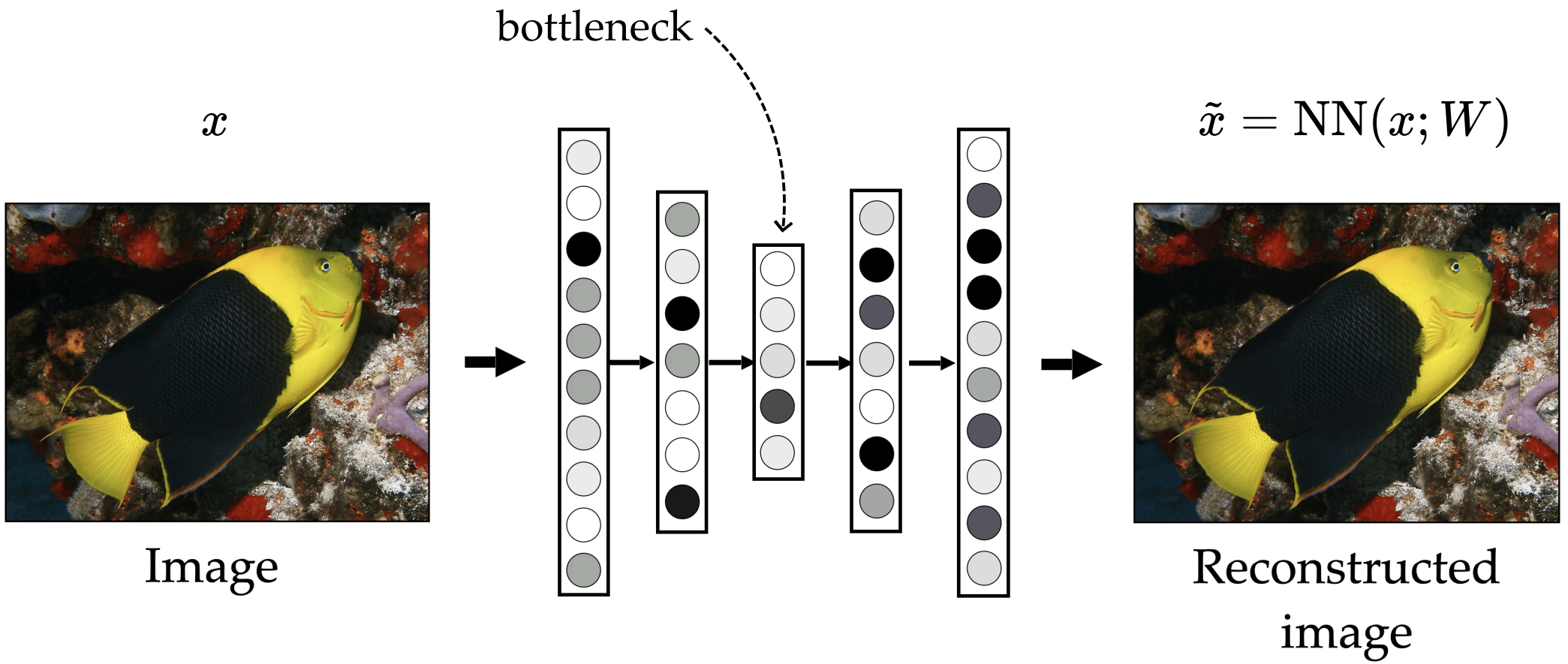
Image



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

Reconstructed image

Auto-encoder



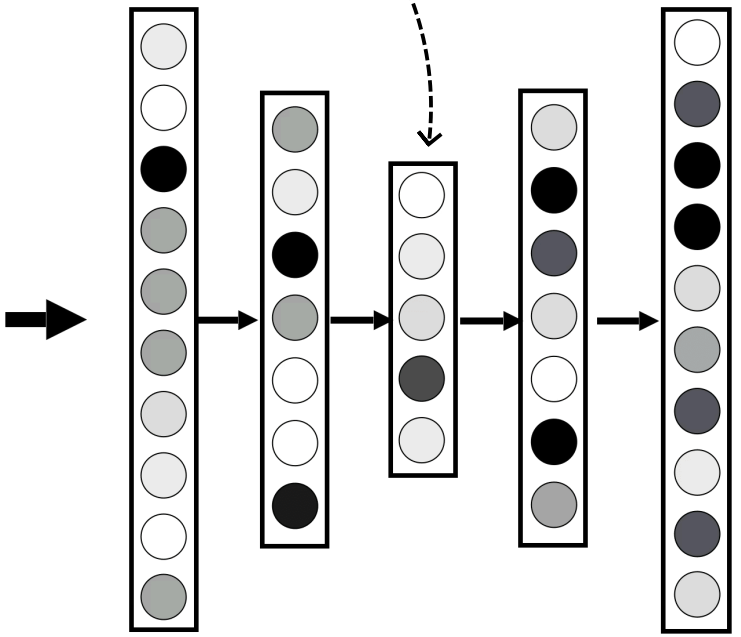
Auto-encoder

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

bottleneck

x

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

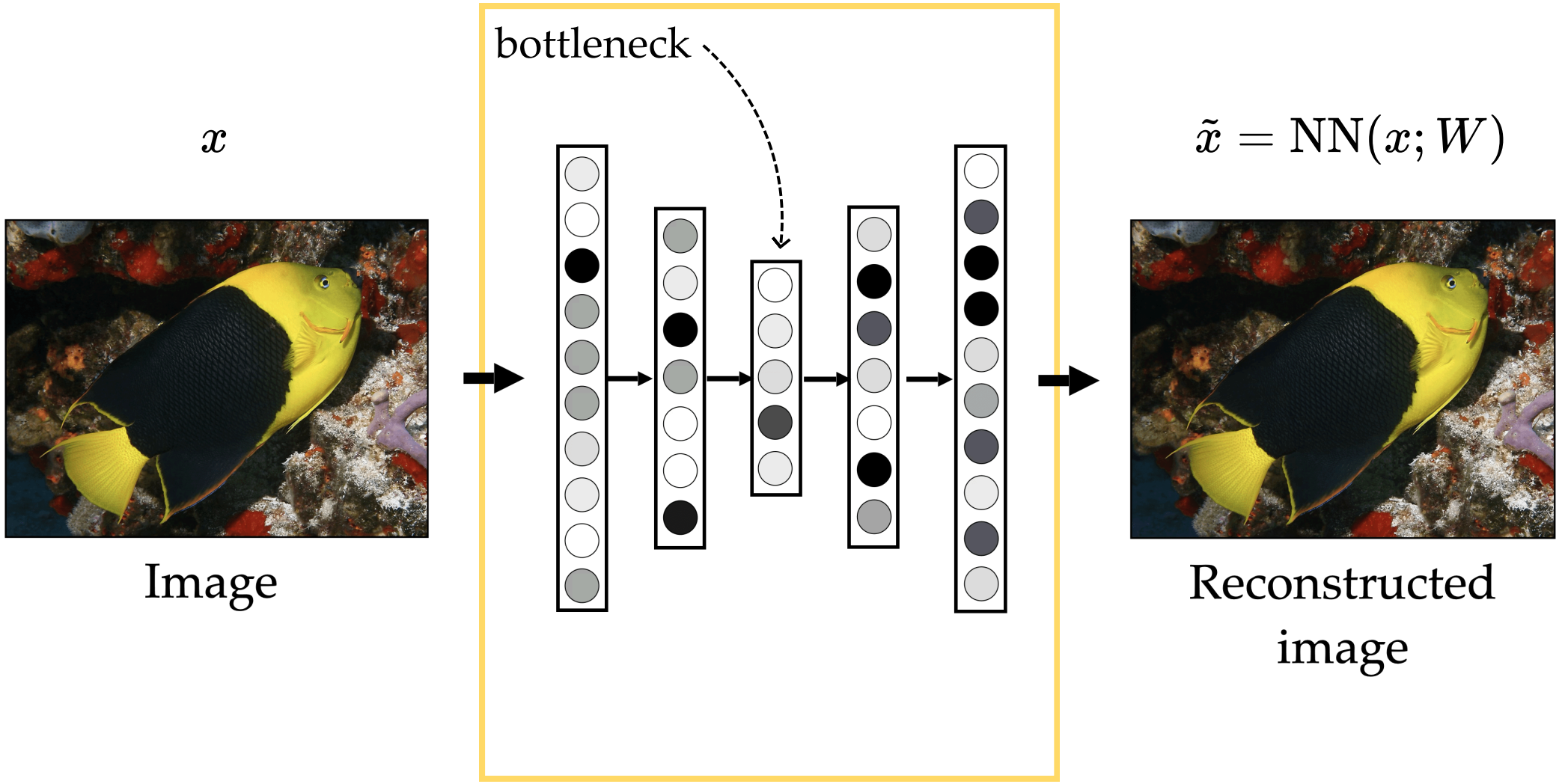


Image

Reconstructed image

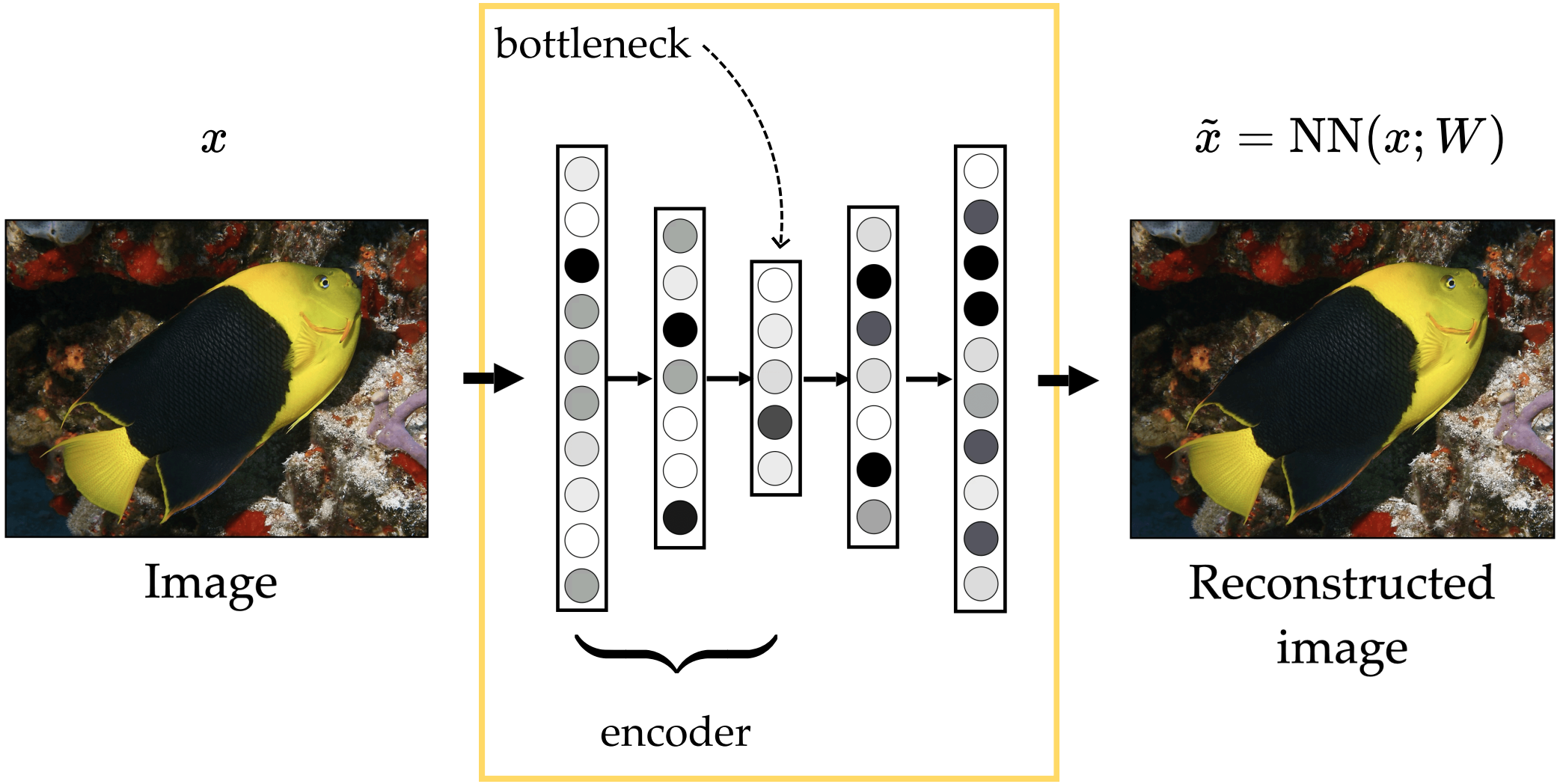
Auto-encoder

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



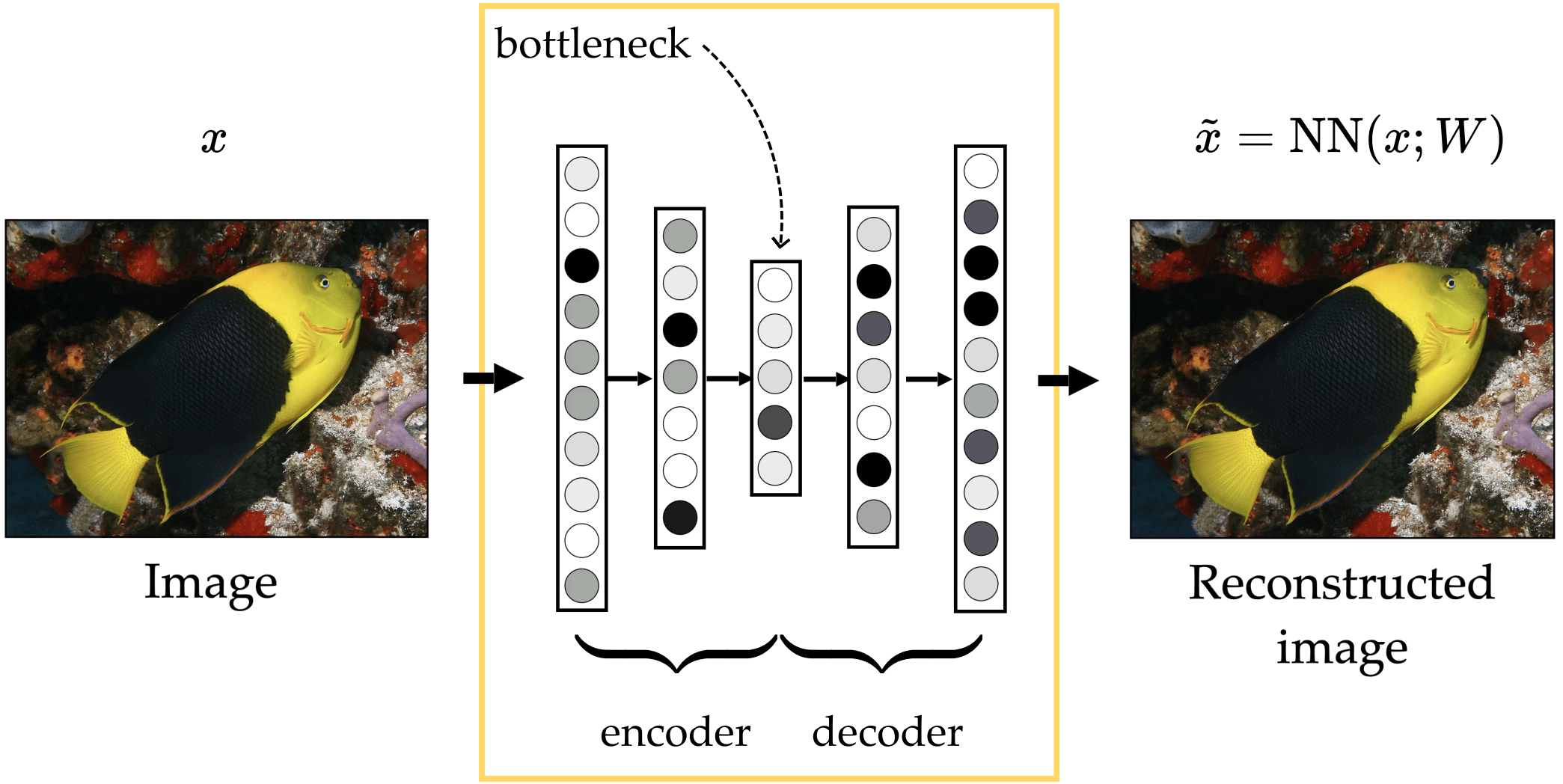
Auto-encoder

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



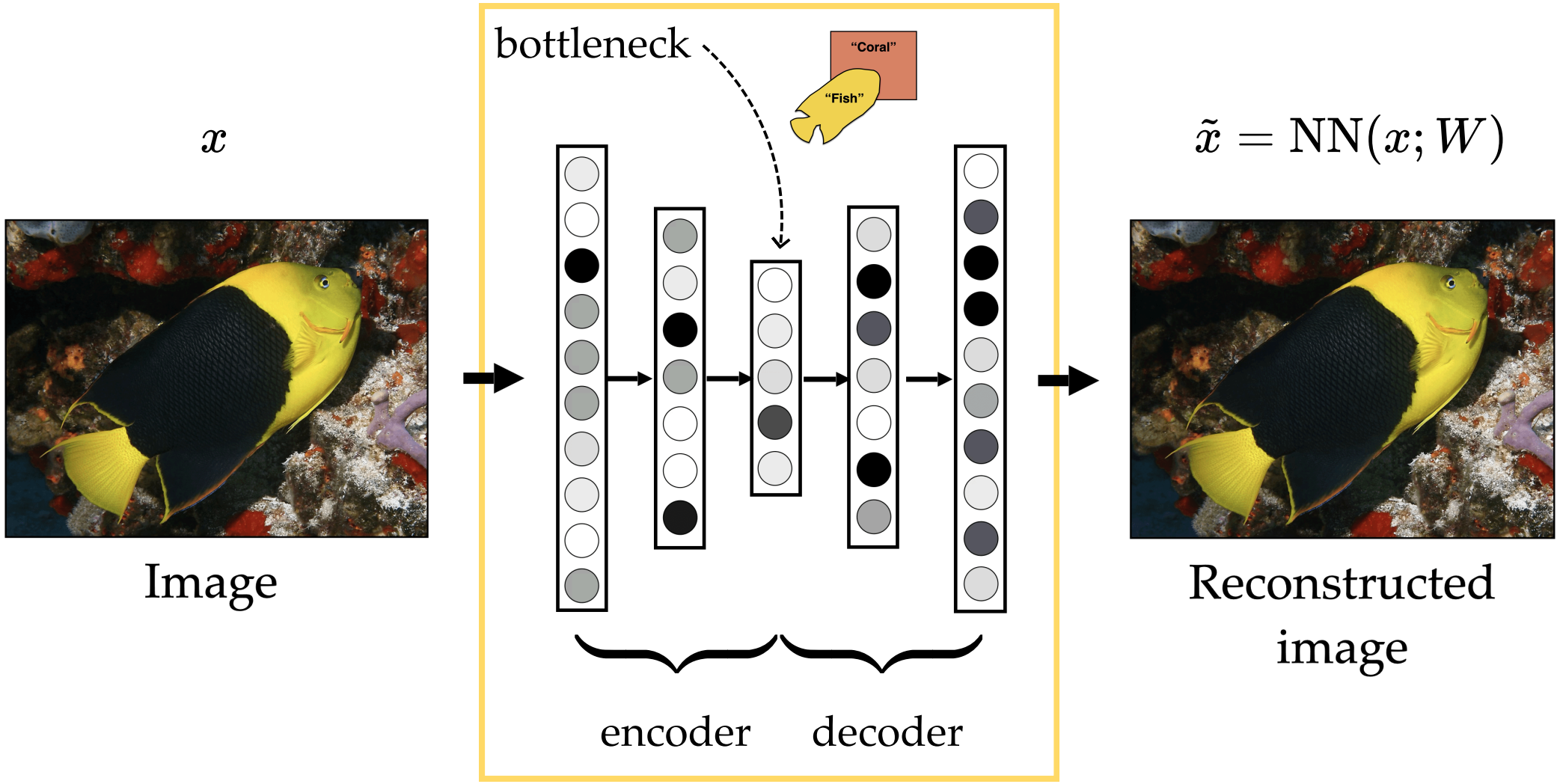
Auto-encoder

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



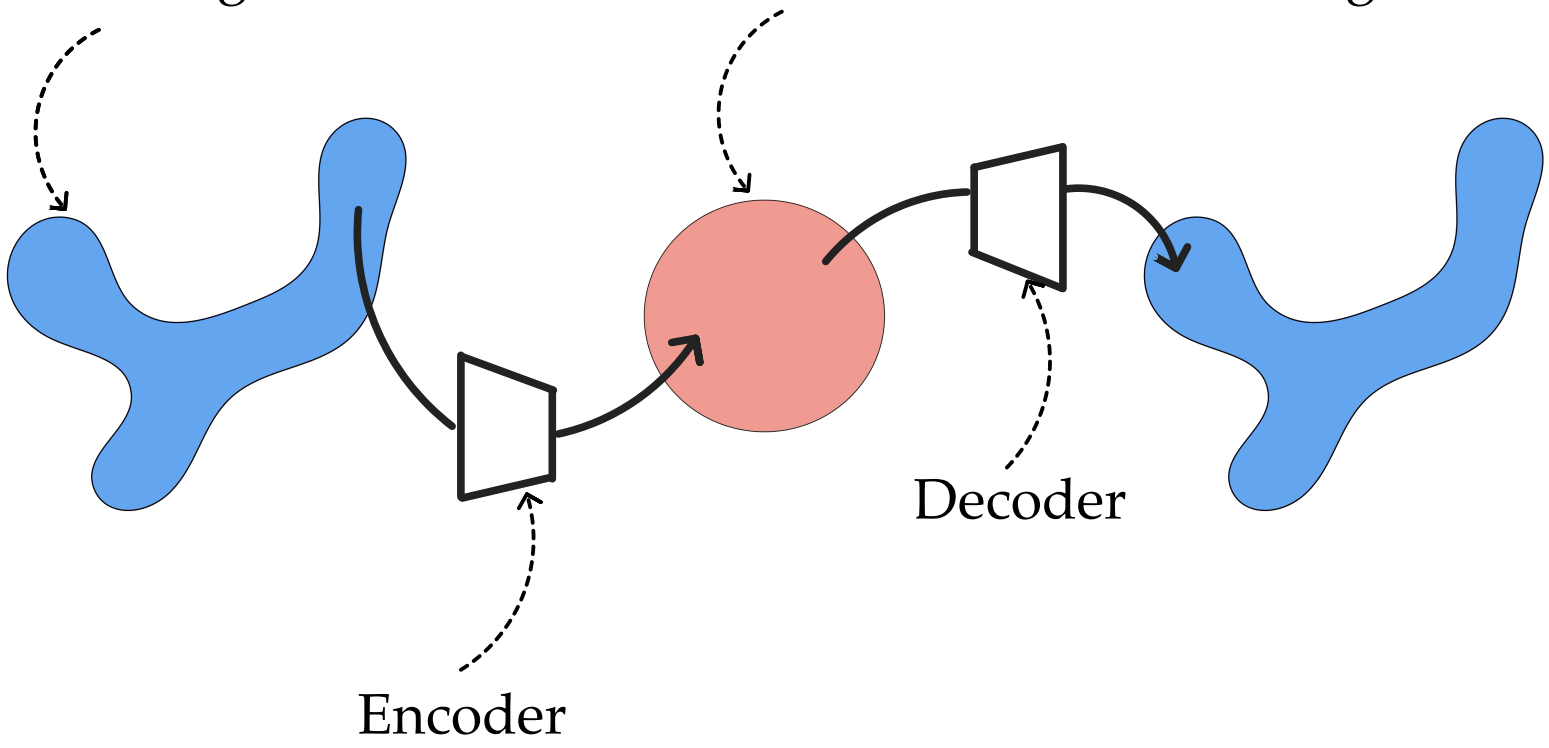
Auto-encoder

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



Data space:
(high dimensional, irregular)

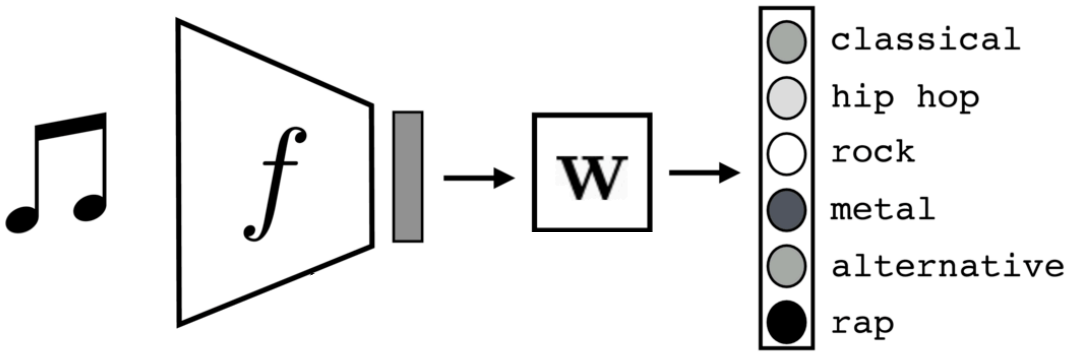
Representation space
(low dimensional, regular)



[Typically, encoders can serve as a translator to get "good representations",
whereas decoders can serve as "generative models"]

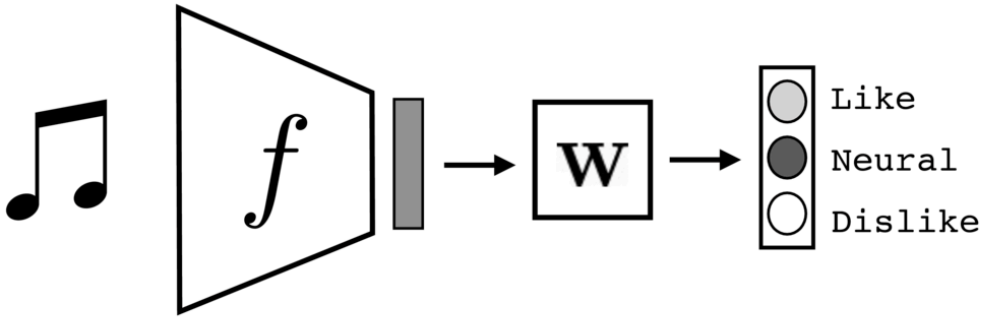
Training

Genre recognition



Testing

Preference prediction

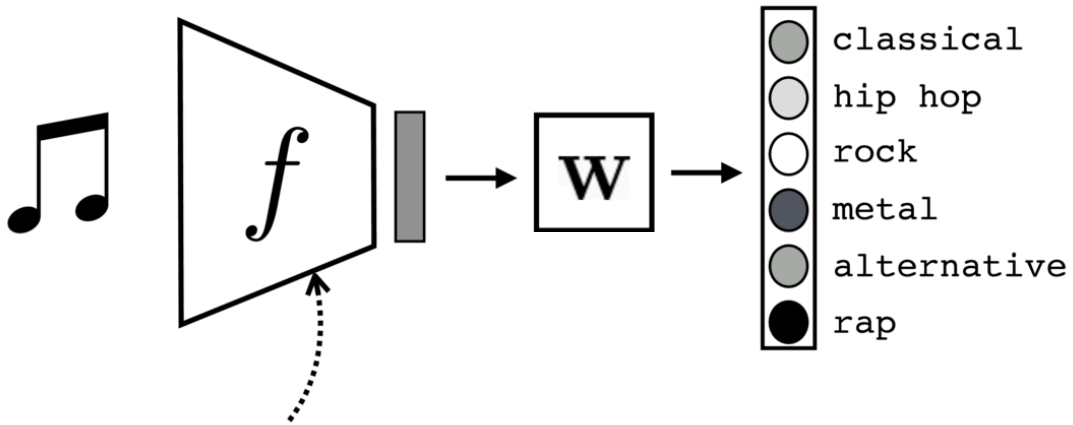


Often, what we will be "tested" on is not what we were trained on.

[images credit: visionbook.mit.edu]

Training

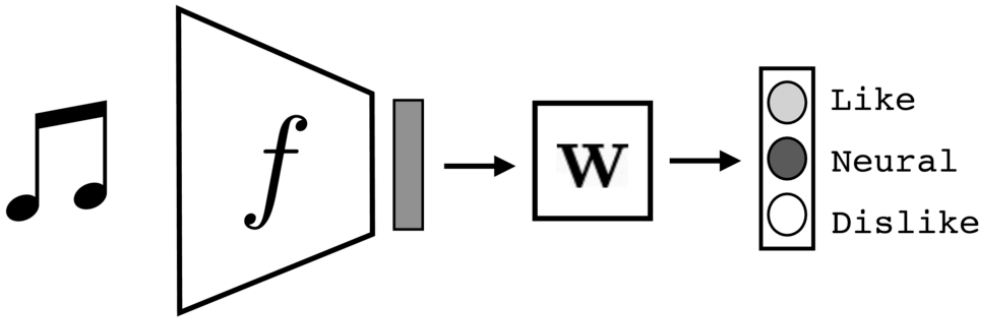
Genre recognition



"common"-sense representation

Testing

Preference prediction

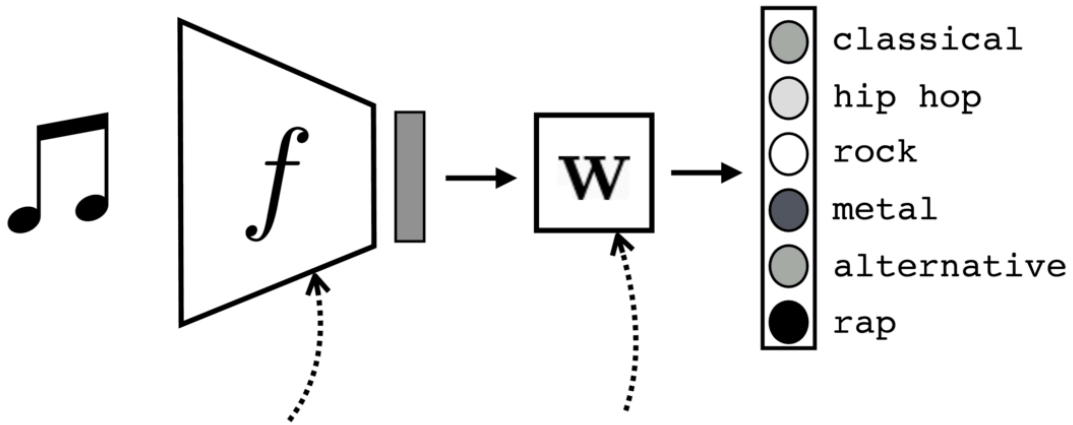


Often, what we will be “tested” on is not what we were trained on.

[images credit: visionbook.mit.edu]

Training

Genre recognition

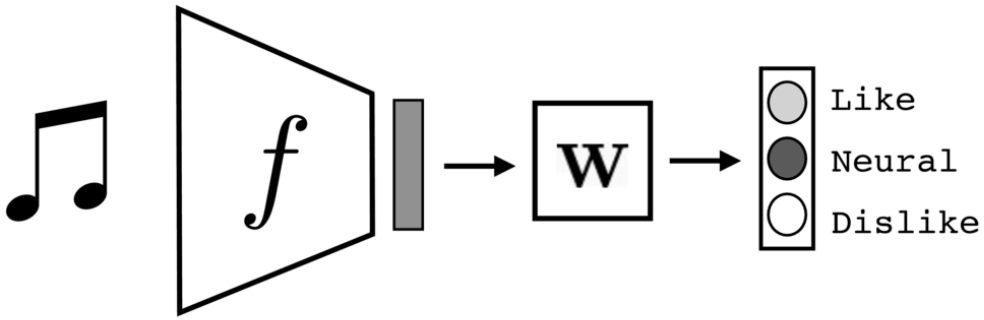


"common"-sense representation

task-specific prediction

Testing

Preference prediction

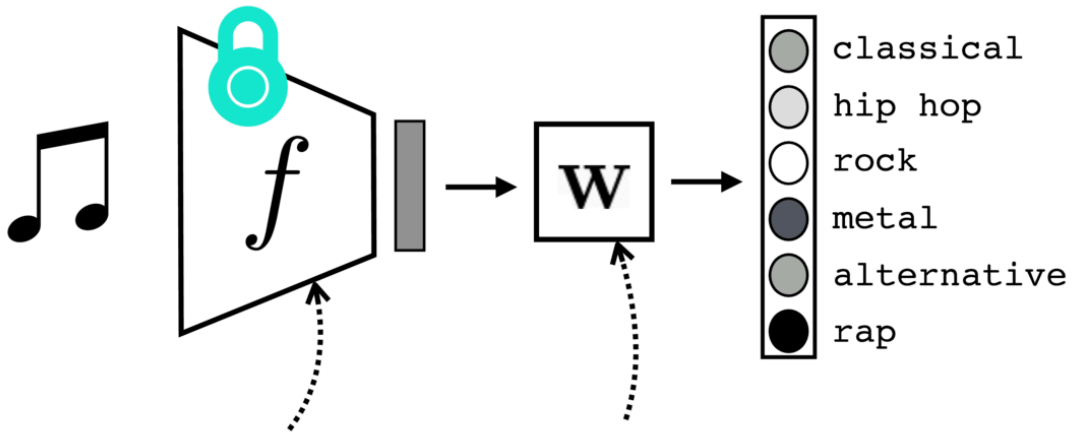


Often, what we will be “tested” on is not what we were trained on.

[images credit: visionbook.mit.edu]

Training

Genre recognition

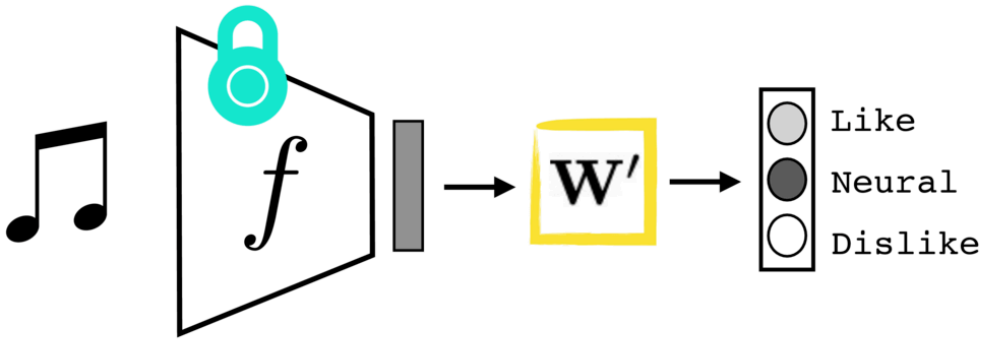


"common"-sense representation

task-specific prediction

Adapting

Preference prediction

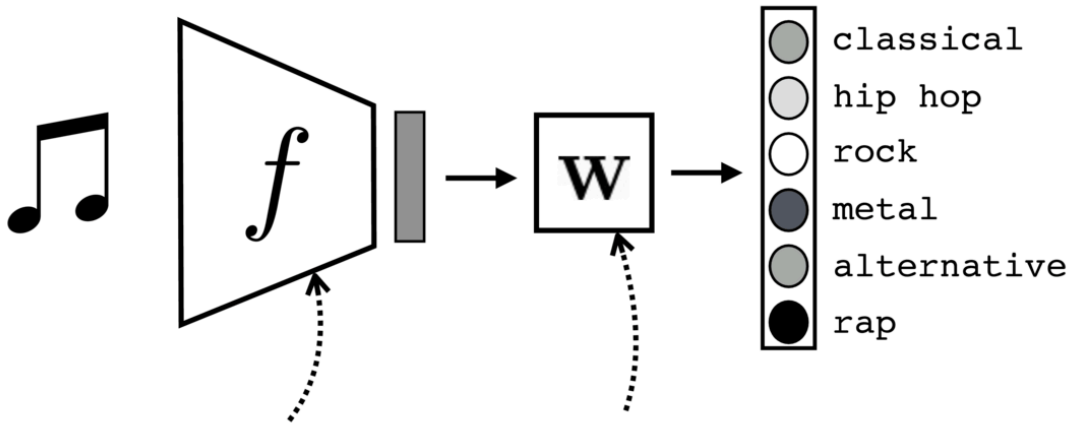


Final-layer adaptation: freeze f, train a new final layer to new target data

[images credit: visionbook.mit.edu]

Training

Genre recognition

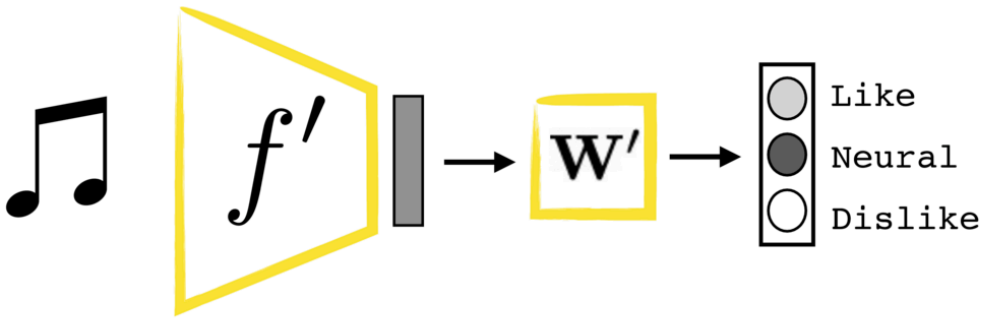


"common"-sense representation

task-specific prediction

Adapting

Preference prediction

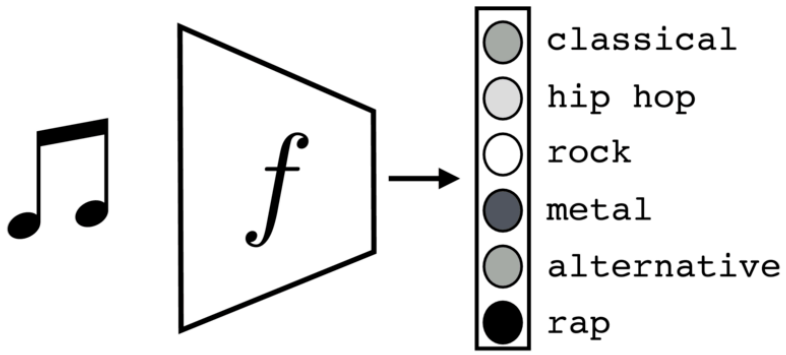


Finetuning: initialize f' as f, then continue training for f' as well, on new target data

[images credit: visionbook.mit.edu]

Pretraining

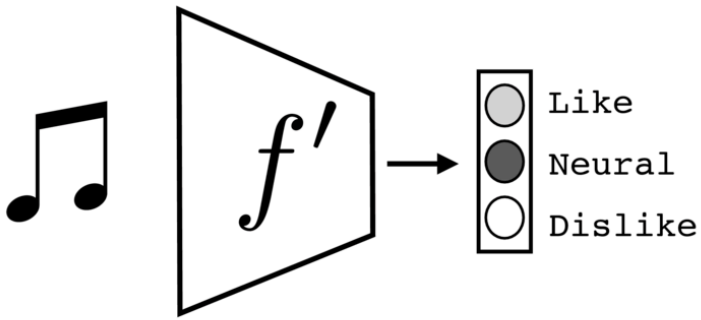
Genre recognition



A lot of data

Adapting

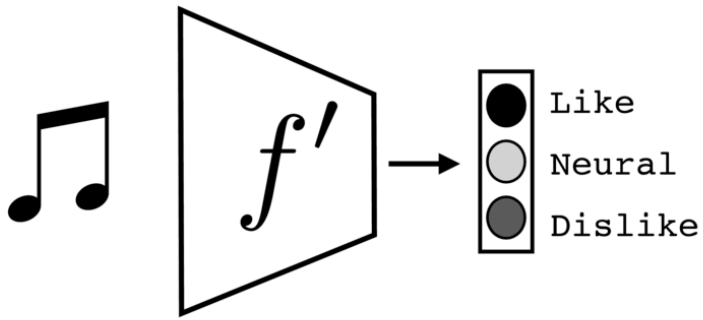
Preference prediction



A little data

Testing

Preference prediction

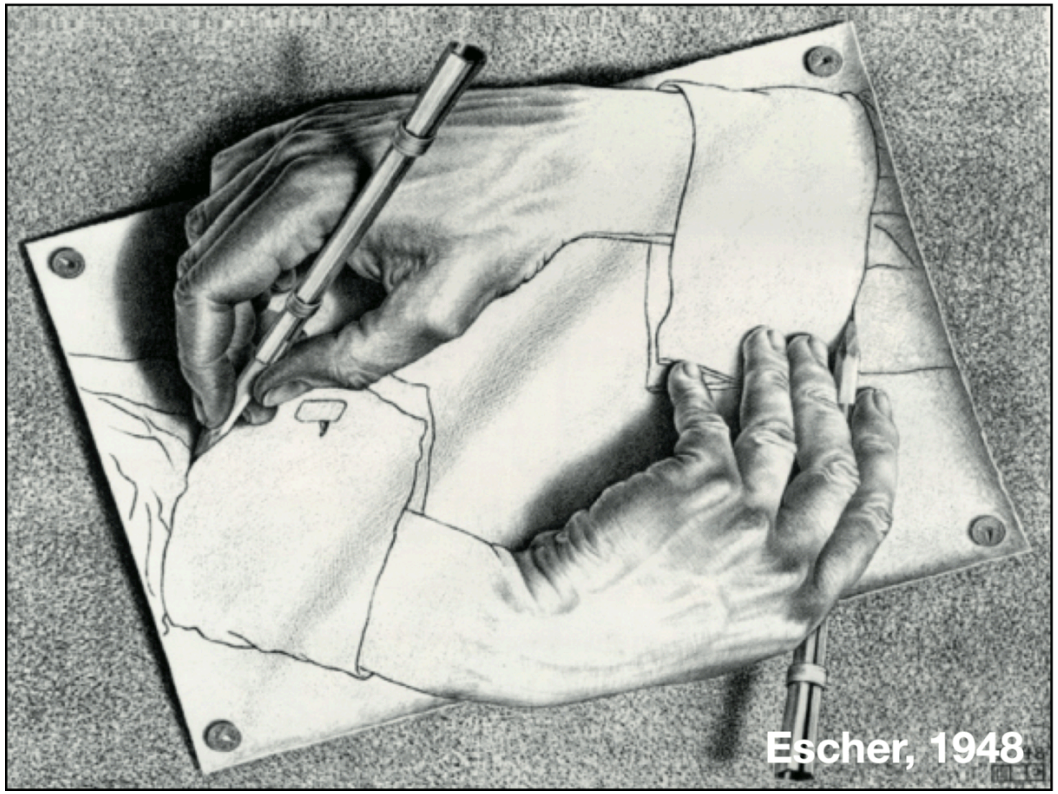


[images credit: visionbook.mit.edu]

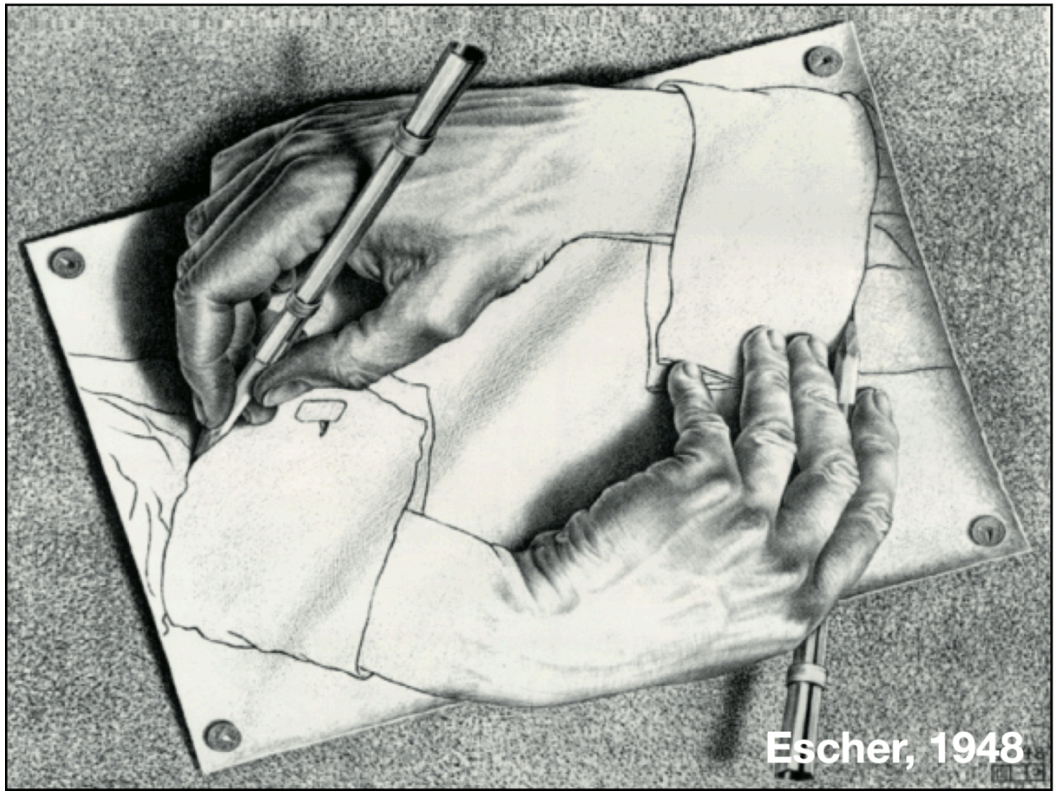
Outline

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

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

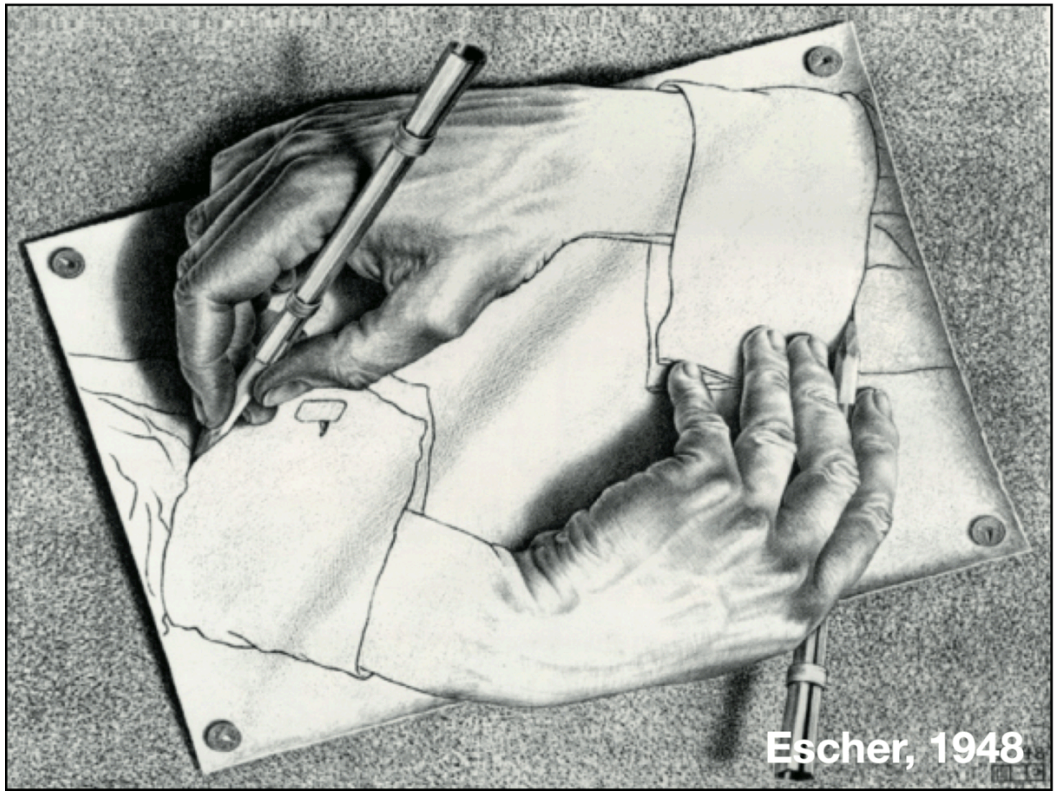


Large Language Models (LLMs) are trained in this self-supervised way



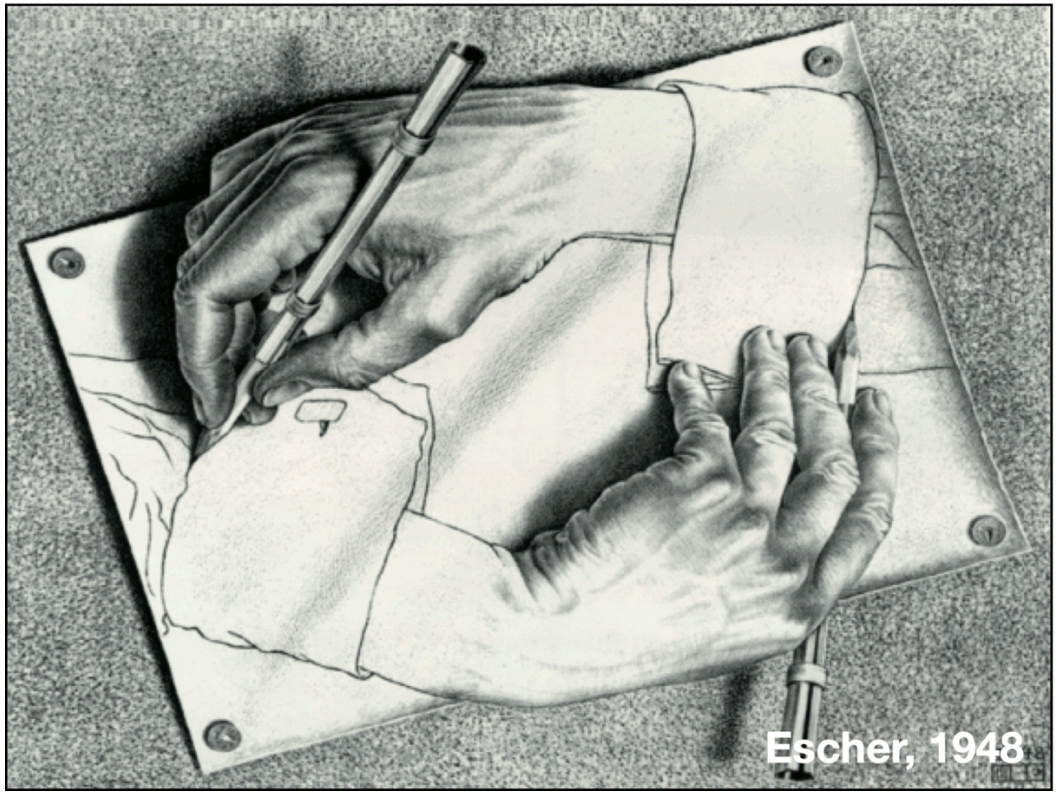
- Scrape the internet for plain texts.

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



- Scrape the internet for plain texts.
- Cook up “labels” (prediction targets) from these texts.

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



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

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

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

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

feature	label
⋮	⋮
To date, the	cleverest

"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 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. "

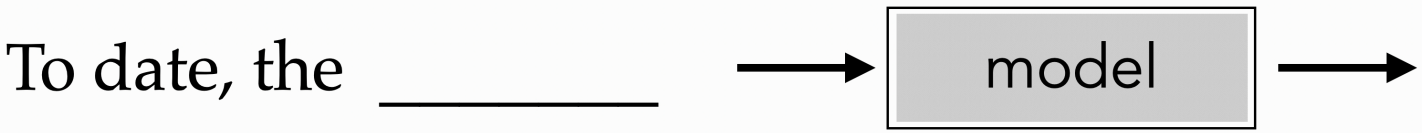
feature	label
⋮	⋮
To date, the	cleverest
To date, the cleverest	thinker
To date, the cleverest thinker	was
⋮	⋮

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

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

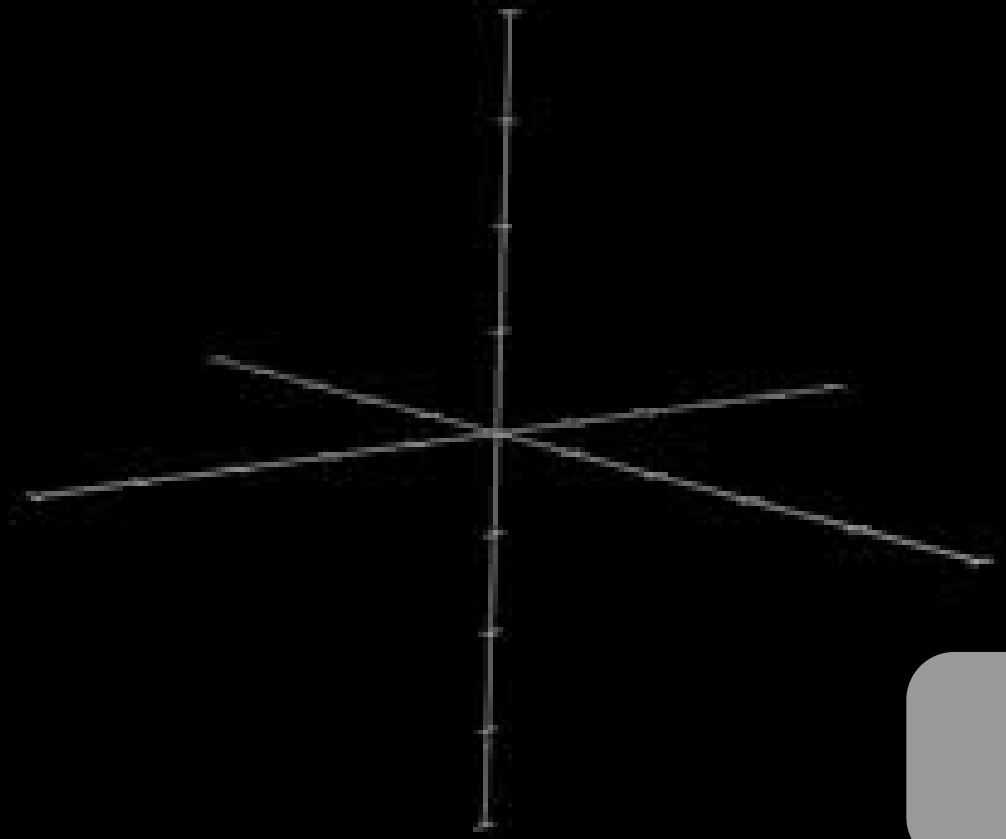
"To date, the cleverest thinker of all time was Issac. "

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



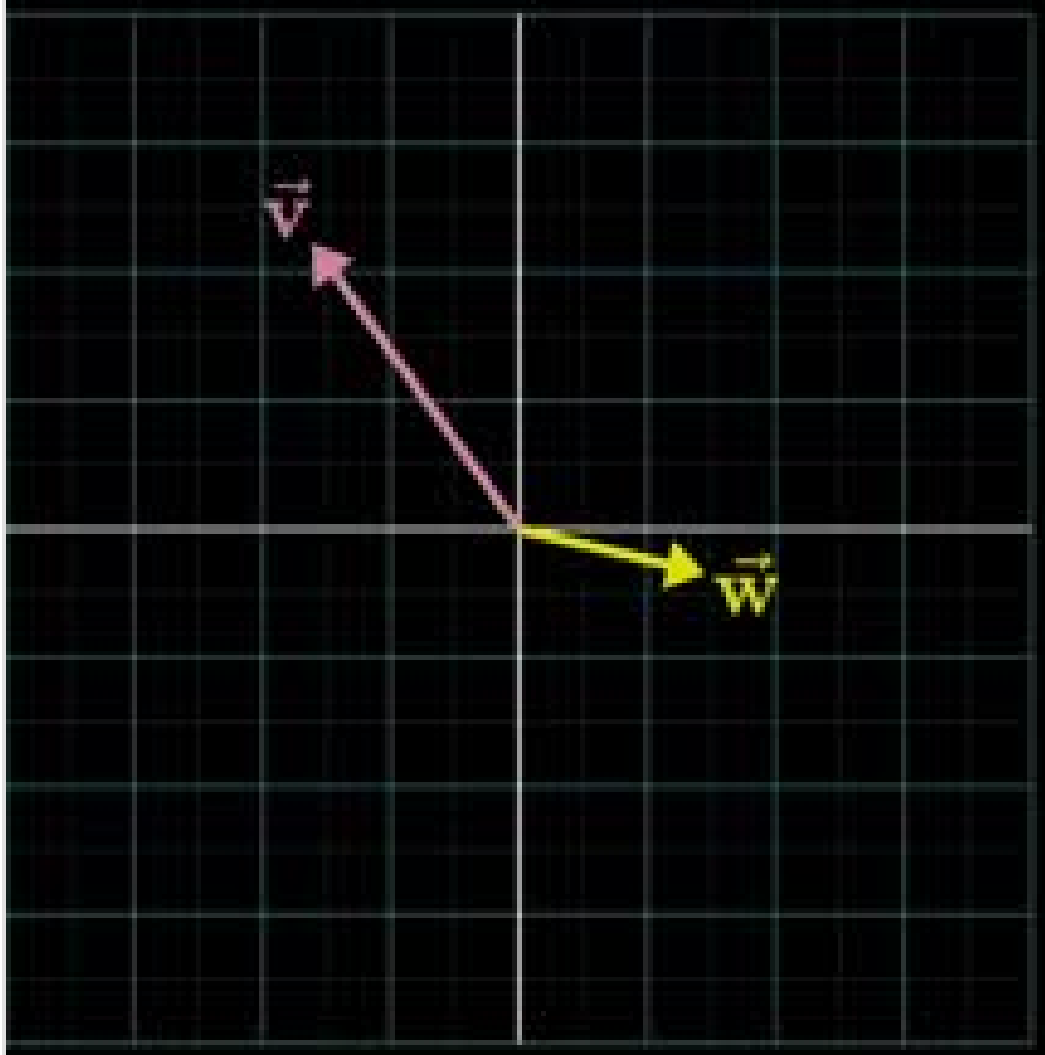
Word embedding

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 word embeddings space is equipped with sensible dot-product similarity

Good word embeddings space is equipped with sensible dot-product similarity

For now, let's look at how good embeddings enable "soft" dictionary look-up:

Good word embeddings space is equipped with sensible dot-product similarity

For now, let's look at how good embeddings enable "soft" dictionary look-up:

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

Good word embeddings space is equipped with sensible dot-product similarity

For now, let's look at how good embeddings enable "soft" dictionary look-up:

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

Key Value

Good word embeddings space is equipped with sensible dot-product similarity

For now, let's look at how good embeddings enable "soft" dictionary look-up:

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

Key Value

apple : pomme

Good word embeddings space is equipped with sensible dot-product similarity

For now, let's look at how good embeddings enable "soft" dictionary look-up:

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

Key Value

apple : pomme

banana : banane

Good word embeddings space is equipped with sensible dot-product similarity

For now, let's look at how good embeddings enable "soft" dictionary look-up:

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

Key Value

apple : pomme

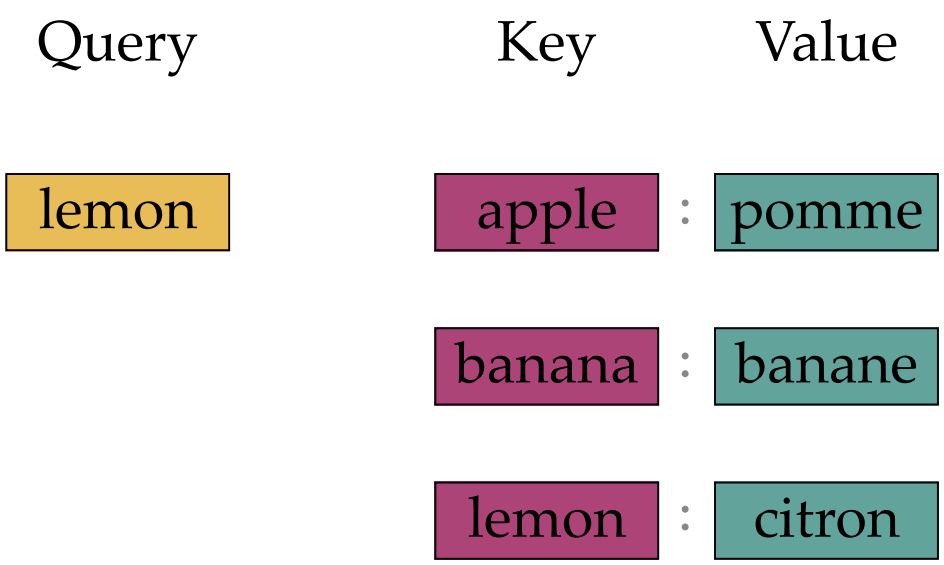
banana : banane

lemon : citron

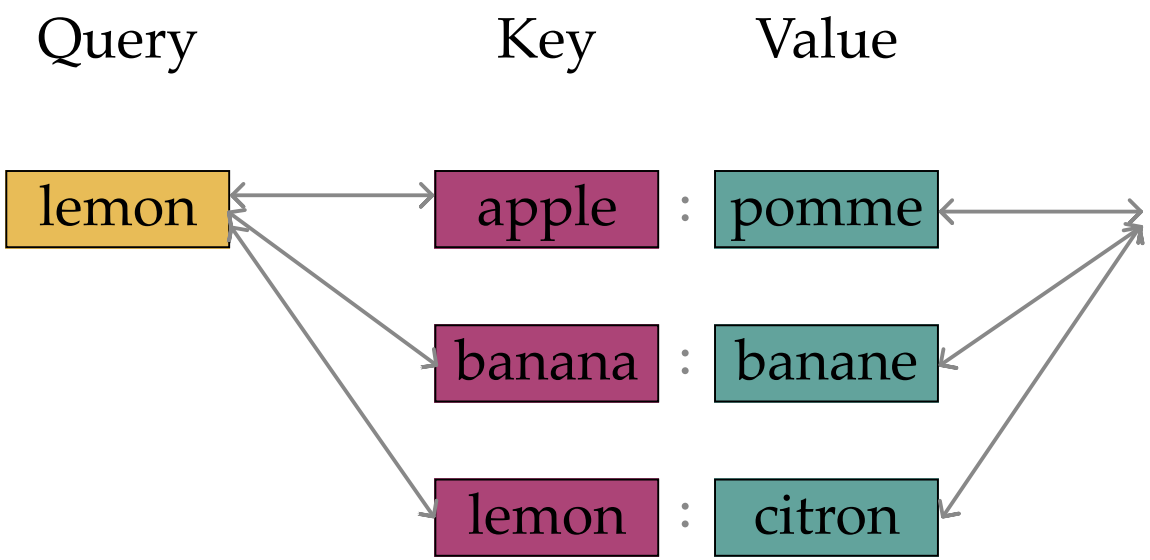
```python
dict_en2fr = {
  "apple" : "pomme",
  "banana": "banane",
  "lemon" : "citron"}

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

Key		Value
apple	:	pomme
banana	:	banane
lemon	:	citron

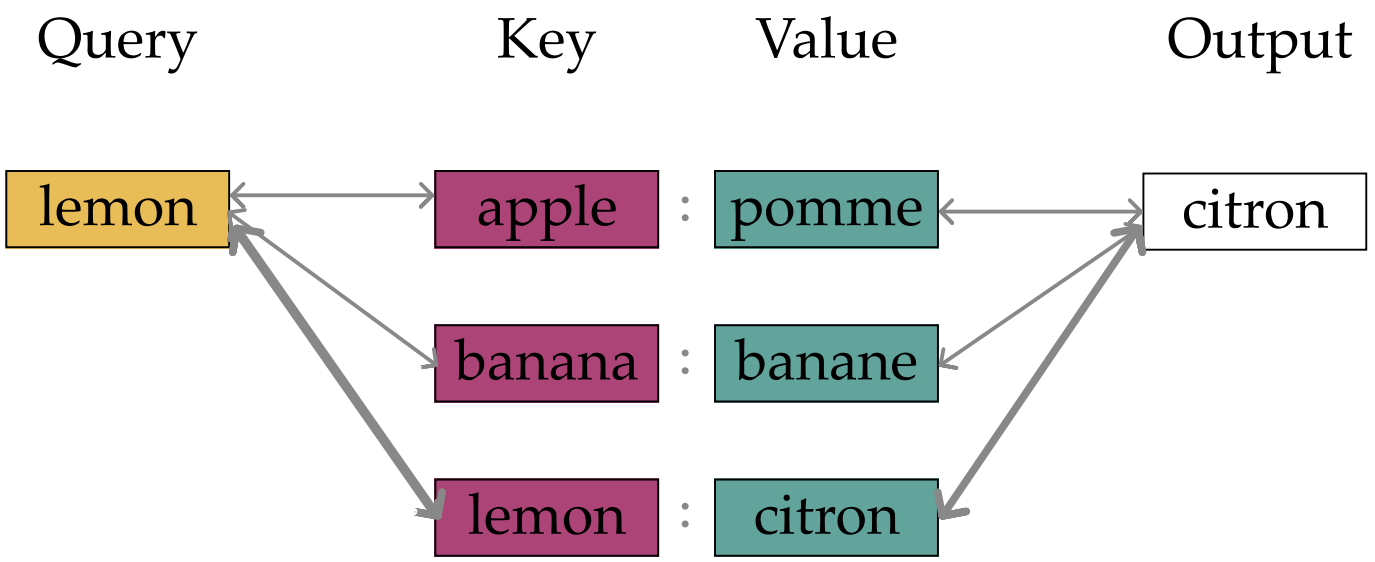
```python
dict_en2fr = {
  "apple" : "pomme",
  "banana": "banane",
  "lemon" : "citron"}

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



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

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

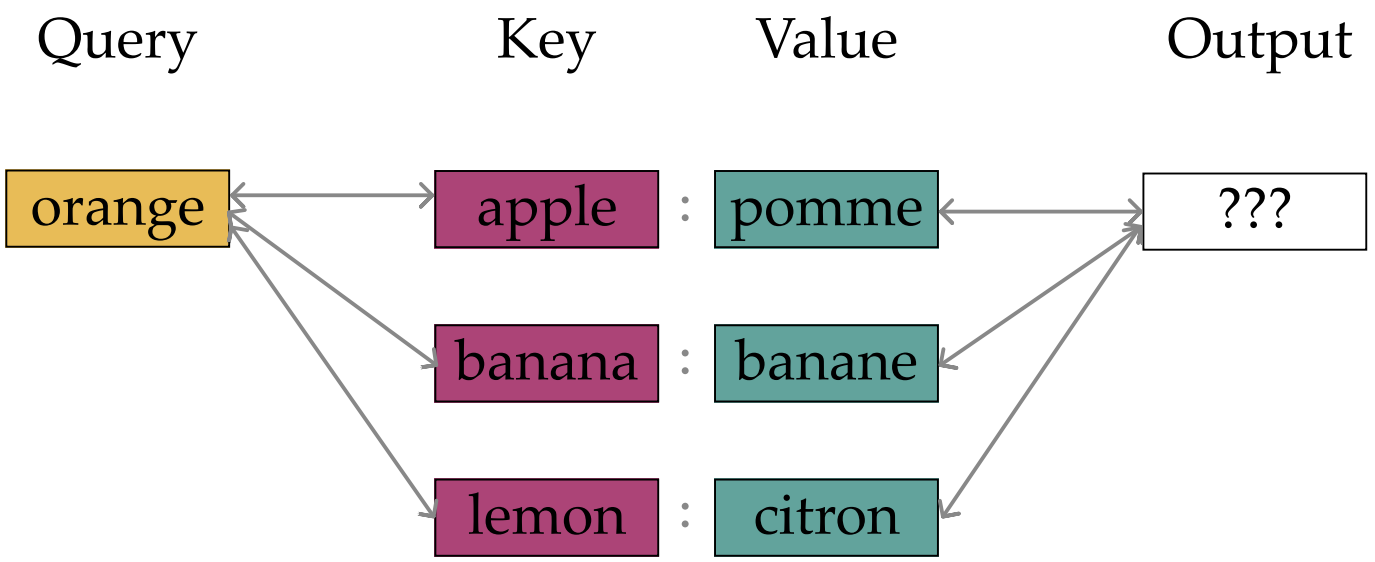


```python
dict_en2fr = {
  "apple" : "pomme",
  "banana": "banane",
  "lemon" : "citron"}

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



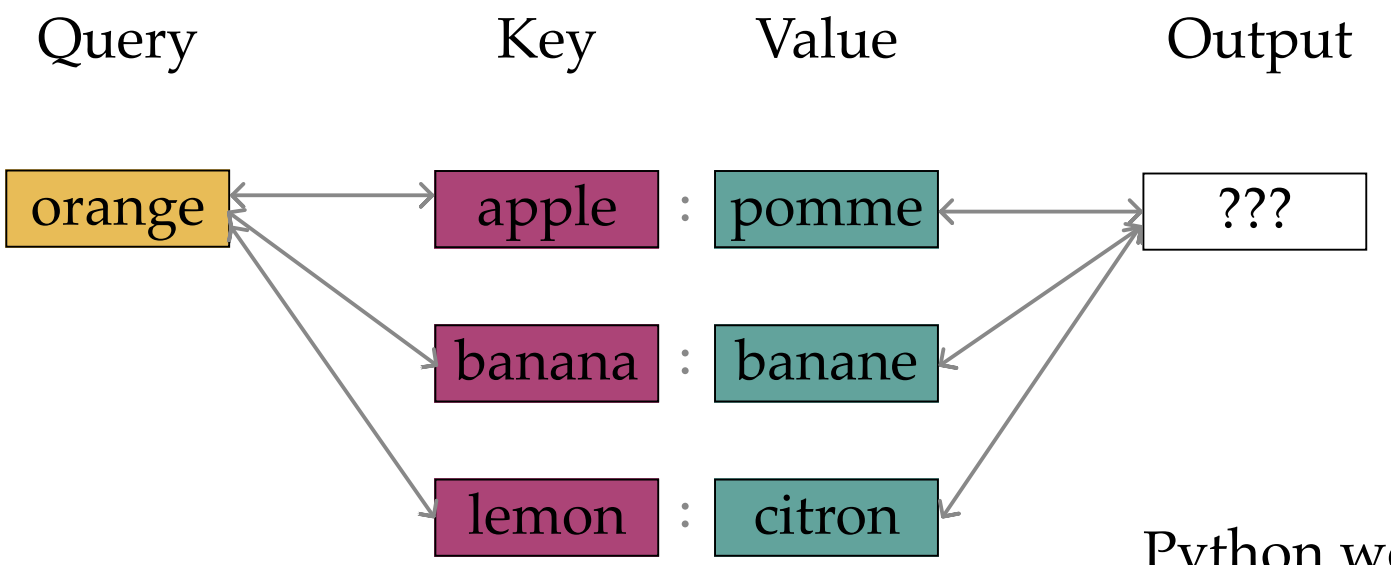
What if we run

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

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



What if we run

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

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

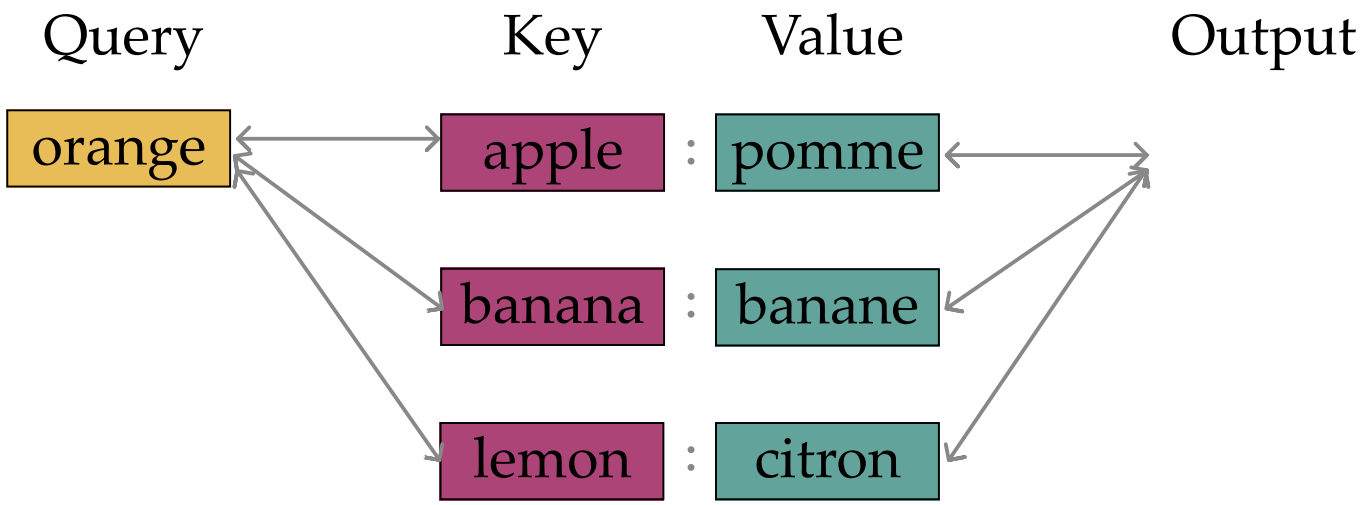


Python would complain. 🤯

What if we run

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

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

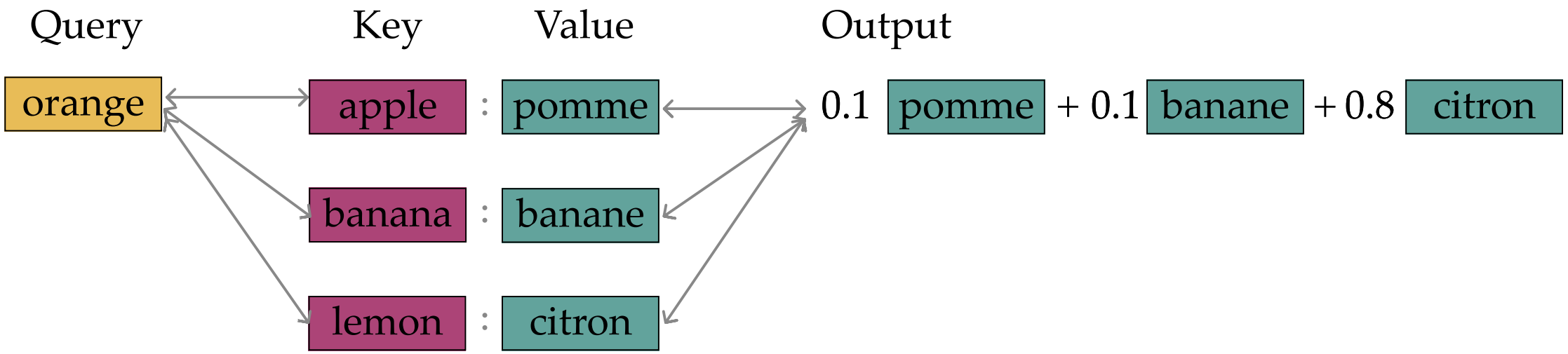
But we can probably see the rationale behind this:



What if we run

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

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

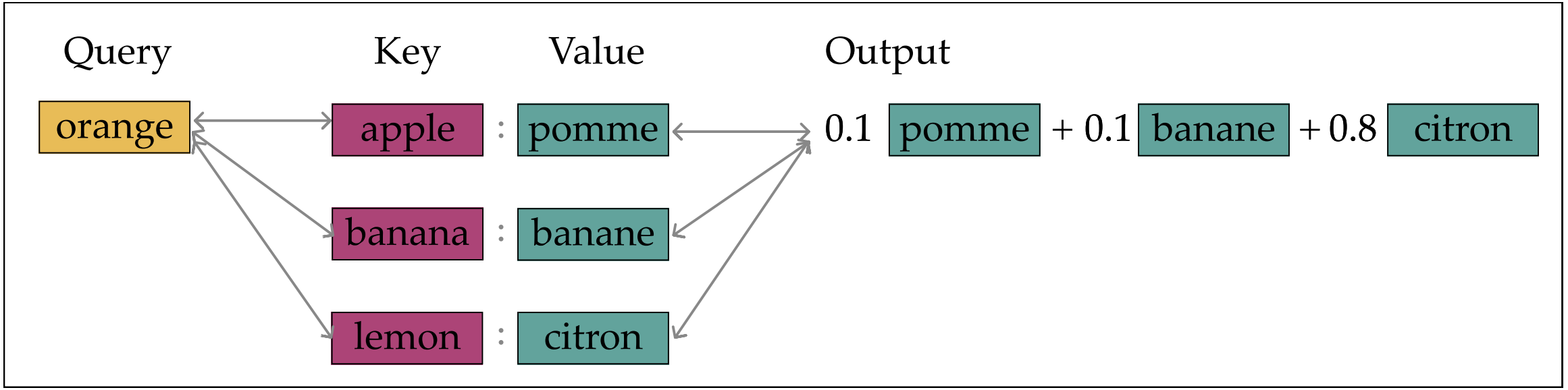
But we can probably see the rationale behind this:

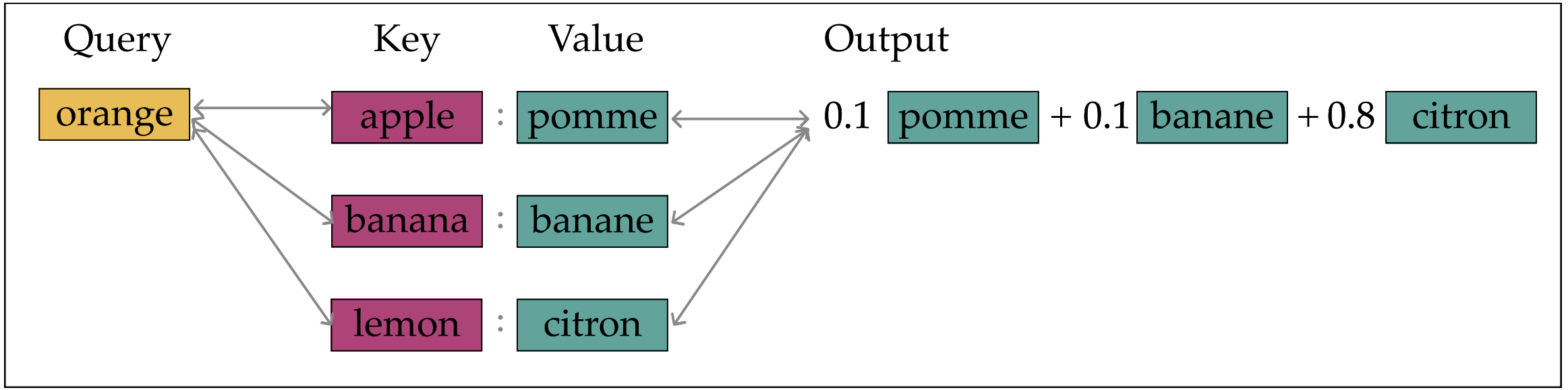


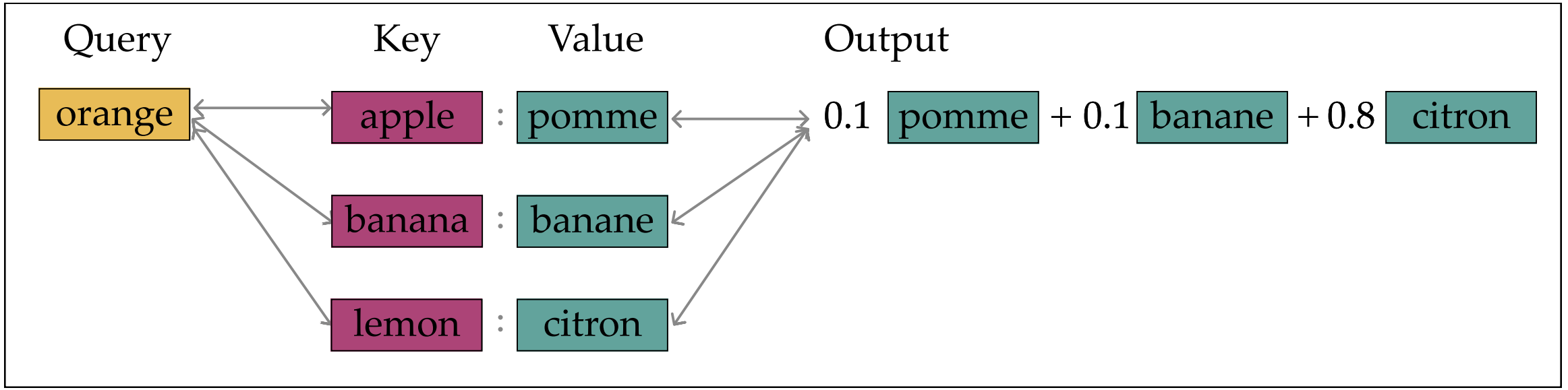
What if we run

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

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

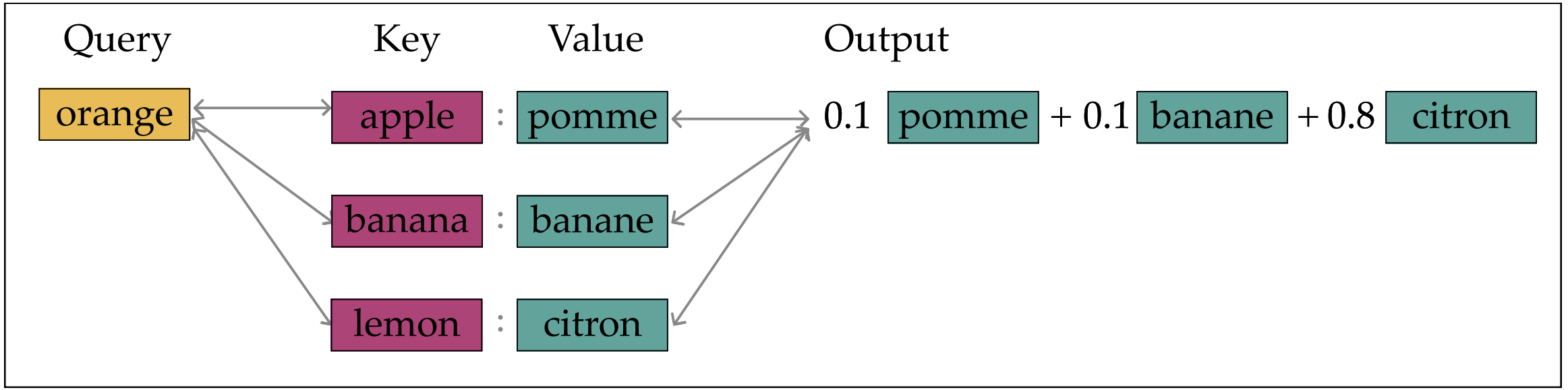
But we can probably see the rationale behind this:




Query
Key
Value
Output
orange
apple
:
pomme
banana
:
banane
lemon
:
citron
0.1
pomme
+ 0.1
banane
+ 0.8
citron

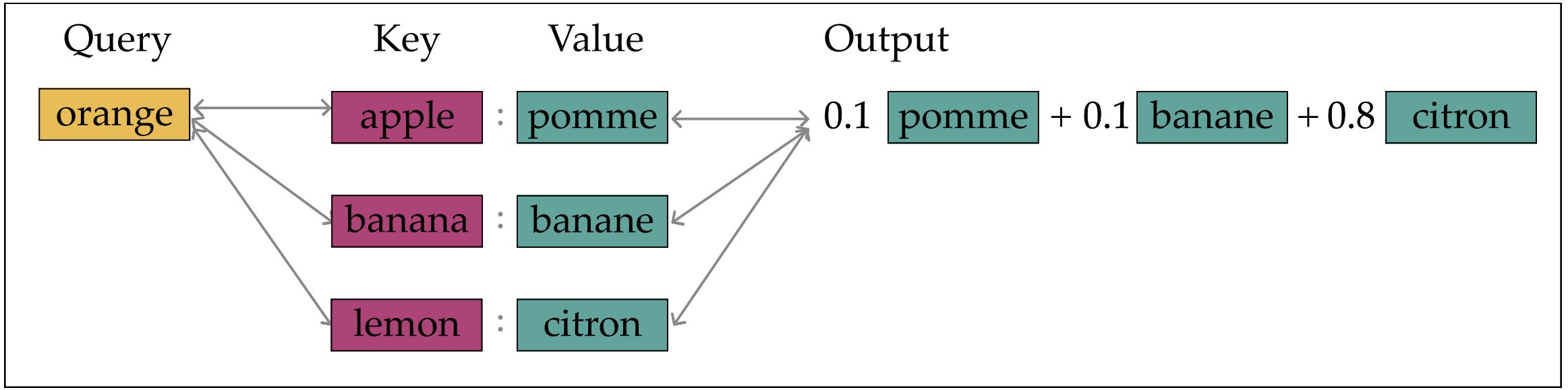


We put (query, key, value) in "good" embeddings in our human brain



We put (query, key, value) in "good" embeddings in our human brain

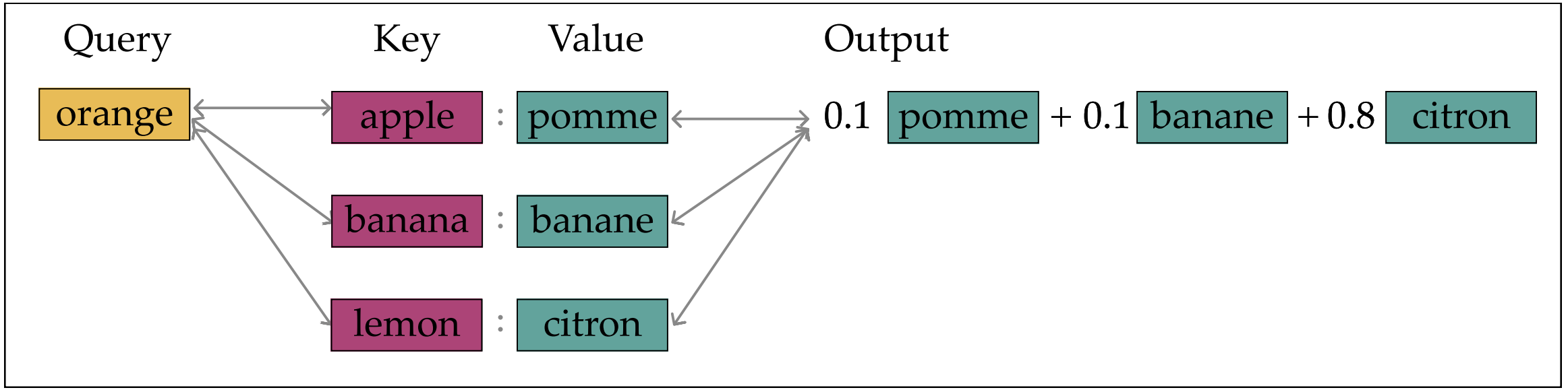
such that "merging" the values 0.1 pomme + 0.1 banane + 0.8 citron



We put (query, key, value) in "good" embeddings in our human brain

such that "merging" the values 0.1 pomme + 0.1 banane + 0.8 citron

via these merging percentages [0.1 0.1 0.8] made sense



We put (query, key, value) in "good" embeddings in our human brain

such that "merging" the values 0.1 pomme + 0.1 banane + 0.8 citron

via these merging percentages [0.1 0.1 0.8] made sense

very roughly, the attention mechanism in transformers automates this process. We'll see the details next week.

Outline

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

[Section slides partially adapted from Kaiming He, Philip Isola, and Andrew Owens]

Outline

- Neural networks are *representation* learners
- Self-supervised learning
- Auto-encoders
- Word embeddings
- (Some recent representation learning ideas)
    - 1. masking: other applications
    - 2. contrastive: similarity

[Section slides partially adapted from Kaiming He, Philip Isola, and Andrew Owens]

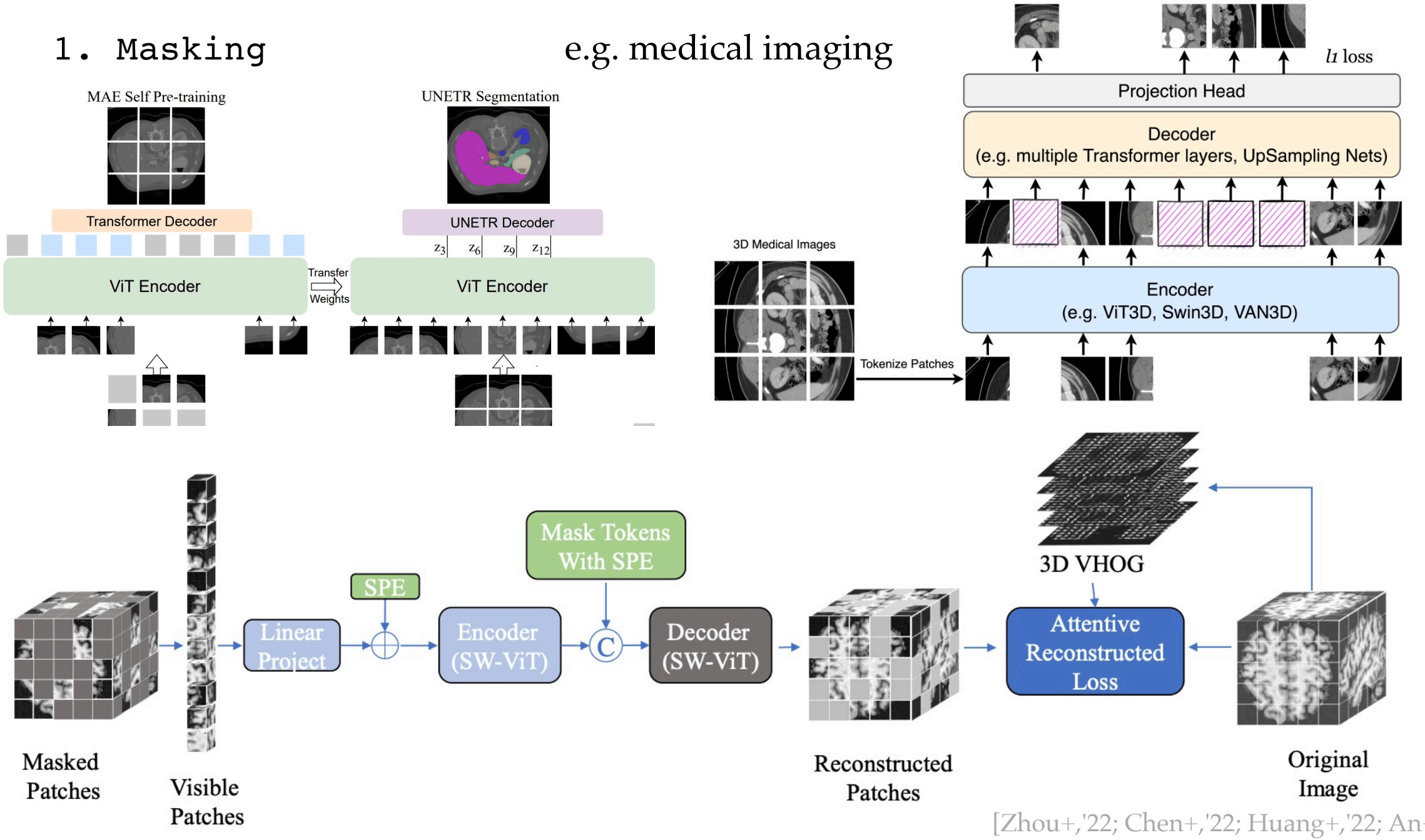
Outline

- Neural networks are *representation* learners
- Self-supervised learning
- Auto-encoders
- Word embeddings
- (Some recent representation learning ideas)
    - 1. masking: other applications
    - 2. contrastive: similarity
    - 3. multi-modality: alignment

[Section slides partially adapted from Kaiming He, Philip Isola, and Andrew Owens]

1. Masking

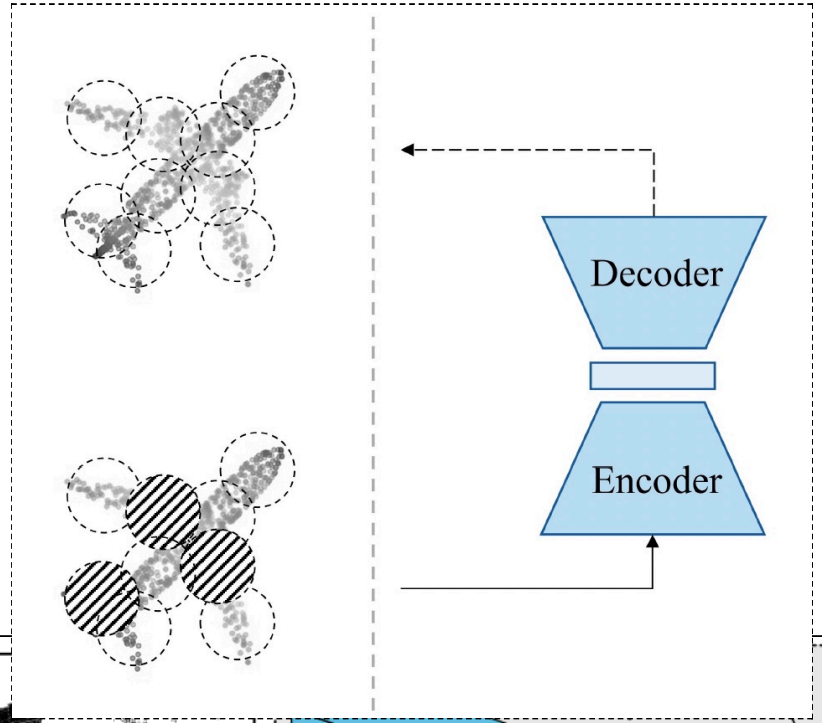
e.g. medical imaging

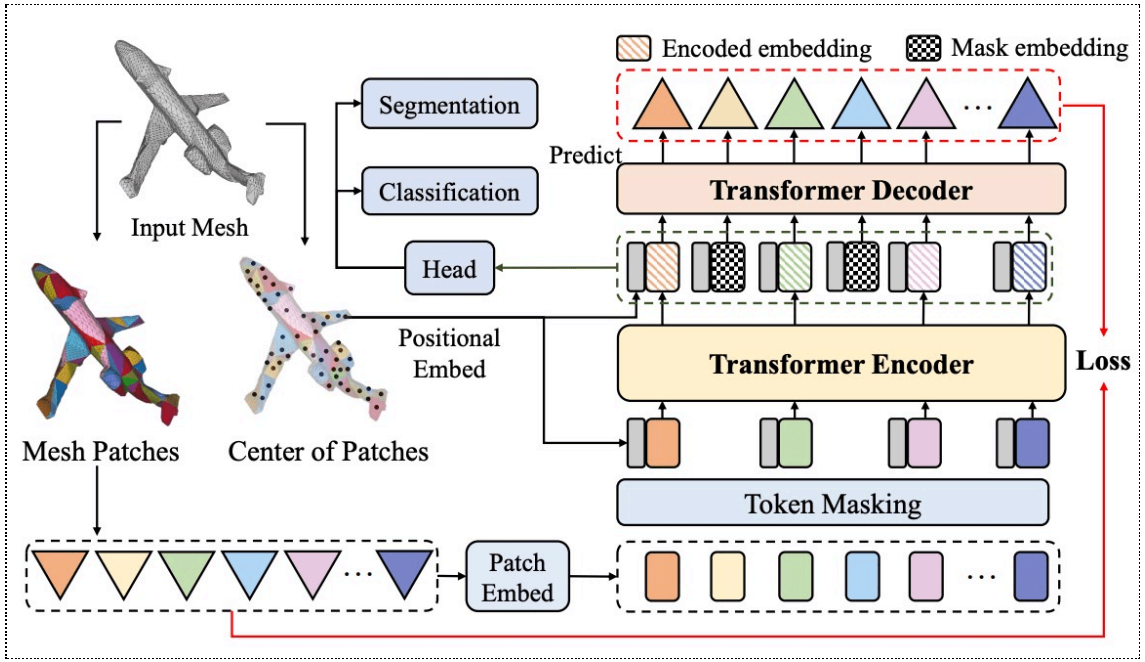


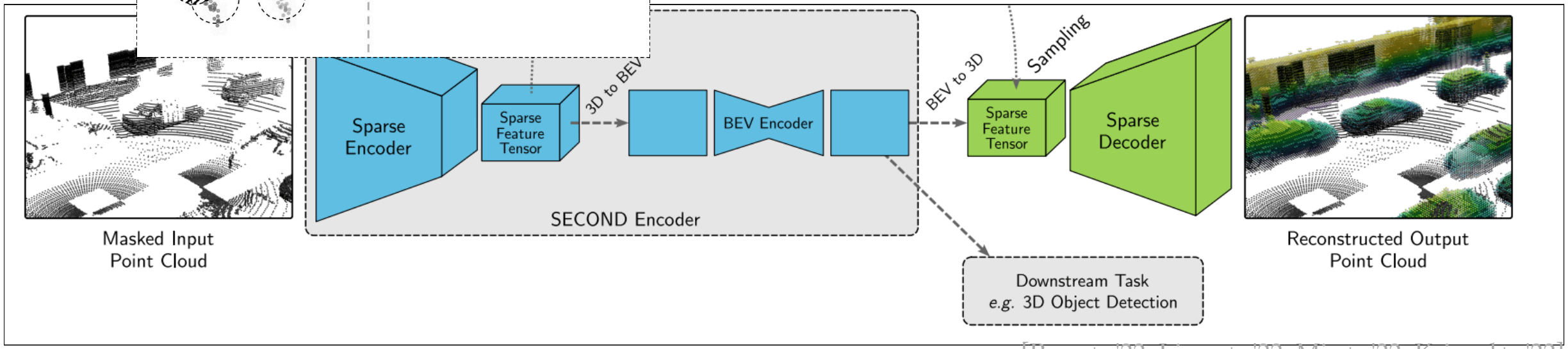
[Zhou+,'22; Chen+,'22; Huang+,'22; An+,'22]

1. Masking

e.g. 3d geometry

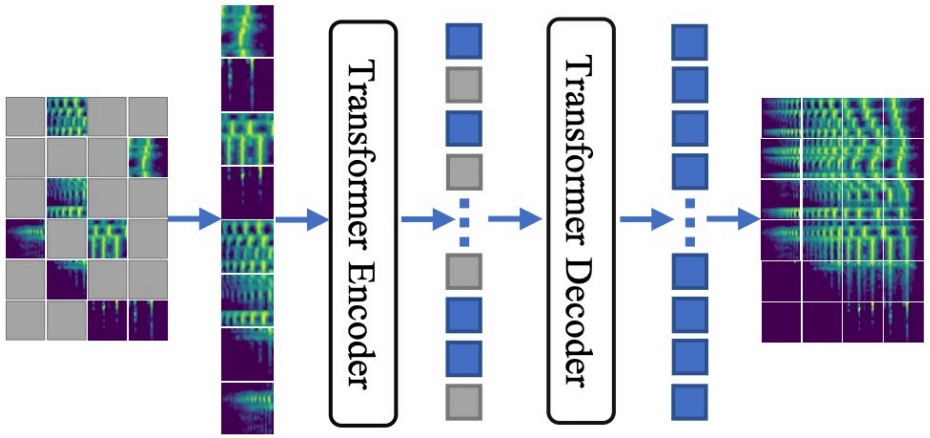




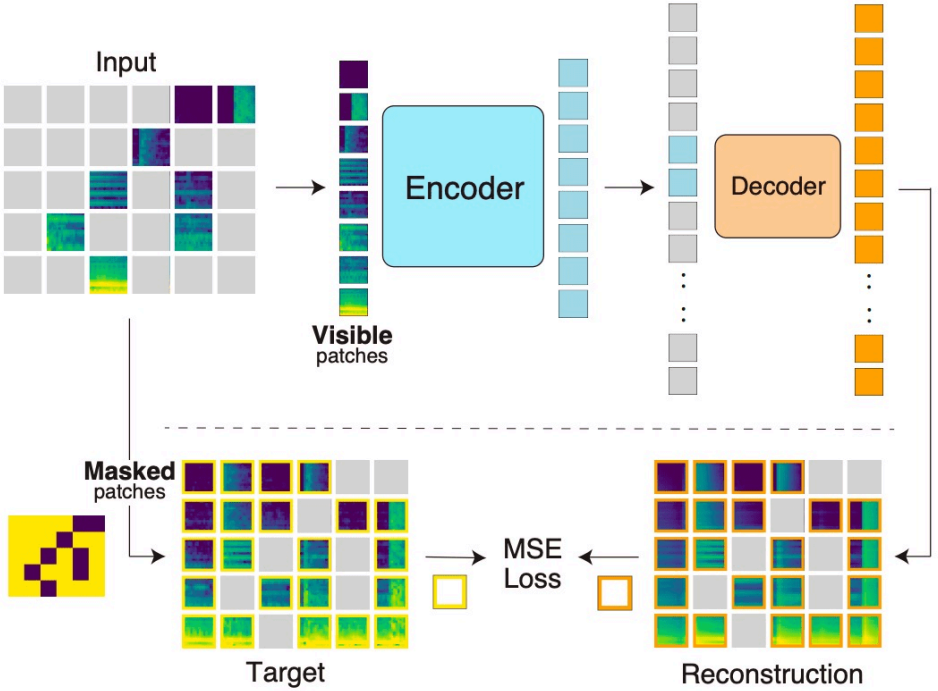


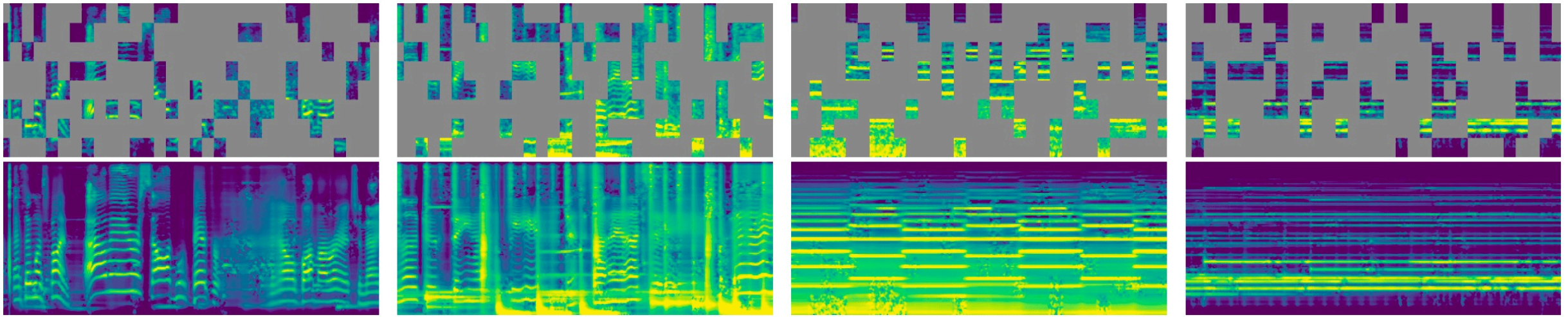
[Pang+, '22; Liang+, '22; Min+, '22; Krispel+, '22]

1. Masking



e.g. audio

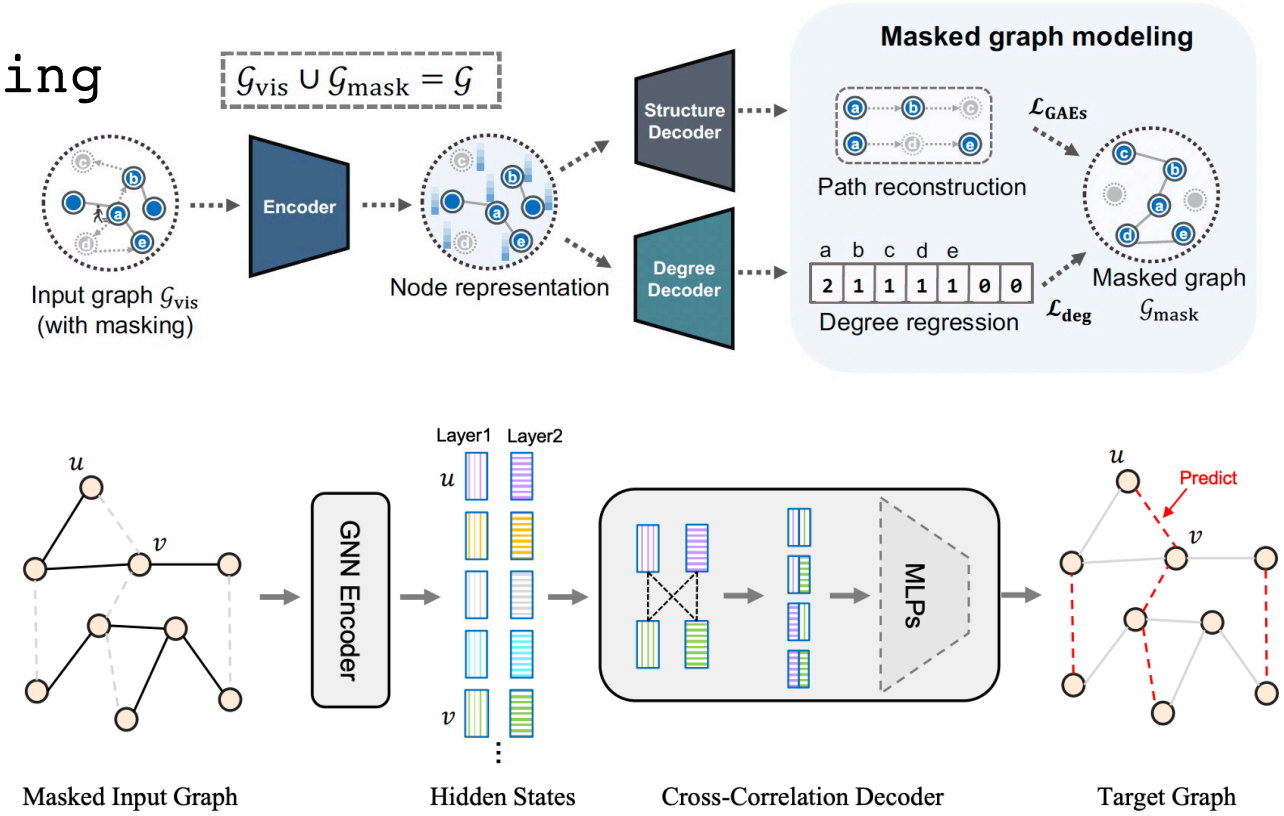


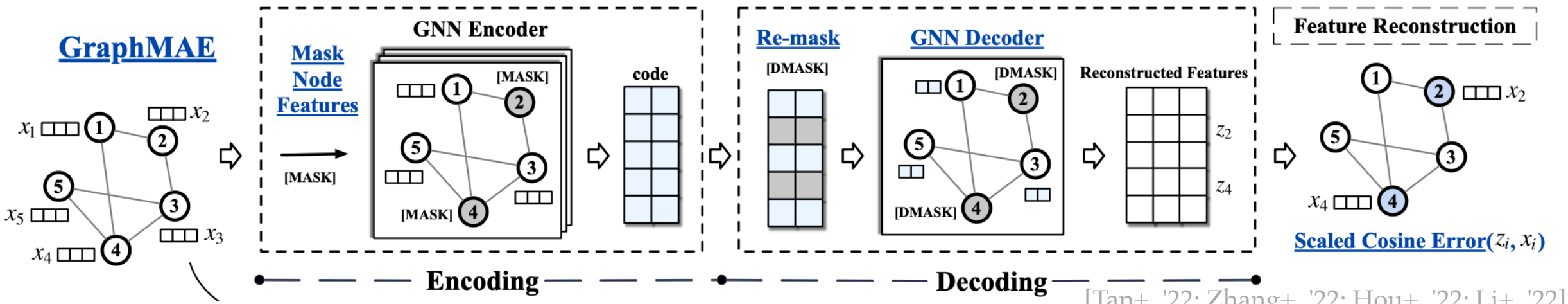


[Baade+, '22; Chong+, '22; Niizumi, '22; Huang+, '22]

1. Masking

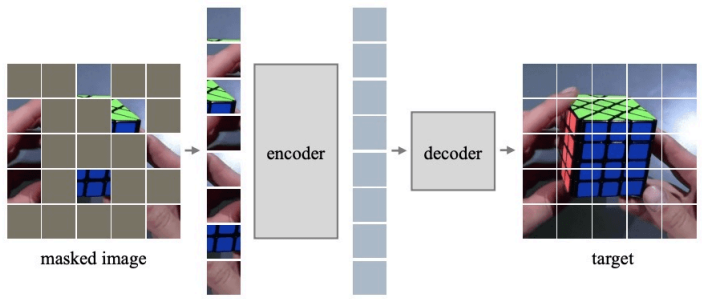
e.g. graphs



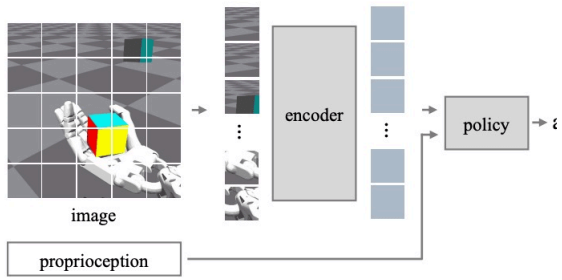


[Tan+, '22; Zhang+, '22; Hou+, '22; Li+, '22]

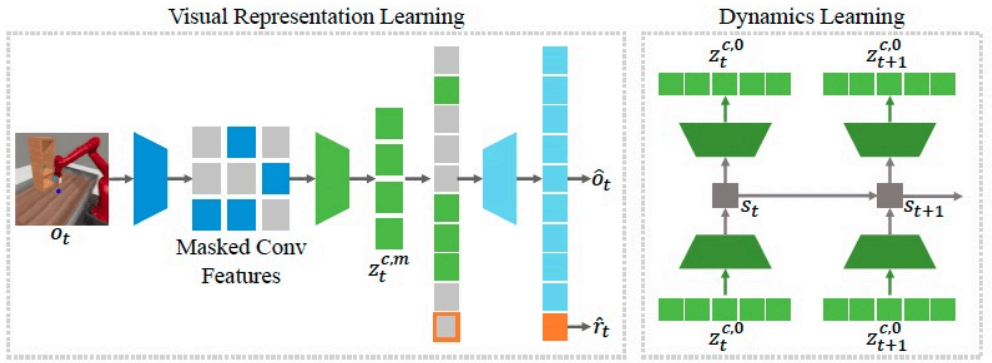
1. Masking e.g. robotics



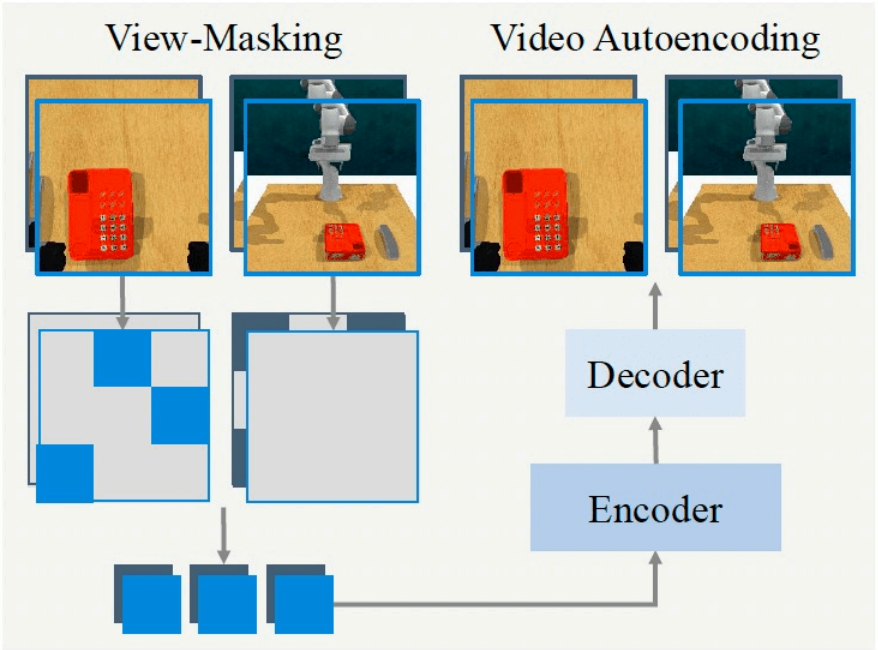
(a) masked visual pretraining



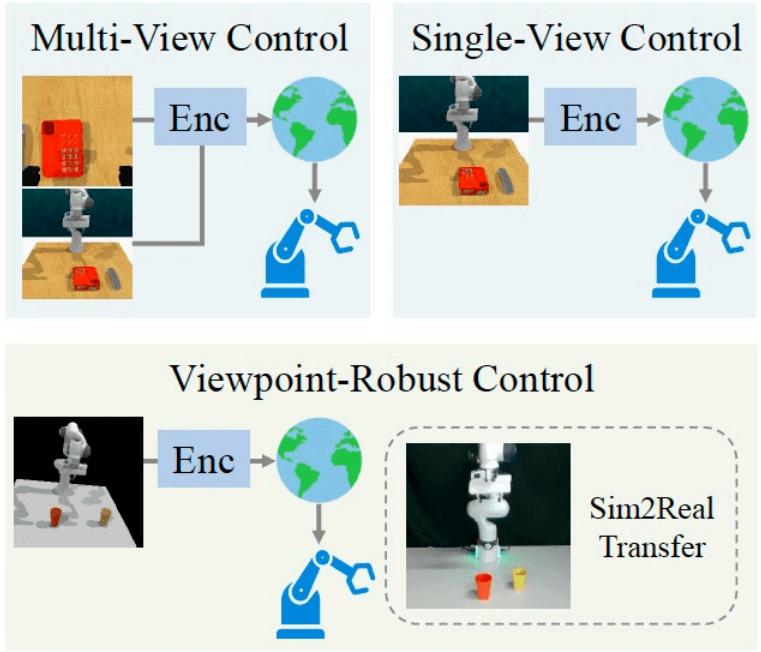
(b) learning motor control



Multi-View Masked Autoencoders

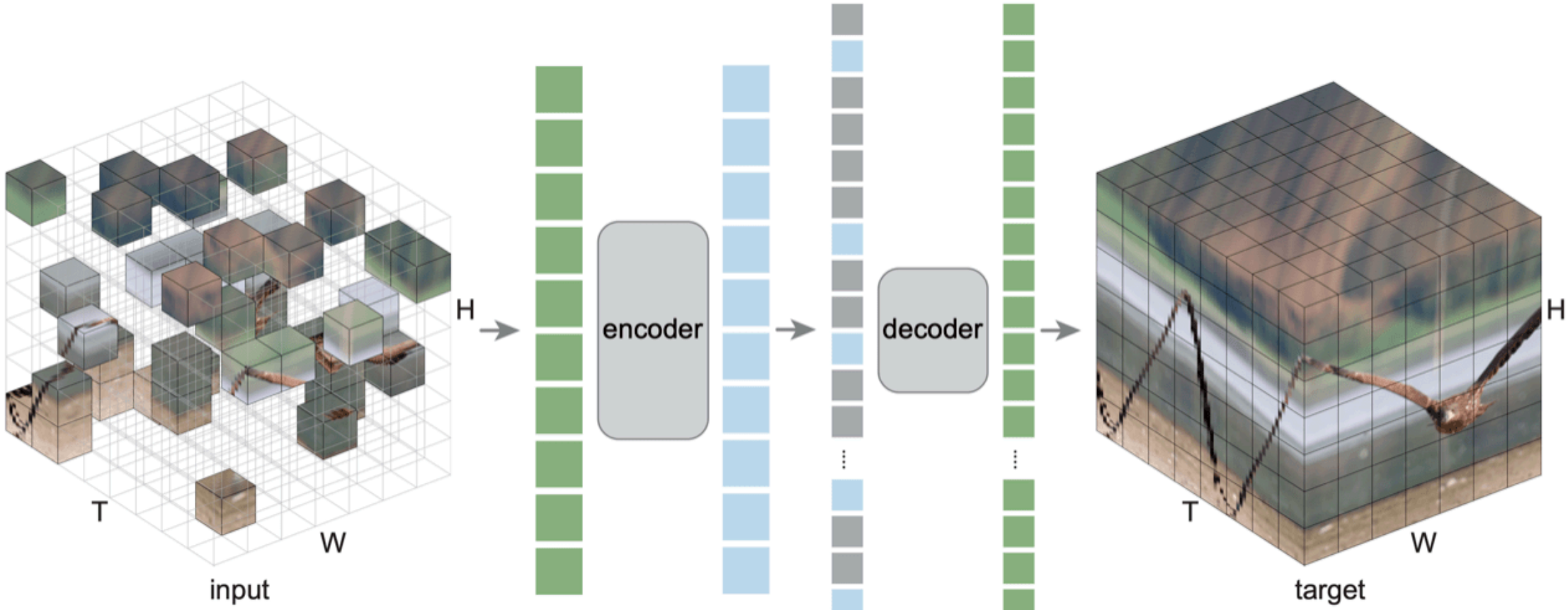


Visual Robotic Manipulation



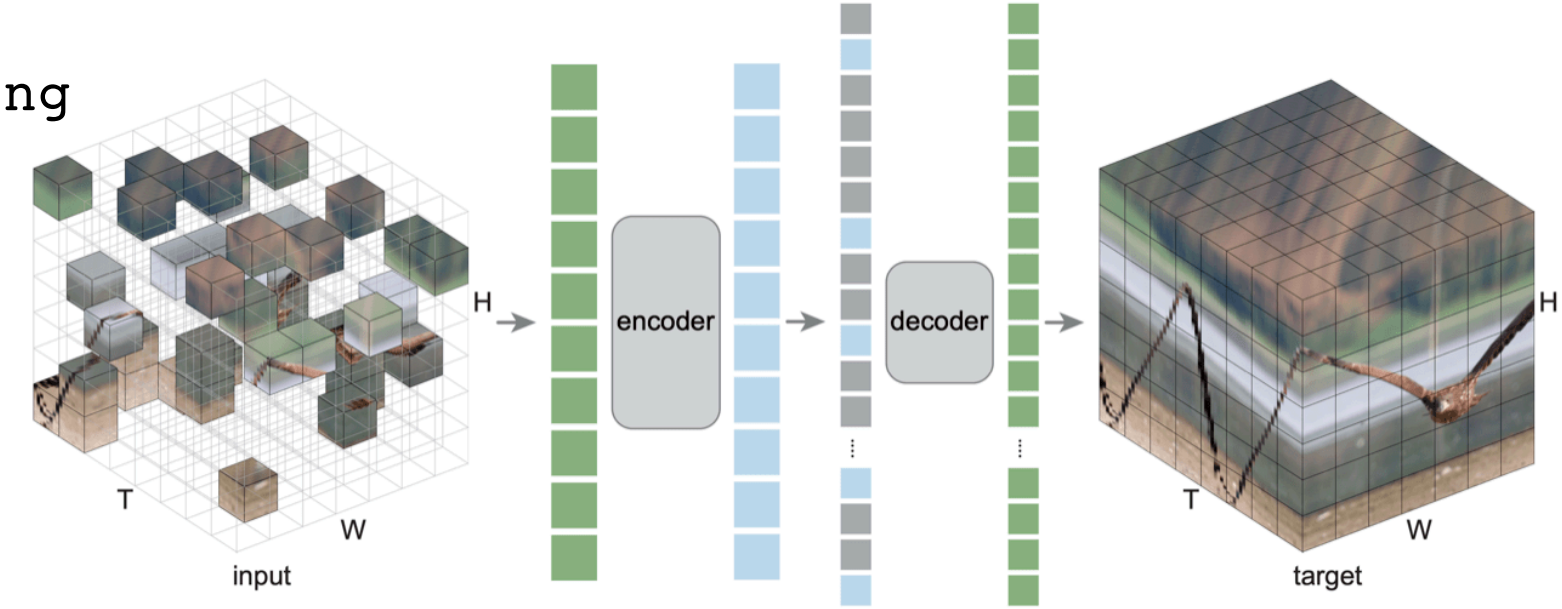
[Xiao+, '22; Radosavovic+, '22; Seo+, '22;]

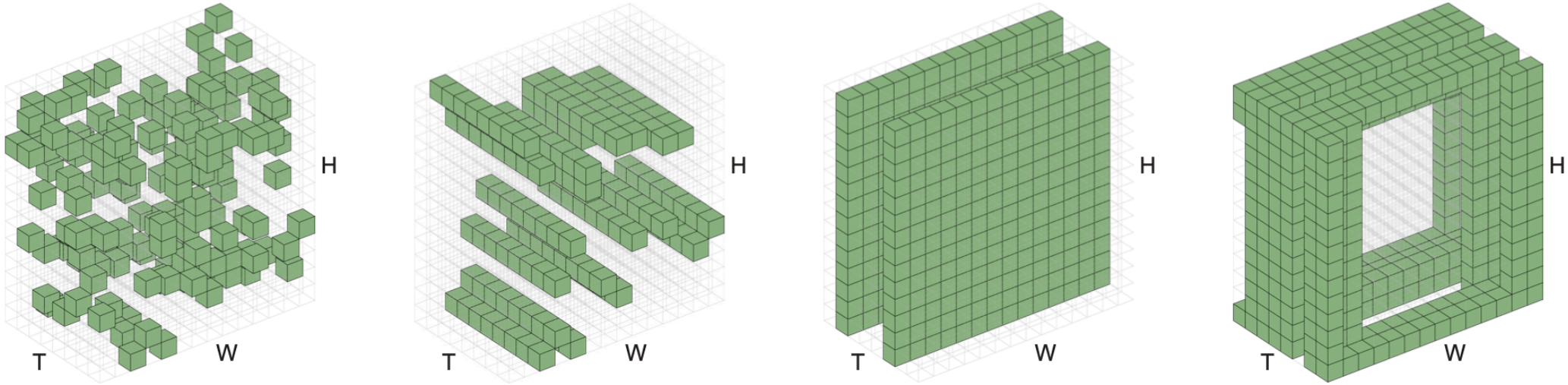
1. Masking



[Feichtenhofer, et al., "Masked Autoencoders As Spatiotemporal Learners", NeurIPS 2022]

1. Masking

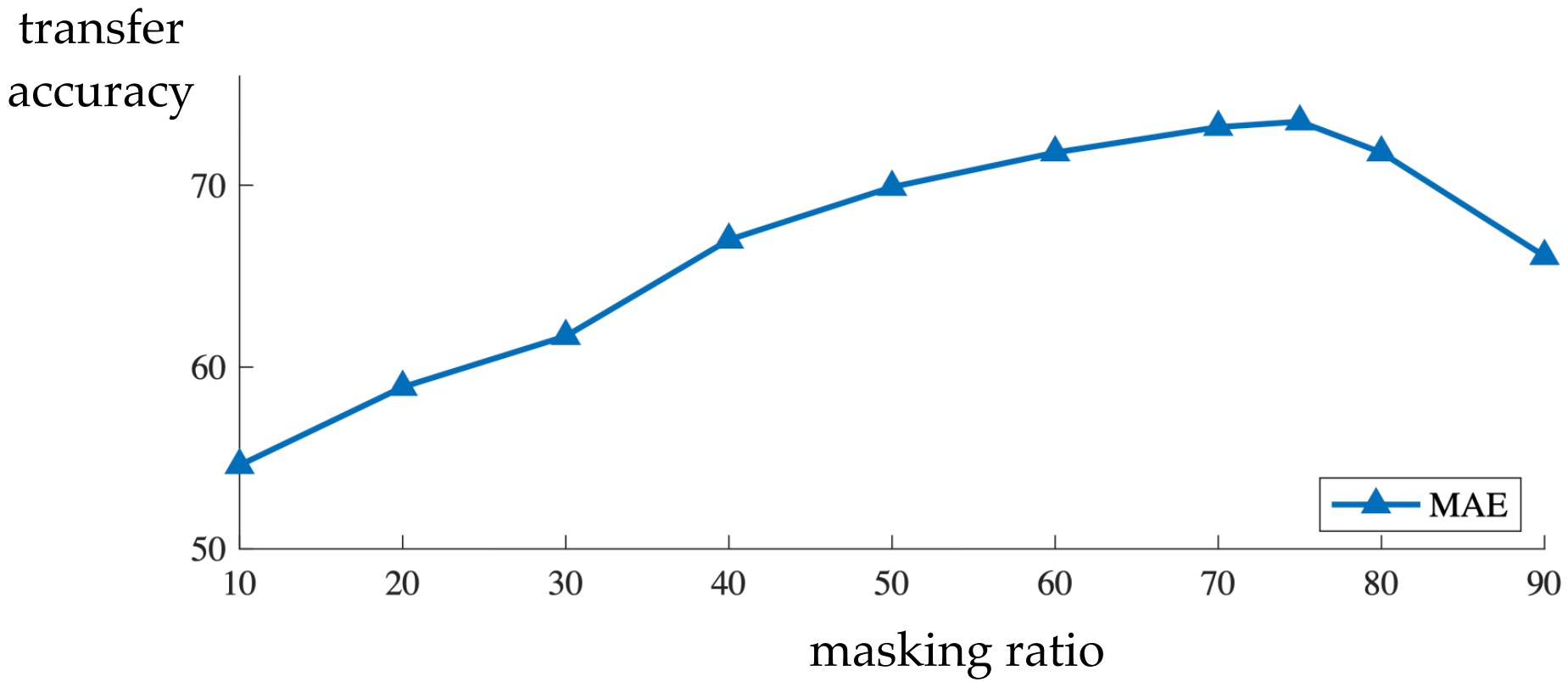




(a) agnostic, 90% (b) space-only, 90% (c) time-only, 75% (d) block, 75%

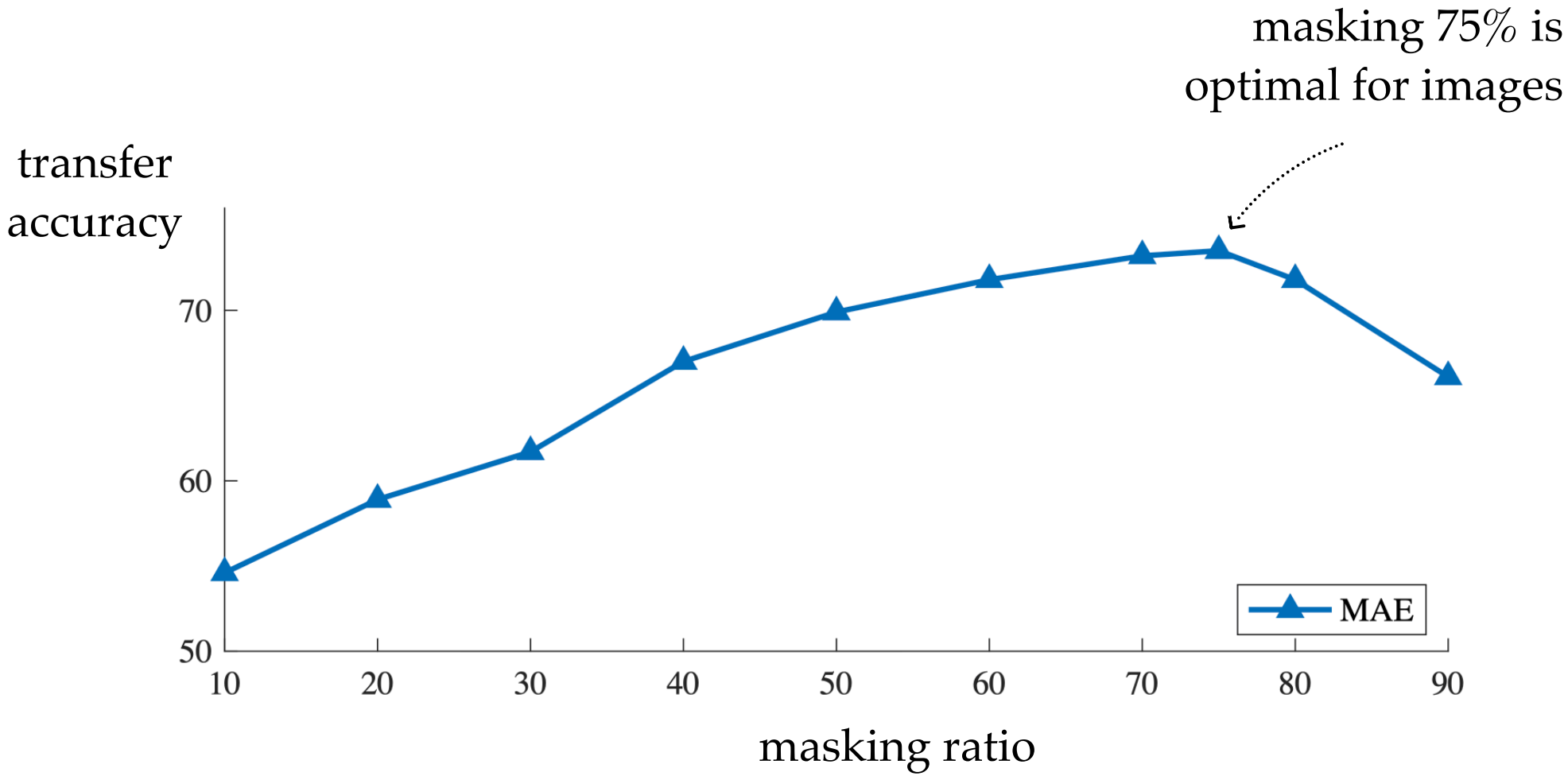
[Feichtenhofer, et al., "Masked Autoencoders As Spatiotemporal Learners", NeurIPS 2022]

1. Masking



[He, et al. Masked Autoencoders Are Scalable Vision Learners, 2021]

1. Masking

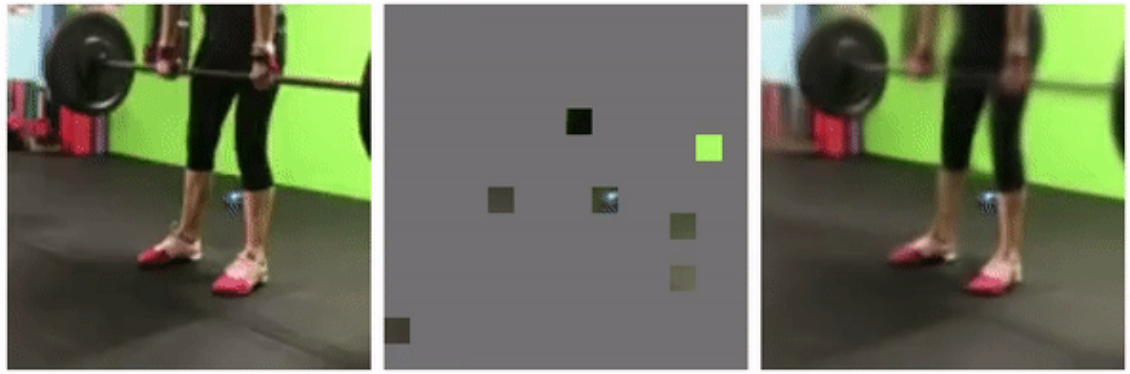


[He, et al. Masked Autoencoders Are Scalable Vision Learners, 2021]

1. Masking

1. Masking

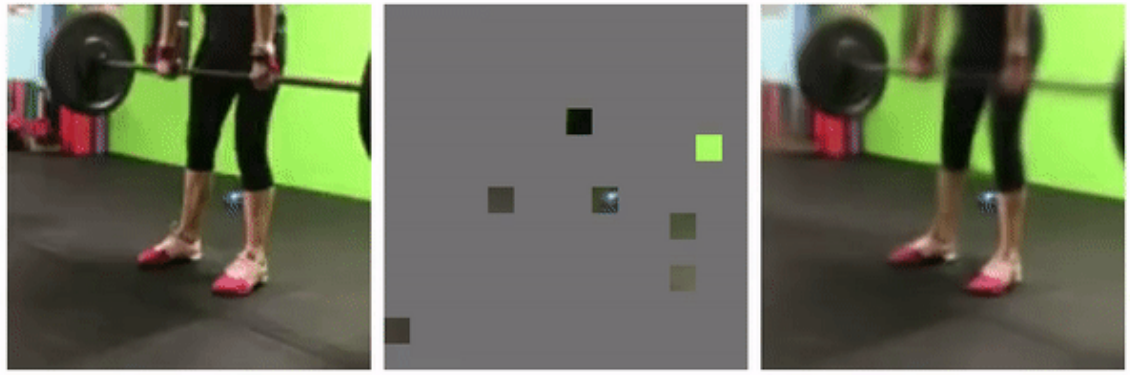
95% masked



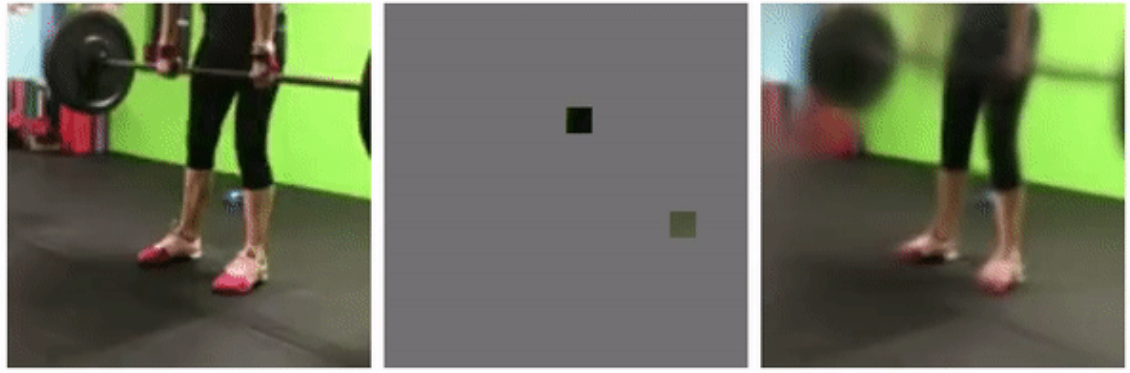
1. Masking

Similar empirical studies shows 15% as optimal for languages, and 95% for videos

95% masked

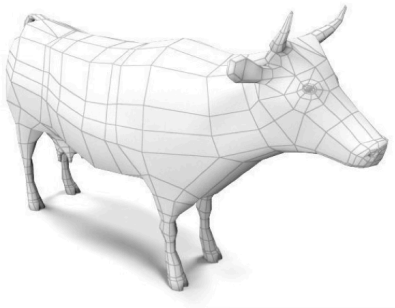


98% masked

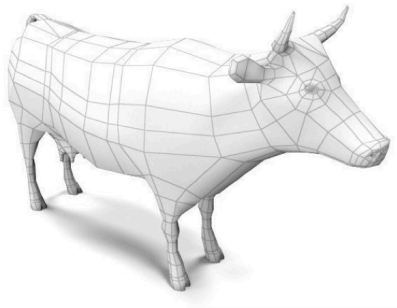


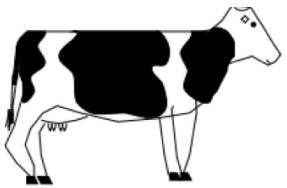
2. Contrastive learning

2. Contrastive learning

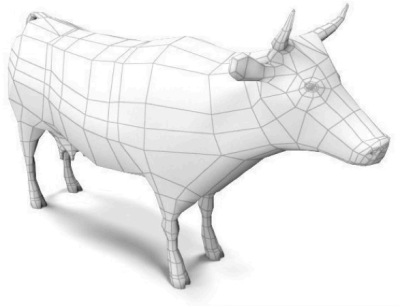


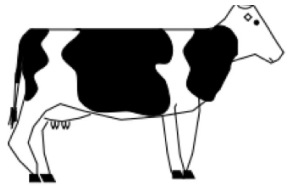
2. Contrastive learning

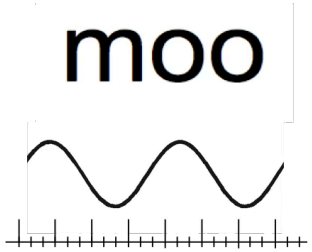




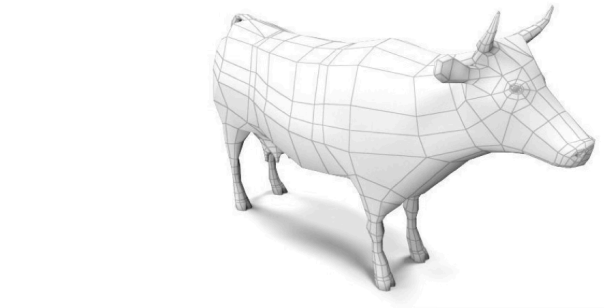
2. Contrastive learning

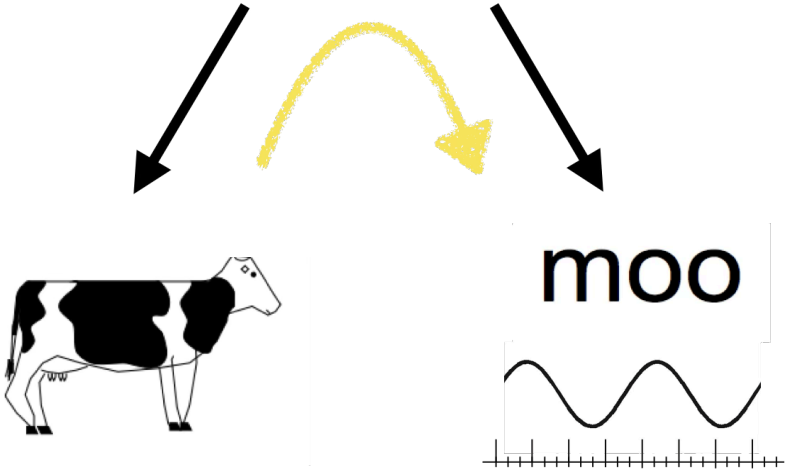




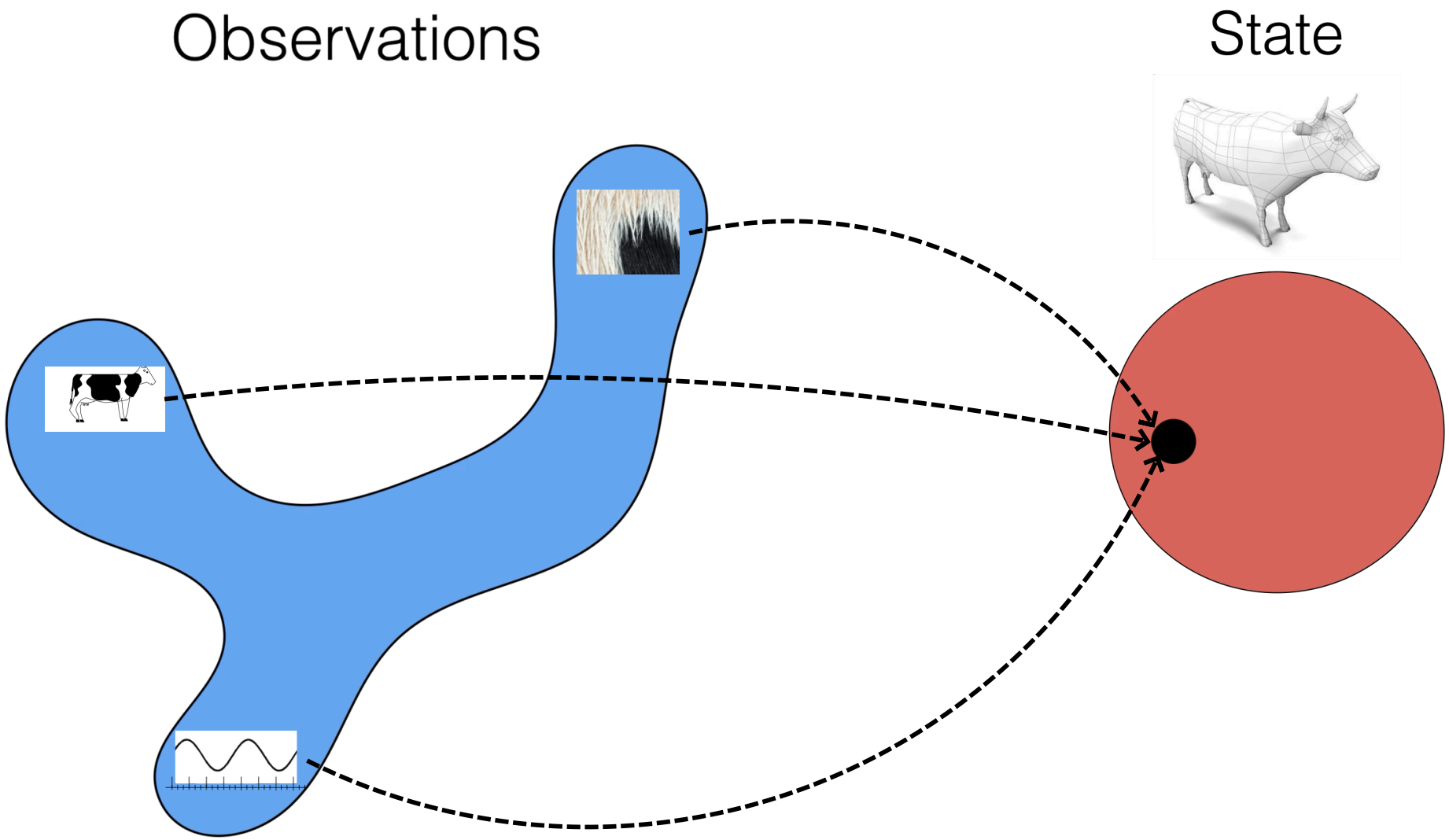


2. Contrastive learning



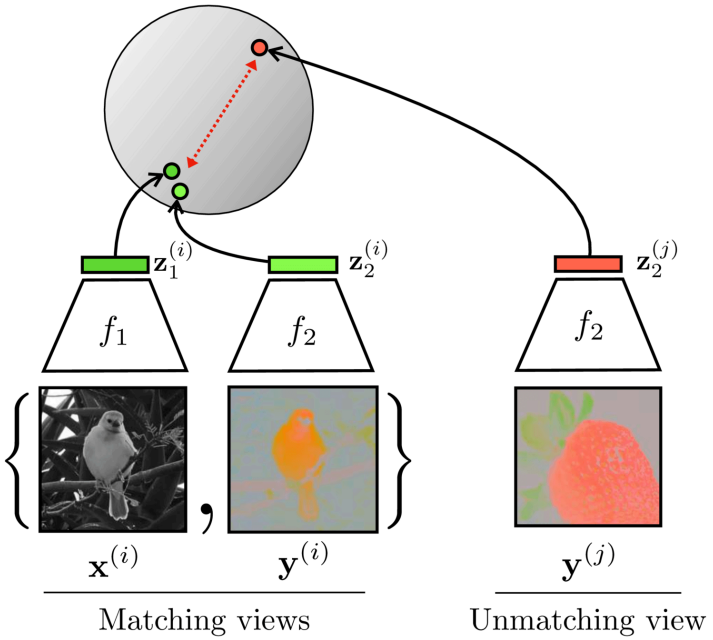


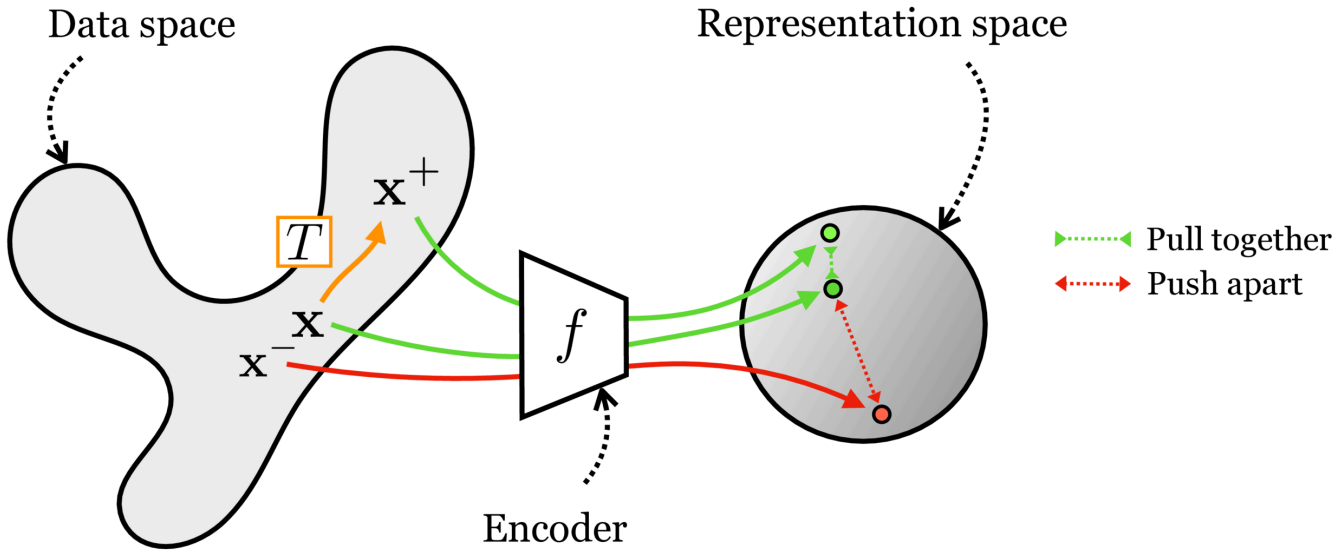
2. Contrastive learning



[images credit: visionbook.mit.edu]

2. Contrastive learning





[images credit: visionbook.mit.edu]

2. Contrastive learning

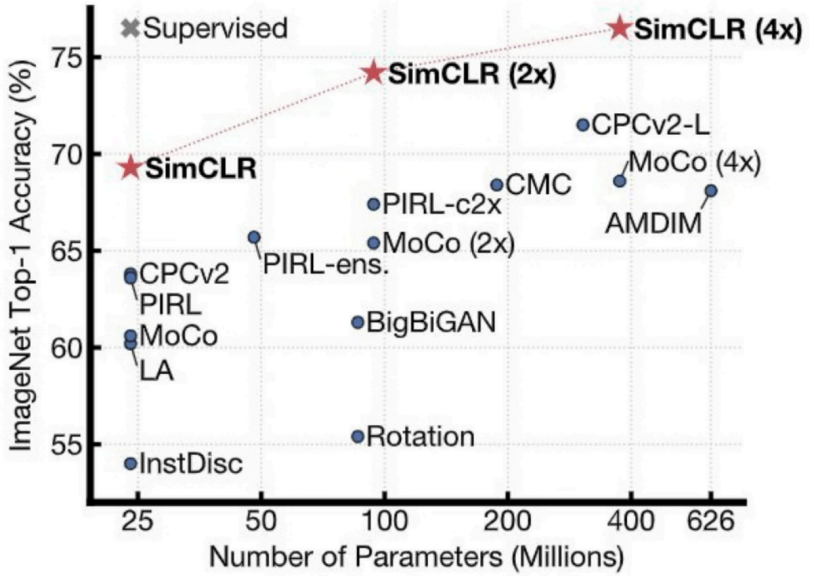
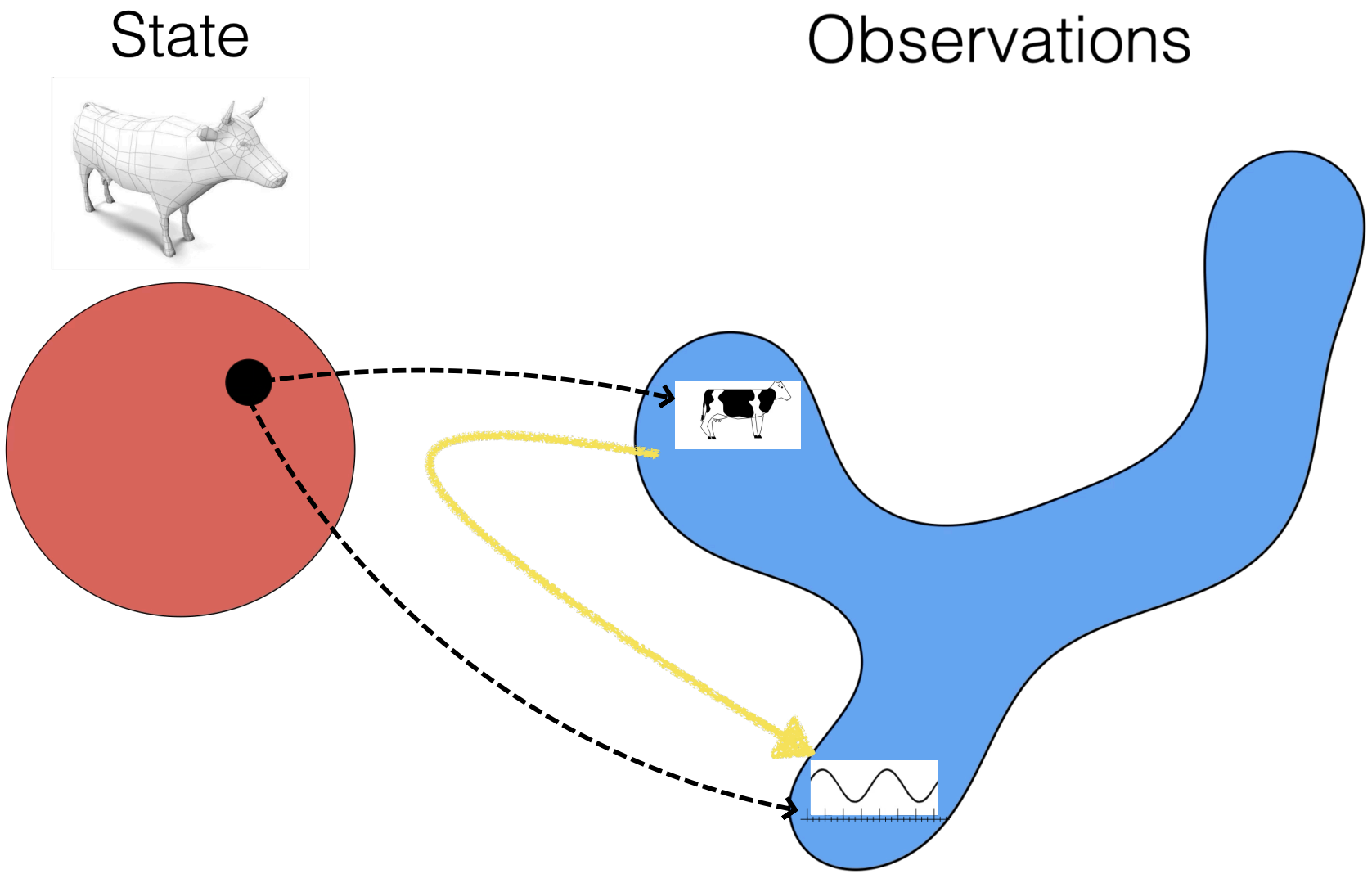


Figure 1. ImageNet top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Our method, SimCLR, is shown in bold.

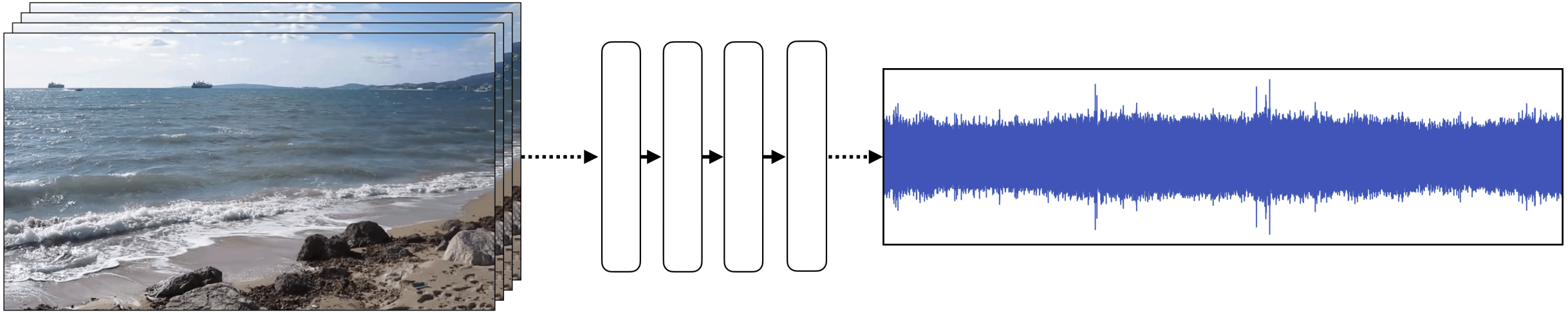
[Chen, Kornblith, Norouzi, Hinton, ICML 2020]

3. Multi-modality



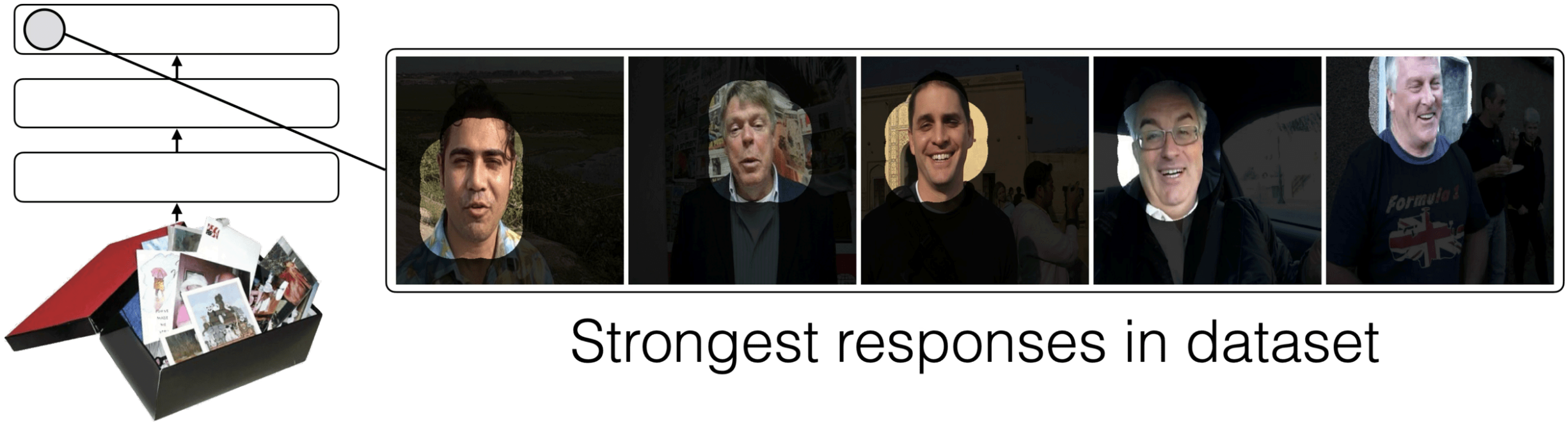
[images credit: visionbook.mit.edu]

e.g. video, audio, images

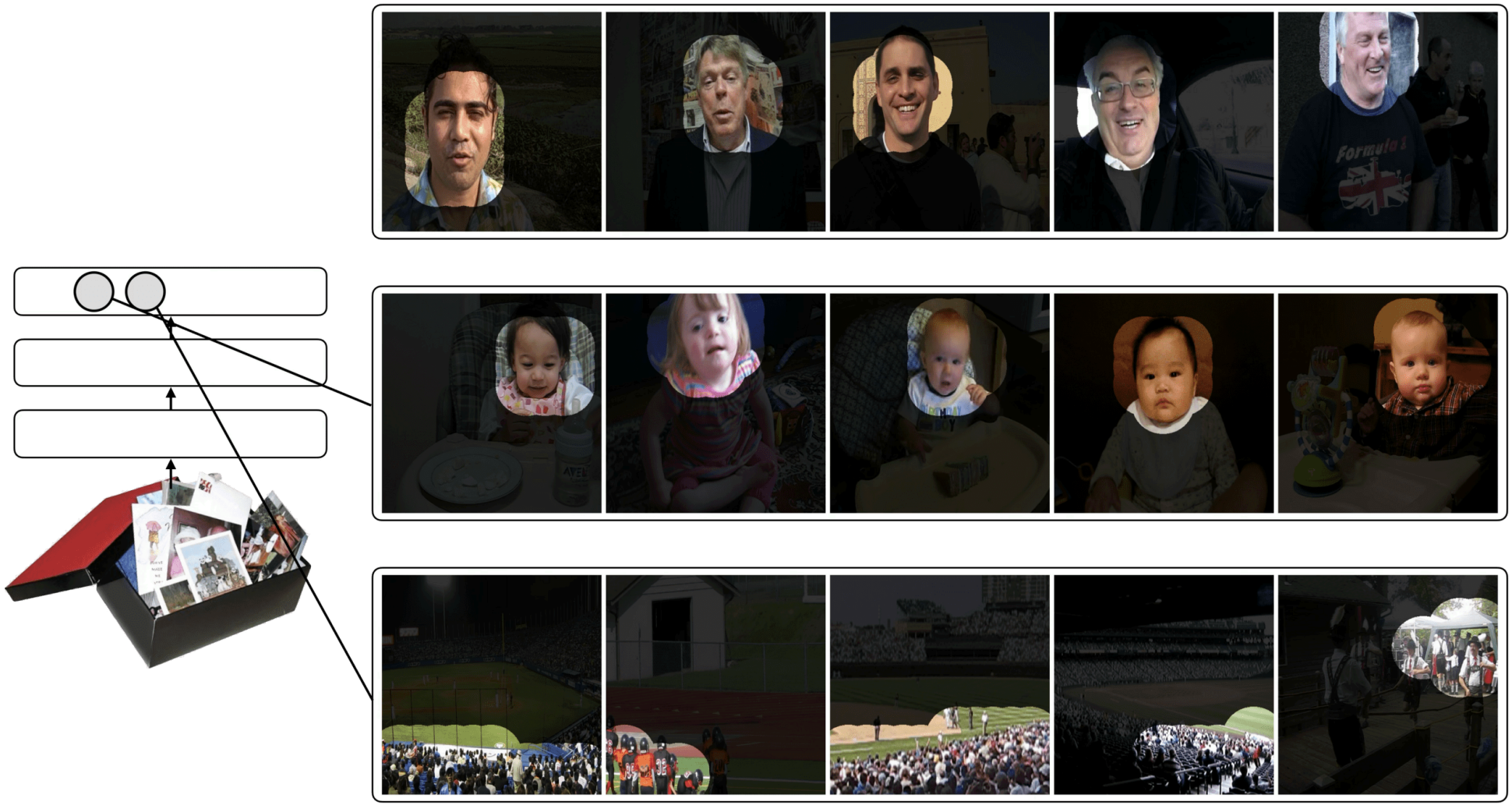


[Owens et al, Ambient Sound Provides Supervision for Visual Learning, ECCV 2016]

What did the model learn?

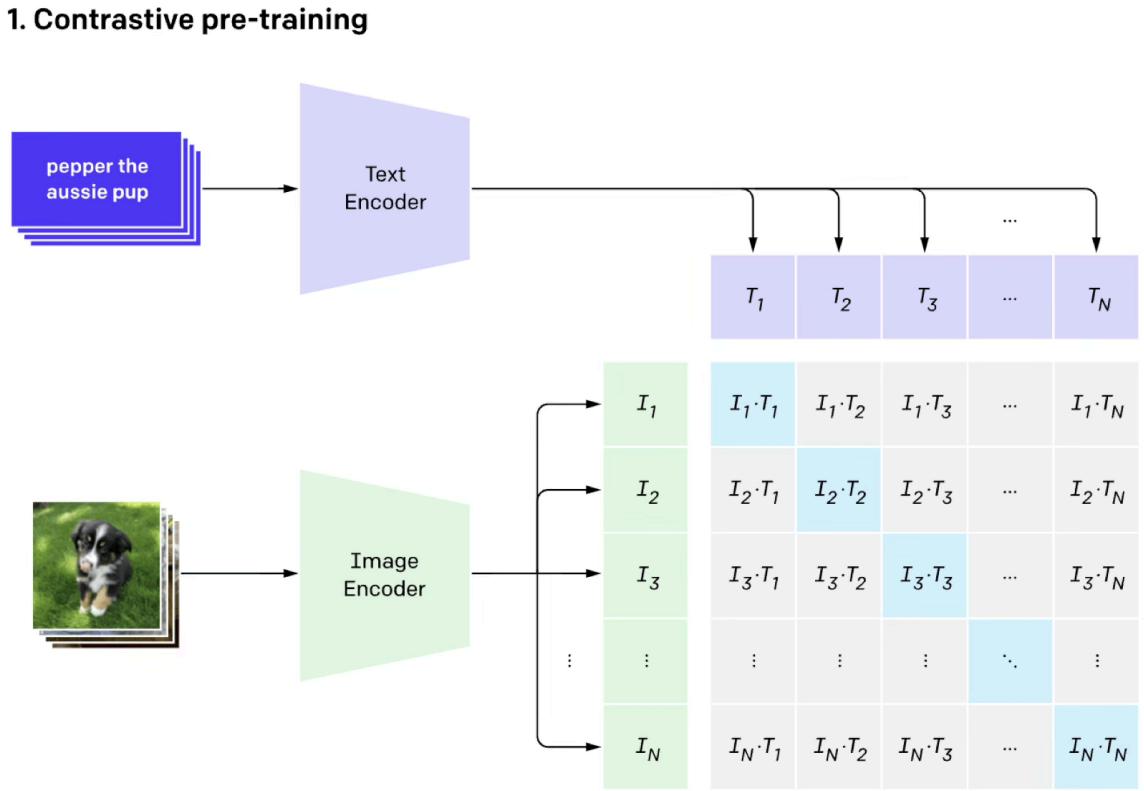


[Owens et al, Ambient Sound Provides Supervision for Visual Learning, ECCV 2016]



[Owens et al, Ambient Sound Provides Supervision for Visual Learning, ECCV 2016]

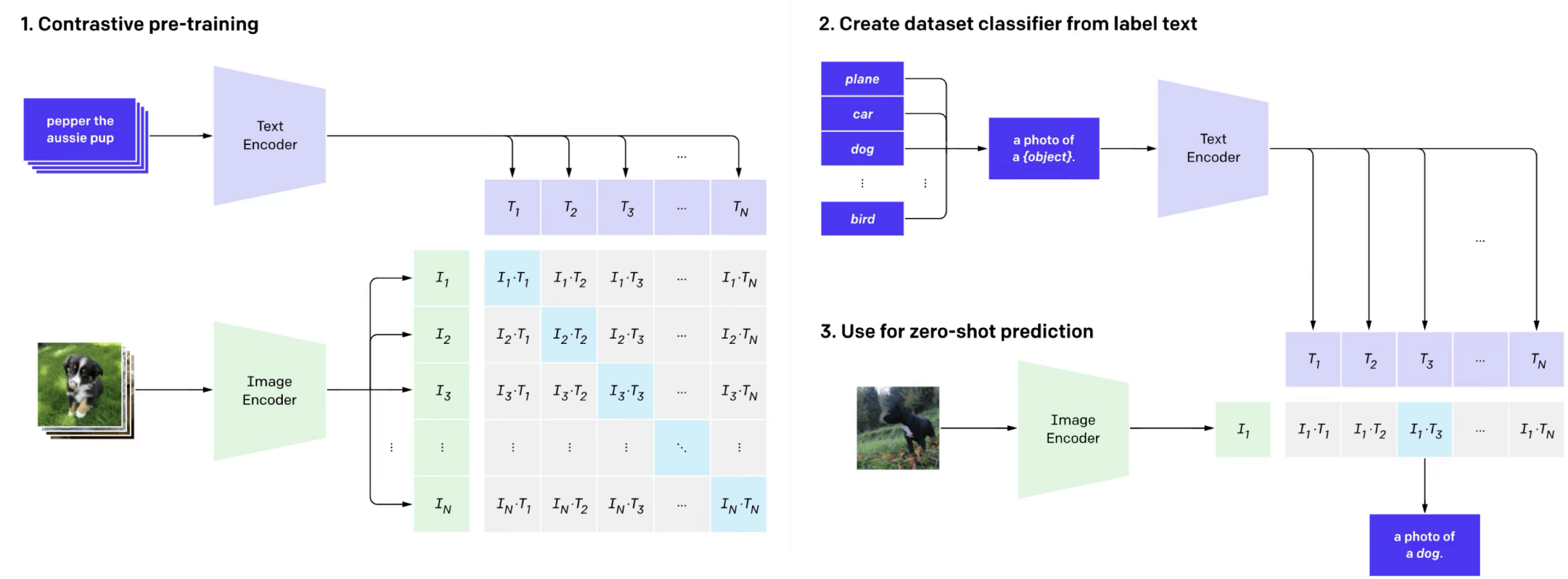
e.g. image classification (done in the contrastive way)




```
# 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
```

[Radford et al, Learning Transferable Visual Models From Natural Language Supervision, ICML, 2011]

e.g. image classification (done in the contrastive way)



[Radford et al, Learning Transferable Visual Models From Natural Language Supervision, ICML, 2011]

e.g Dall-E: text-image generation

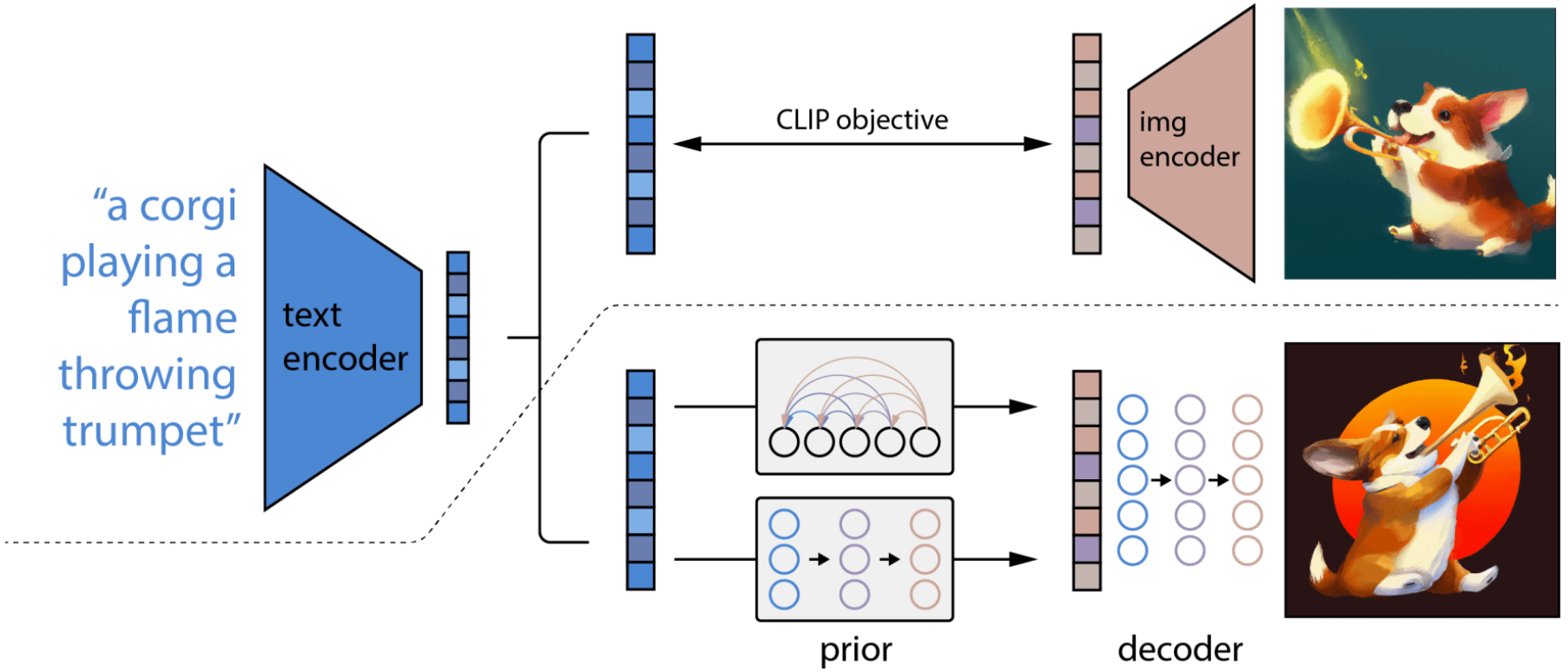


Figure 2: A high-level overview of unCLIP. Above the dotted line, we depict the CLIP training process, through which we learn a joint representation space for text and images. Below the dotted line, we depict our text-to-image generation process: a CLIP text embedding is first fed to an autoregressive 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 NN. Today we see they 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:
  - representations as compressed codes, auto-encoder with bottleneck
  - (representations that are predictive of their context)
  - (representations that are shared across sensory modalities)