Carnegie Mellon University
Heinz College

Introduction to Deep Learning

most of the slides here are by George Chen (CMU)

some slides are by Phillip Isola (OpenAI, UC Berkeley)

CMU 95-865 Fall 2017

IMAGENET

Over 10 million images, 1000 object classes

Russakovsky et al. ImageNet Large Scale Visual Recognition Challenge. IJCV 2015.

IMAGENET

Over 10 million images, 1000 object classes

2011: Traditional computer vision achieves accuracy ~74%

Russakovsky et al. ImageNet Large Scale Visual Recognition Challenge. IJCV 2015.

IMAGENET

Over 10 million images, 1000 object classes

2011: Traditional computer vision achieves accuracy ~74%

2012: Initial deep neural network approach accuracy ~84%

Russakovsky et al. ImageNet Large Scale Visual Recognition Challenge. IJCV 2015.

Over 10 million images, 1000 object classes

2011: Traditional computer vision achieves accuracy ~74%

2012: Initial deep neural network approach accuracy ~84%

2015 onwards: Deep learning achieves accuracy 96%+

Russakovsky et al. ImageNet Large Scale Visual Recognition Challenge. IJCV 2015.

Deep Learning Takeover

Deep Learning Takeover

Academia:

Deep Learning Takeover

Academia:

- Top computer vision conferences (CVPR, ICCV, ECCV) are now nearly all about deep learning

Deep Learning Takeover

Academia:

- Top computer vision conferences (CVPR, ICCV, ECCV) are now nearly all about deep learning
- Top machine learning conferences (ICML, NIPS) have *heavily* been taken over by deep learning

Deep Learning Takeover

Academia:

- Top computer vision conferences (CVPR, ICCV, ECCV) are now nearly all about deep learning
- Top machine learning conferences (ICML, NIPS) have *heavily* been taken over by deep learning

Heavily dominated by industry now!

Google
facebook
amazon

Deep Learning Takeover

Academia:

- Top computer vision conferences (CVPR, ICCV, ECCV) are now nearly all about deep learning
- Top machine learning conferences (ICML, NIPS) have *heavily* been taken over by deep learning

Heavily dominated by industry now!

Extremely useful in practice:

Google
facebook
amazon

Deep Learning Takeover

Academia:

- Top computer vision conferences (CVPR, ICCV, ECCV) are now nearly all about deep learning
- Top machine learning conferences (ICML, NIPS) have *heavily* been taken over by deep learning

Heavily dominated by industry now!

Extremely useful in practice:

- Near human level image classification (including handwritten digit recognition)

Google
facebook
amazon

Deep Learning Takeover

Academia:

- Top computer vision conferences (CVPR, ICCV, ECCV) are now nearly all about deep learning
- Top machine learning conferences (ICML, NIPS) have *heavily* been taken over by deep learning

Heavily dominated by industry now!

Extremely useful in practice:

- Near human level image classification (including handwritten digit recognition)
- Near human level speech recognition

Deep Learning Takeover

Academia:

- Top computer vision conferences (CVPR, ICCV, ECCV) are now nearly all about deep learning
- Top machine learning conferences (ICML, NIPS) have *heavily* been taken over by deep learning

Heavily dominated by industry now!

Extremely useful in practice:

- Near human level image classification (including handwritten digit recognition)
- Near human level speech recognition
- Improvements in machine translation, text-to-speech

Deep Learning Takeover

Academia:

- Top computer vision conferences (CVPR, ICCV, ECCV) are now nearly all about deep learning
- Top machine learning conferences (ICML, NIPS) have *heavily* been taken over by deep learning

Heavily dominated by industry now!

Extremely useful in practice:

- Near human level image classification (including handwritten digit recognition)
- Near human level speech recognition
- Improvements in machine translation, text-to-speech
- Self-driving cars

Deep Learning Takeover

Academia:

- Top computer vision conferences (CVPR, ICCV, ECCV) are now nearly all about deep learning
- Top machine learning conferences (ICML, NIPS) have *heavily* been taken over by deep learning

Heavily dominated by industry now!

Extremely useful in practice:

- Near human level image classification (including handwritten digit recognition)
- Near human level speech recognition
- Improvements in machine translation, text-to-speech
- Self-driving cars
- *Better* than humans at playing Go

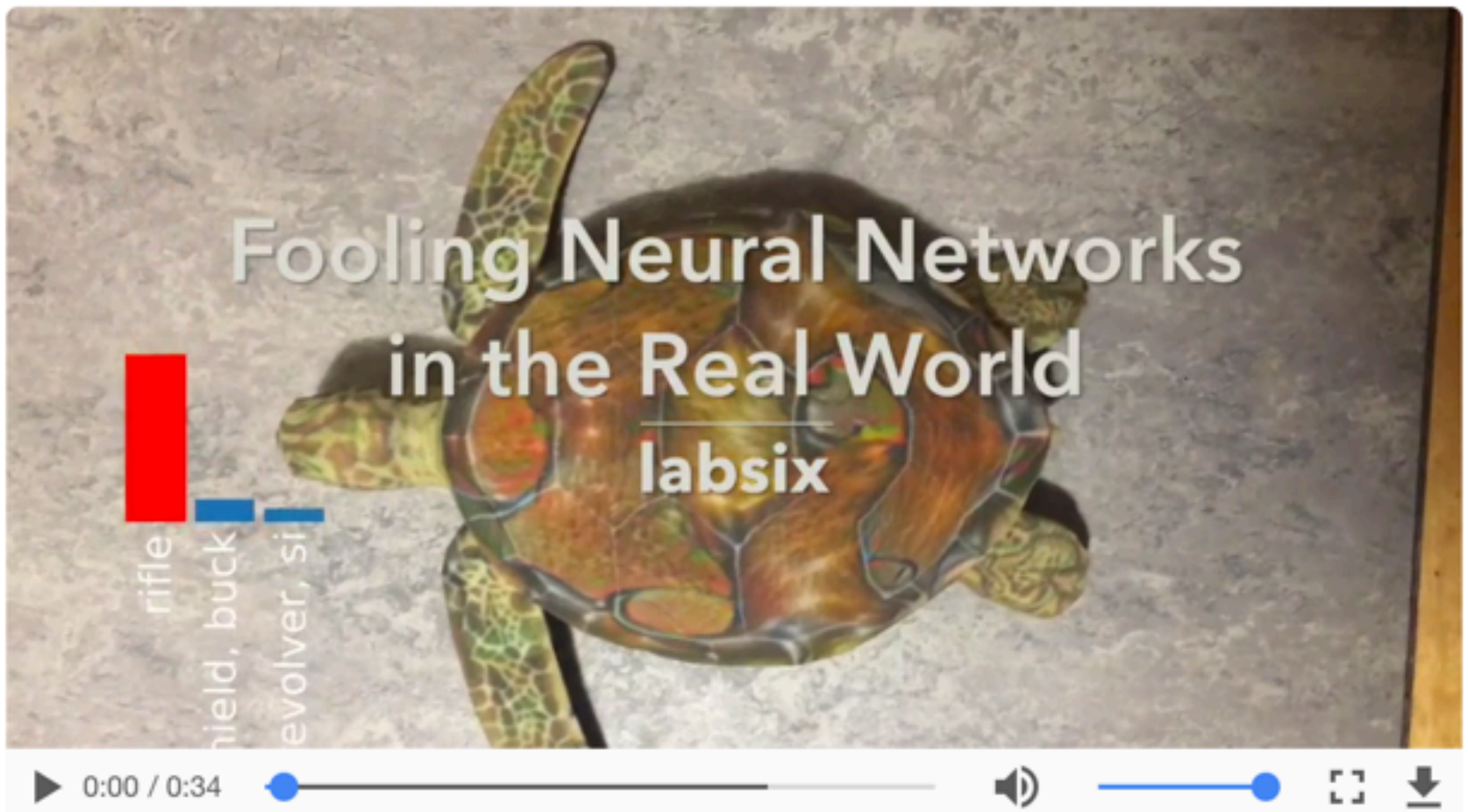
Google DeepMind’s AlphaGo vs Lee Sedol, 2016

Is it all hype?

Fooling Neural Networks in the Physical World with 3D Adversarial Objects

31 Oct 2017 · 3 min read — shared on Hacker News, Lobsters, Reddit, Twitter

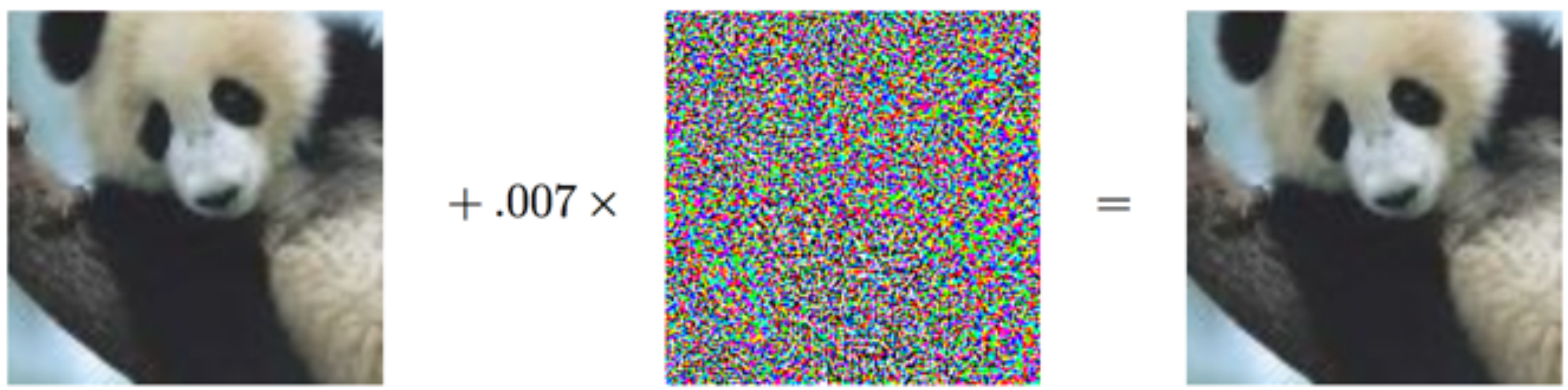
We've developed an approach to generate *3D adversarial objects* that reliably fool neural networks in the real world, no matter how the objects are looked at.



Neural network based classifiers reach near-human performance in many tasks, and they're used in high risk, real world systems. Yet, these same neural networks are particularly vulnerable to *adversarial examples*, carefully perturbed inputs that cause

Source: labsix

Source: Gizmodo article "This Neural Network's Hilariously Bad Image Descriptions Are Still Advanced AI". September 16, 2015. (They're using the NeuralTalk image-to-caption software.)



panda
~58% confidence

adversarial
noise

gibbon
~99% confidence

Source: Goodfellow, Shlens, and Szegedy. Explaining and Harnessing Adversarial Examples. ICLR 2015.

Another AI Winter?

Another AI Winter?

~1970’s: First AI winter over symbolic AI

Another AI Winter?

~1970's: First AI winter over symbolic AI

~1980's: Second AI winter over "expert systems"

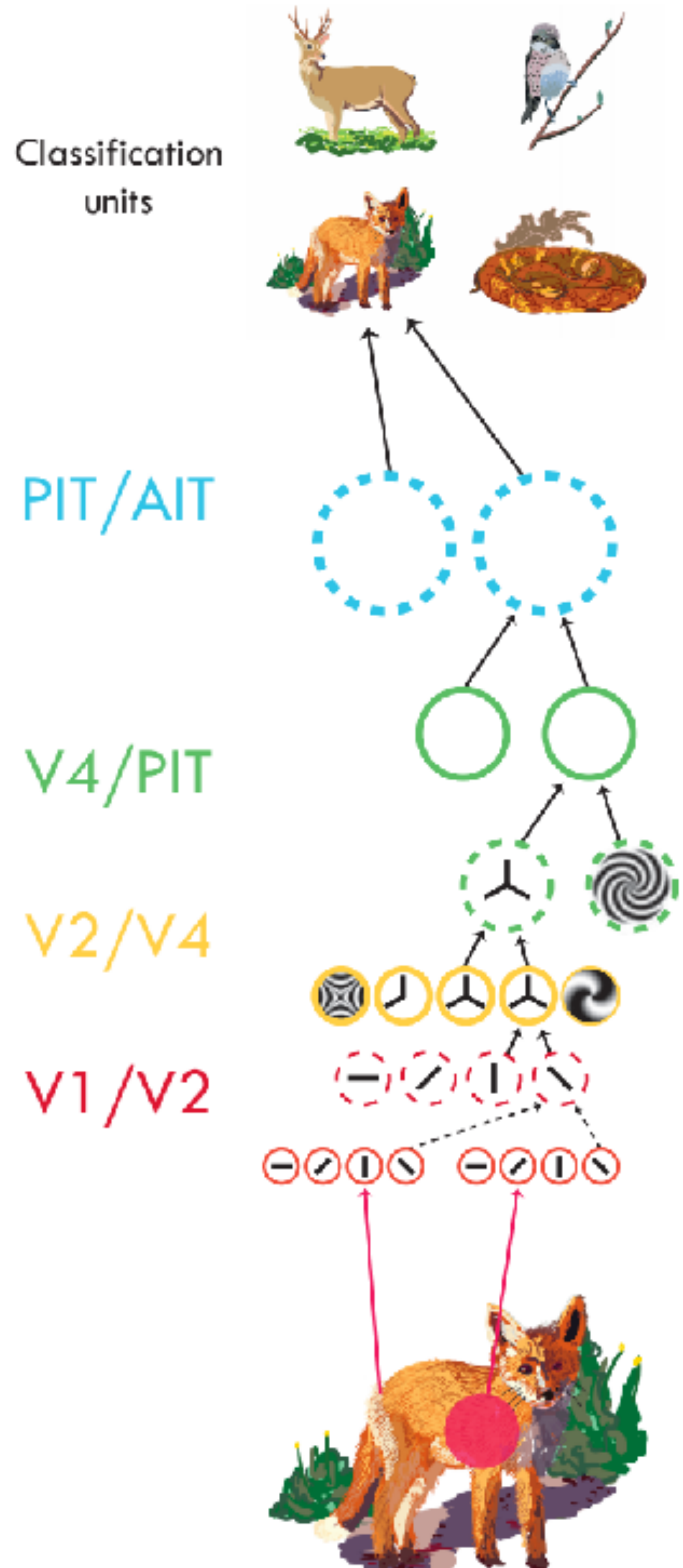
Another AI Winter?

~1970’s: First AI winter over symbolic AI

~1980’s: Second AI winter over “expert systems”

Every time: Lots of hype, explosion in funding, then bubble bursts

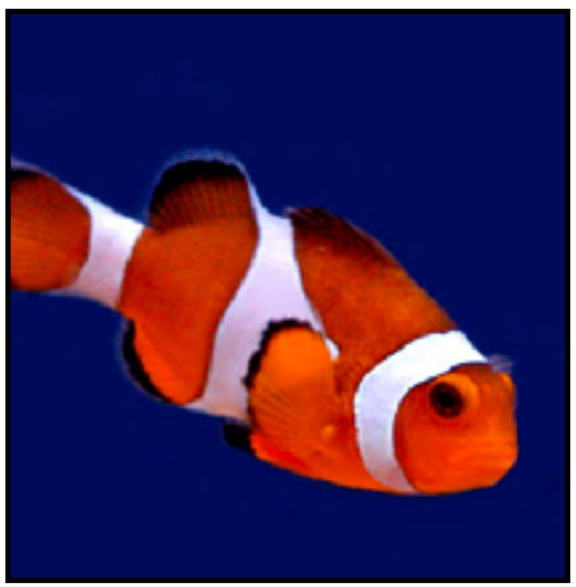
What is deep learning?



Slide by Phillip Isola

Serre, 2014

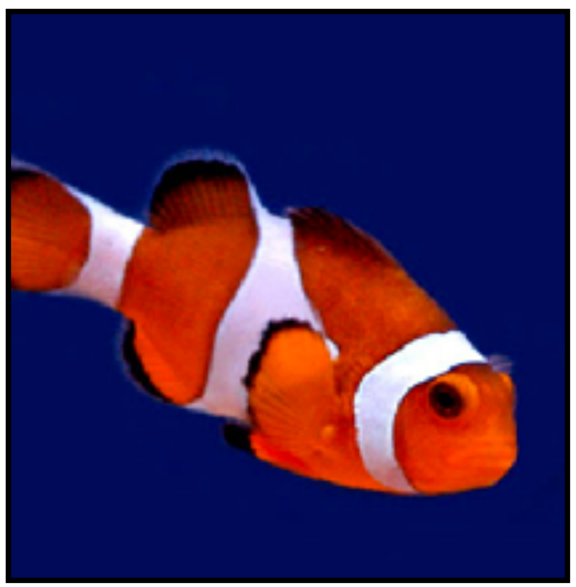
Basic Idea



→ Brain/Machine → "clown fish"

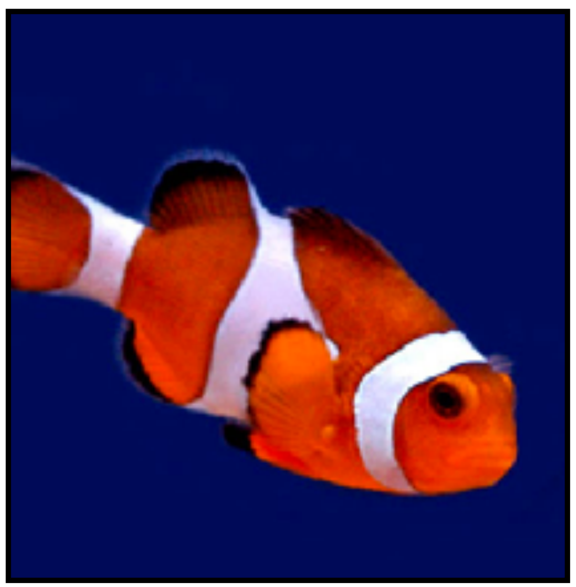
Slide by Phillip Isola

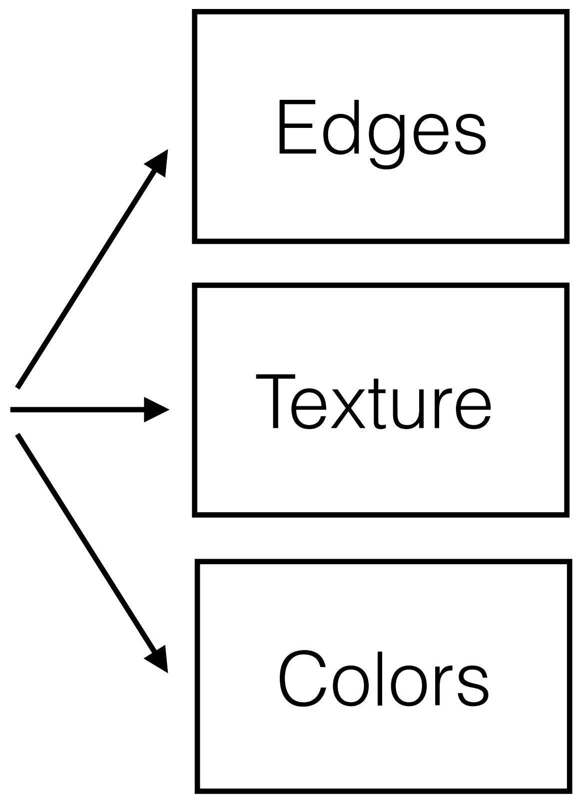
Object Recognition



Slide by Phillip Isola

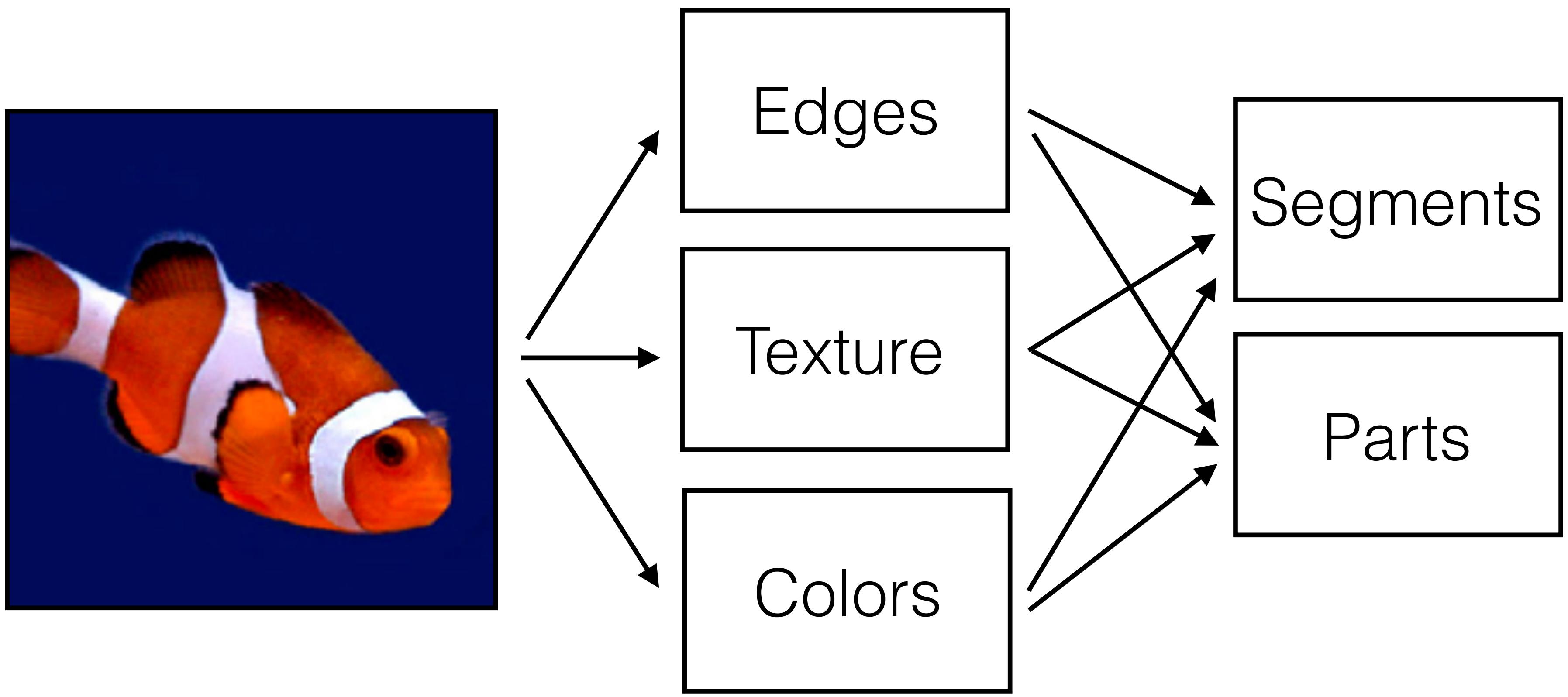
Object Recognition





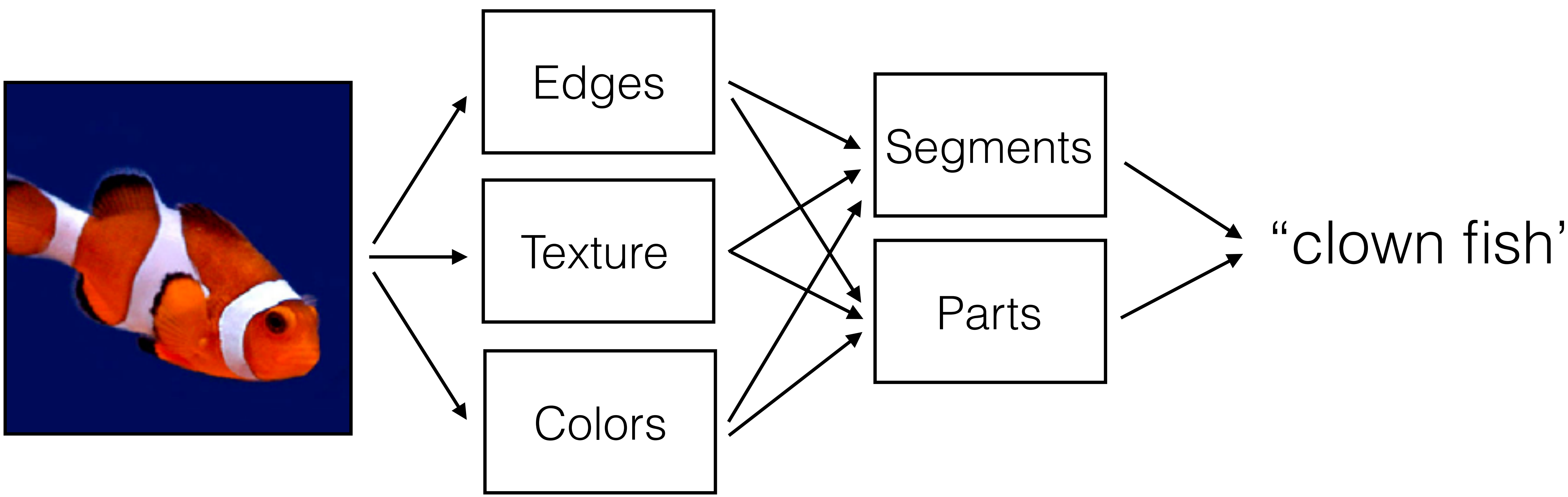
Slide by Phillip Isola

Object Recognition



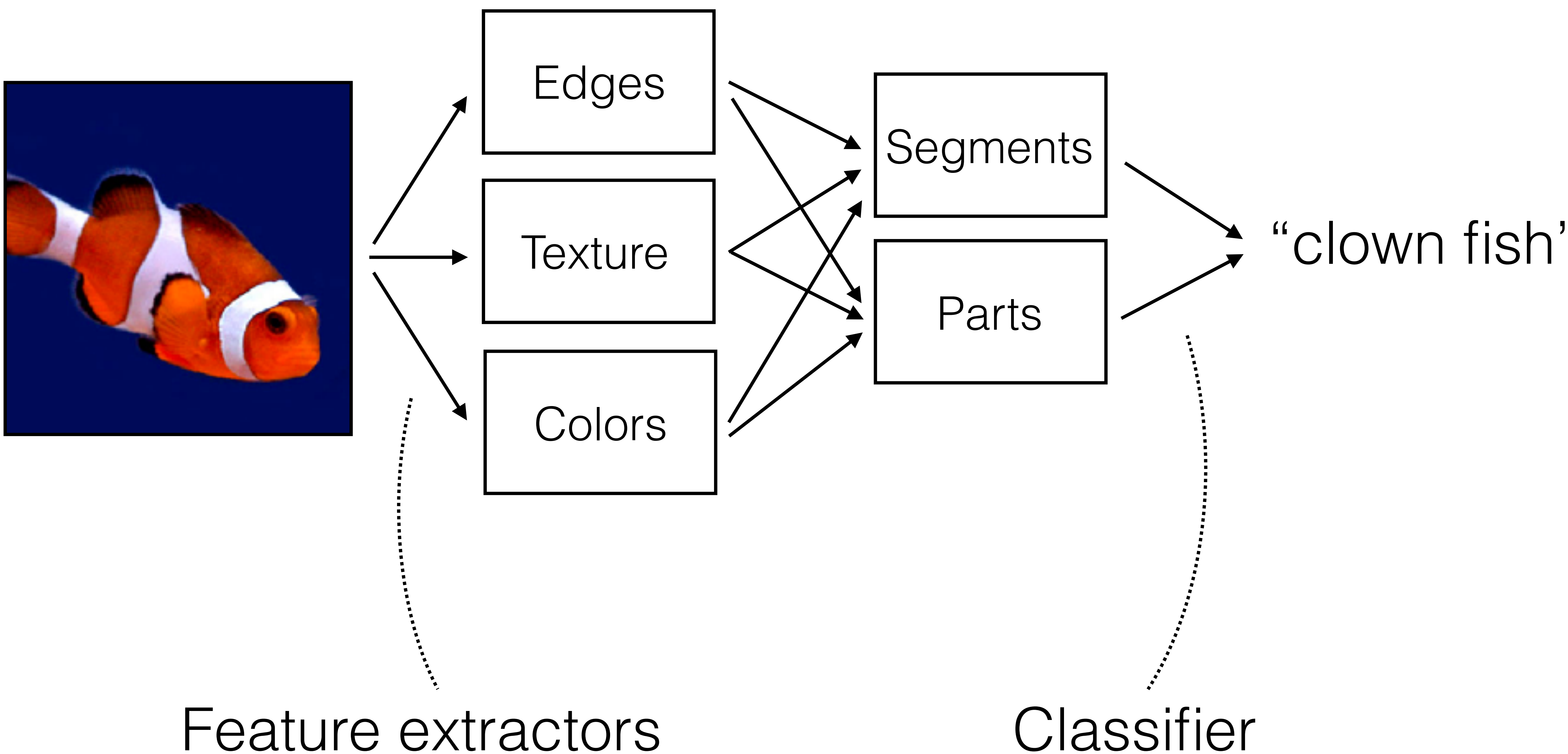
Slide by Phillip Isola

Object Recognition



Slide by Phillip Isola

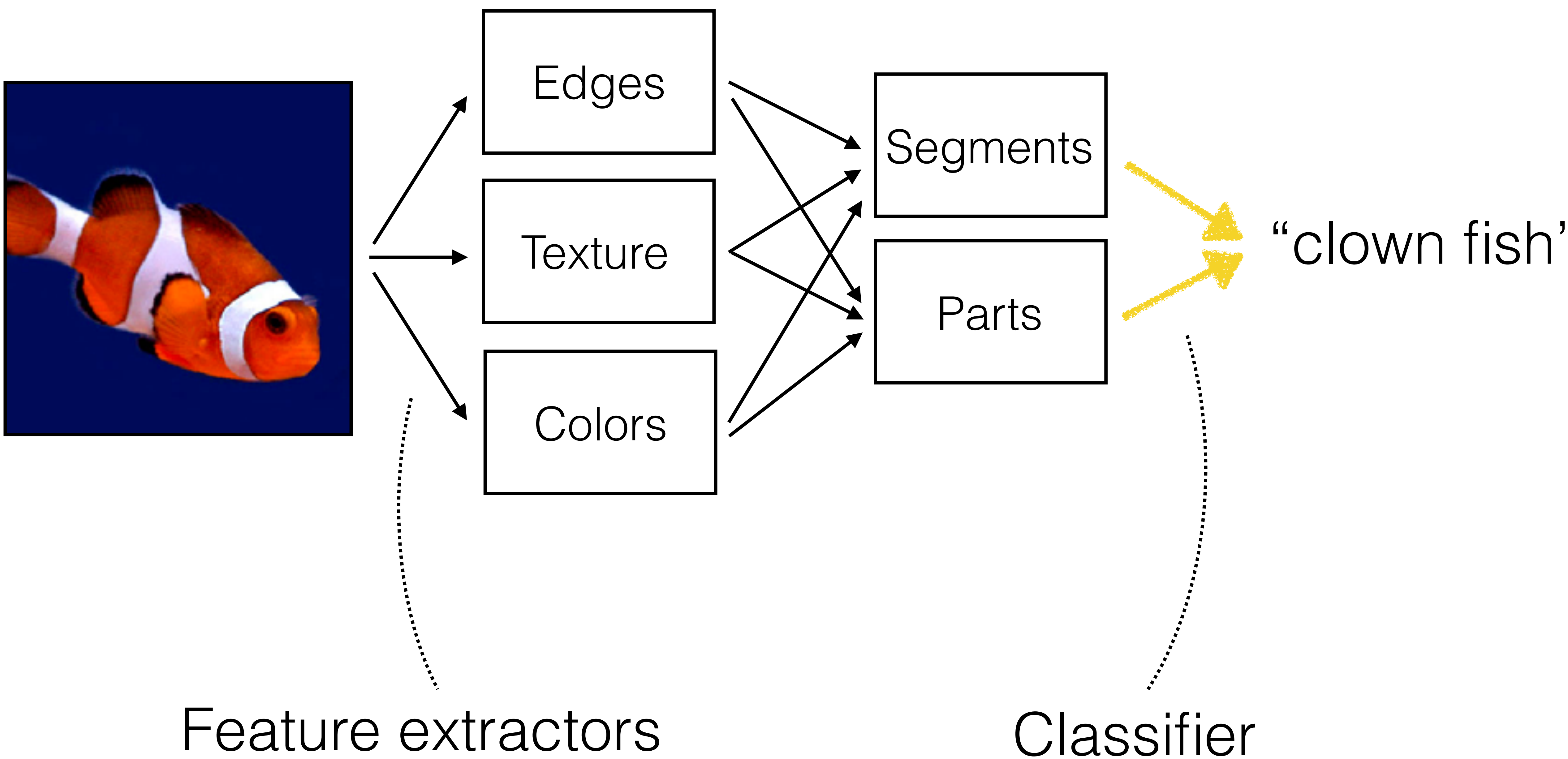
Object Recognition



Slide by Phillip Isola

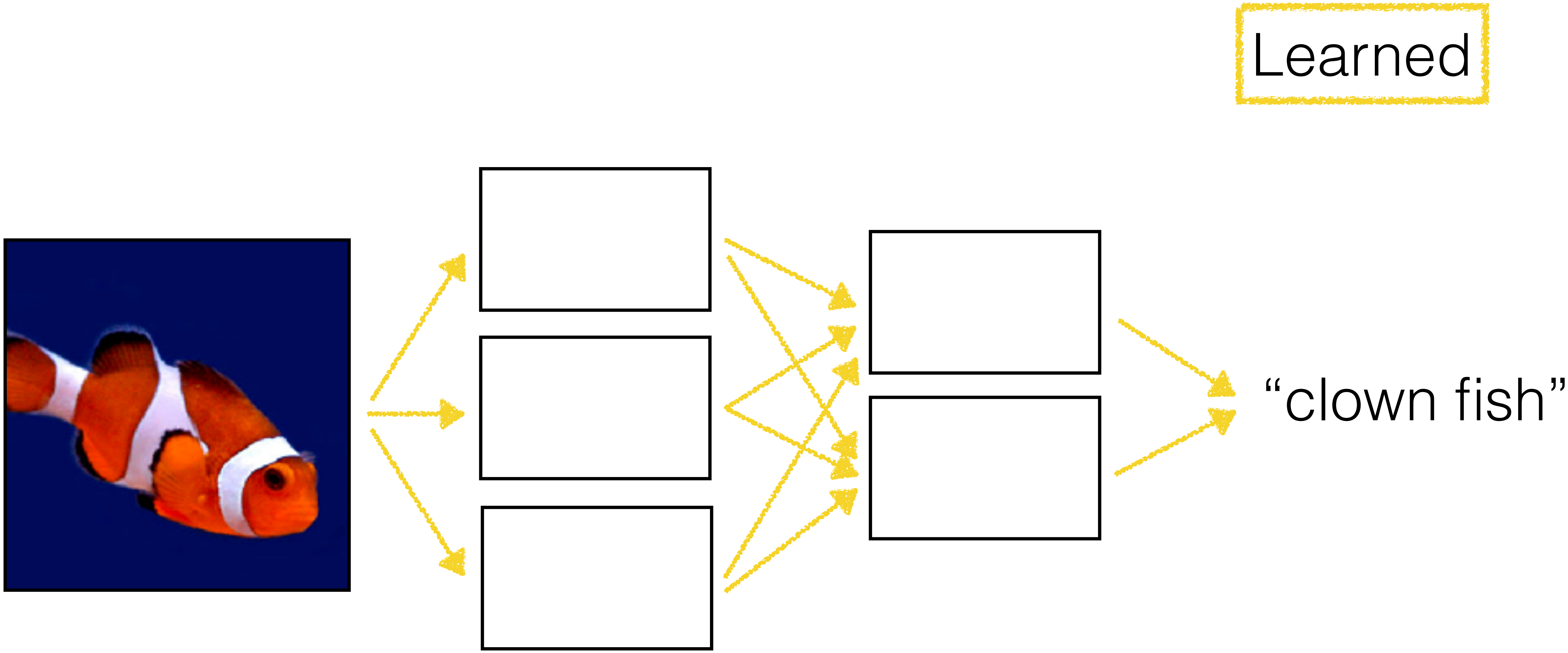
Object Recognition

Learned



Slide by Phillip Isola

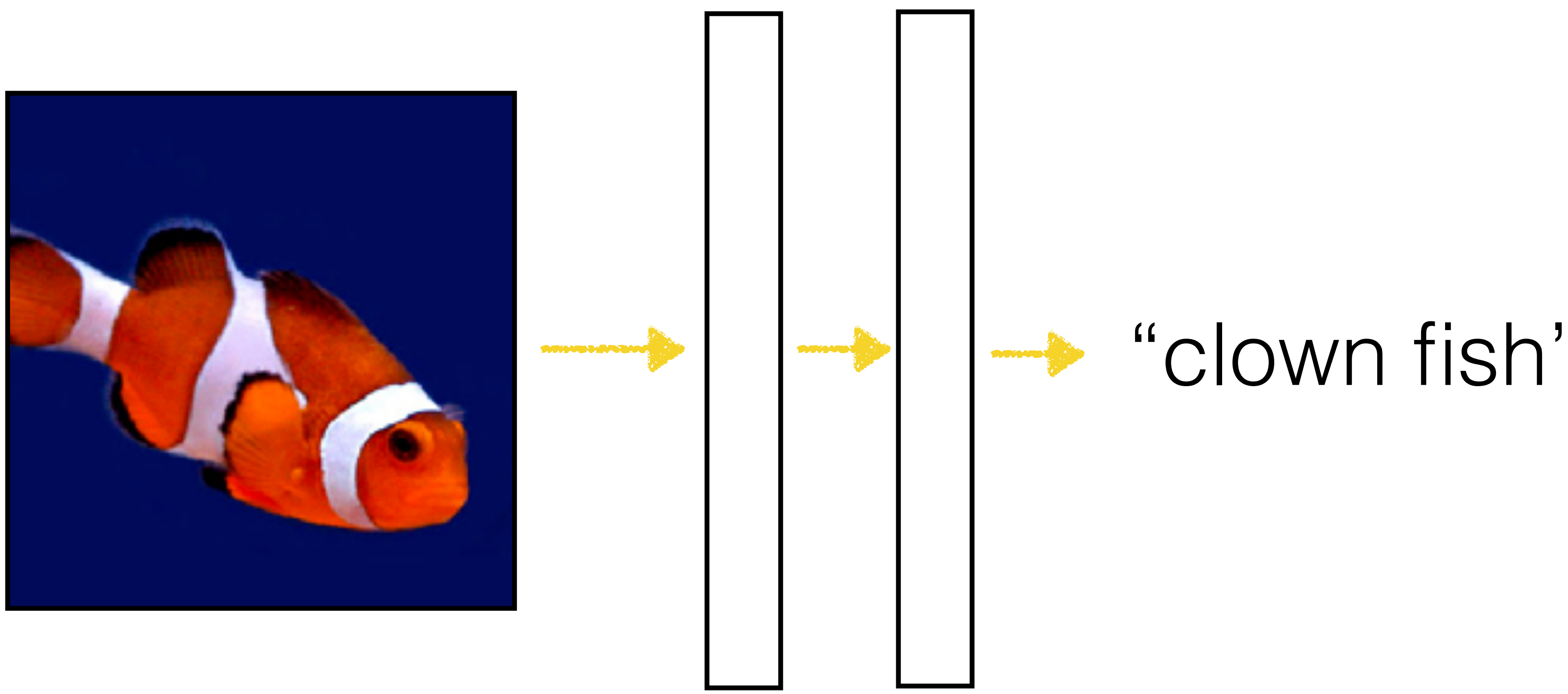
Neural Network



Slide by Phillip Isola

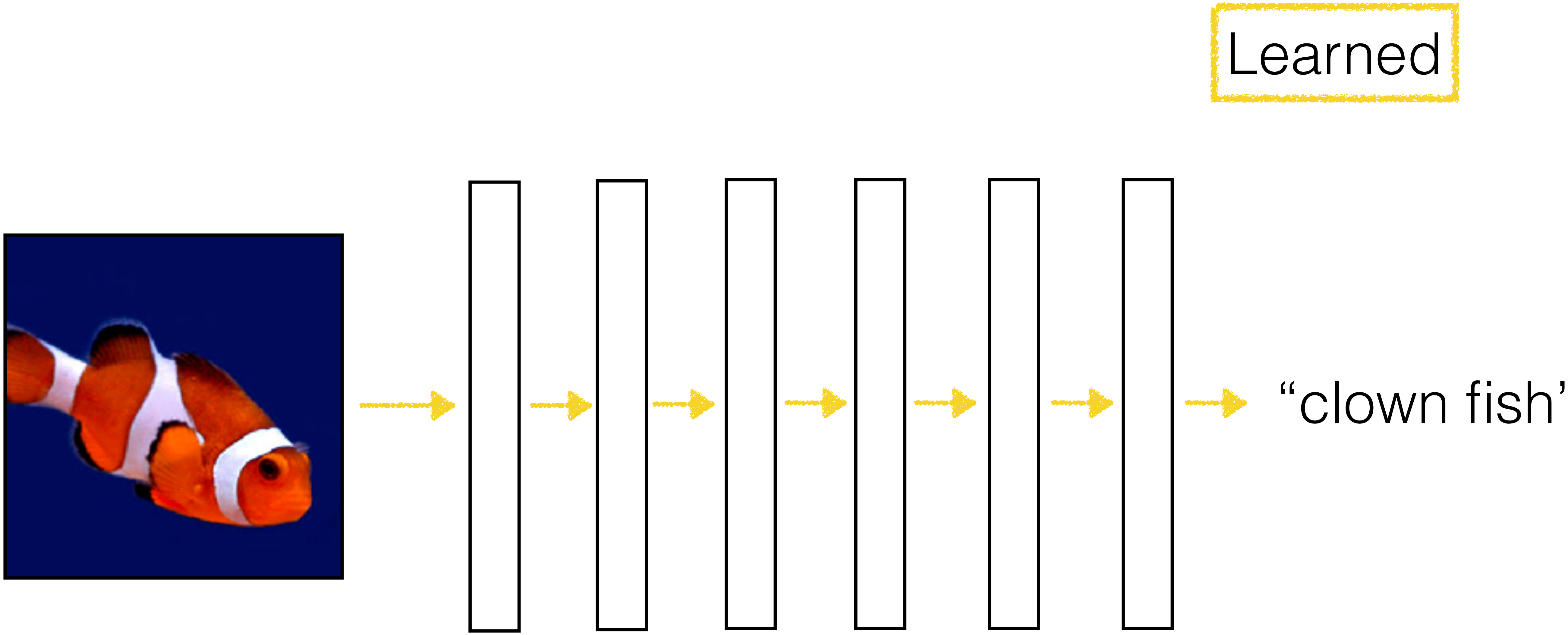
Neural Network

Learned



Slide by Phillip Isola

Deep Neural Network



Slide by Phillip Isola

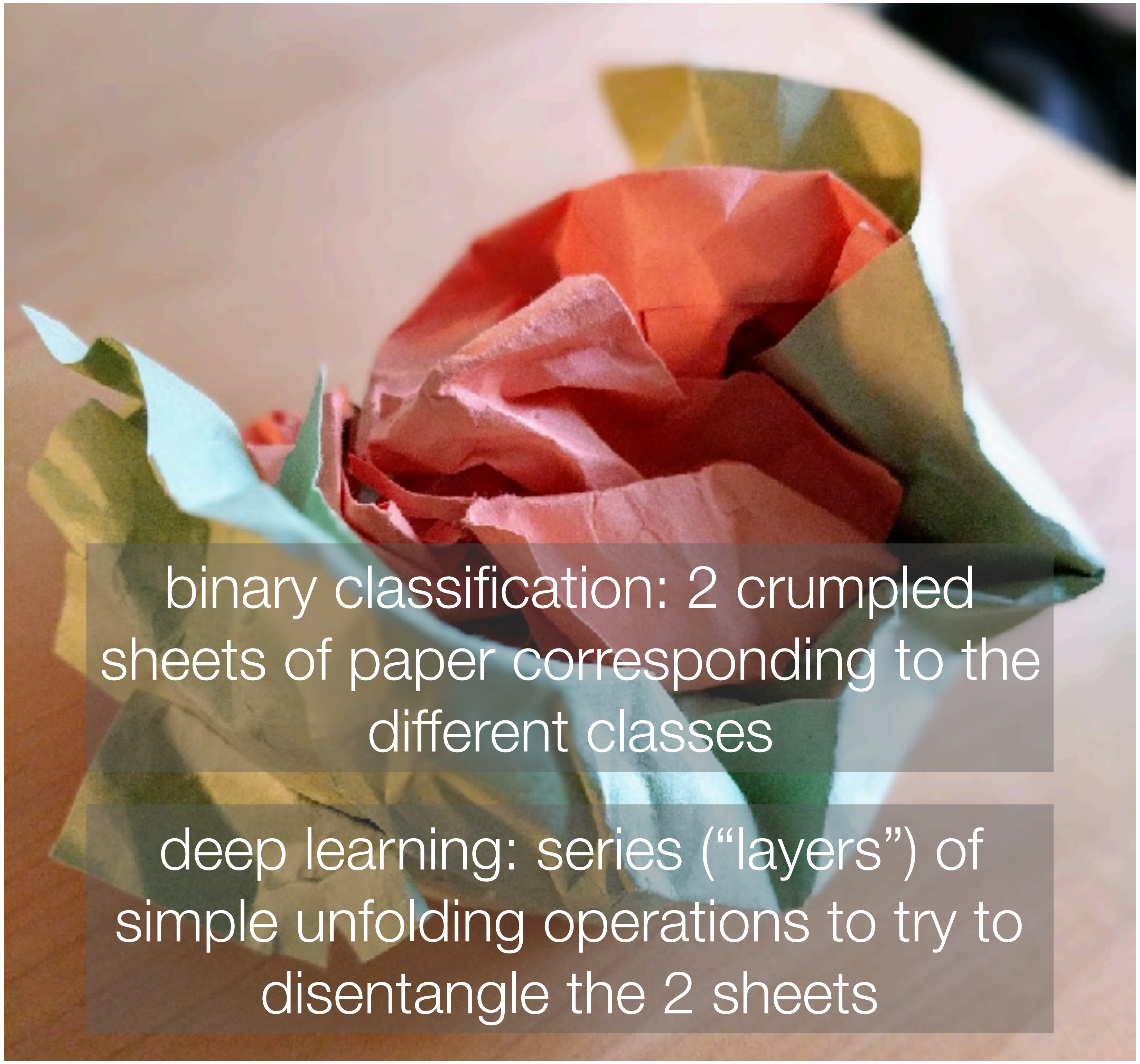
Crumpled Paper Analogy

Analogy: Francois Chollet, photo: George Chen

Crumpled Paper Analogy

Analogy: Francois Chollet, photo: George Chen

Crumpled Paper Analogy



Analogy: Francois Chollet, photo: George Chen

Structure Present in Data Matters

Structure Present in Data Matters

The best performing neural network architectures account for the kind of data they process!

Structure Present in Data Matters

The best performing neural network architectures account for the kind of data they process!

- **Image analysis:** convolutional neural networks (convnets) neatly incorporates intuitive image processing ideas (for example: if a car appears in an image, even if you shift it over by many pixels, it’s still a car)

Structure Present in Data Matters

The best performing neural network architectures account for the kind of data they process!

- **Image analysis:** convolutional neural networks (convnets) neatly incorporates intuitive image processing ideas (for example: if a car appears in an image, even if you shift it over by many pixels, it's still a car)

- **Time series analysis:** recurrent neural networks (RNNs) incorporates ability to remember and forget things over time (note: text naturally comprise of time series as words appear one after another in a meaningful sequence!)

Why Does Deep Learning Work?

Why Does Deep Learning Work?

Actually the ideas behind deep learning are old (~1980’s)

Why Does Deep Learning Work?

Actually the ideas behind deep learning are old (~1980's)

- Big data

Why Does Deep Learning Work?

Actually the ideas behind deep learning are old (~1980's)

- Big data

Why Does Deep Learning Work?

Actually the ideas behind deep learning are old (~1980's)

- Big data

- Better hardware

Why Does Deep Learning Work?

Actually the ideas behind deep learning are old (~1980's)

- Big data
- Better hardware

CPU's
& Moore's law

Why Does Deep Learning Work?

Actually the ideas behind deep learning are old (~1980’s)

- Big data

amazon.com

lyft

NETFLIX fitbit

UPMC
LIFE CHANGING MEDICINE

- Better hardware

intel

AMD

CPU’s
& Moore’s law

GPU’s

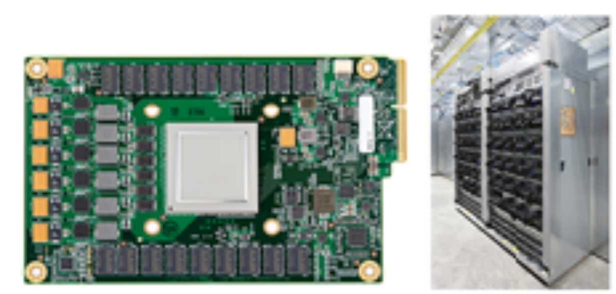
Why Does Deep Learning Work?

Actually the ideas behind deep learning are old (~1980's)

- Big data
- Better hardware

CPU's
& Moore's law

GPU's



TPU's

Why Does Deep Learning Work?

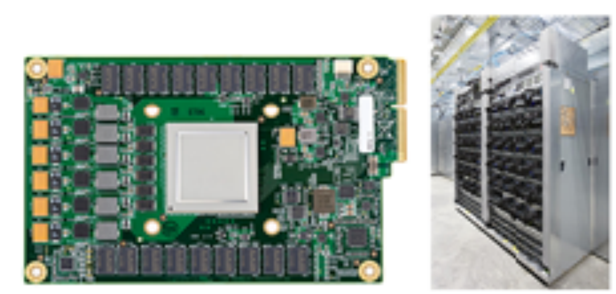
Actually the ideas behind deep learning are old (~1980's)

- Big data

- Better hardware

CPU's
& Moore's law

GPU's



TPU's

- Better algorithms

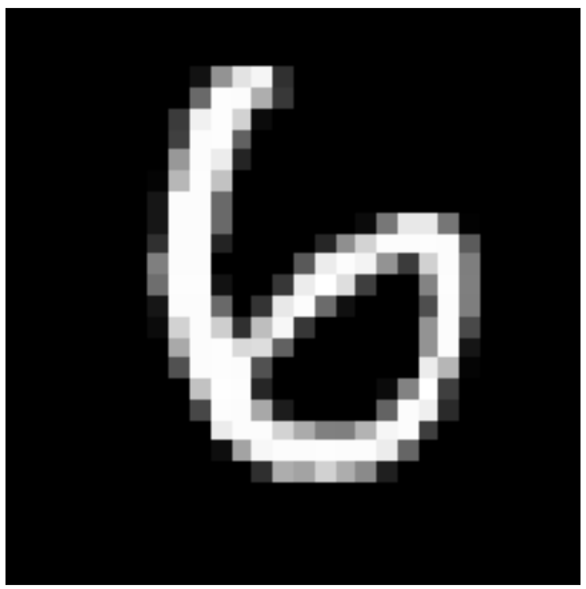
Handwritten Digit Recognition Example

Handwritten Digit Recognition Example

Walkthrough of building a 1-layer and then a 2-layer neural net

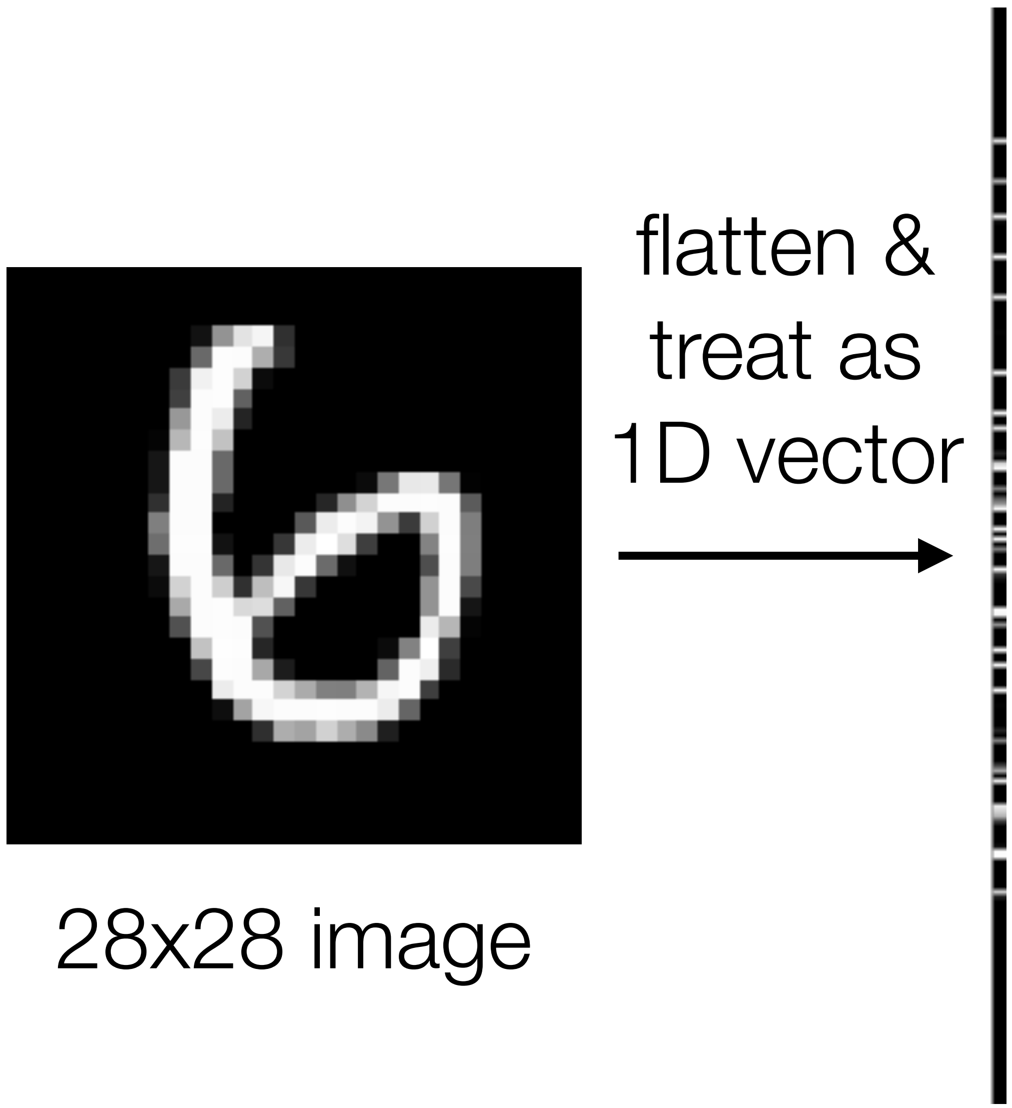
Handwritten Digit Recognition

Handwritten Digit Recognition

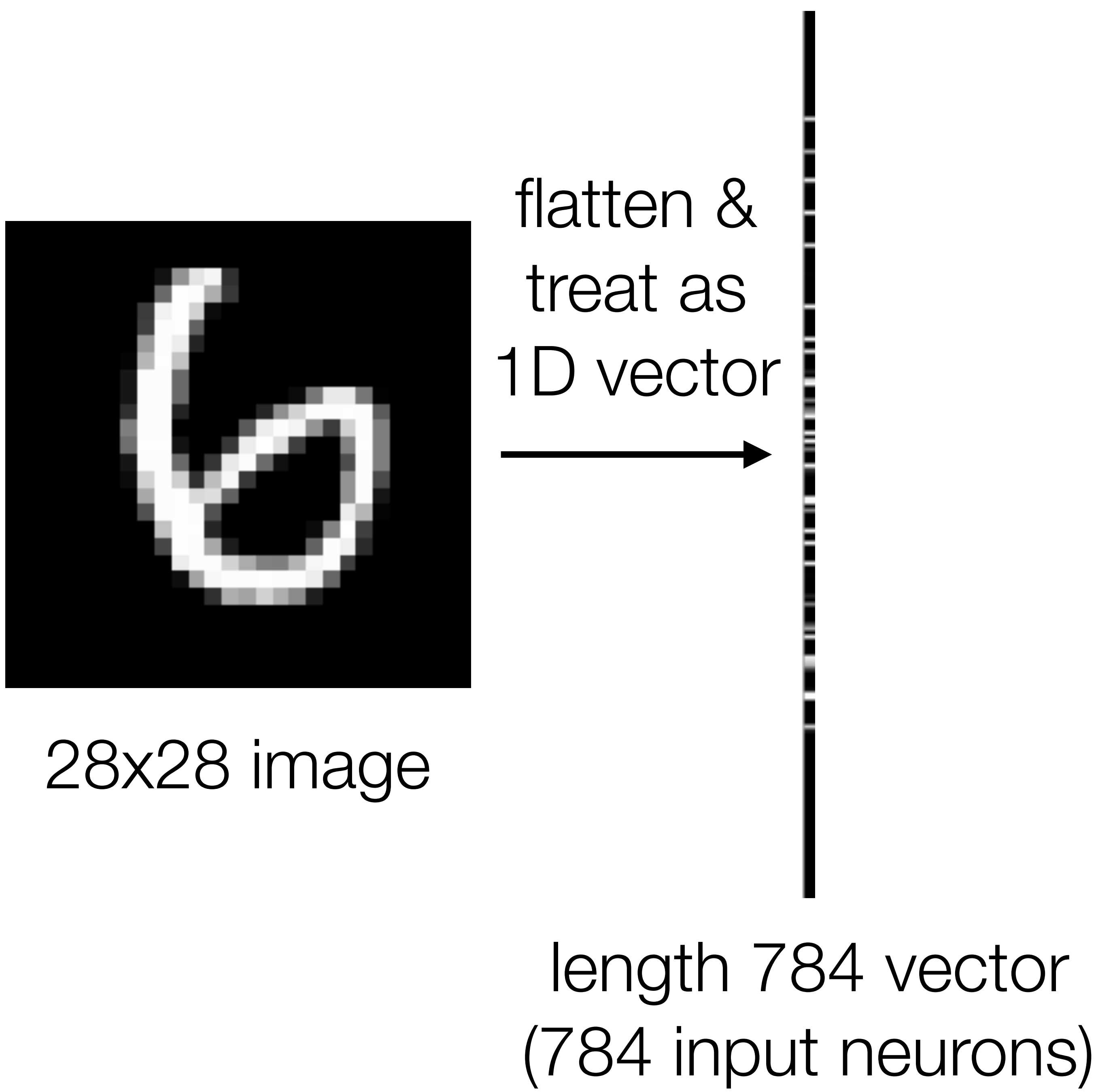


28x28 image

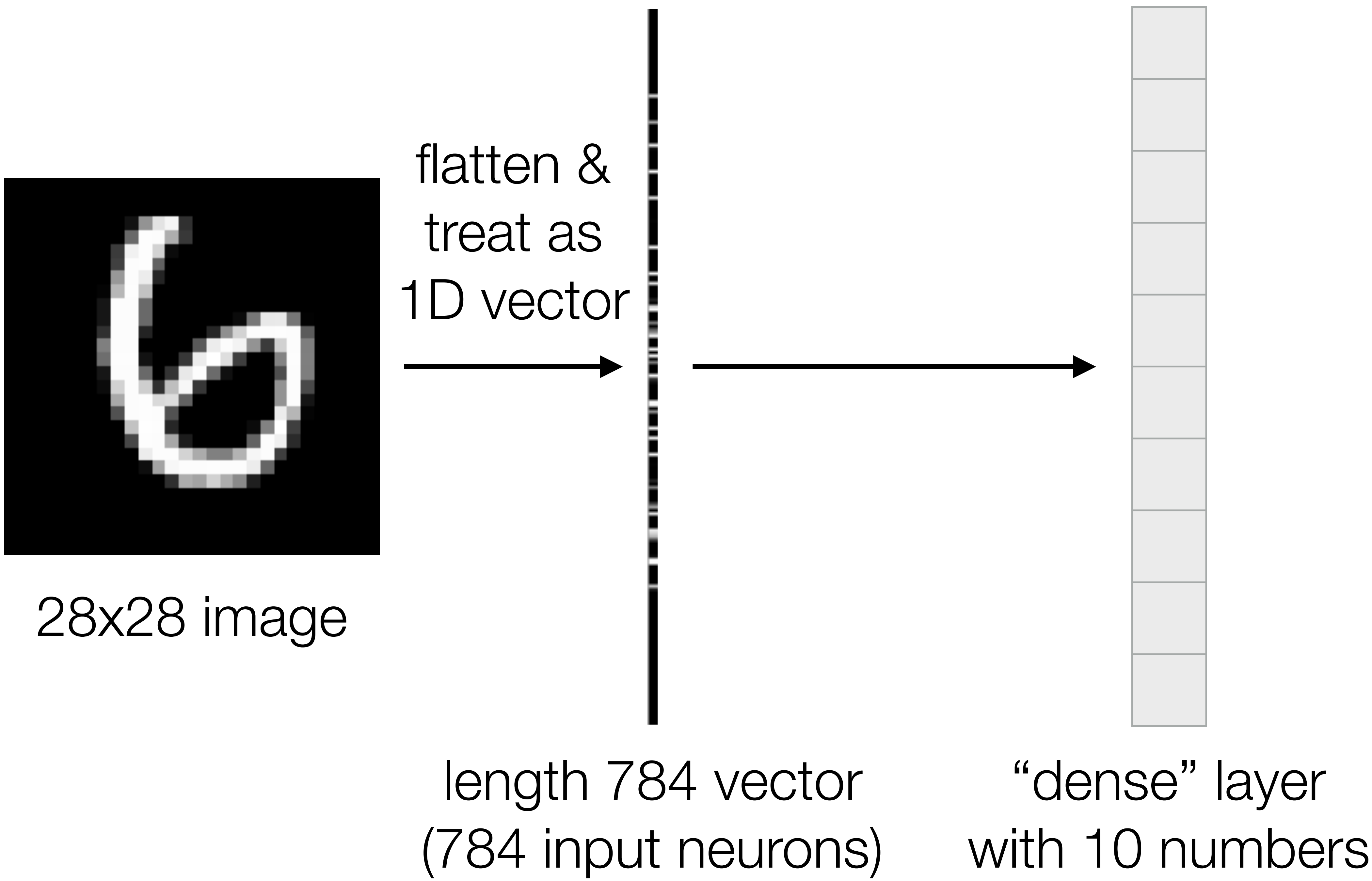
Handwritten Digit Recognition



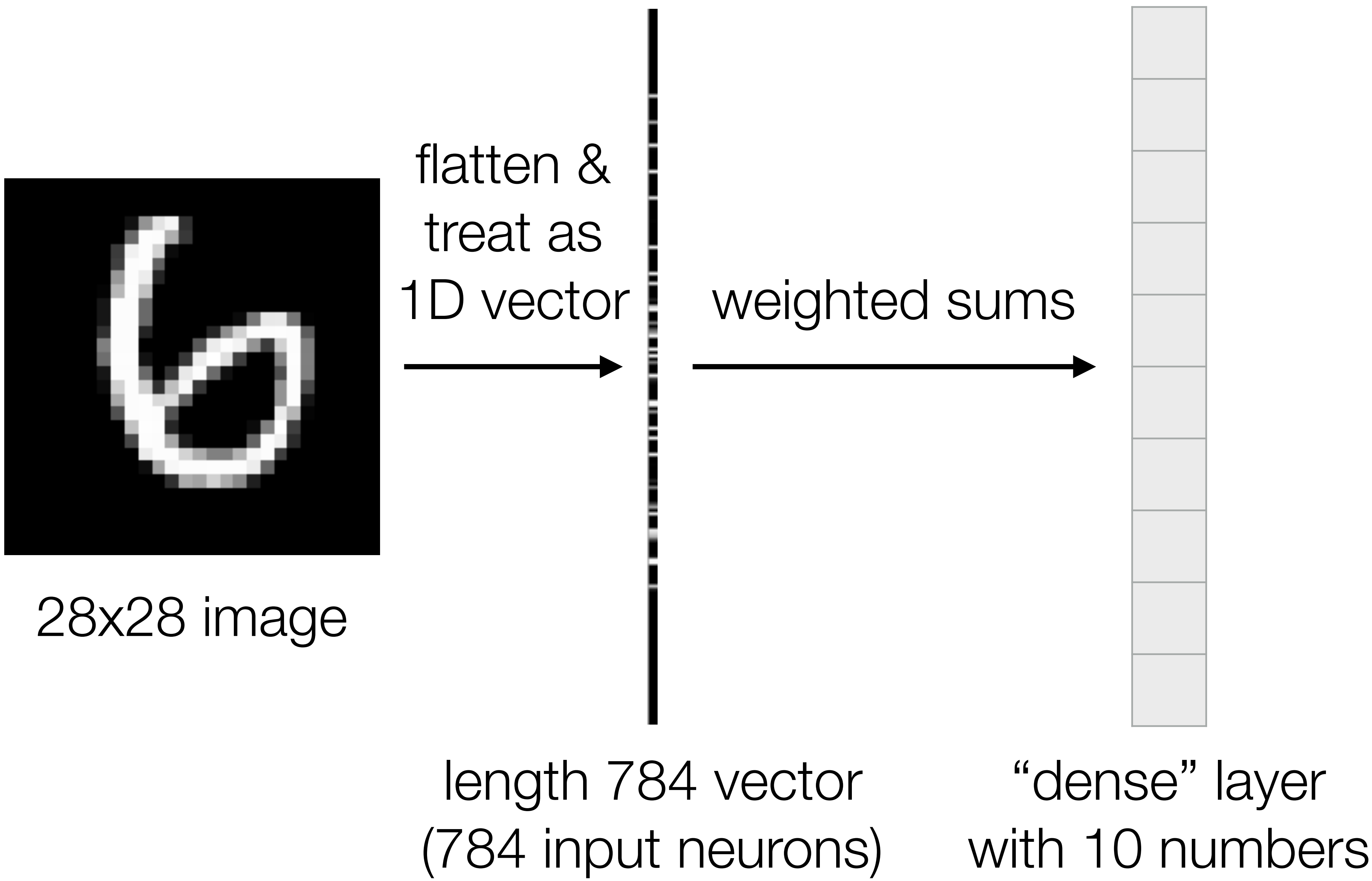
Handwritten Digit Recognition



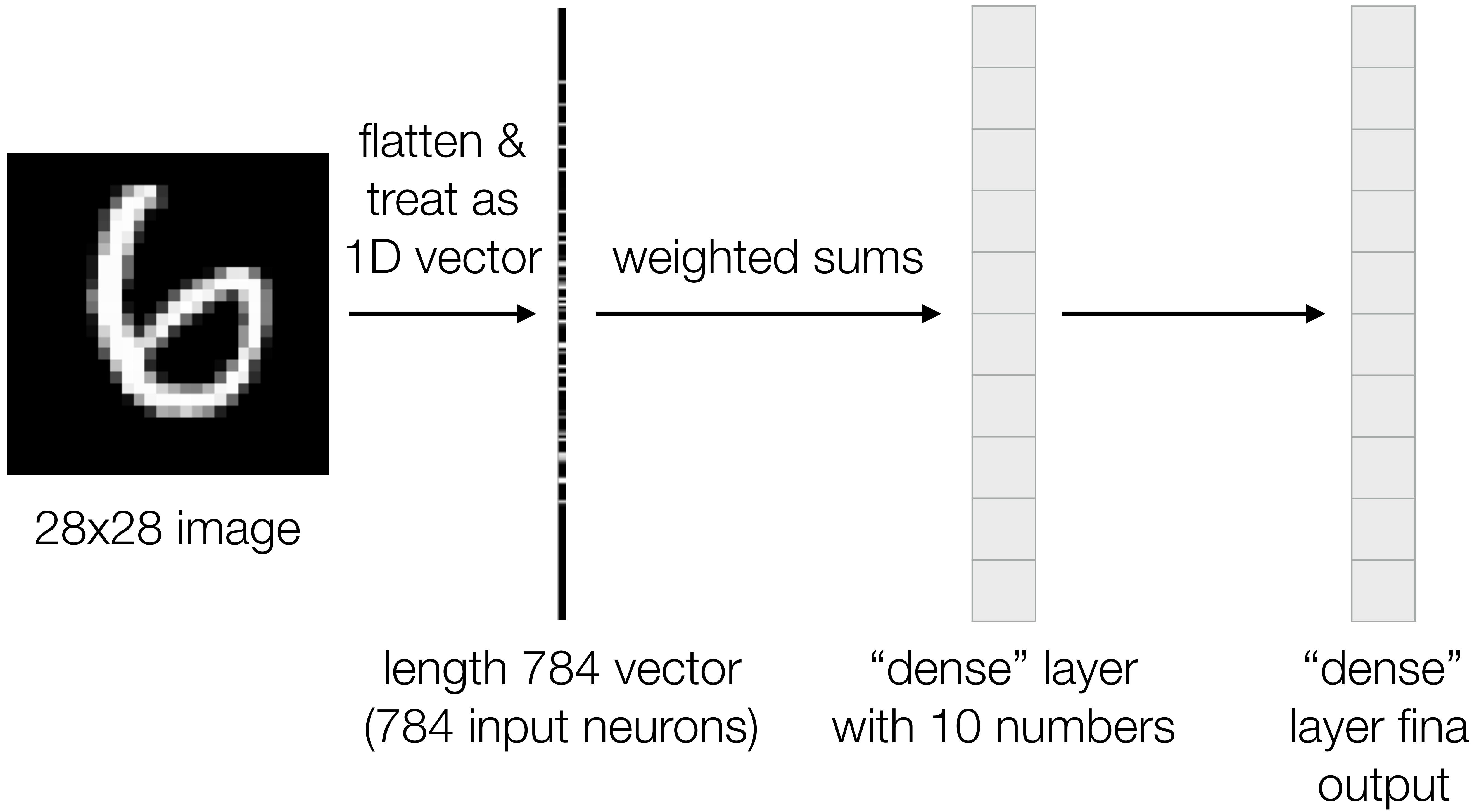
Handwritten Digit Recognition



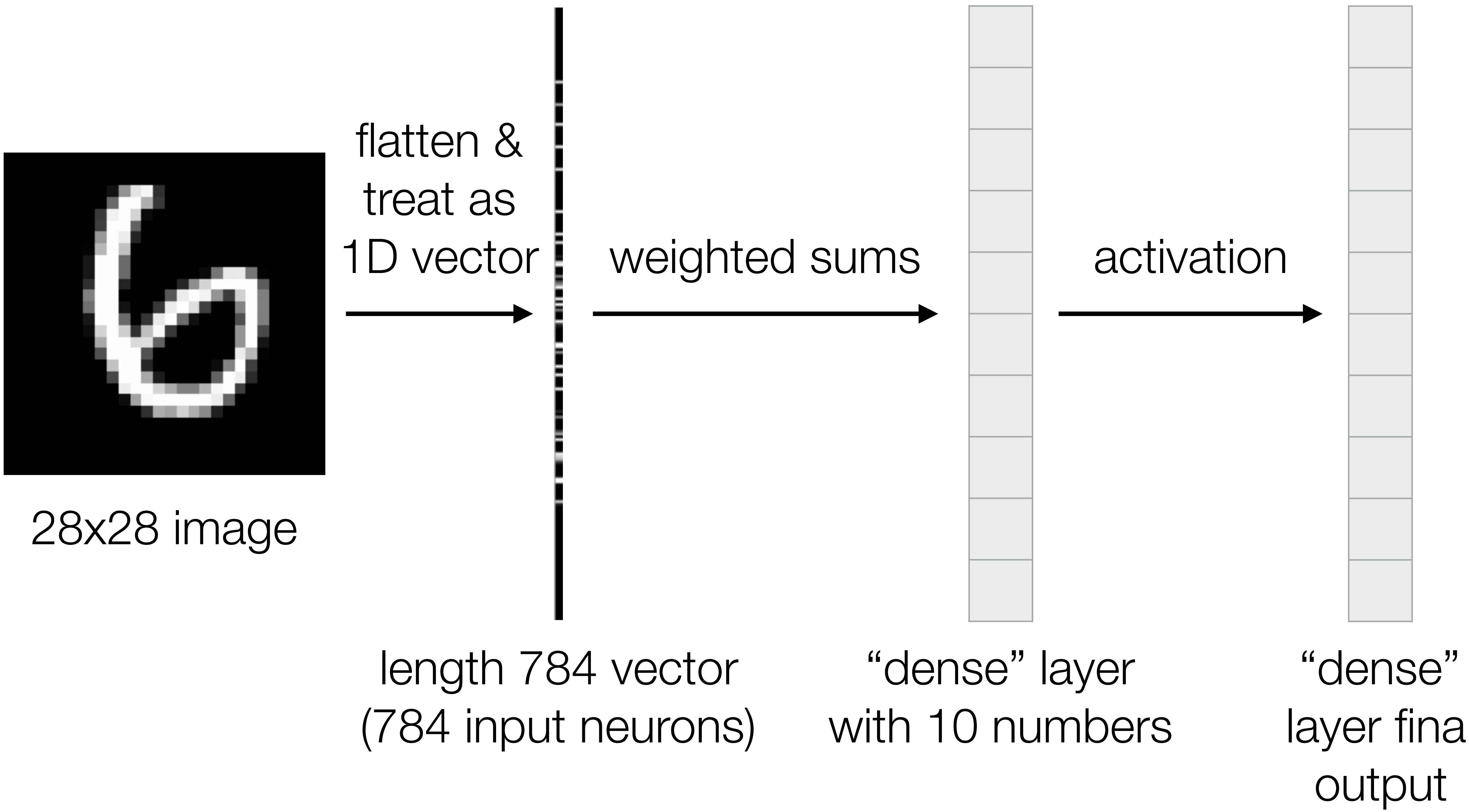
Handwritten Digit Recognition



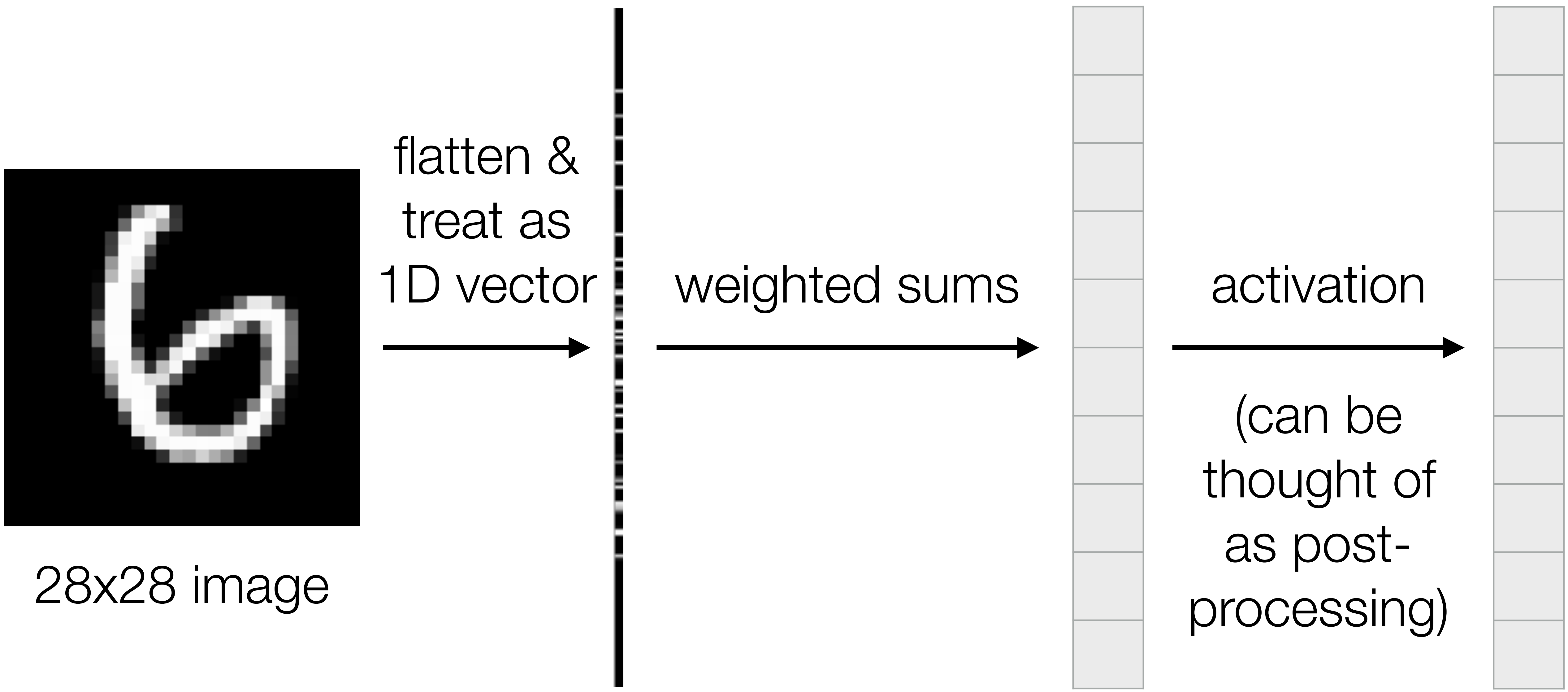
Handwritten Digit Recognition



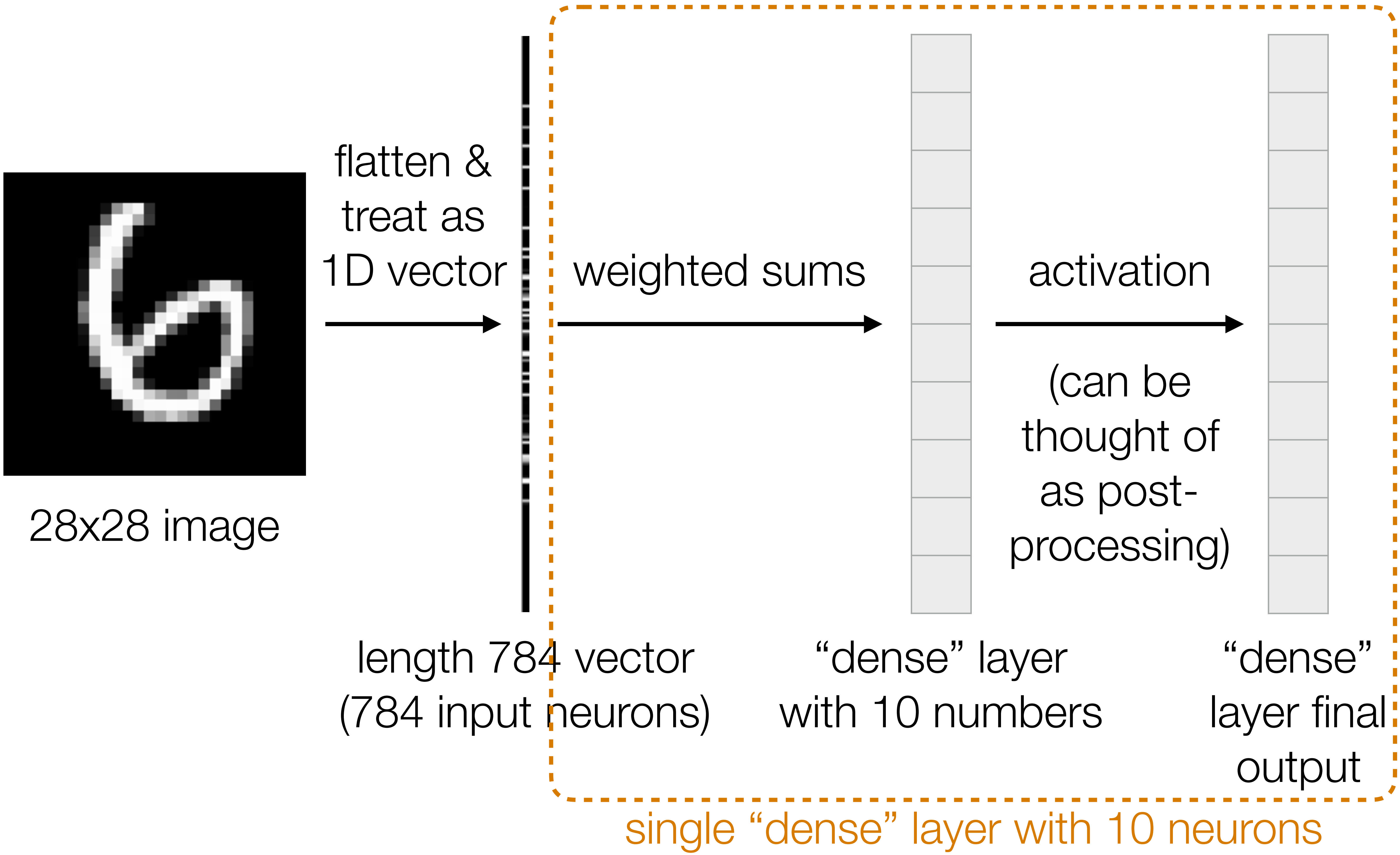
Handwritten Digit Recognition



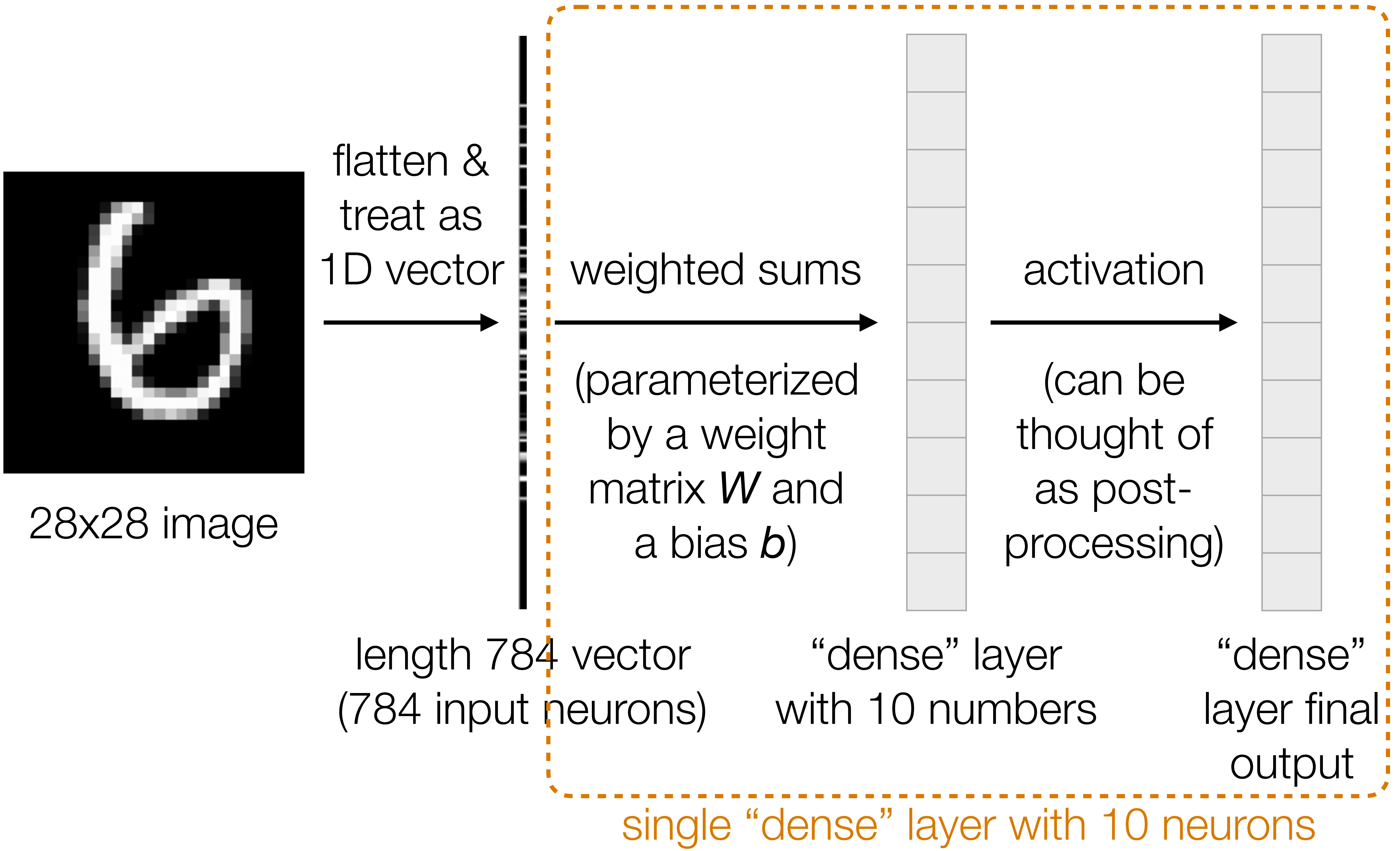
Handwritten Digit Recognition



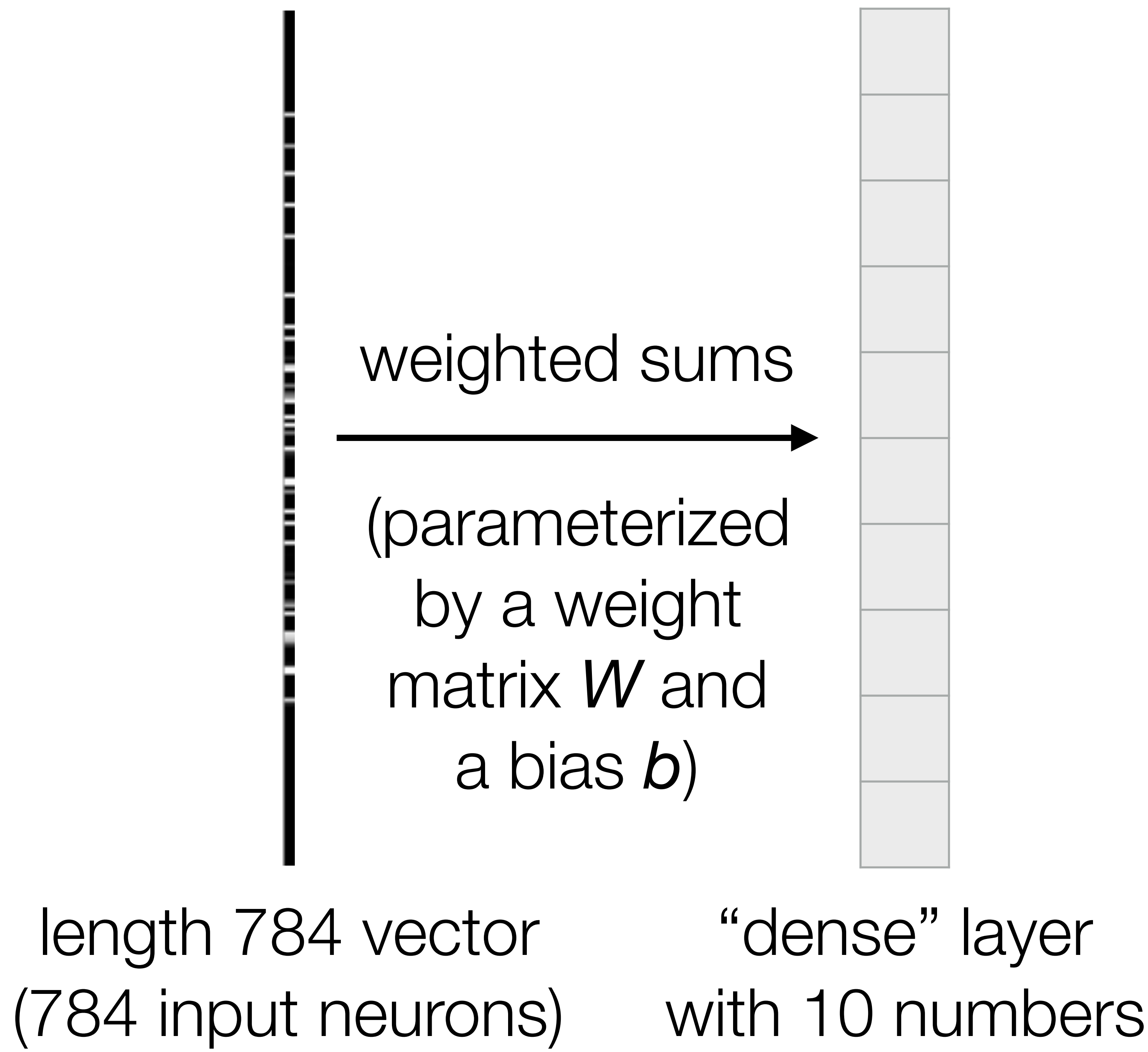
Handwritten Digit Recognition



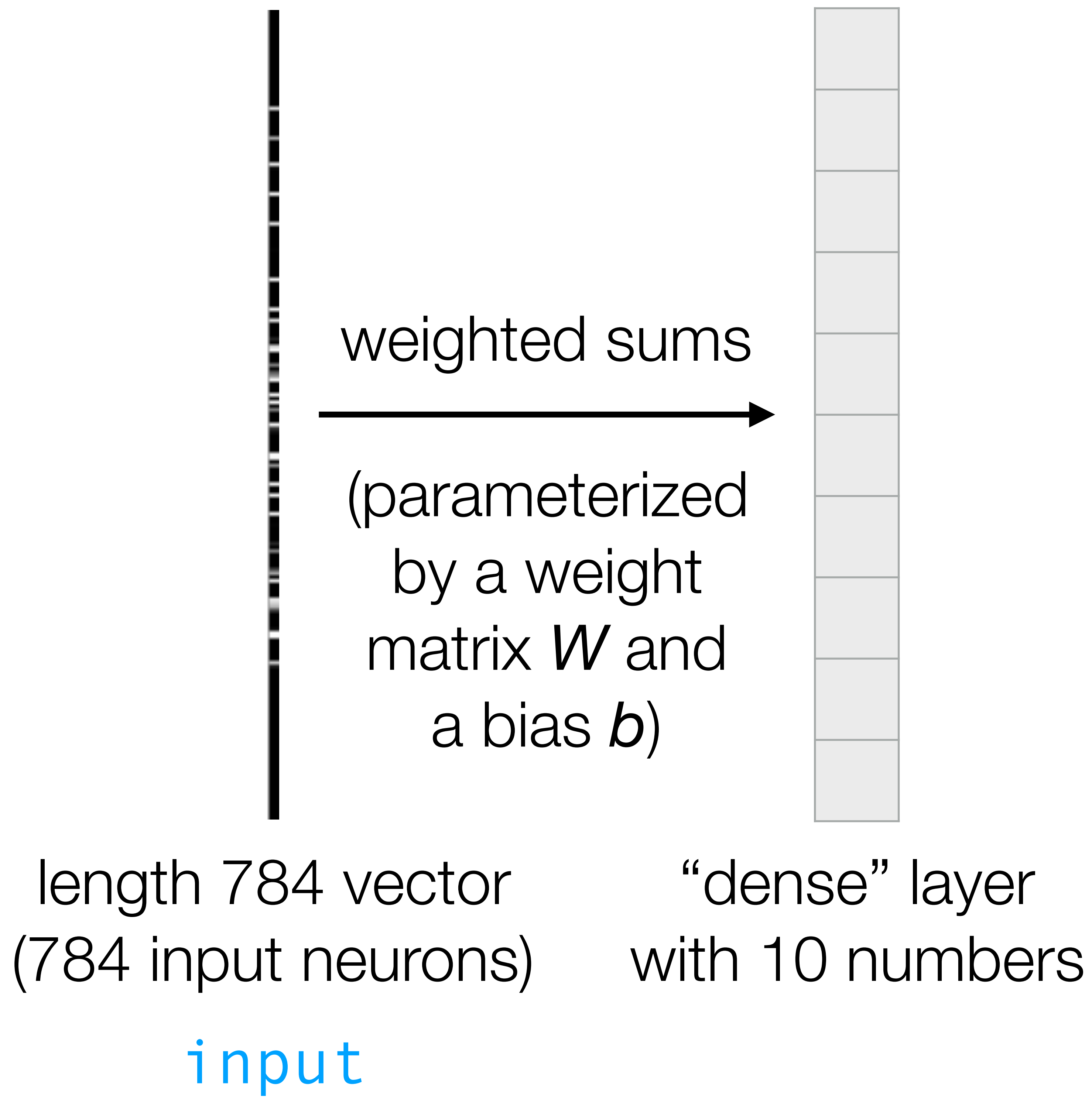
Handwritten Digit Recognition



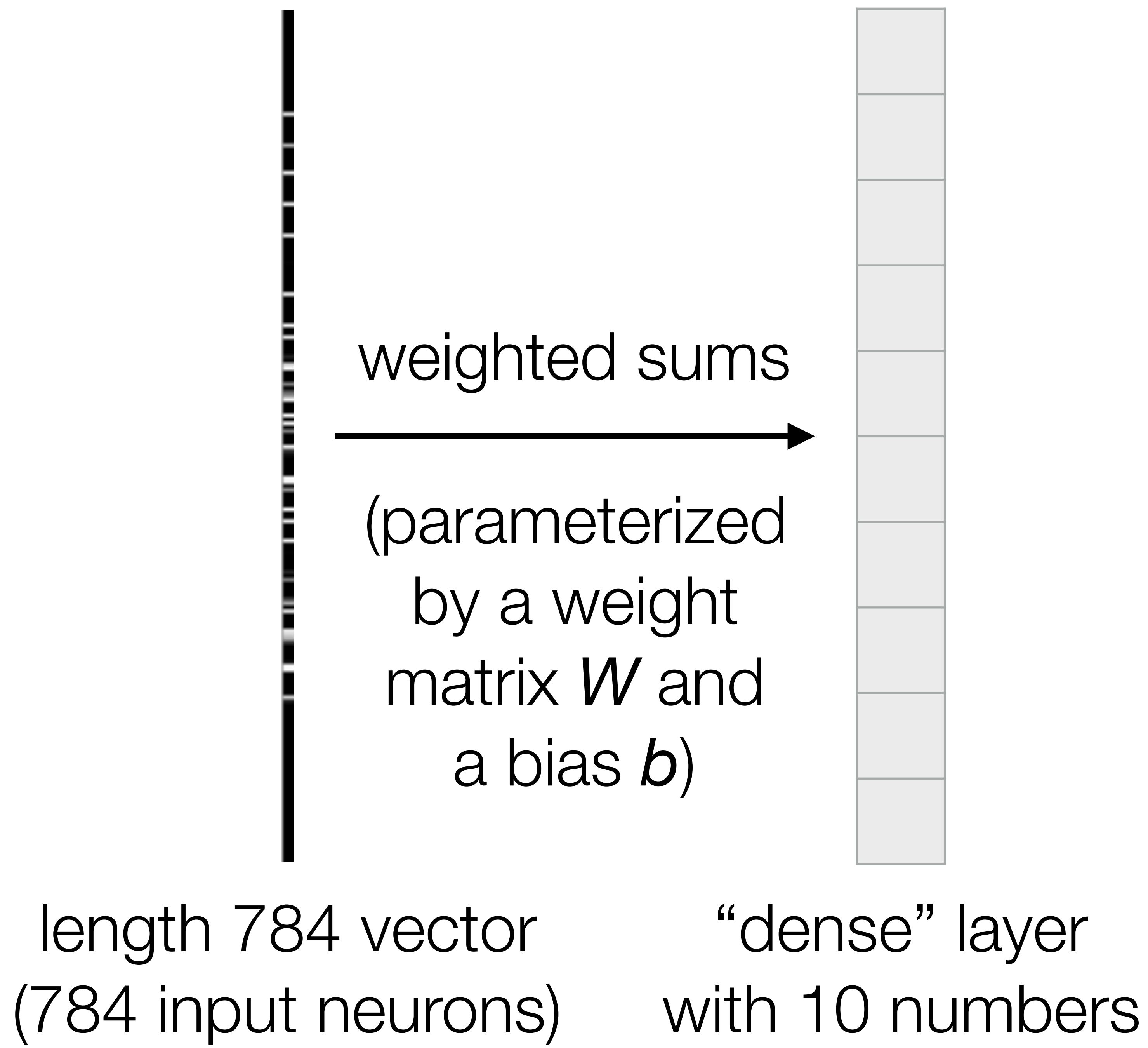
Handwritten Digit Recognition



Handwritten Digit Recognition



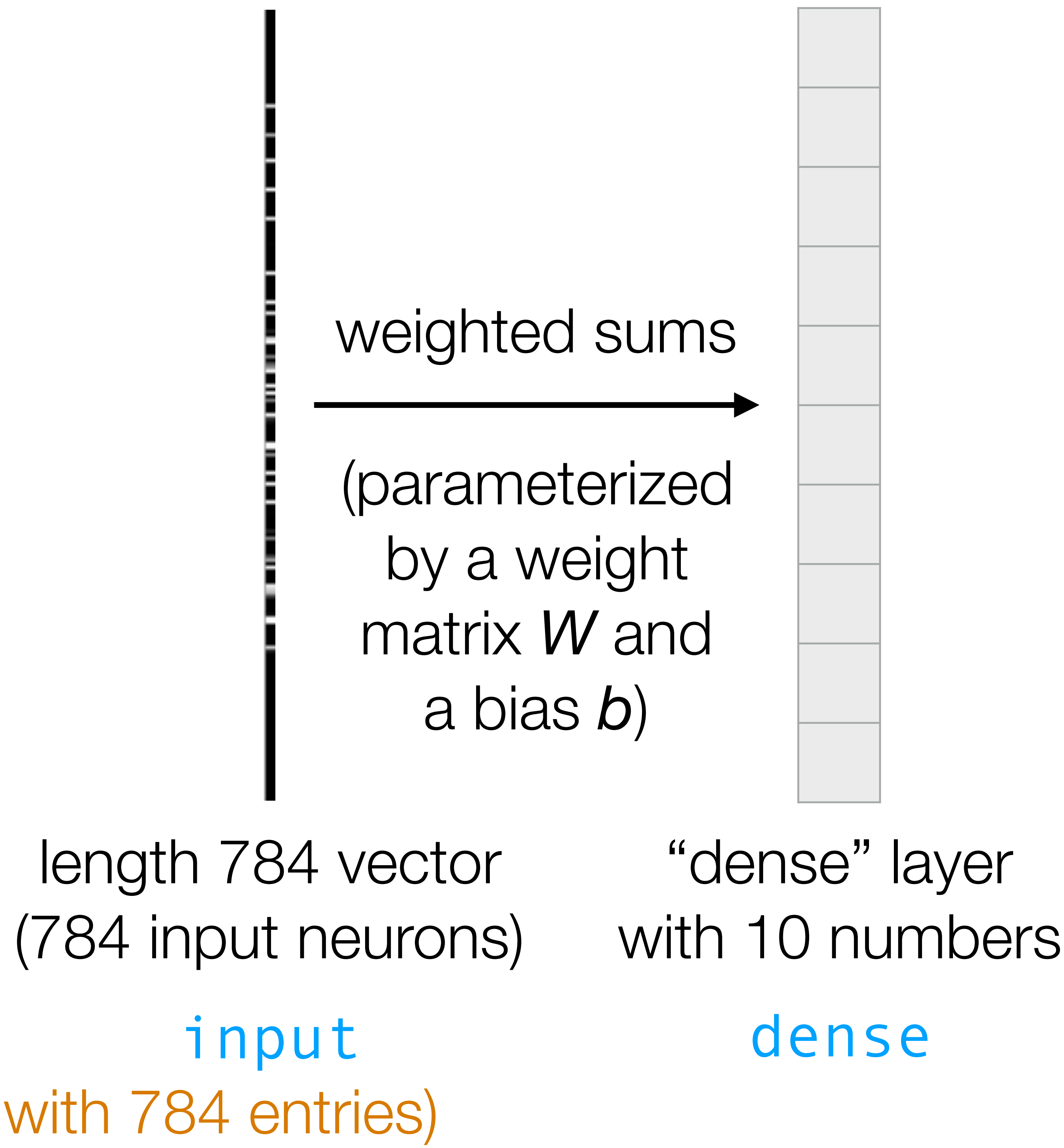
Handwritten Digit Recognition



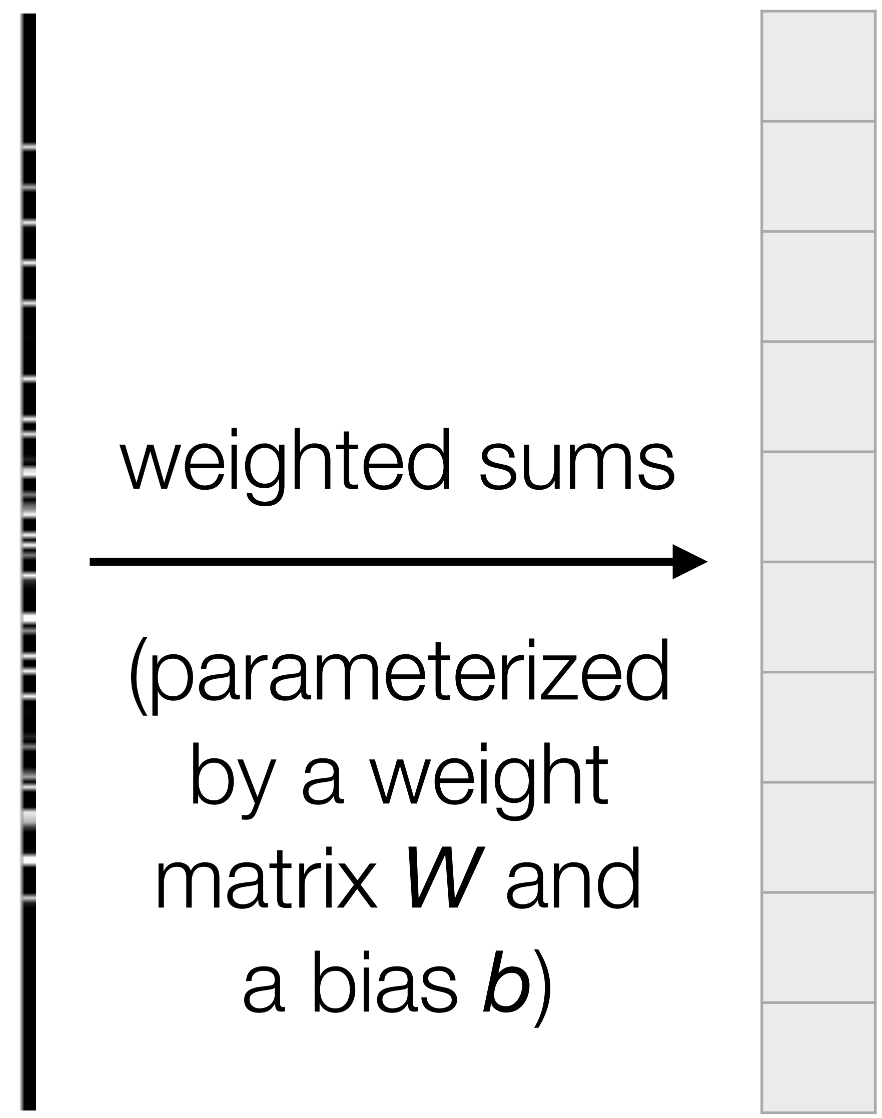
`input`

(1D numpy array with 784 entries)

Handwritten Digit Recognition



Handwritten Digit Recognition



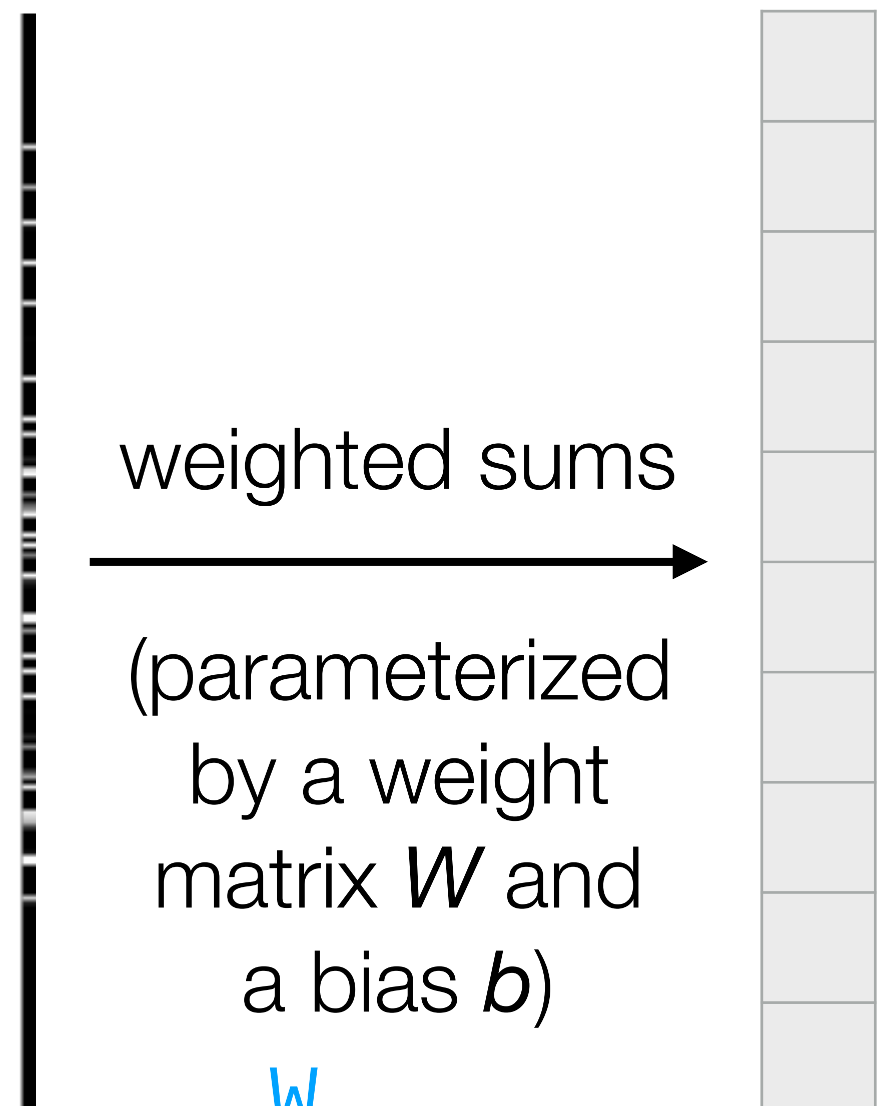
length 784 vector
(784 input neurons)

"dense" layer
with 10 numbers

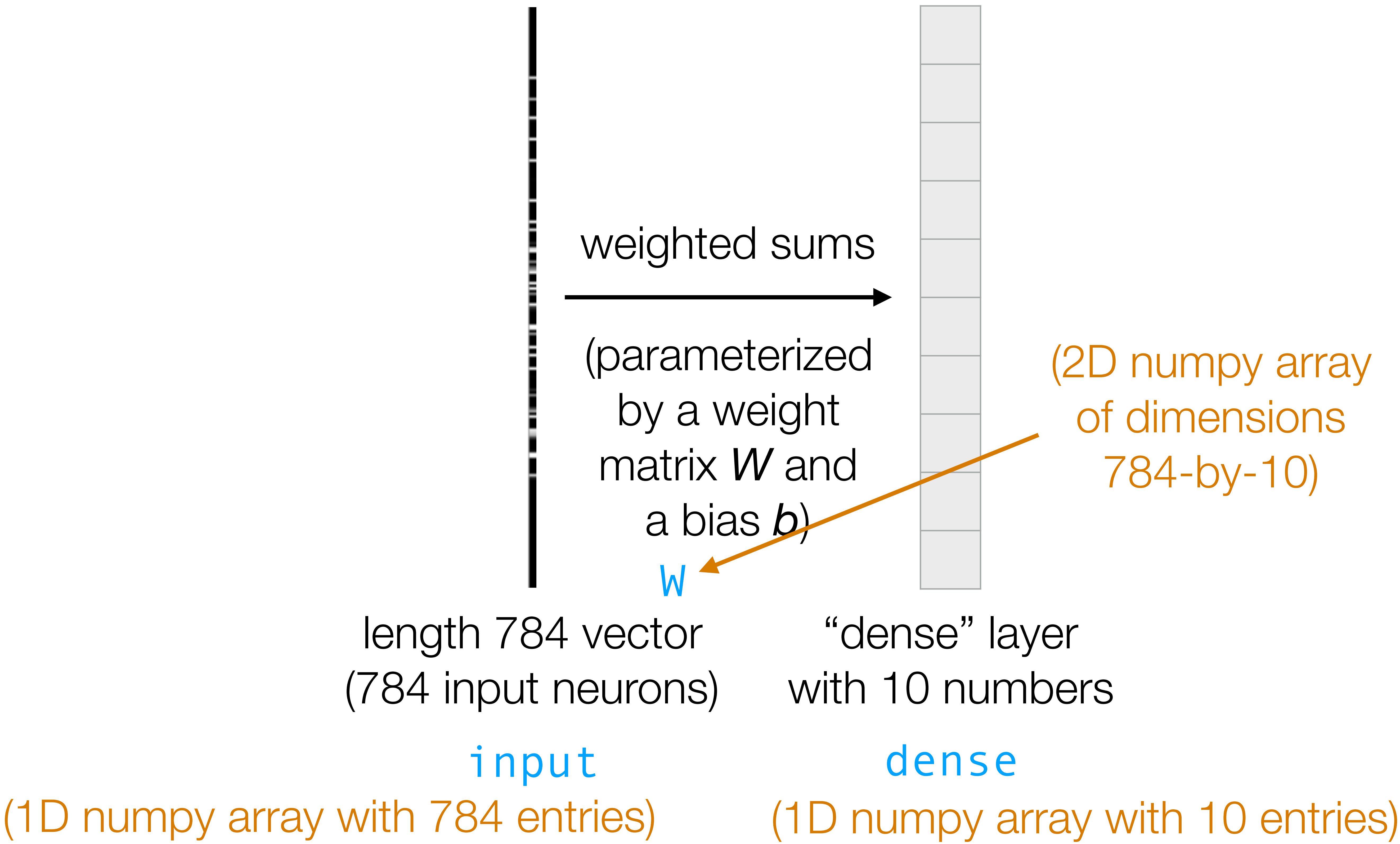
`input`
(1D numpy array with 784 entries)

`dense`
(1D numpy array with 10 entries)

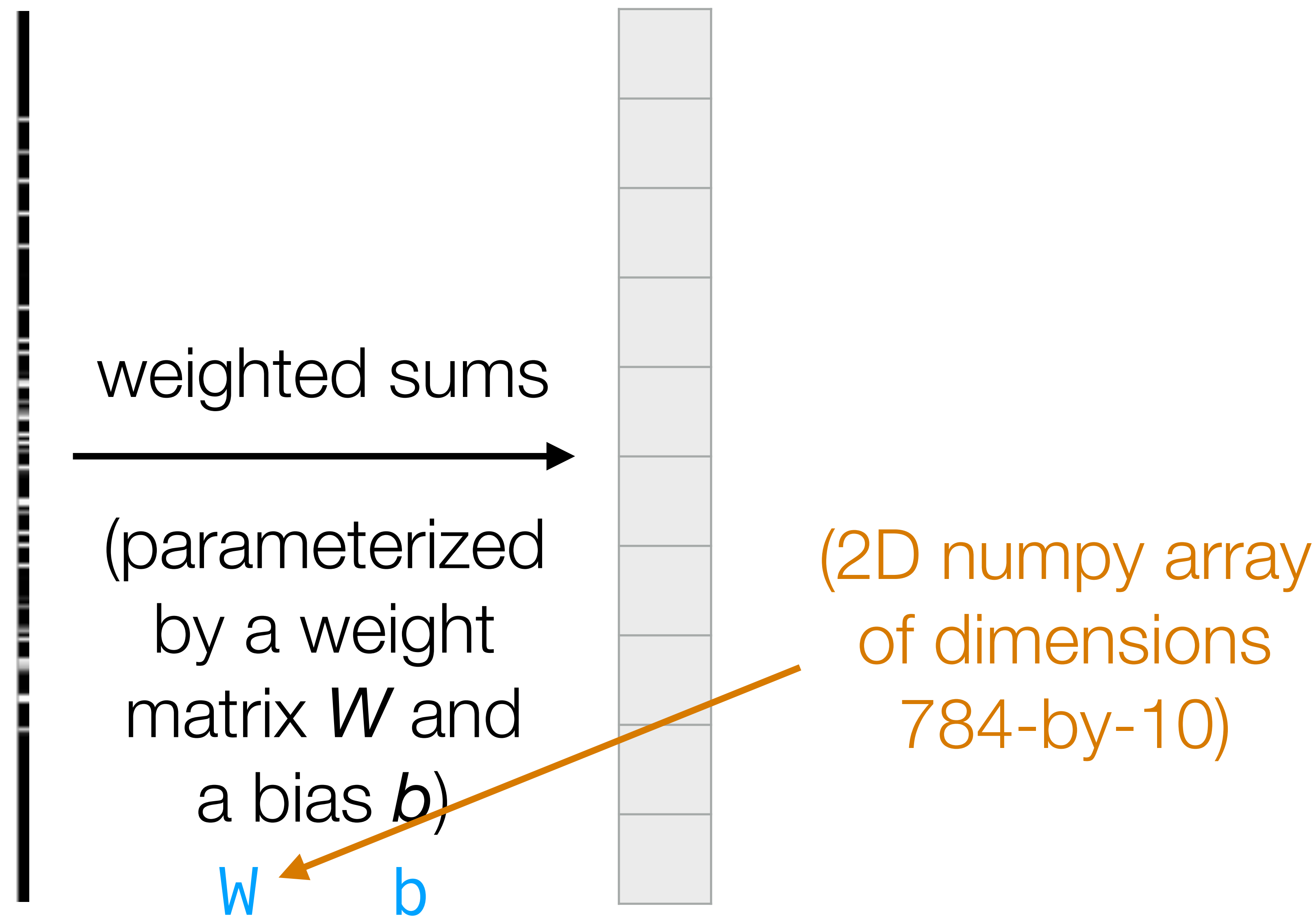
Handwritten Digit Recognition



Handwritten Digit Recognition



Handwritten Digit Recognition



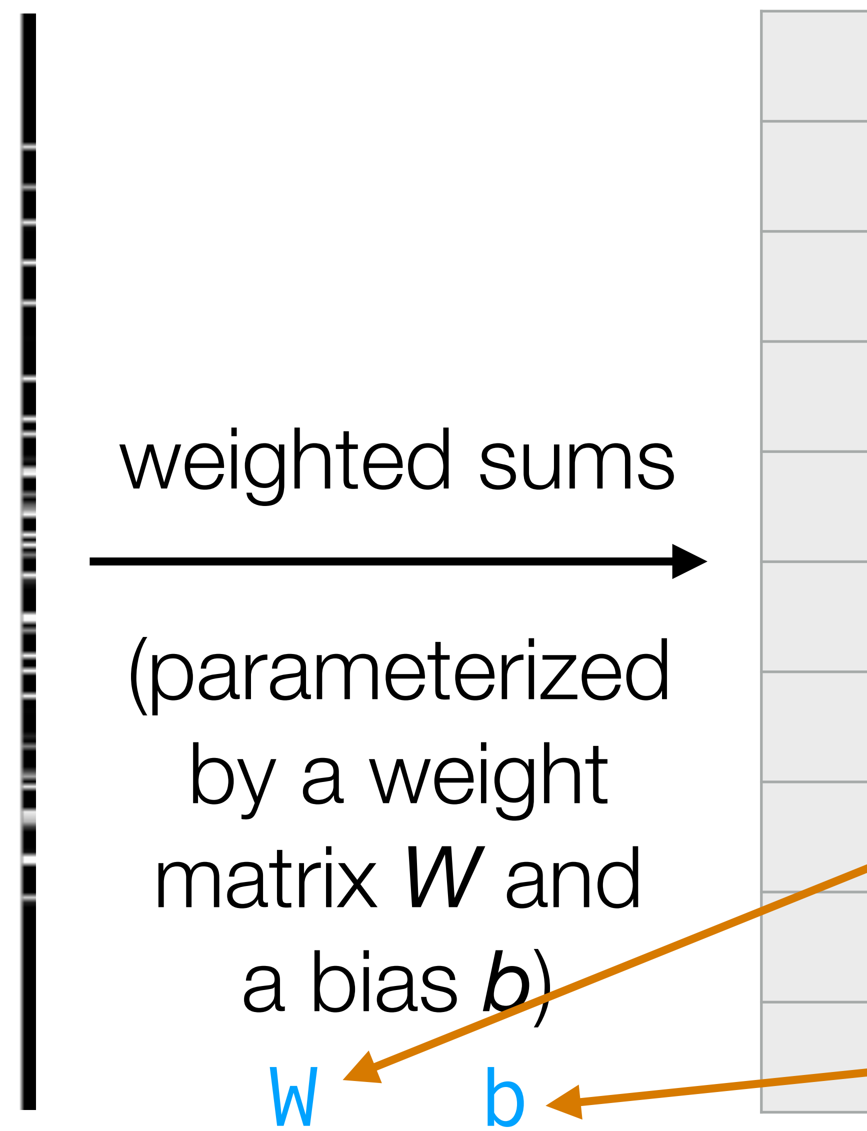
length 784 vector
(784 input neurons)

“dense” layer
with 10 numbers

`input`
(1D numpy array with 784 entries)

`dense`
(1D numpy array with 10 entries)

Handwritten Digit Recognition



(2D numpy array of dimensions 784-by-10)

(1D numpy array with 10 entries)

length 784 vector (784 input neurons)

“dense” layer with 10 numbers

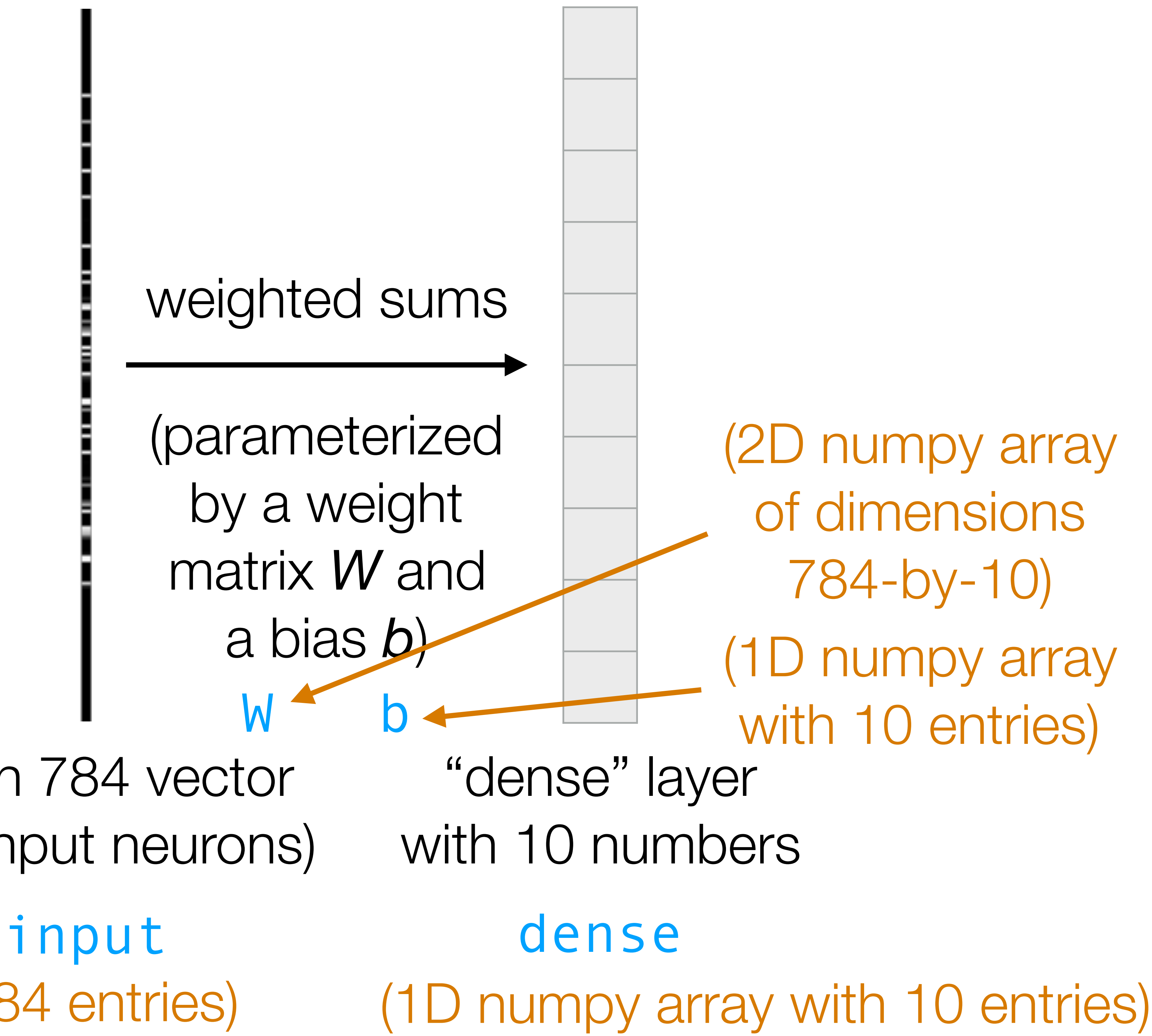
`input`

(1D numpy array with 784 entries)

`dense`

(1D numpy array with 10 entries)

Handwritten Digit Recognition



Handwritten Digit Recognition

```
dense[0] = np.dot(input, W[:, 0]) + b[0]
```

weighted sums

(parameterized by a weight matrix *W* and a bias *b*)

W b

(2D numpy array of dimensions 784-by-10)

(1D numpy array with 10 entries)

n 784 vector
nput neurons)

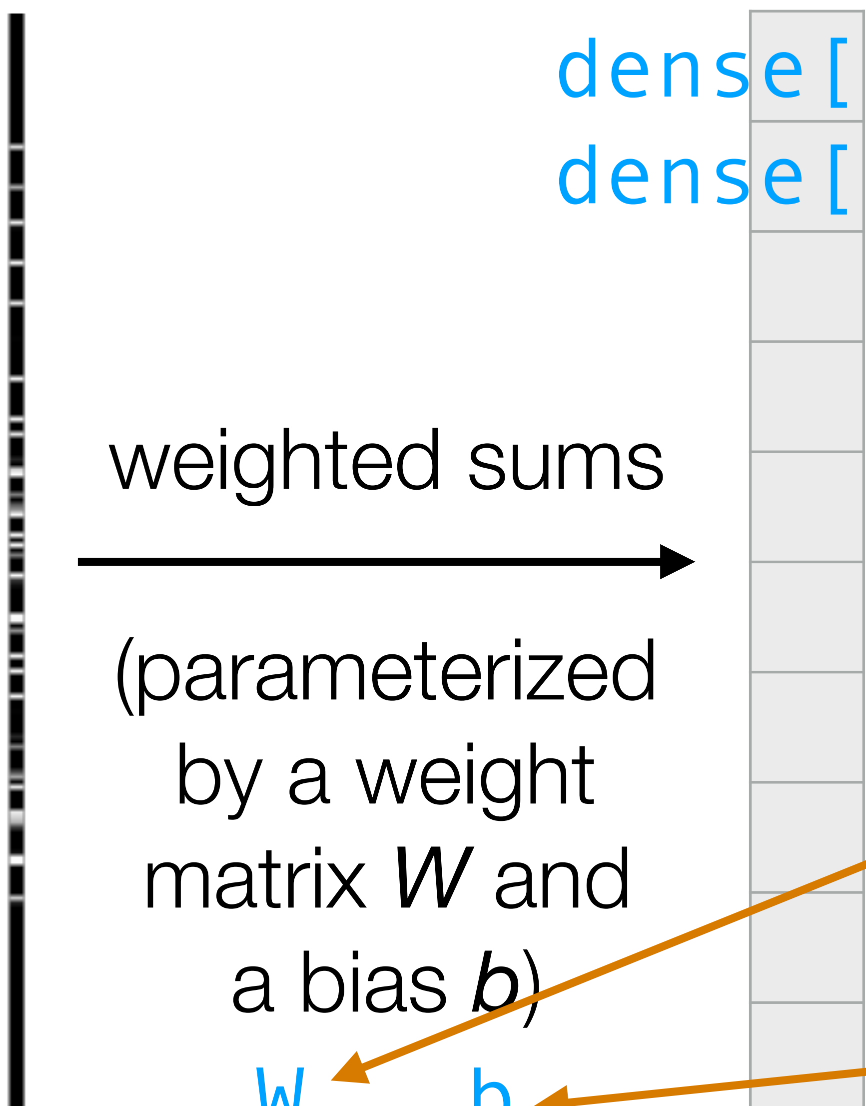
“dense” layer
with 10 numbers

`input`
34 entries)

`dense`
(1D numpy array with 10 entries)

Handwritten Digit Recognition

```
dense[0] = np.dot(input, W[:, 0]) + b[0]
dense[1] = np.dot(input, W[:, 1]) + b[1]
```



weighted sums

(parameterized by a weight matrix *W* and a bias *b*)

`W` `b`

(2D numpy array of dimensions 784-by-10)

(1D numpy array with 10 entries)

n 784 vector
nput neurons)

“dense” layer
with 10 numbers

`input`
84 entries)

`dense`
(1D numpy array with 10 entries)

Handwritten Digit Recognition

```
dense[0] = np.dot(input, W[:, 0]) + b[0]
dense[1] = np.dot(input, W[:, 1]) + b[1]
```

⋮

weighted sums

(parameterized by a weight matrix *W* and a bias *b*)

`W` `b`

(2D numpy array of dimensions 784-by-10)

(1D numpy array with 10 entries)

n 784 vector
nput neurons)

`input`

34 entries)

"dense" layer
with 10 numbers

`dense`

(1D numpy array with 10 entries)

Handwritten Digit Recognition

```
dense[0] = np.dot(input, W[:, 0]) + b[0]
dense[1] = np.dot(input, W[:, 1]) + b[1]
```

⋮

$$\texttt{dense[j]} = \sum_{i=0}^{783} \texttt{input[i]} \times \texttt{W[i, j]} + \texttt{b[j]}$$

weighted sums

(parameterized by a weight matrix *W* and a bias *b*)

`W` `b`

(2D numpy array of dimensions 784-by-10)

(1D numpy array with 10 entries)

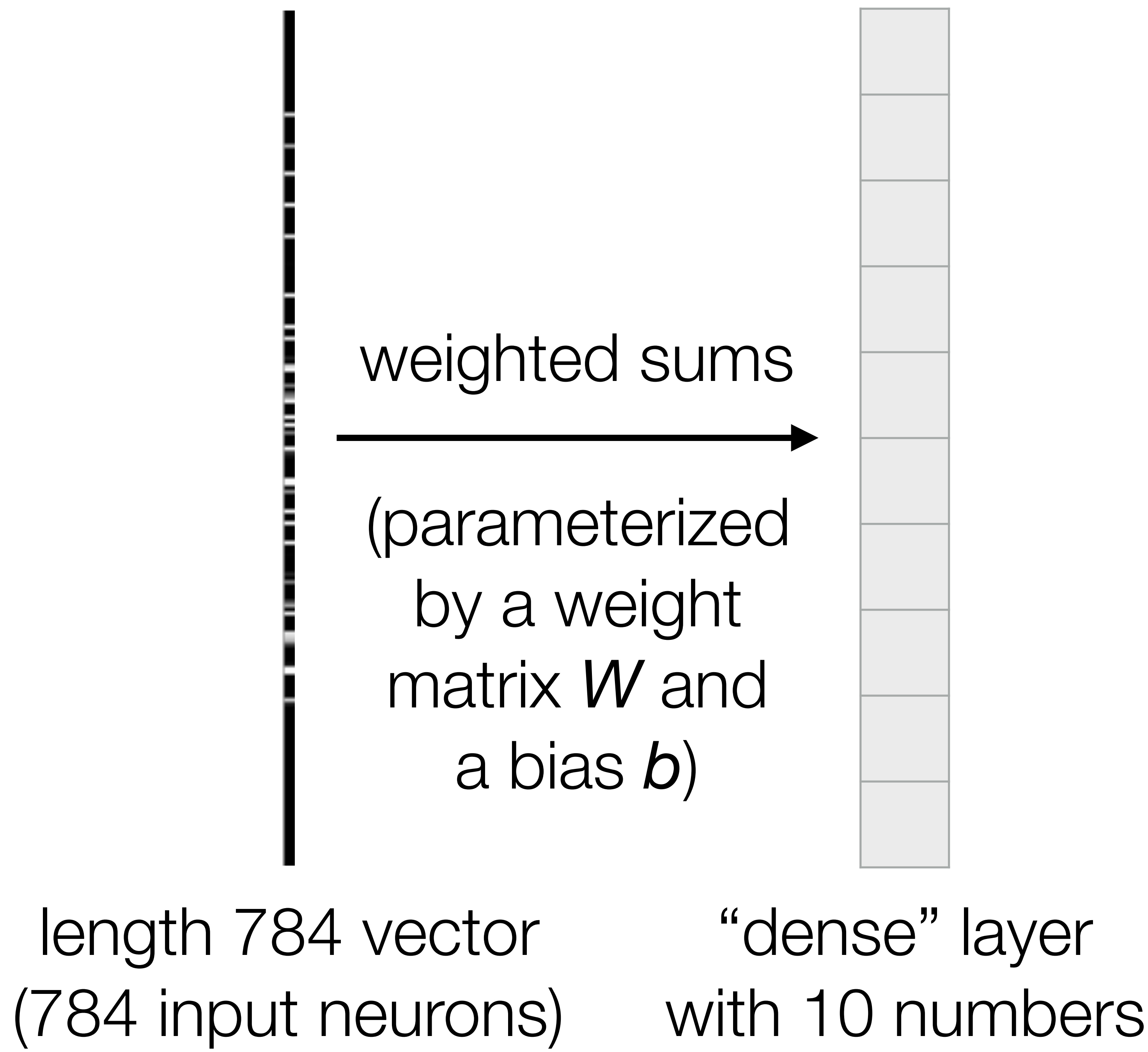
n 784 vector
nput neurons)

“dense” layer
with 10 numbers

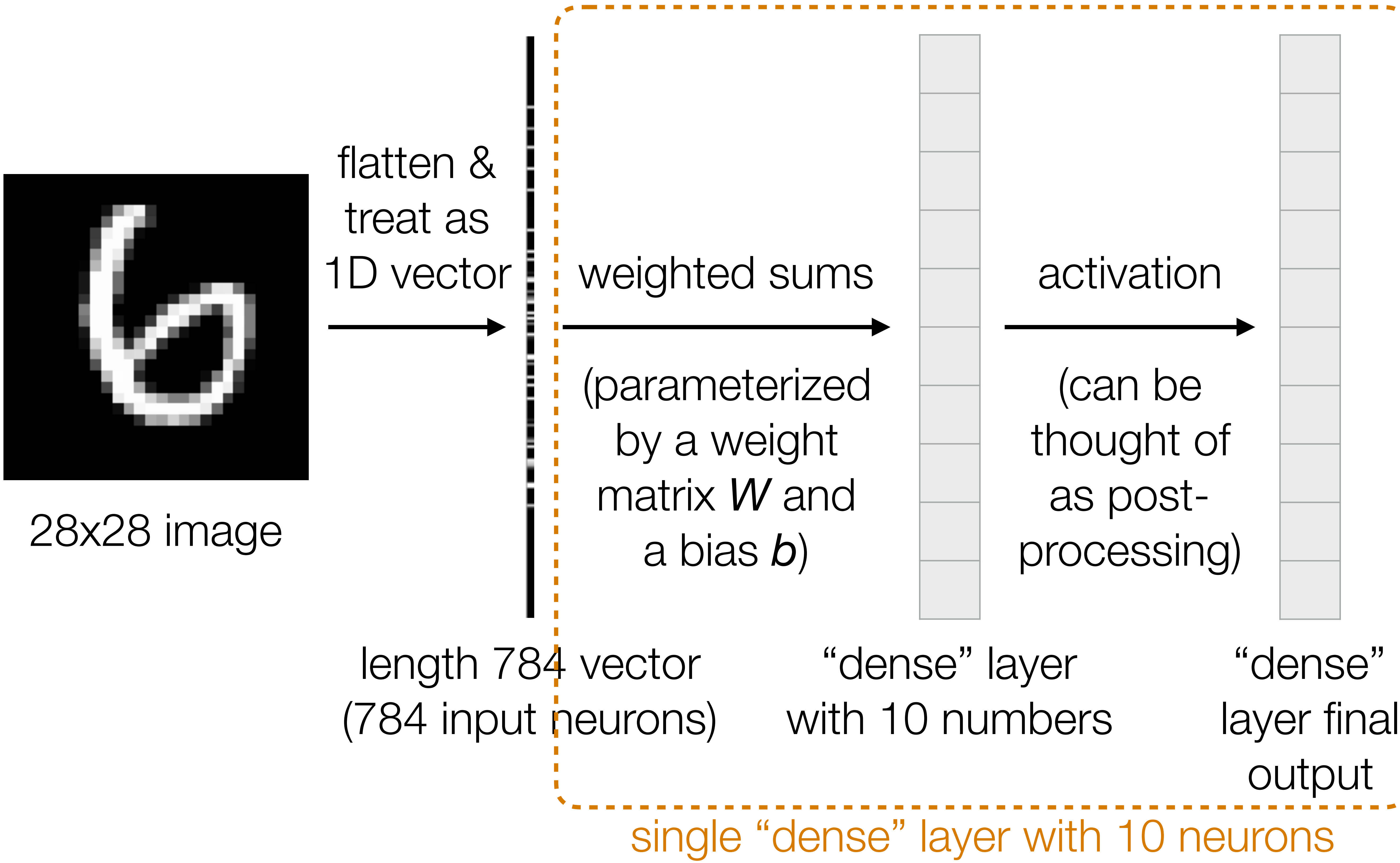
`input`
34 entries)

`dense`
(1D numpy array with 10 entries)

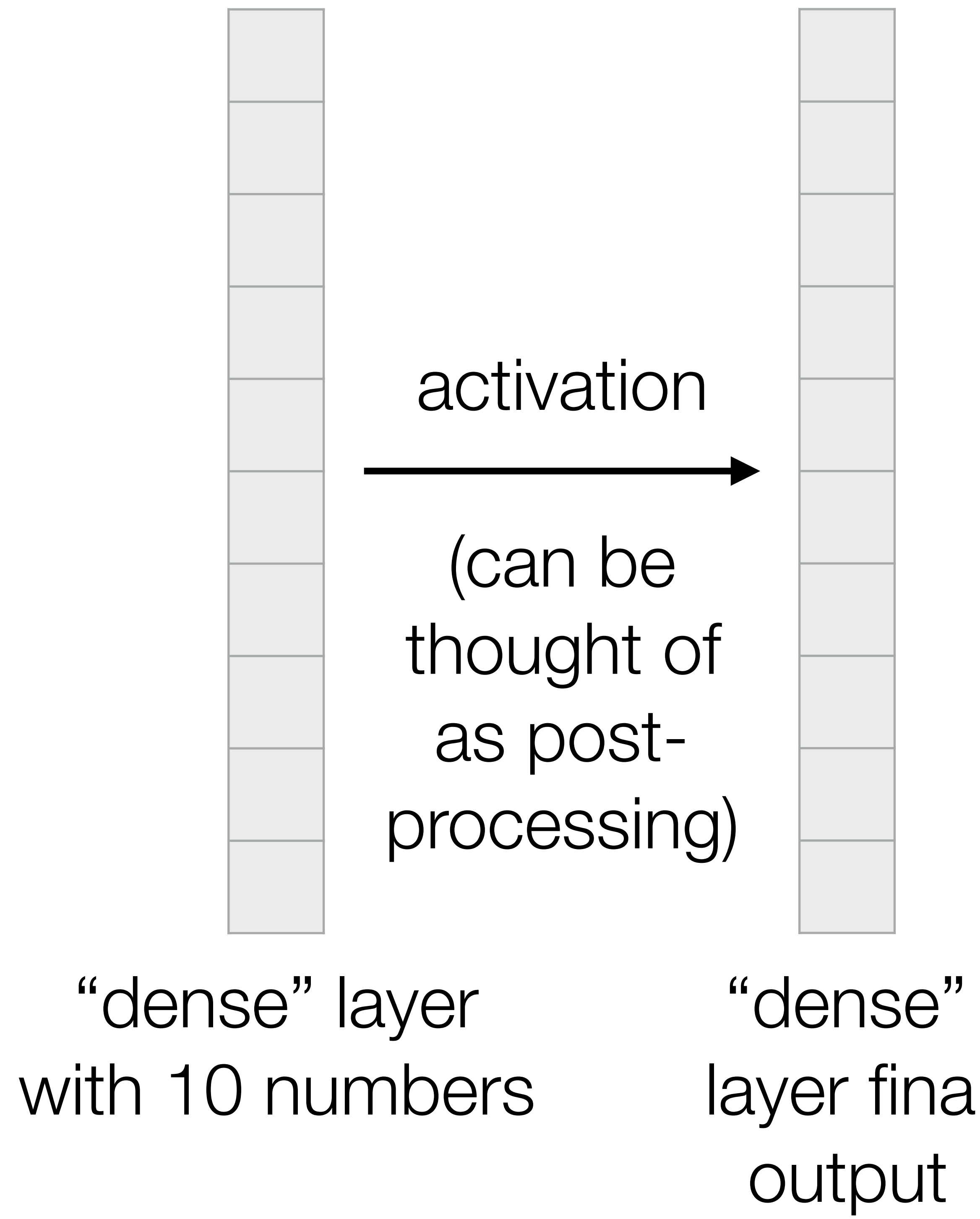
Handwritten Digit Recognition



Handwritten Digit Recognition

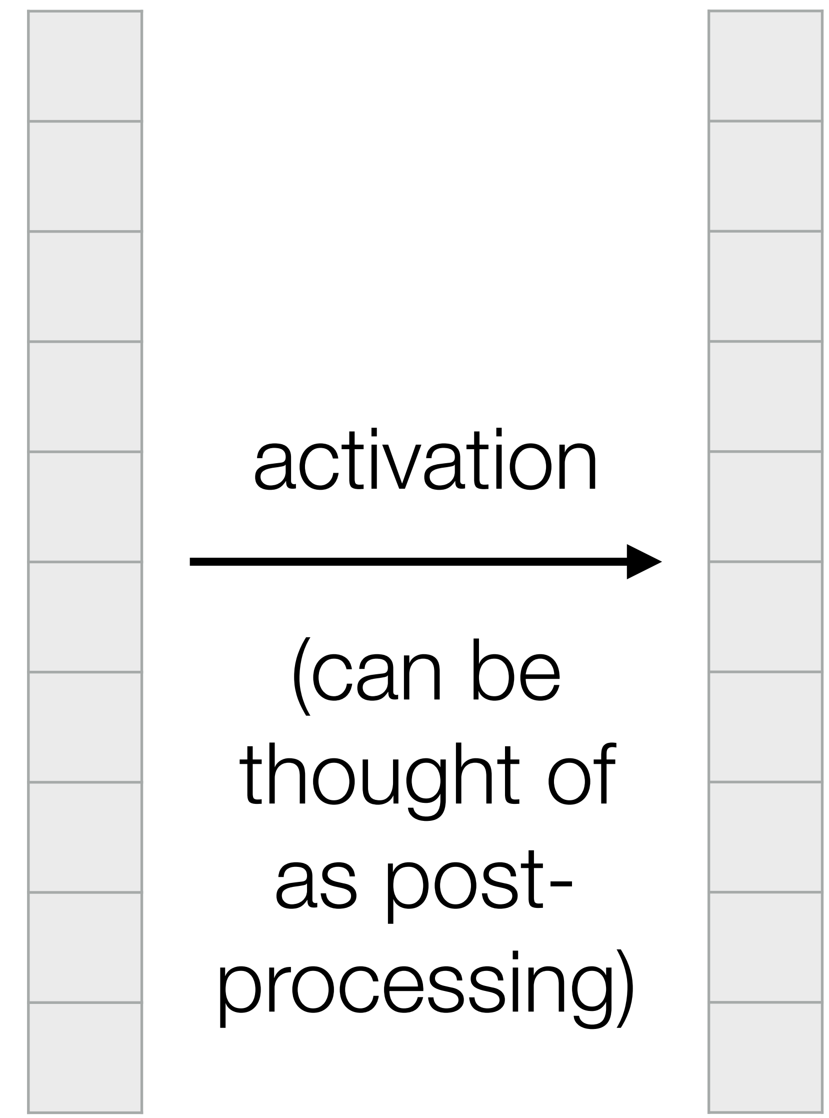


Handwritten Digit Recognition



Handwritten Digit Recognition

Many different activation functions possible



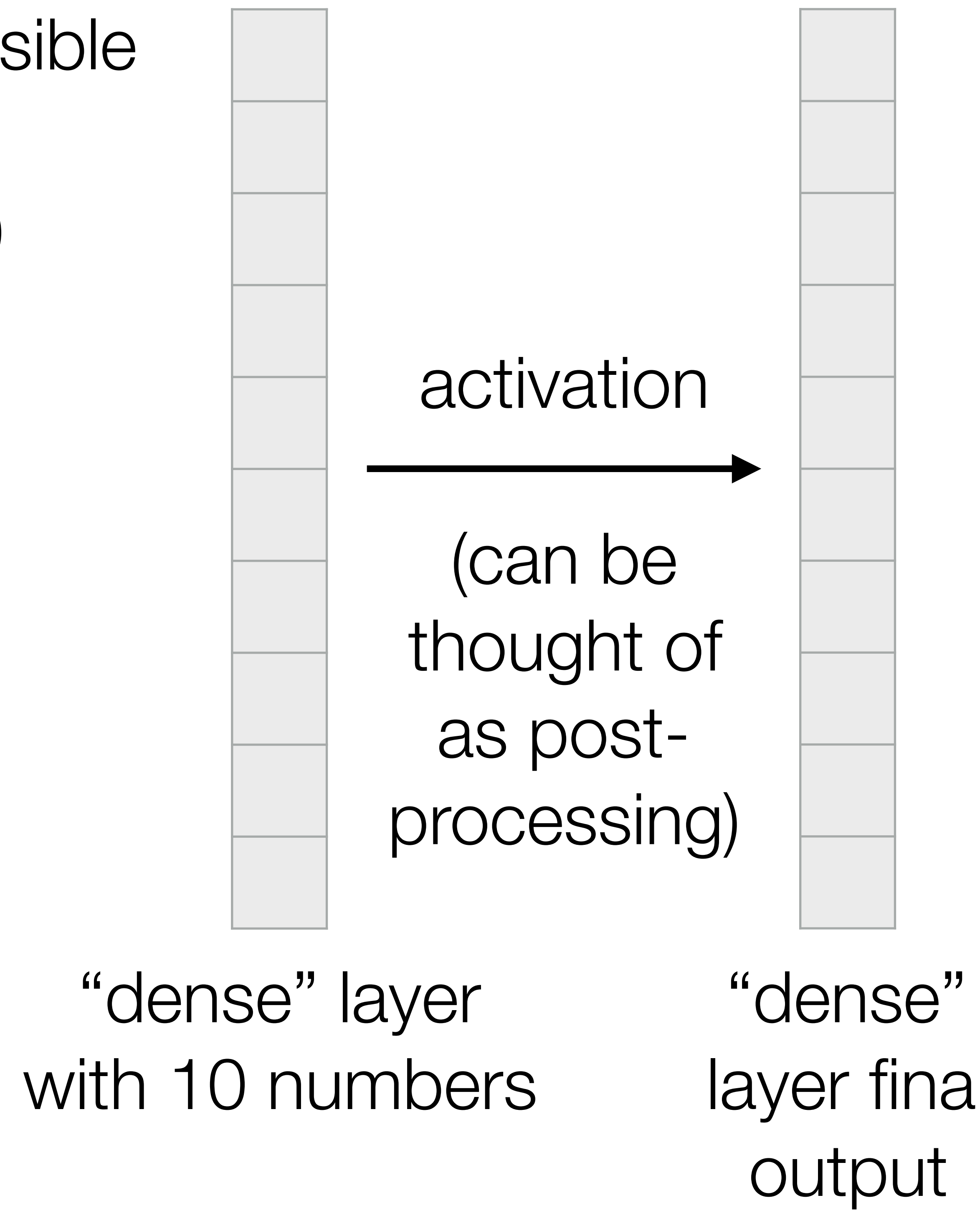
"dense" layer with 10 numbers

"dense" layer final output

Handwritten Digit Recognition

Many different activation functions possible

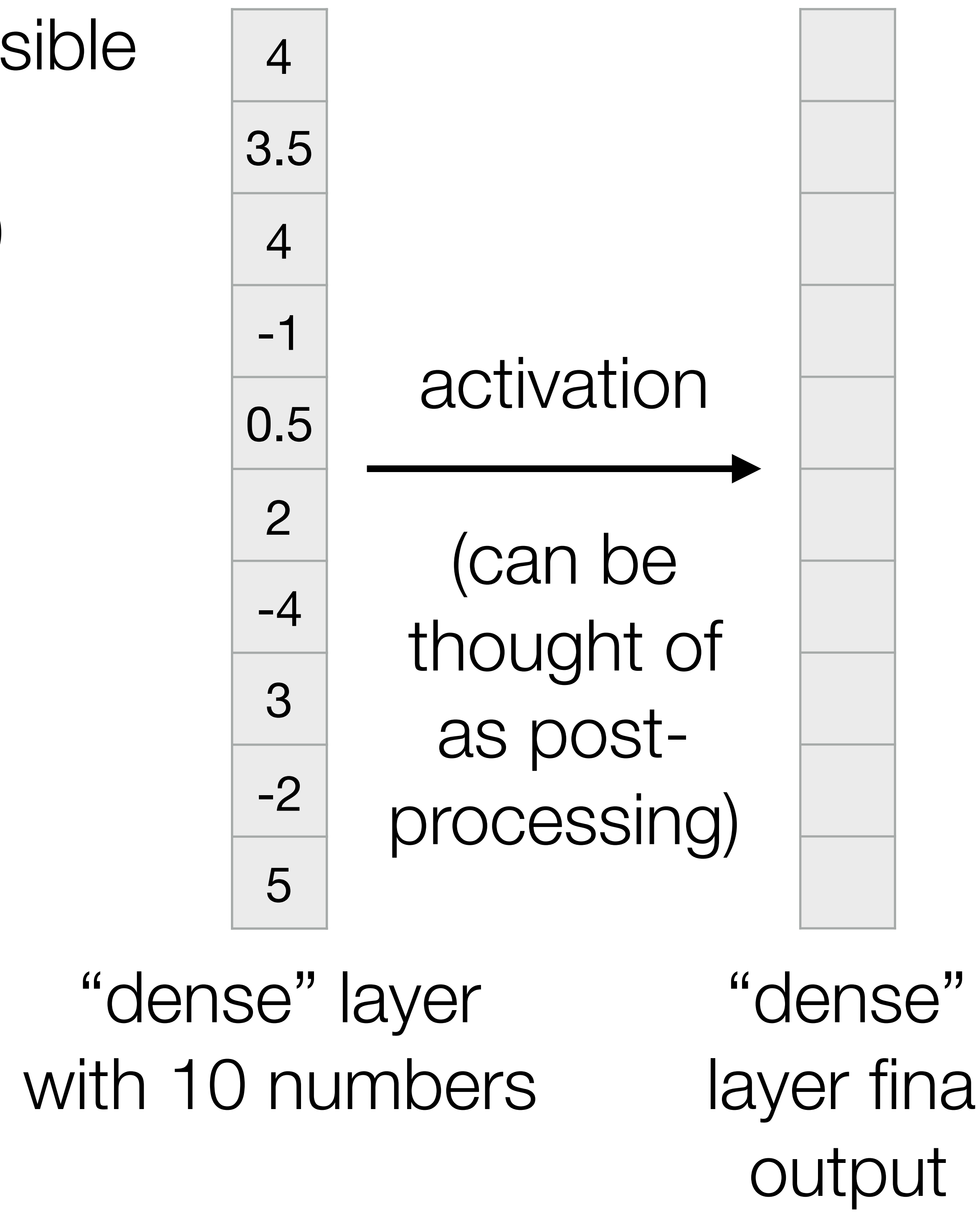
Example: **Rectified linear unit (ReLU)**
zeros out entries that are negative



Handwritten Digit Recognition

Many different activation functions possible

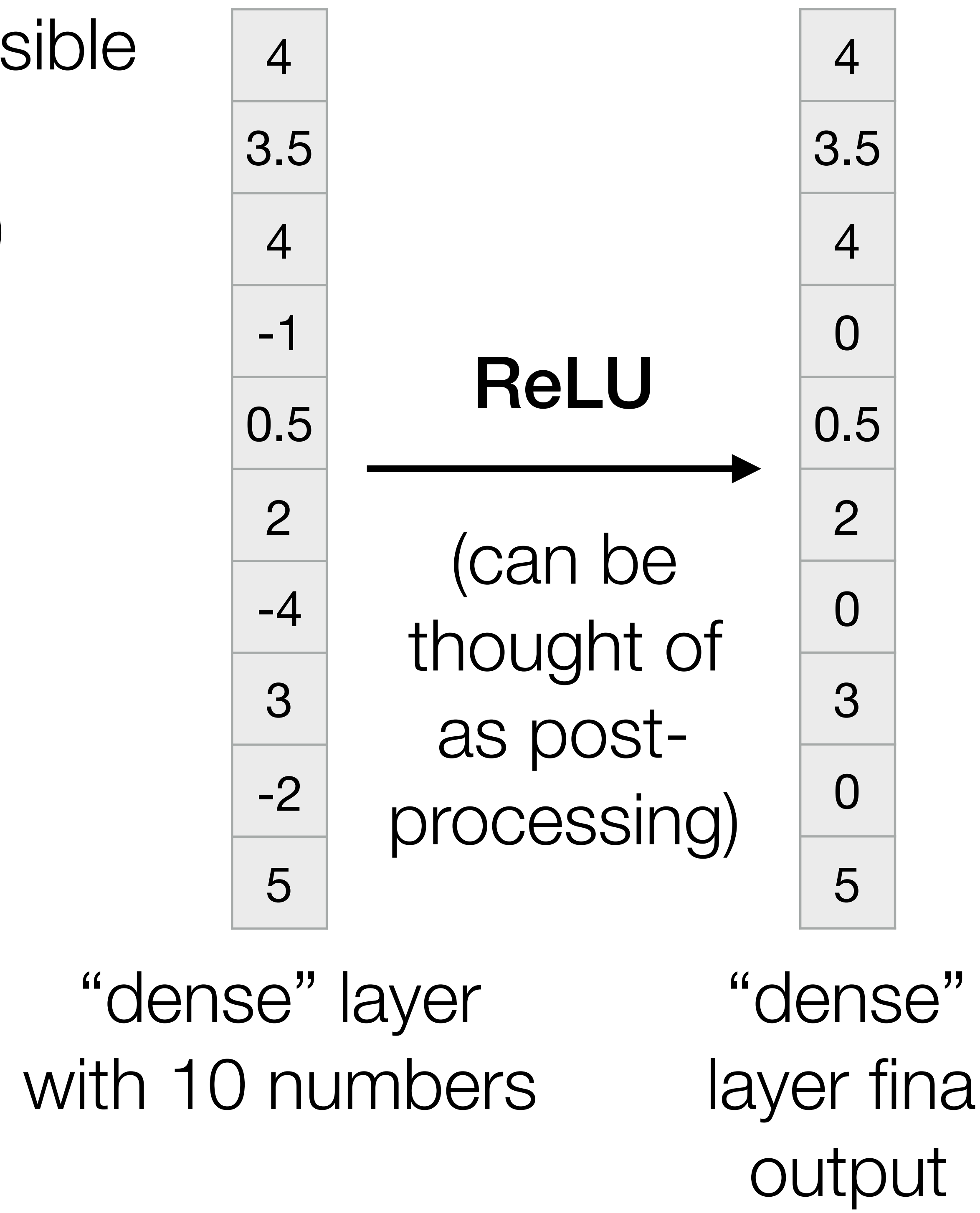
Example: **Rectified linear unit (ReLU)** zeros out entries that are negative



Handwritten Digit Recognition

Many different activation functions possible

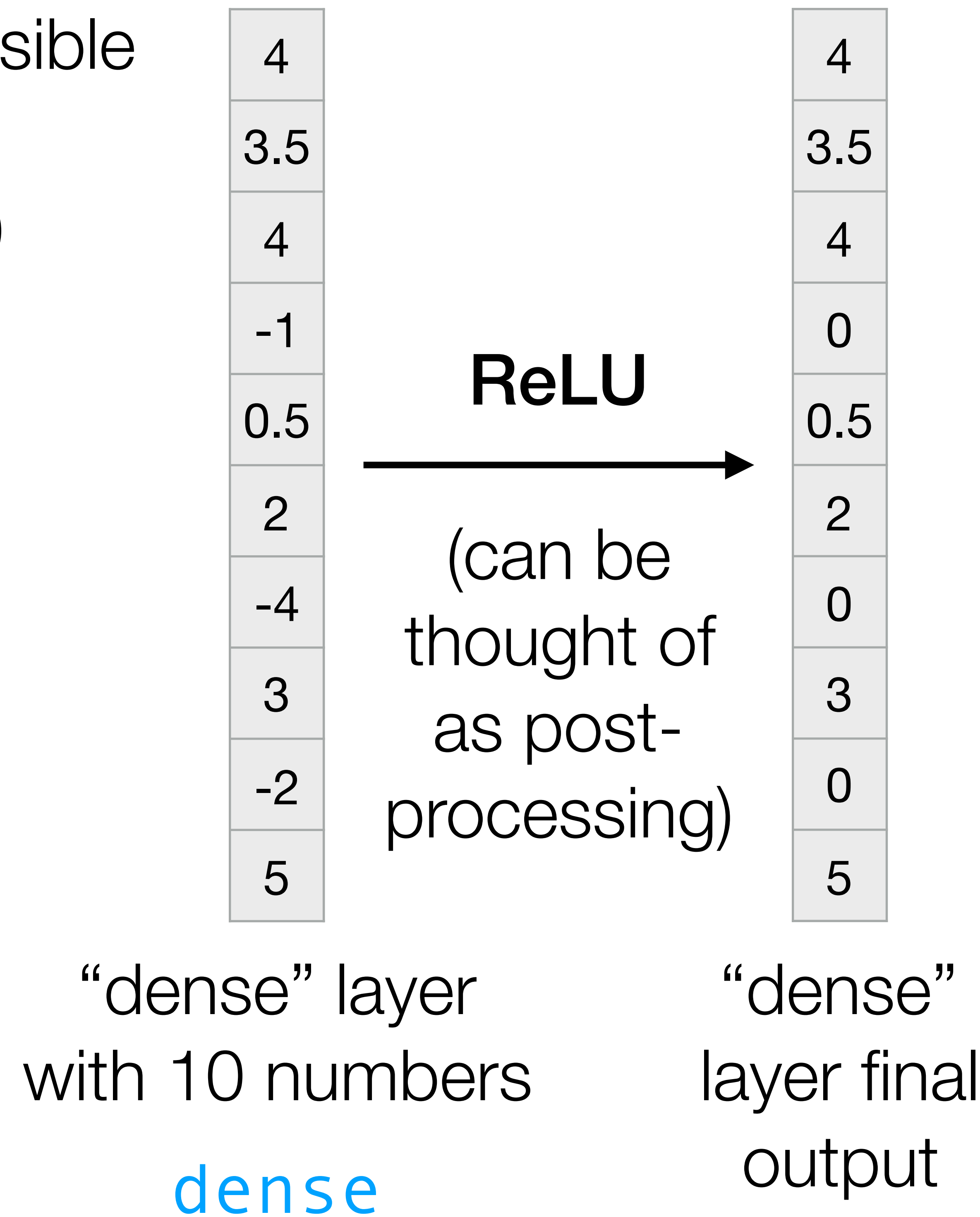
Example: **Rectified linear unit (ReLU)** zeros out entries that are negative



Handwritten Digit Recognition

Many different activation functions possible

Example: **Rectified linear unit (ReLU)** zeros out entries that are negative



Handwritten Digit Recognition

Many different activation functions possible

Example: **Rectified linear unit (ReLU)** zeros out entries that are negative

4		4
3.5		3.5
4		4
-1		0
0.5		0.5
2		2
-4		0
3		3
-2		0
5		5

ReLU

(can be thought of as post-processing)

“dense” layer with 10 numbers

`dense`

“dense” layer final output

`dense_final`

Handwritten Digit Recognition

Many different activation functions possible

Example: **Rectified linear unit (ReLU)** zeros out entries that are negative

```
dense_final = np.maximum(0, dense)
```

"dense" layer with 10 numbers (`dense`)	"dense" layer final output (`dense_final`)
4	4
3.5	3.5
4	4
-1	0
0.5	0.5
2	2
-4	0
3	3
-2	0
5	5

$\xrightarrow{\textbf{ReLU}}$

(can be thought of as post-processing)

"dense" layer with 10 numbers

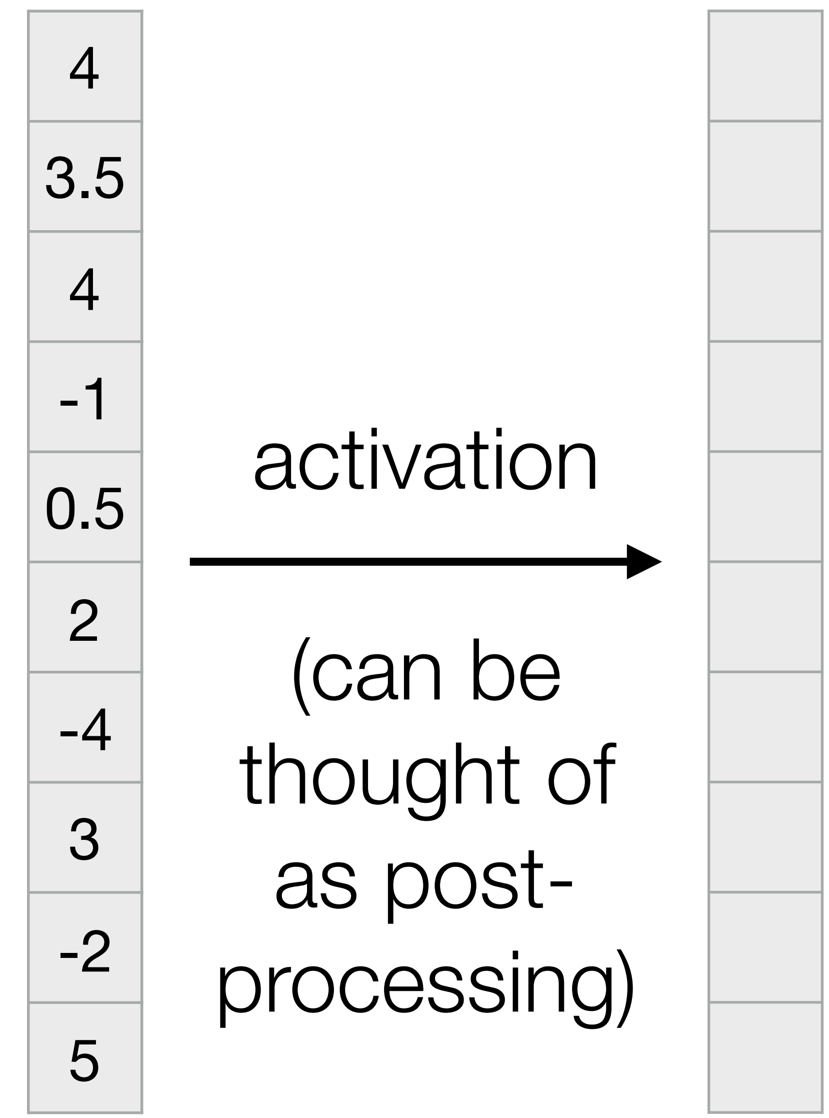
`dense`

"dense" layer final output

`dense_final`

Handwritten Digit Recognition

Many different activation functions possible



“dense” layer with 10 numbers

`dense`

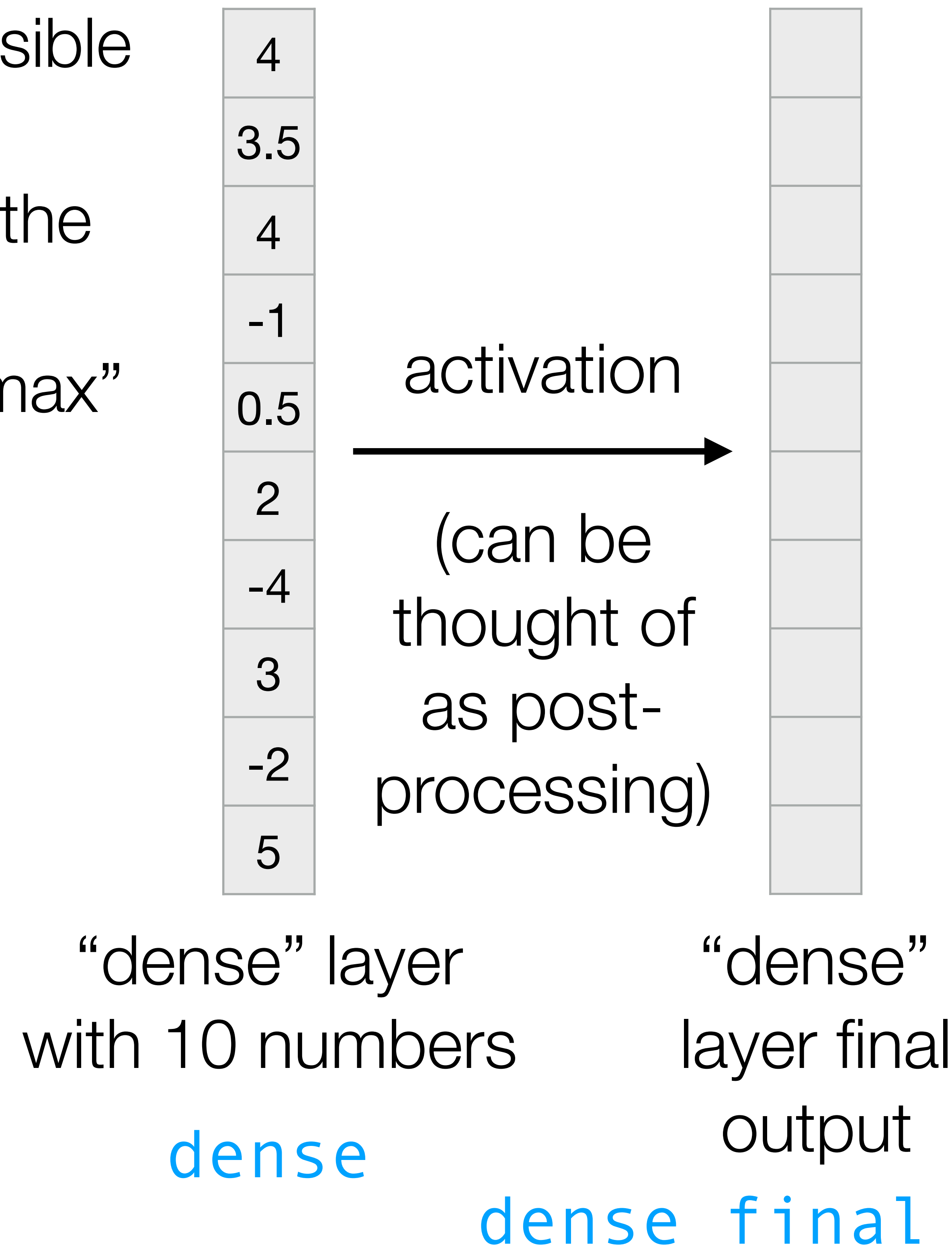
“dense” layer final output

`dense_final`

Handwritten Digit Recognition

Many different activation functions possible

Example: **softmax** turns the entries in the dense layer (prior to activation) into a probability distribution (using the “softmax” transformation)

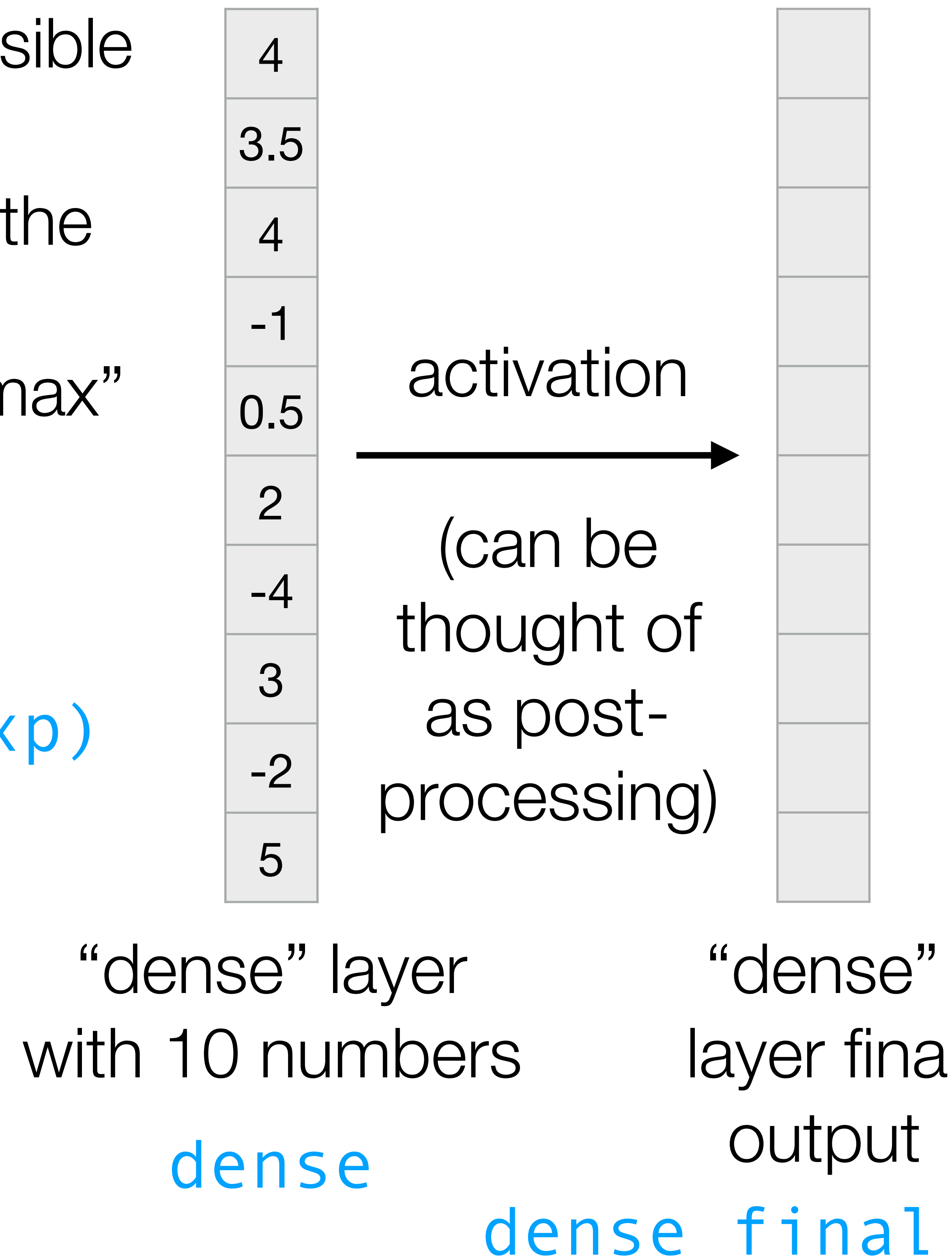


Handwritten Digit Recognition

Many different activation functions possible

Example: **softmax** turns the entries in the dense layer (prior to activation) into a probability distribution (using the "softmax" transformation)

```
dense_exp = np.exp(dense)
dense_exp /= np.sum(dense_exp)
dense_final = dense_exp
```

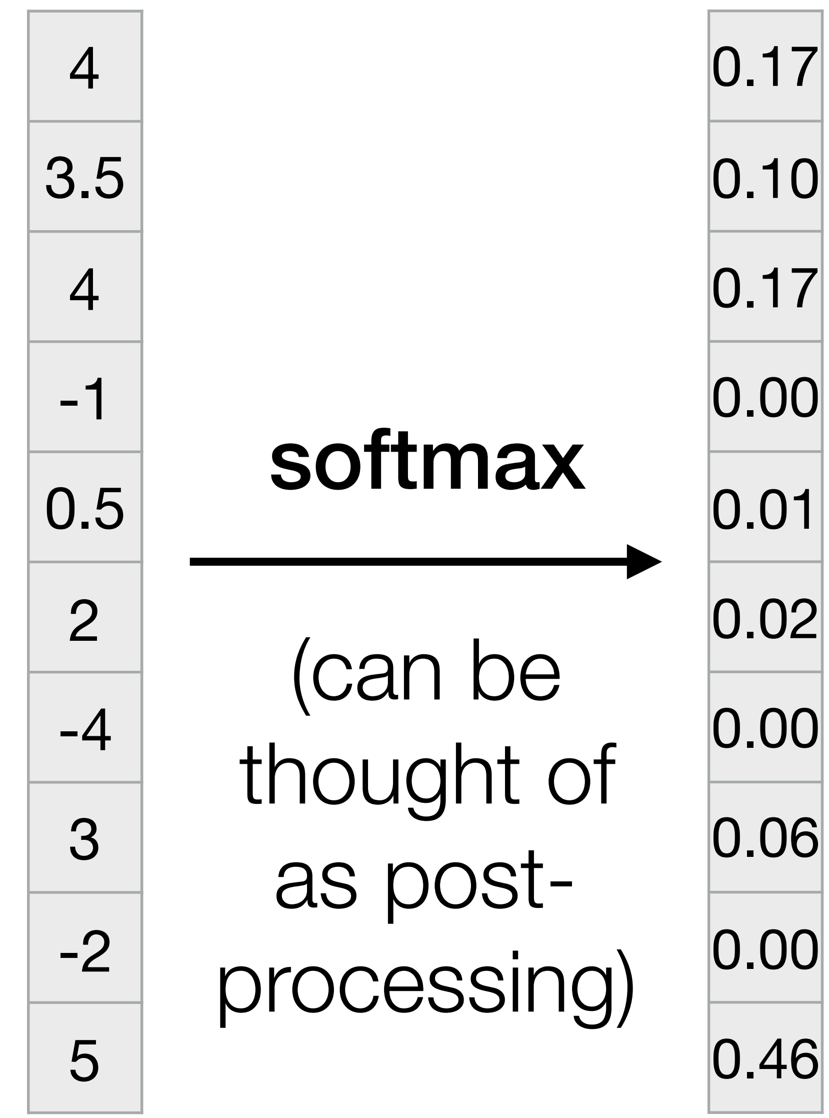


Handwritten Digit Recognition

Many different activation functions possible

Example: **softmax** turns the entries in the dense layer (prior to activation) into a probability distribution (using the “softmax” transformation)

```
dense_exp = np.exp(dense)
dense_exp /= np.sum(dense_exp)
dense_final = dense_exp
```



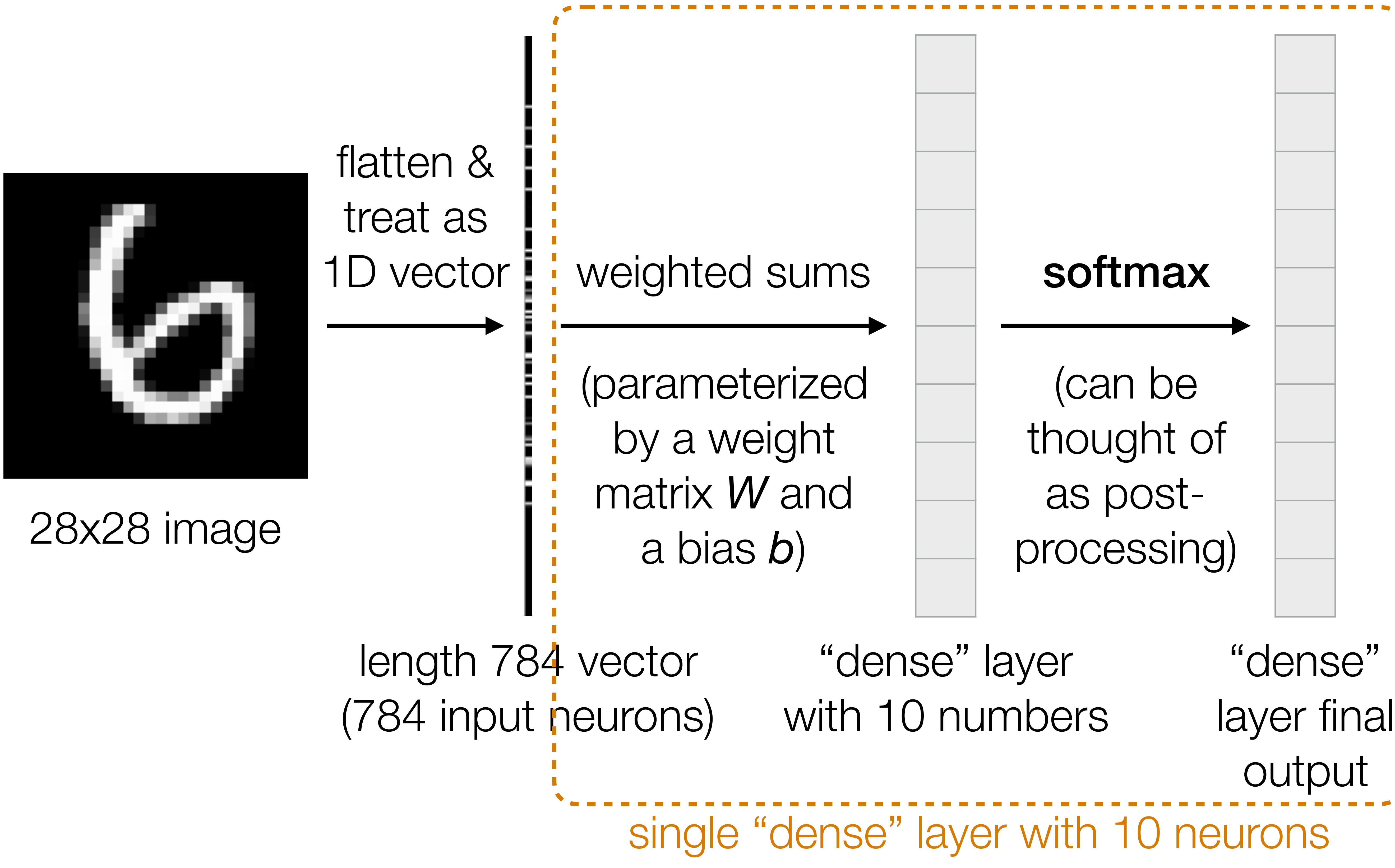
“dense” layer with 10 numbers

`dense`

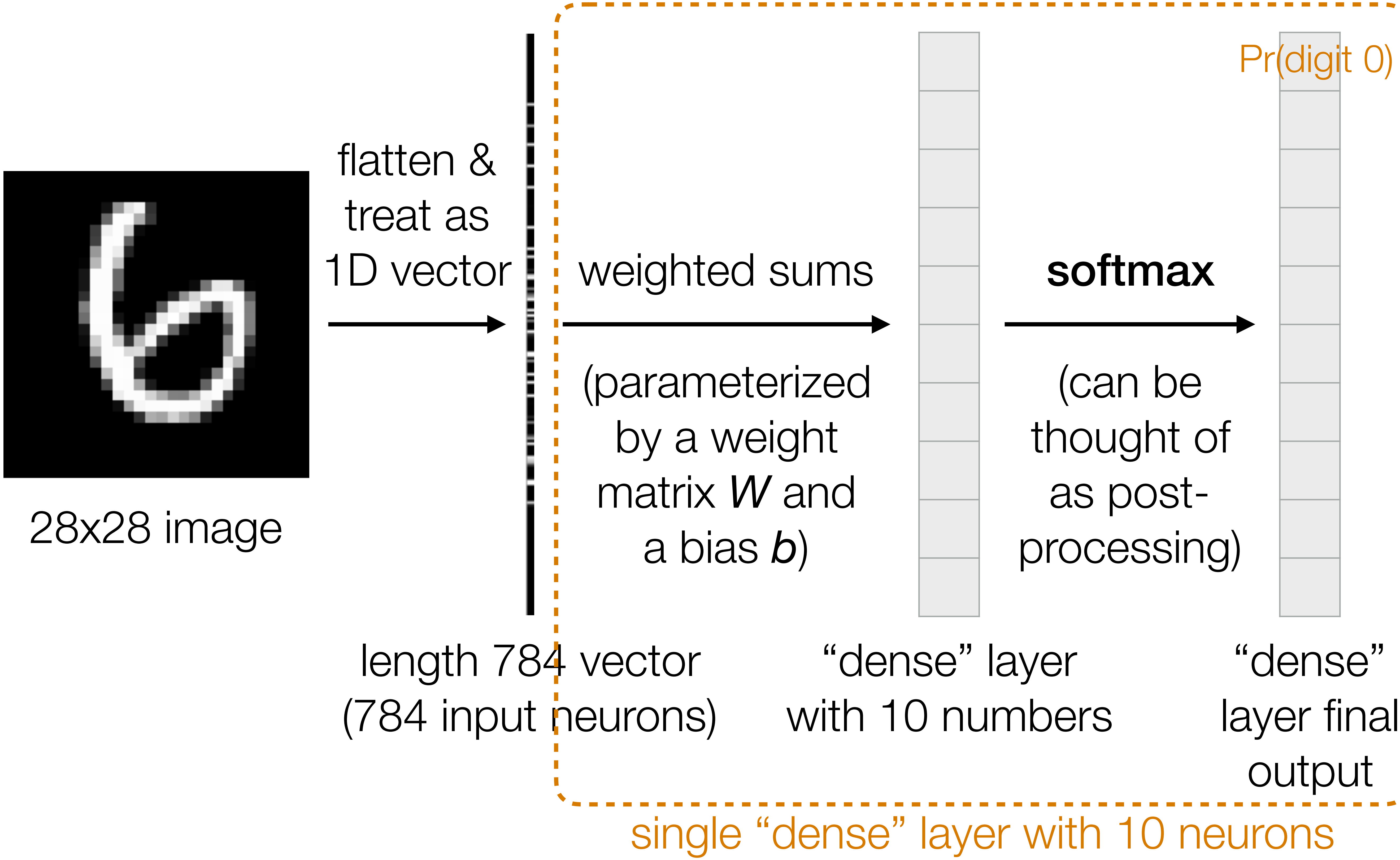
“dense” layer final output

`dense_final`

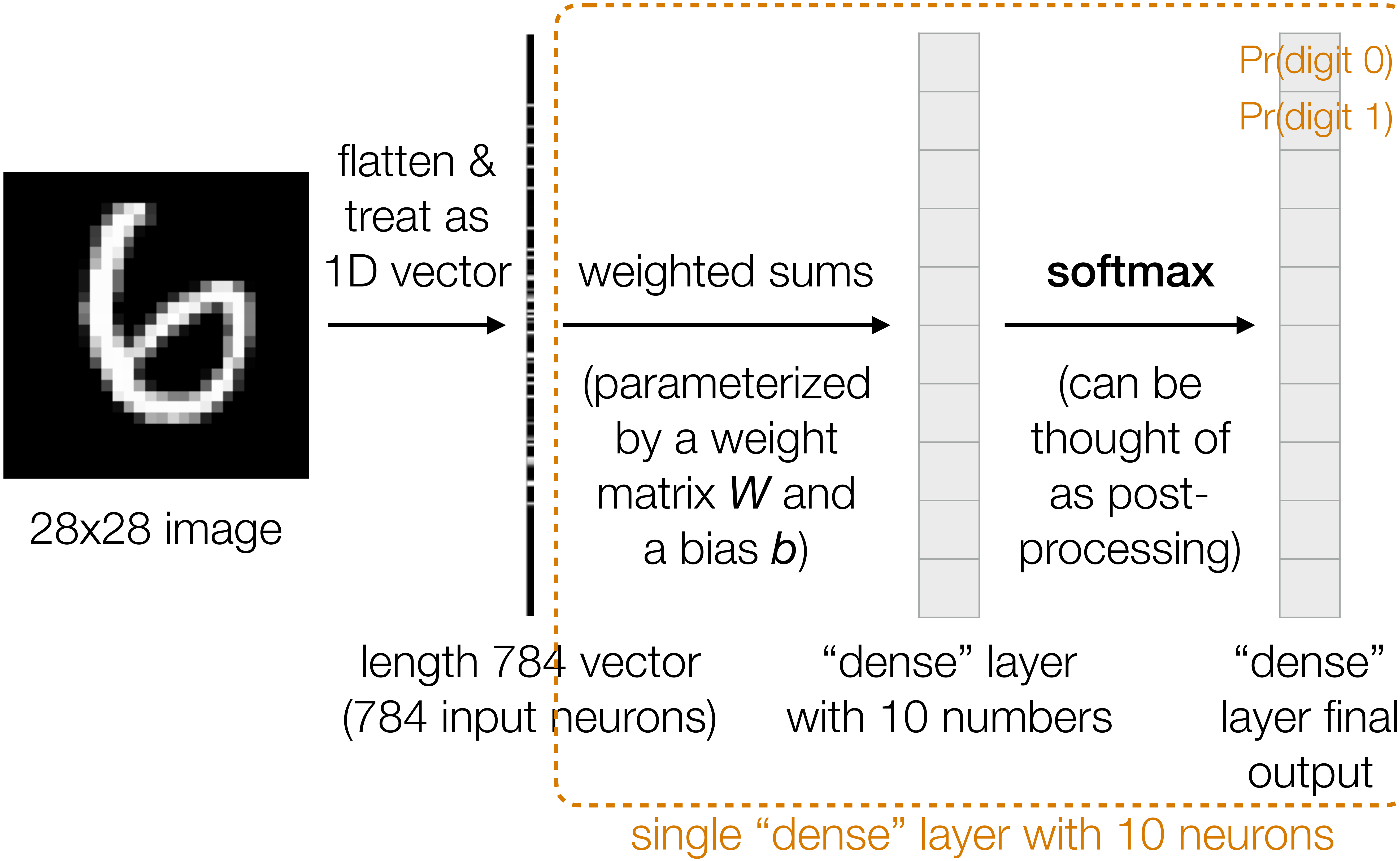
Handwritten Digit Recognition



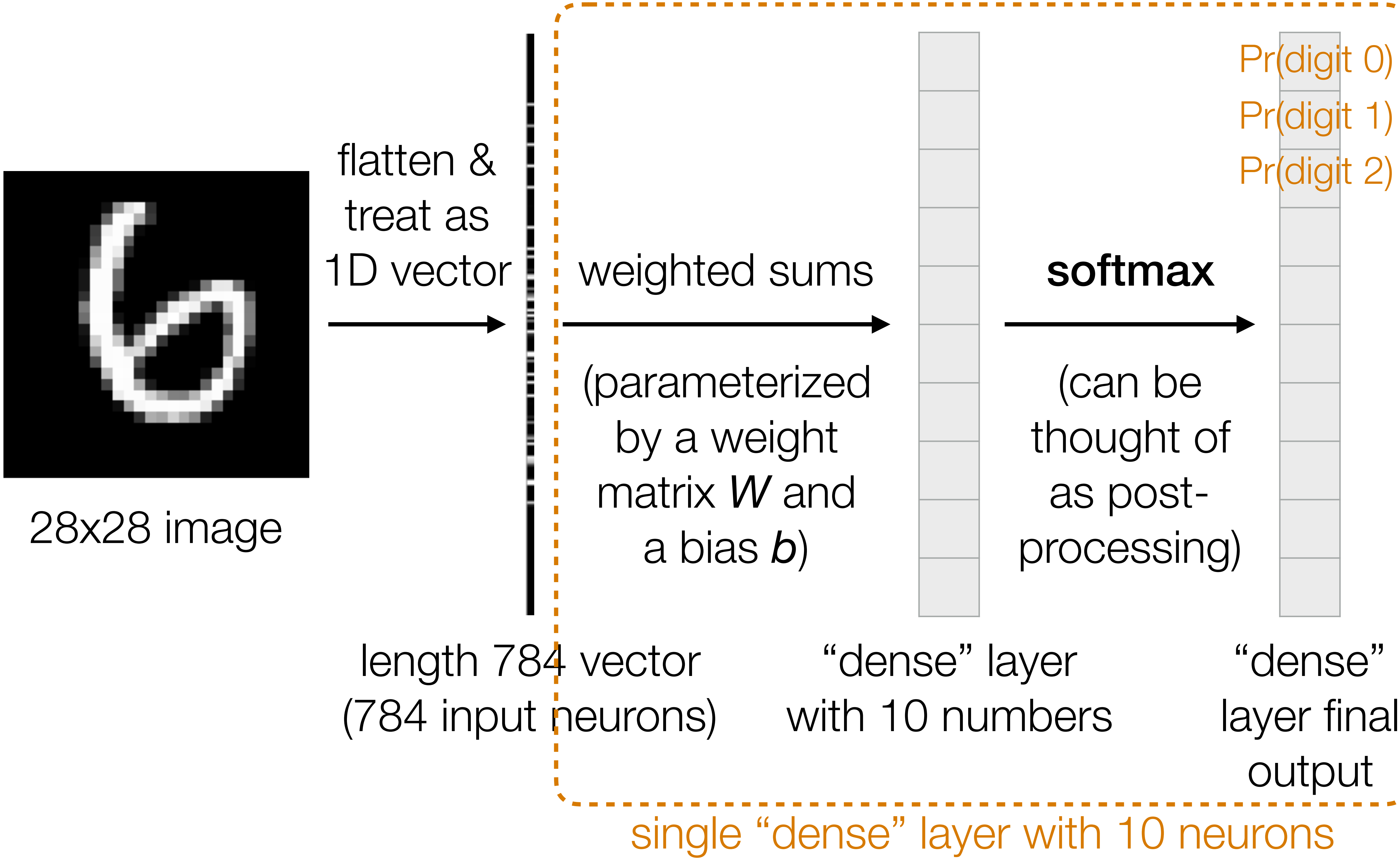
Handwritten Digit Recognition



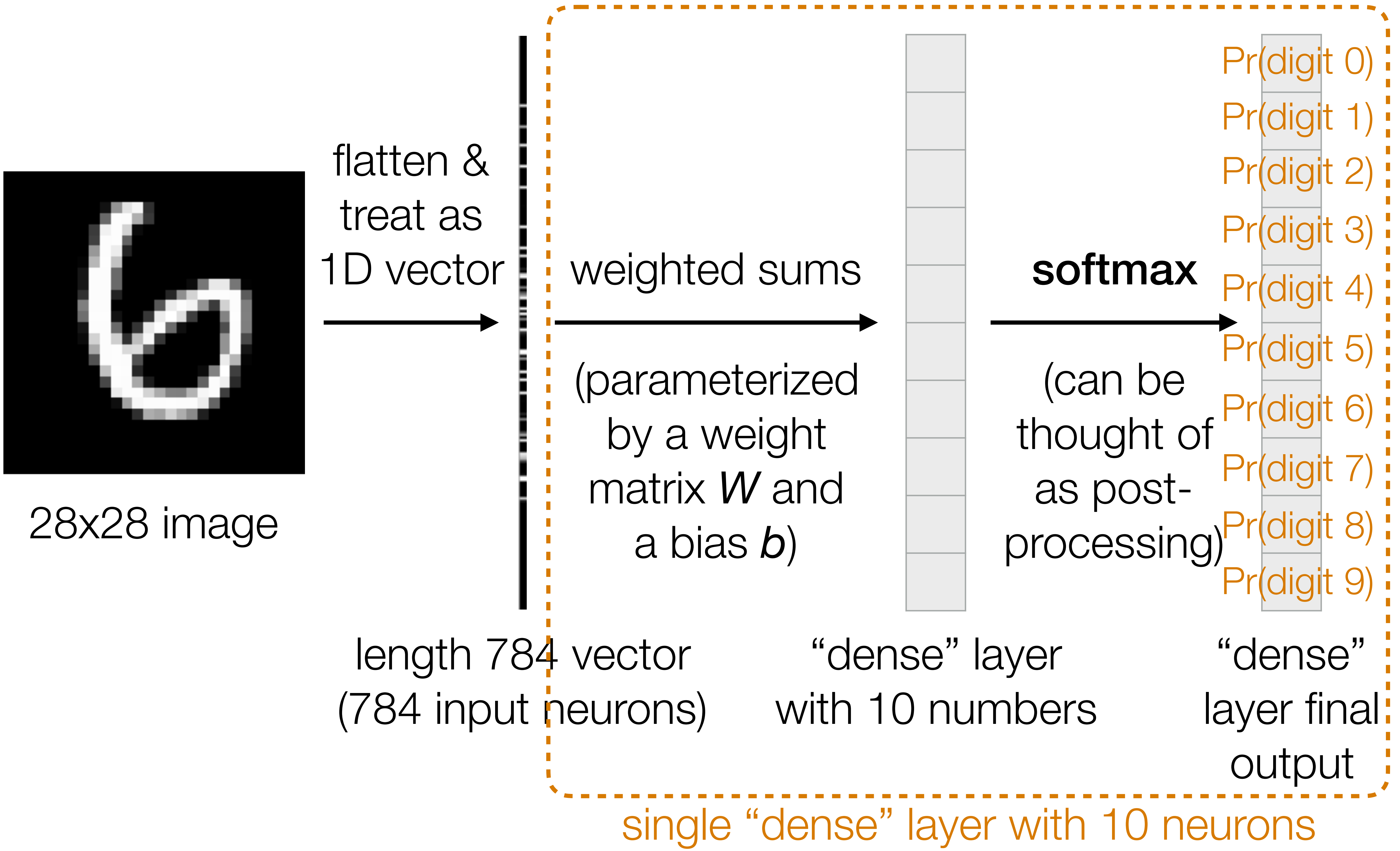
Handwritten Digit Recognition



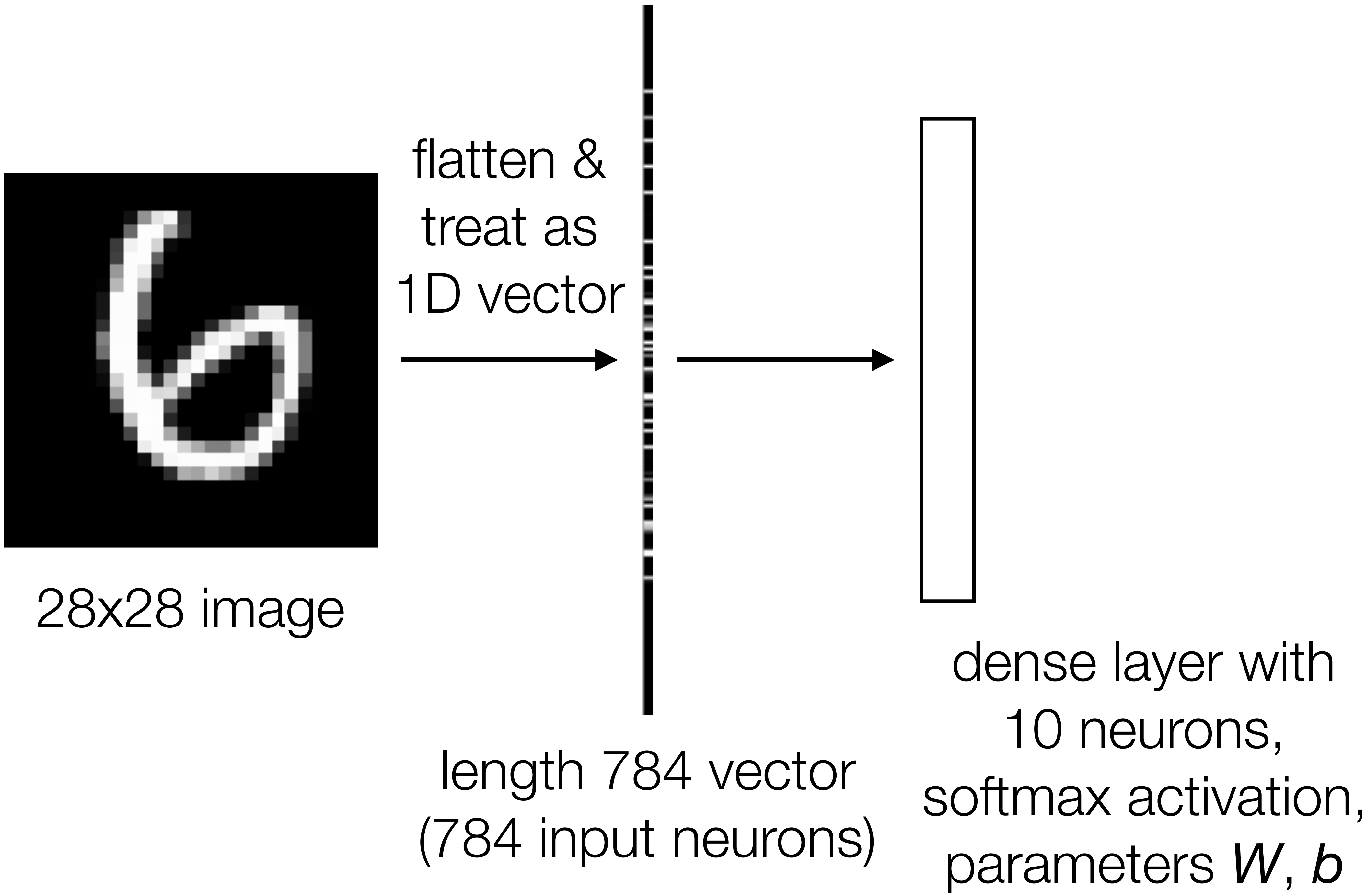
Handwritten Digit Recognition



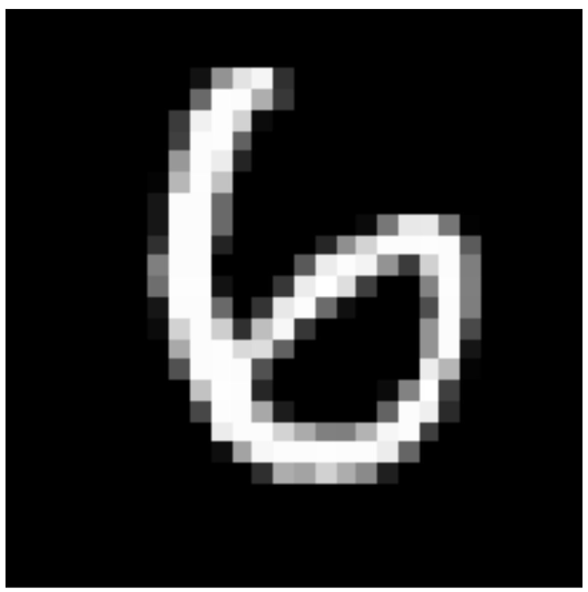
Handwritten Digit Recognition



Handwritten Digit Recognition



Handwritten Digit Recognition



28x28 image

flatten &
treat as
1D vector

length 784 vector
(784 input neurons)

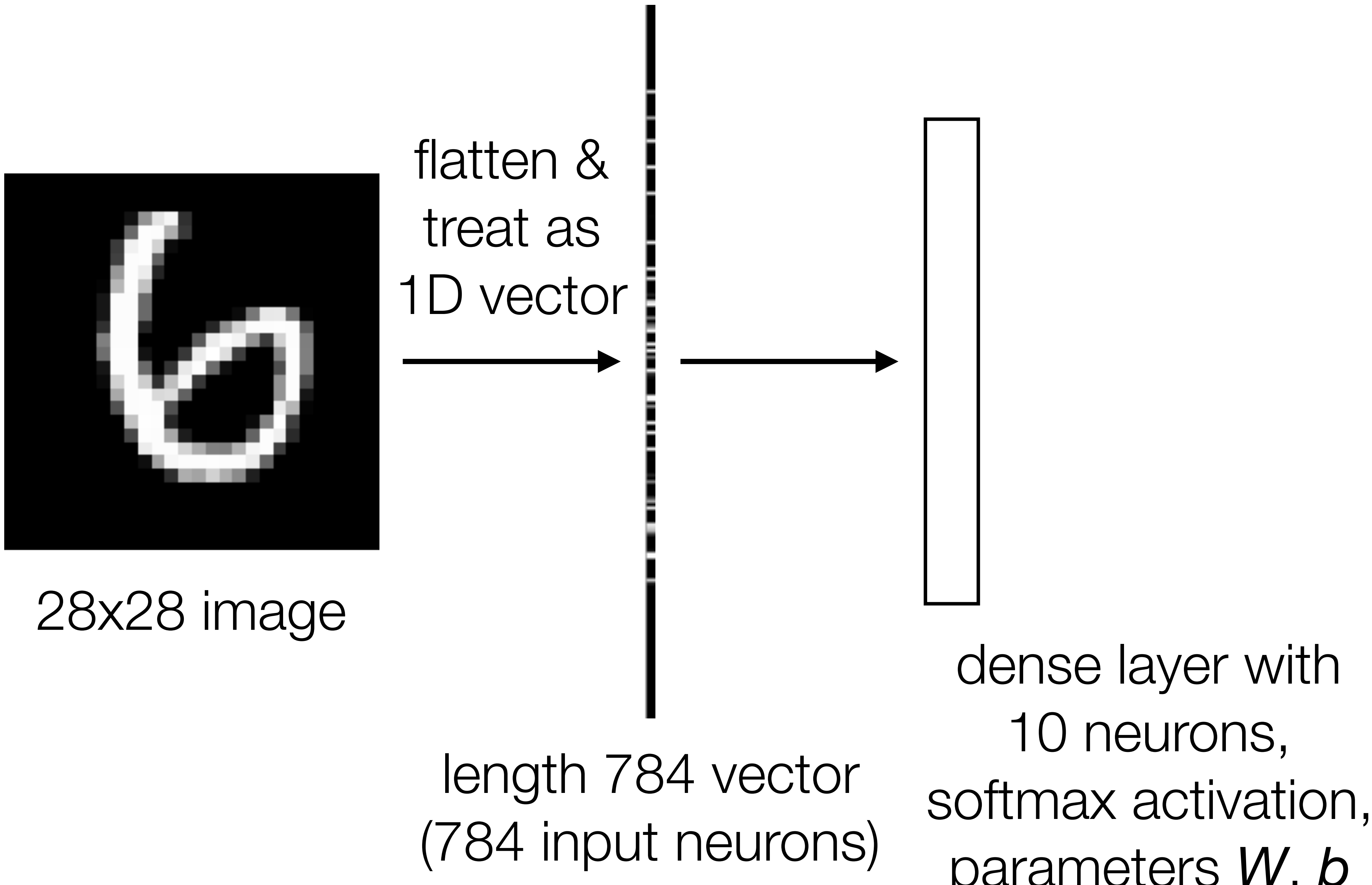
dense layer with
10 neurons,
softmax activation,
parameters *W*, *b*

We want the output of the dense layer to encode probabilities for whether the input image is a 0, 1, 2, …, 9 *but as of now we aren't providing any sort of information to enforce this*

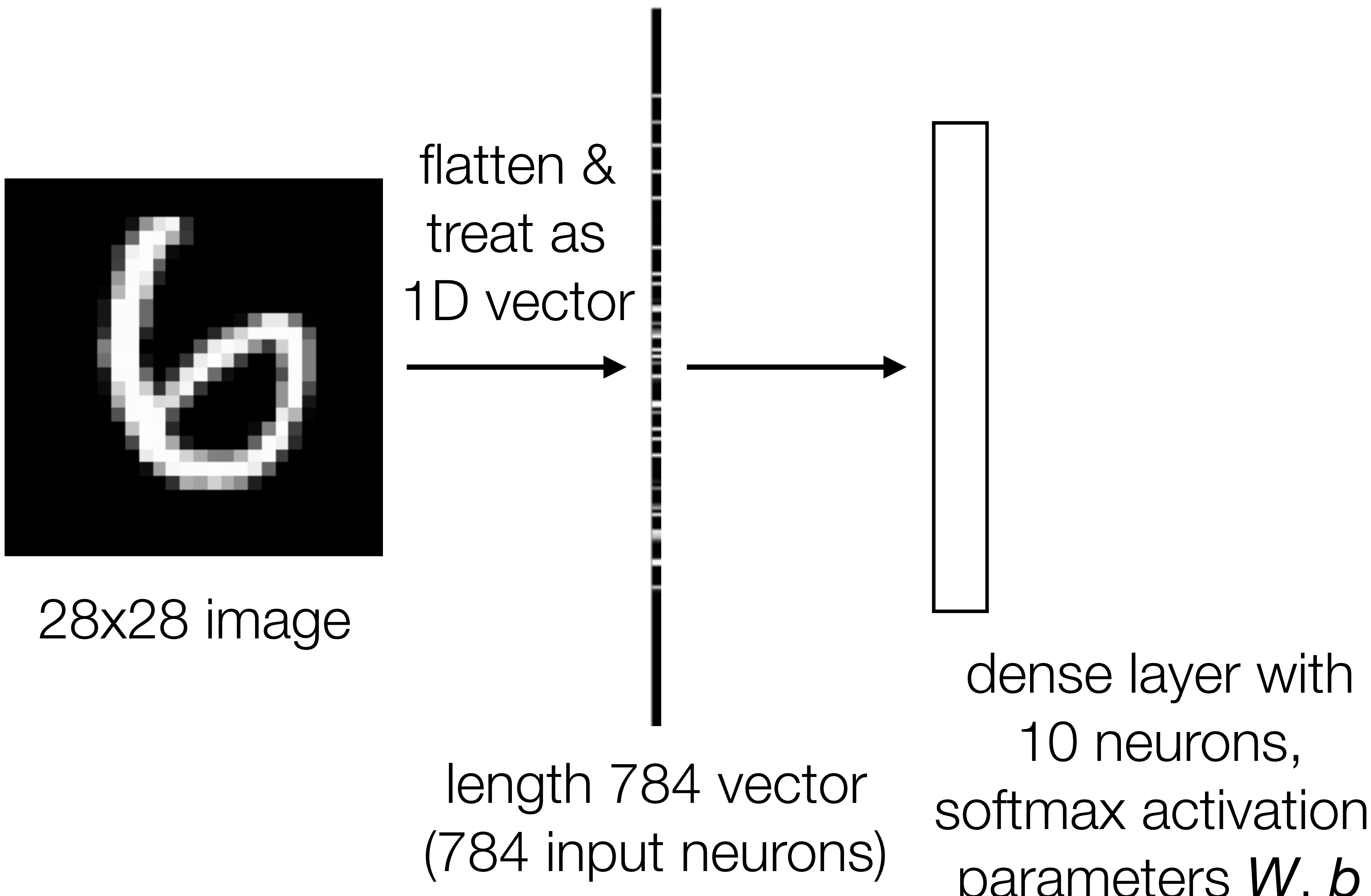
Handwritten Digit Recognition

Demo part 1

Handwritten Digit Recognition

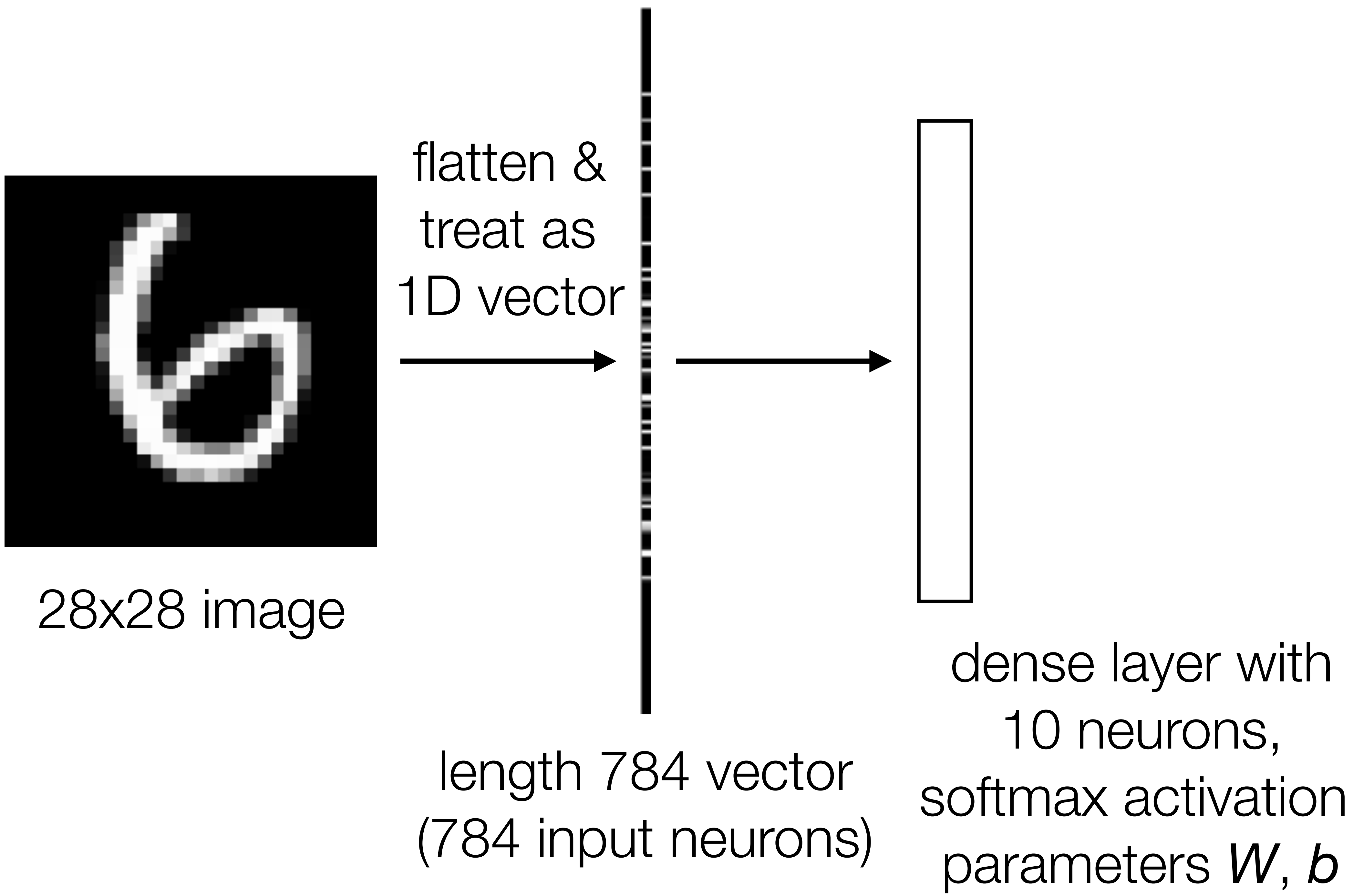


Handwritten Digit Recognition

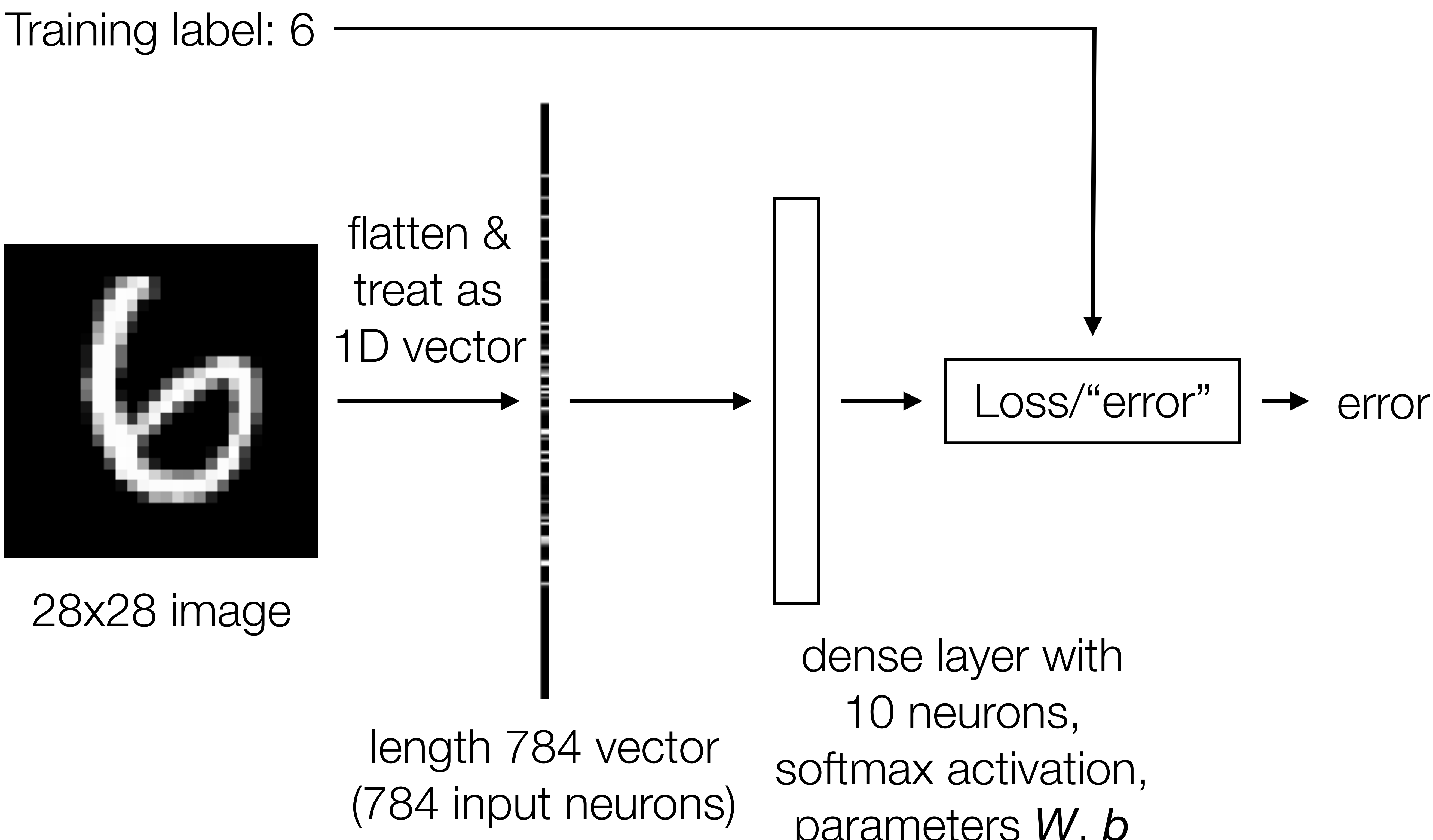


Handwritten Digit Recognition

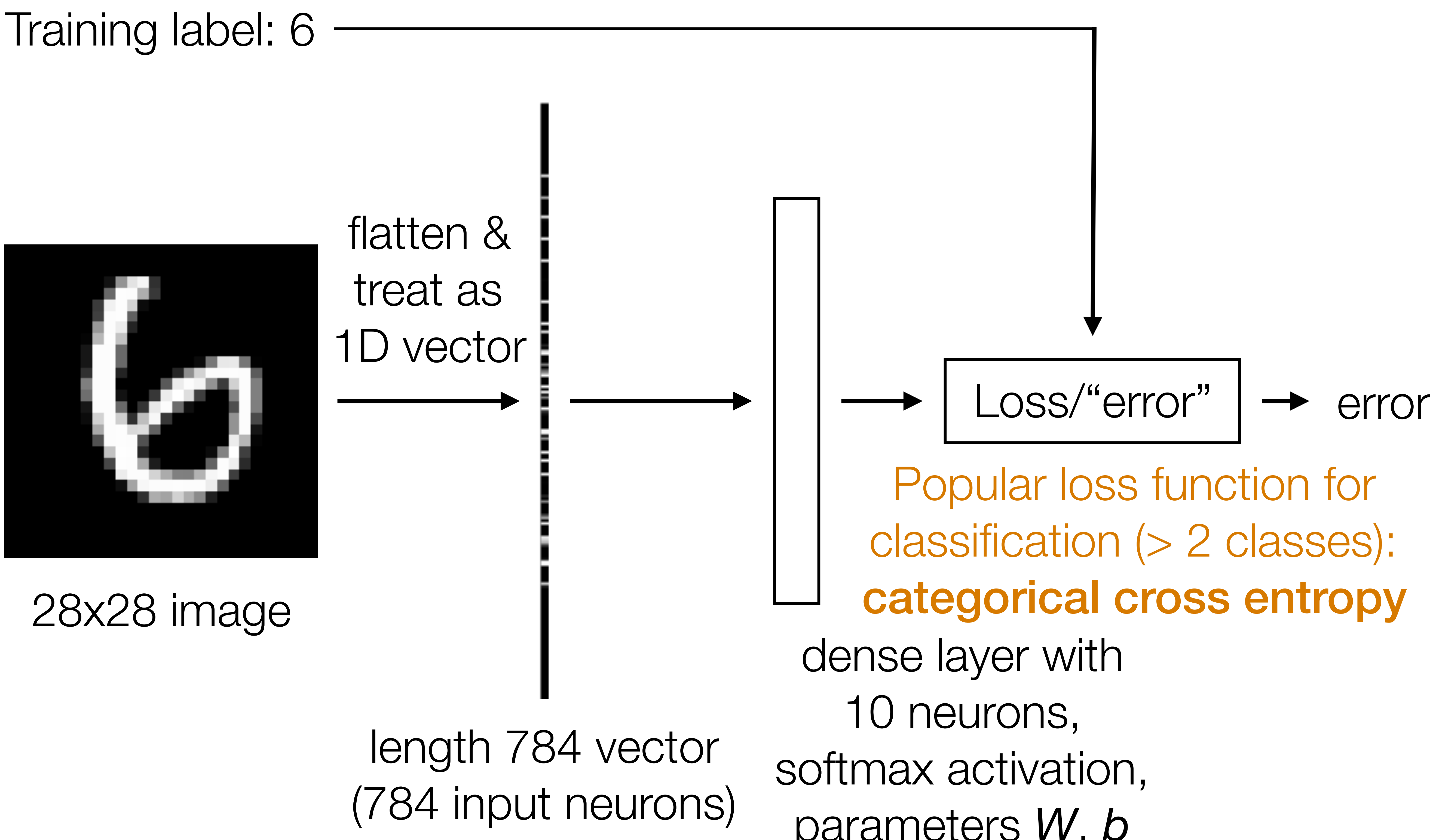
Training label: 6



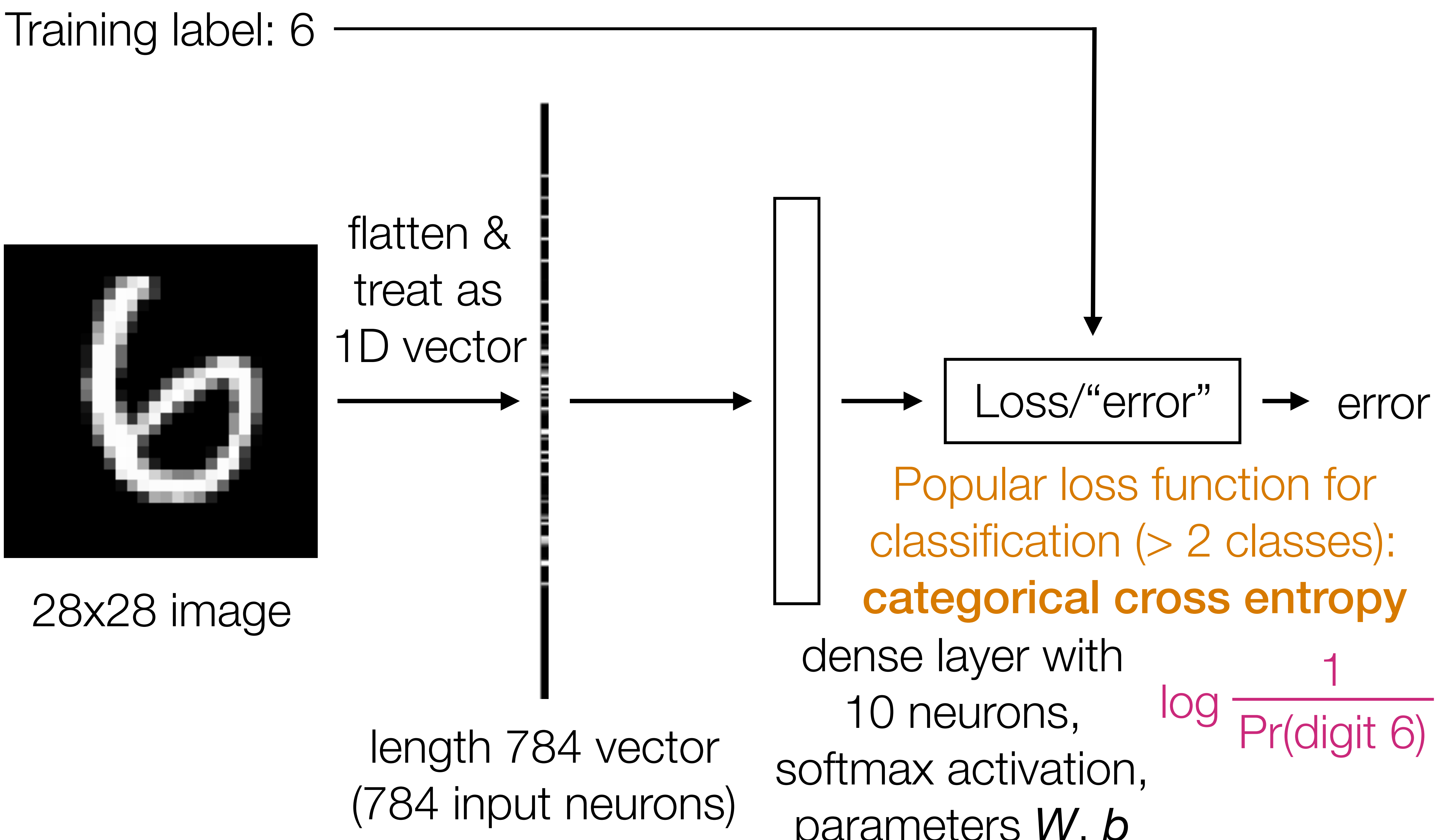
Handwritten Digit Recognition



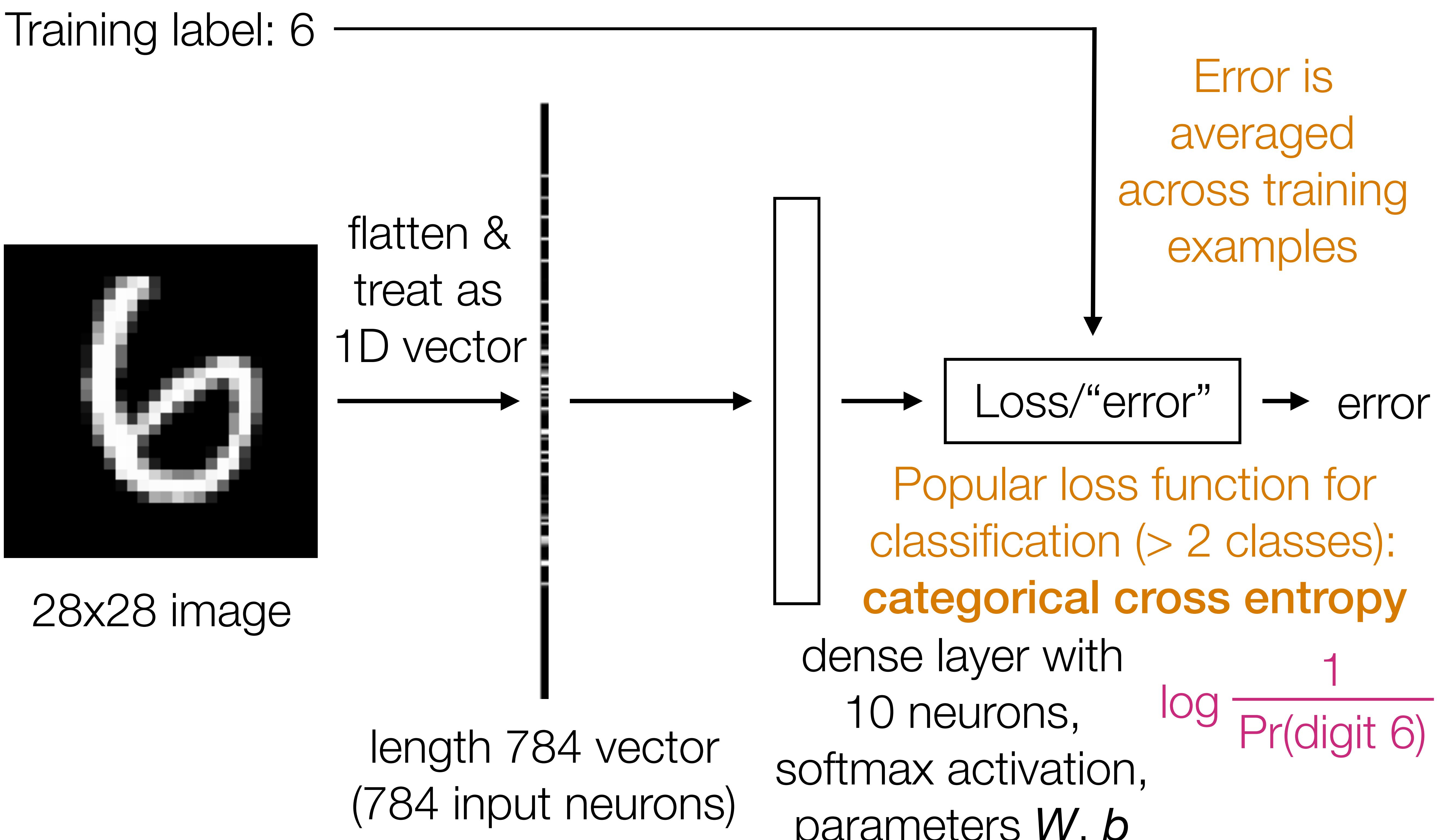
Handwritten Digit Recognition



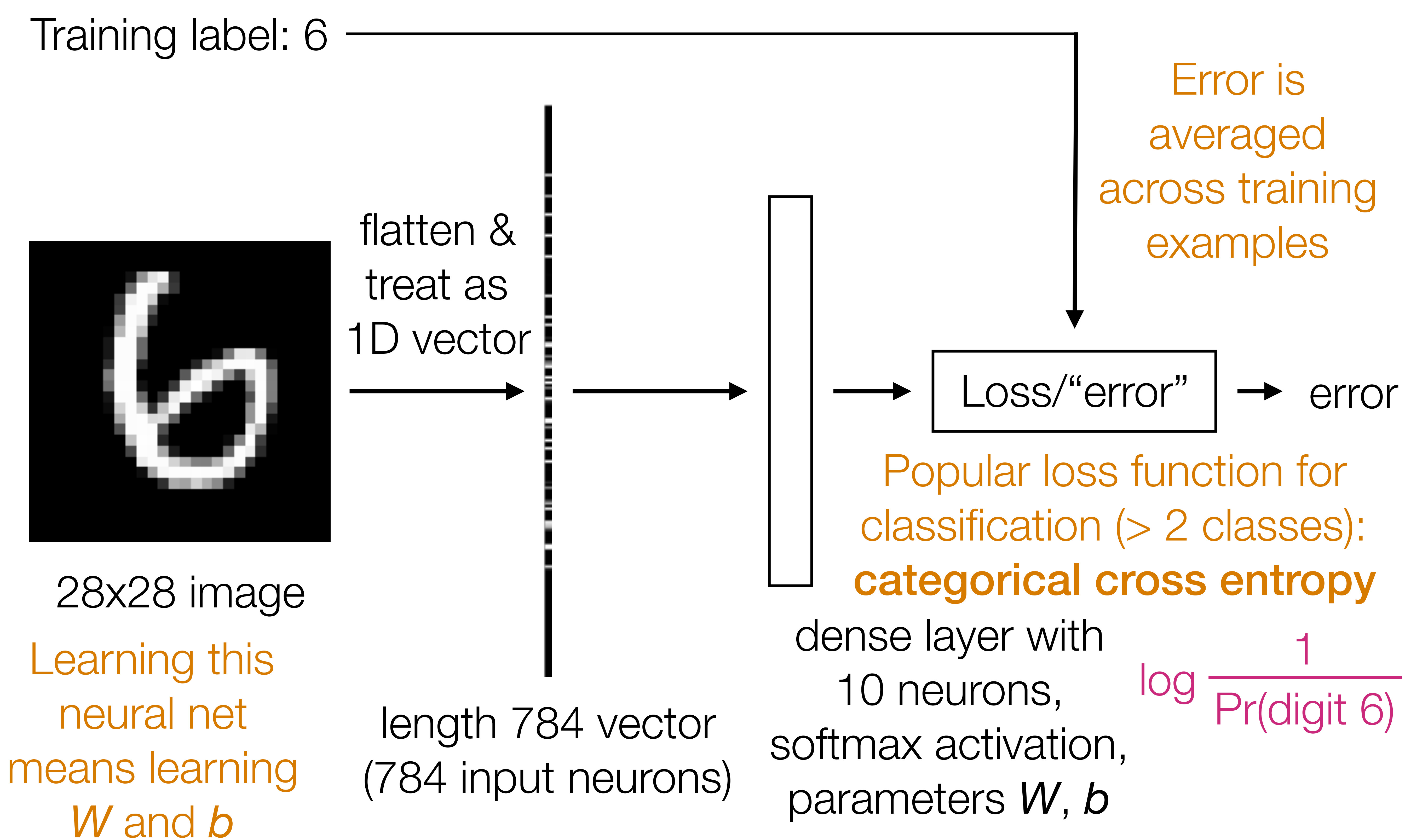
Handwritten Digit Recognition



Handwritten Digit Recognition



Handwritten Digit Recognition

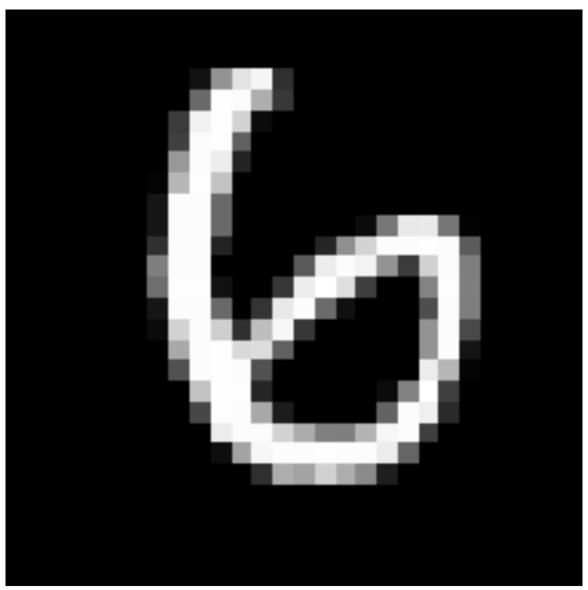


Handwritten Digit Recognition

Demo part 2

Handwritten Digit Recognition

Training label: 6



28x28 image

Learning this neural net means learning *W* and *b*

flatten & treat as 1D vector

length 784 vector (784 input neurons)

dense layer with 10 neurons, softmax activation, parameters *W*, *b*

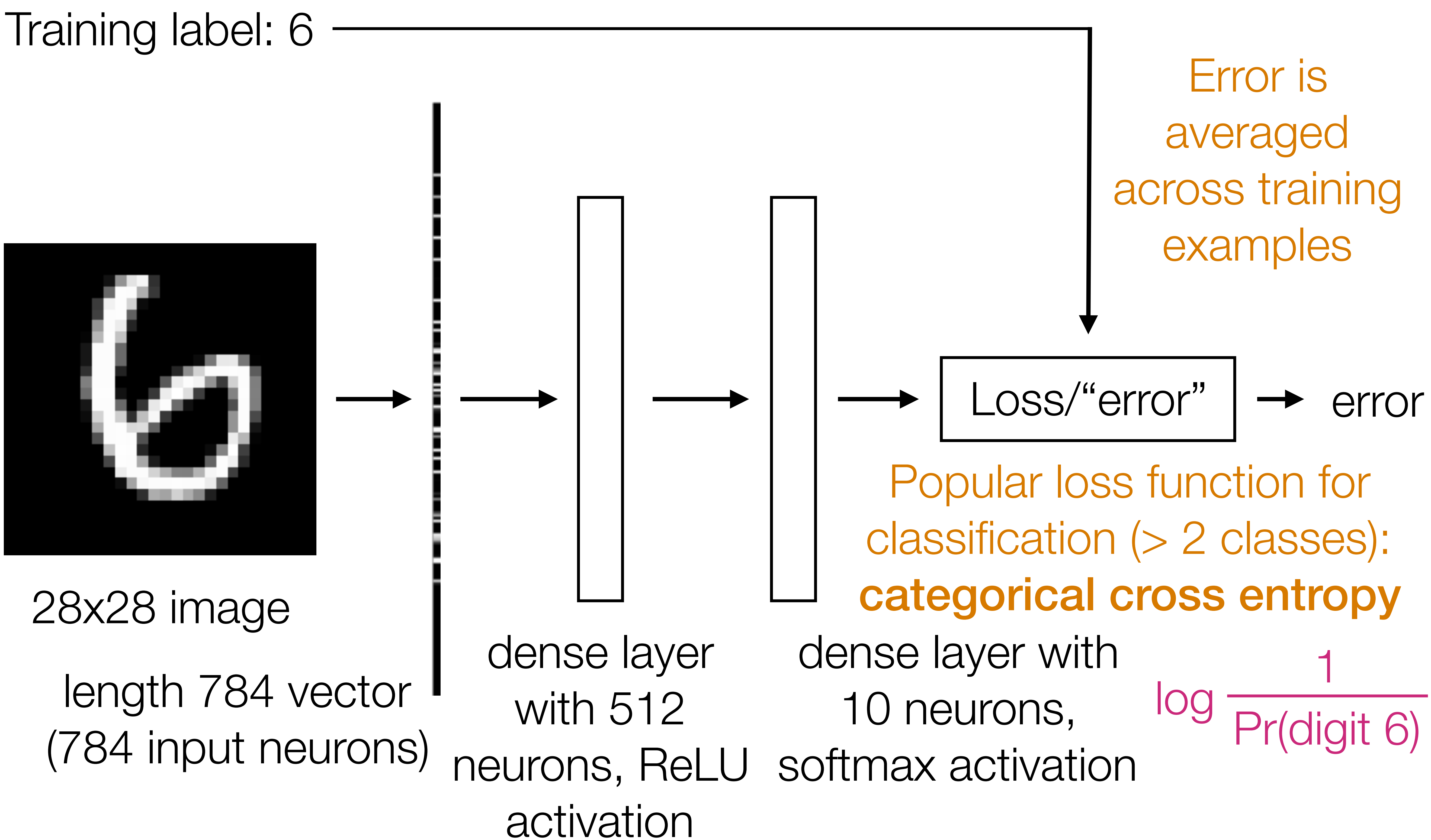
Loss/"error" → error

Error is averaged across training examples

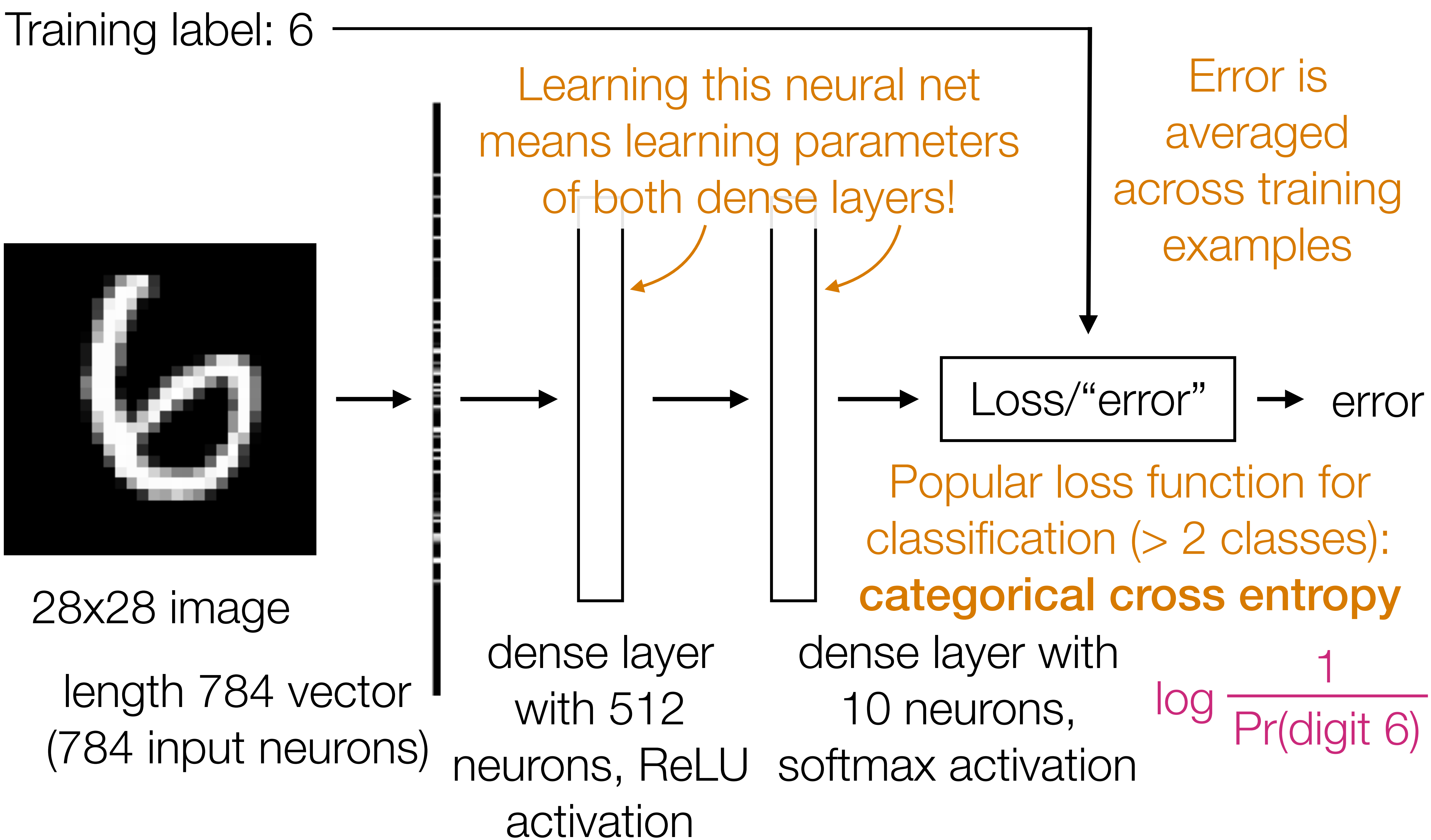
Popular loss function for classification (> 2 classes): **categorical cross entropy**

$$\log \frac{1}{\Pr(\text{digit } 6)}$$

Handwritten Digit Recognition



Handwritten Digit Recognition



Handwritten Digit Recognition

Demo part 3

Architecting Neural Nets

Architecting Neural Nets

- Increasing number of layers (depth) makes neural net more complex

Architecting Neural Nets

- Increasing number of layers (depth) makes neural net more complex
    - Can approximate more functions

Architecting Neural Nets

- Increasing number of layers (depth) makes neural net more complex
  - Can approximate more functions
  - More parameters needed

Architecting Neural Nets

- Increasing number of layers (depth) makes neural net more complex
  - Can approximate more functions
  - More parameters needed
    - More training data may be needed

Architecting Neural Nets

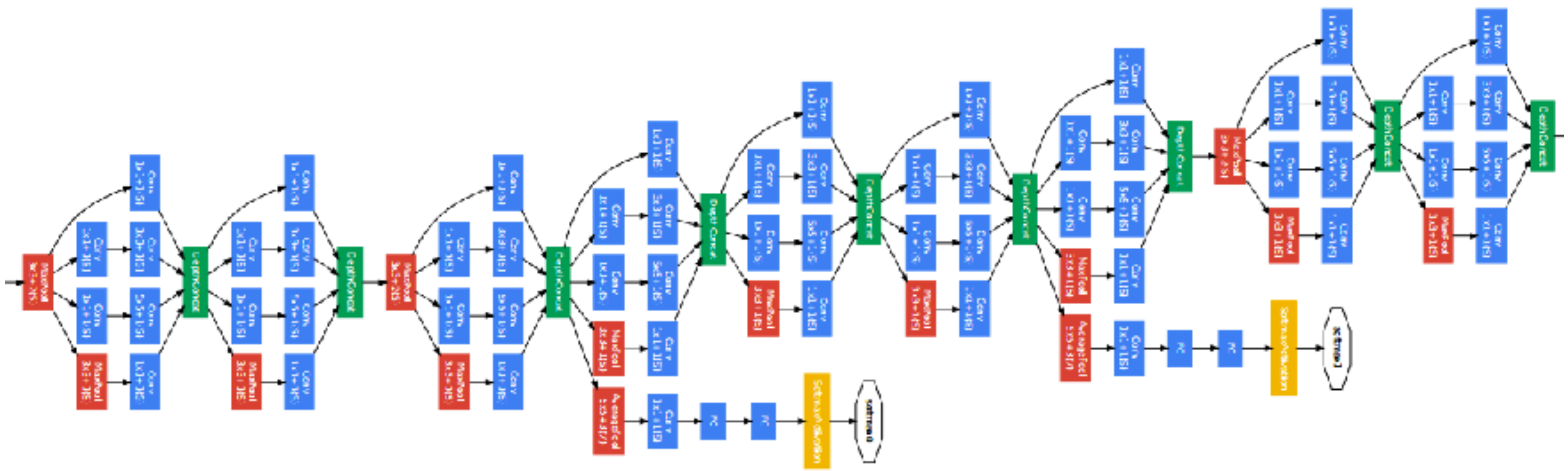
- Increasing number of layers (depth) makes neural net more complex
  - Can approximate more functions
  - More parameters needed
    - More training data may be needed
- Designing neural net architectures is a bit of an art

Architecting Neural Nets

- Increasing number of layers (depth) makes neural net more complex
  - Can approximate more functions
  - More parameters needed
    - More training data may be needed
- Designing neural net architectures is a bit of an art
  - How to select the number of neurons for intermediate layers?

Architecting Neural Nets

- Increasing number of layers (depth) makes neural net more complex
  - Can approximate more functions
  - More parameters needed
    - More training data may be needed
- Designing neural net architectures is a bit of an art
  - How to select the number of neurons for intermediate layers?
  - Very common in practice: modify existing architectures that are known to work well (e.g., VGG-16 for computer vision/image processing)



GoogLeNet 2014

Deep Learning

Deep Learning

- Inspired by biological neural nets *but otherwise not the same at all* (biological neural nets do *not* work like deep nets)

Deep Learning

- Inspired by biological neural nets *but otherwise not the same at all* (biological neural nets do *not* work like deep nets)

- Learns a layered representation

Deep Learning

- Inspired by biological neural nets *but otherwise not the same at all* (biological neural nets do *not* work like deep nets)
- Learns a layered representation
  - Tries to get rid of manual feature engineering

Deep Learning

- Inspired by biological neural nets *but otherwise not the same at all* (biological neural nets do *not* work like deep nets)
- Learns a layered representation
  - Tries to get rid of manual feature engineering
- Upcoming: enforce structure using special layers

Deep Learning

- Inspired by biological neural nets *but otherwise not the same at all* (biological neural nets do *not* work like deep nets)

- Learns a layered representation

  - Tries to get rid of manual feature engineering

- Upcoming: enforce structure using special layers

  - Can think of this as constraining what features are learned