ANN

Name	Dharini Baskaran
Identity Key	dhba5060

	Level	Completed
	Beginner	11
	Intermediate	6
	Advanced	1
	Expert	0

Goal	
4722	14
5722	16
Total Completed	
18	

Artificial Neural Network (ANN)

CSCI 5722/4722: Computer Vision

Spring 2024

Dr. Tom Yeh

Dr. Mehdi Moghari

Graphical Representation

CSCI 5722/4722 Computer Vision

University of Colorado Boulder

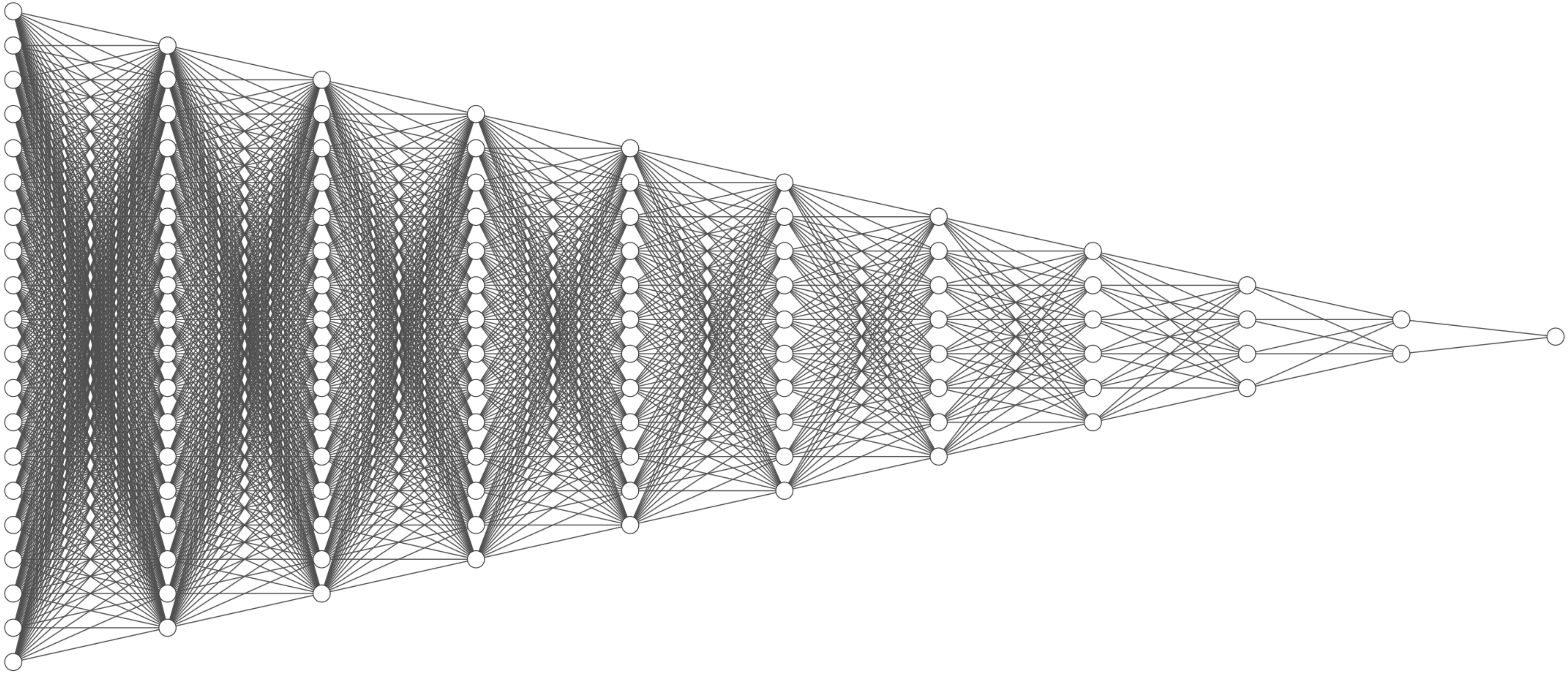


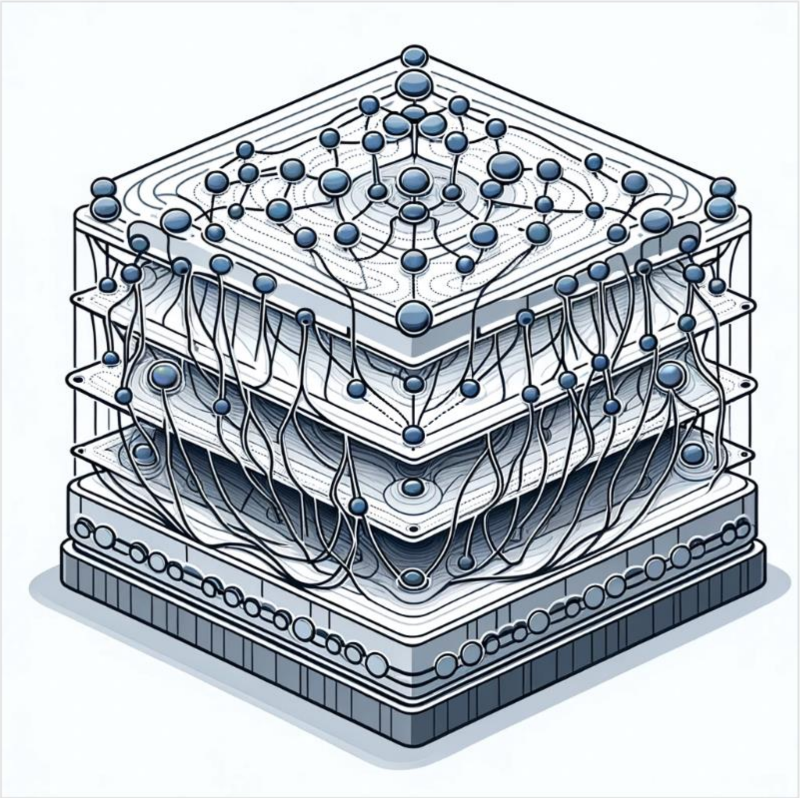
Image Generated by: http://alexlenail.me/NN-SVG

ChatGPT

You

generate a graphic representation of an artificial neural network

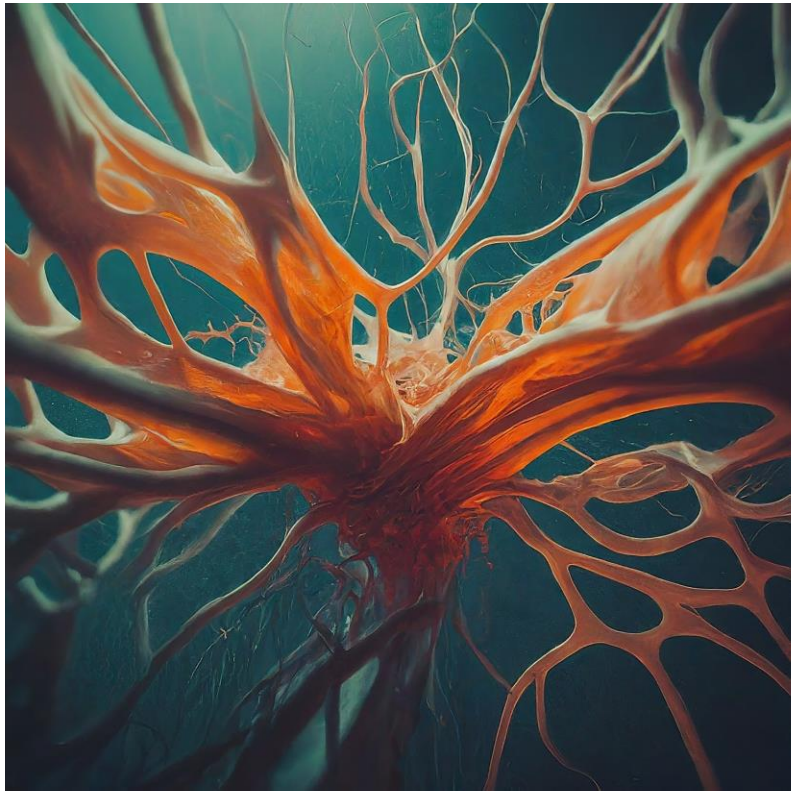
ChatGPT



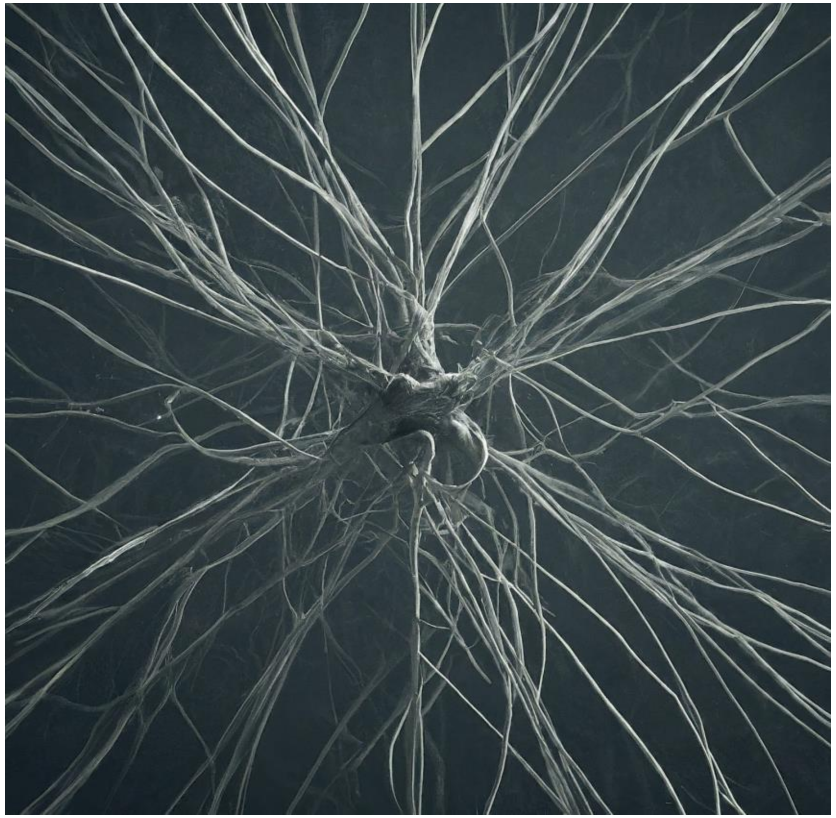
Here is a graphic representation of an artificial neural network. This illustration shows the typical structure with an input layer, multiple hidden layers, and an output layer, all connected by lines representing the connections between neurons. Each layer is composed of circles symbolizing neurons. The input, hidden, and output layers are appropriately labeled for clarity.

Prompt ChatGPT to draw an ANN, as complex or absurd as possible

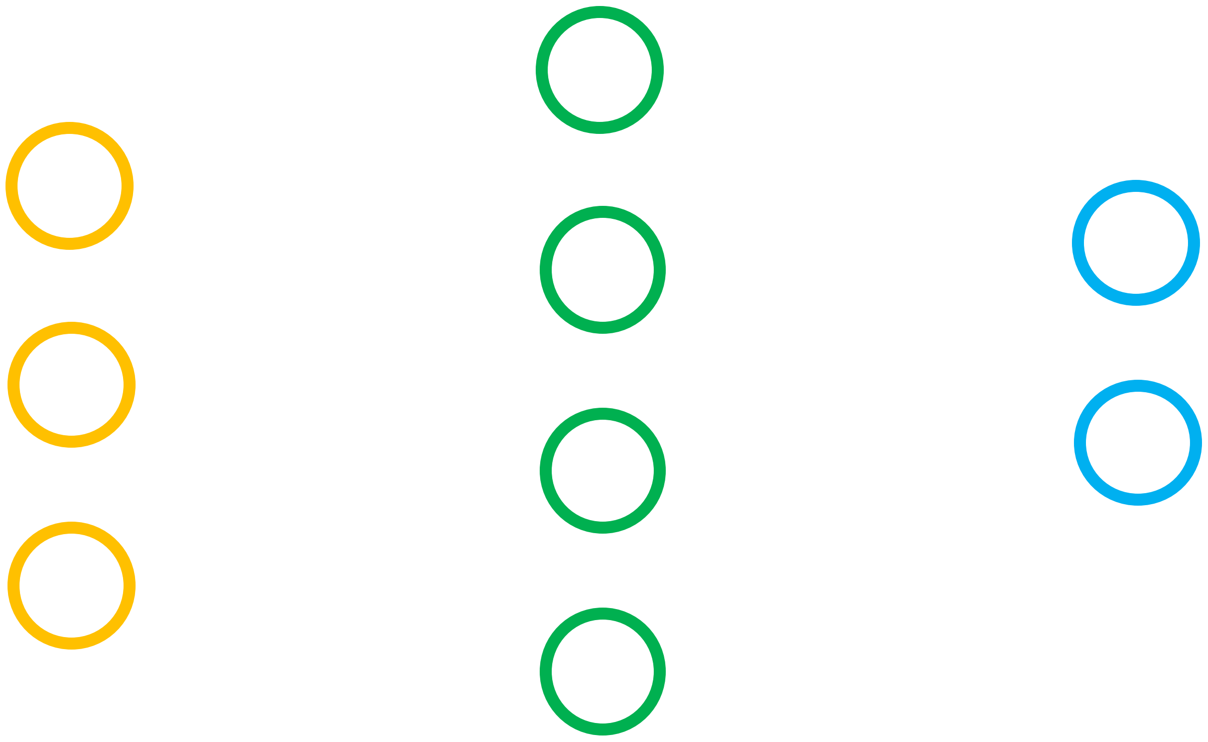
- ANN in a cartesian plane

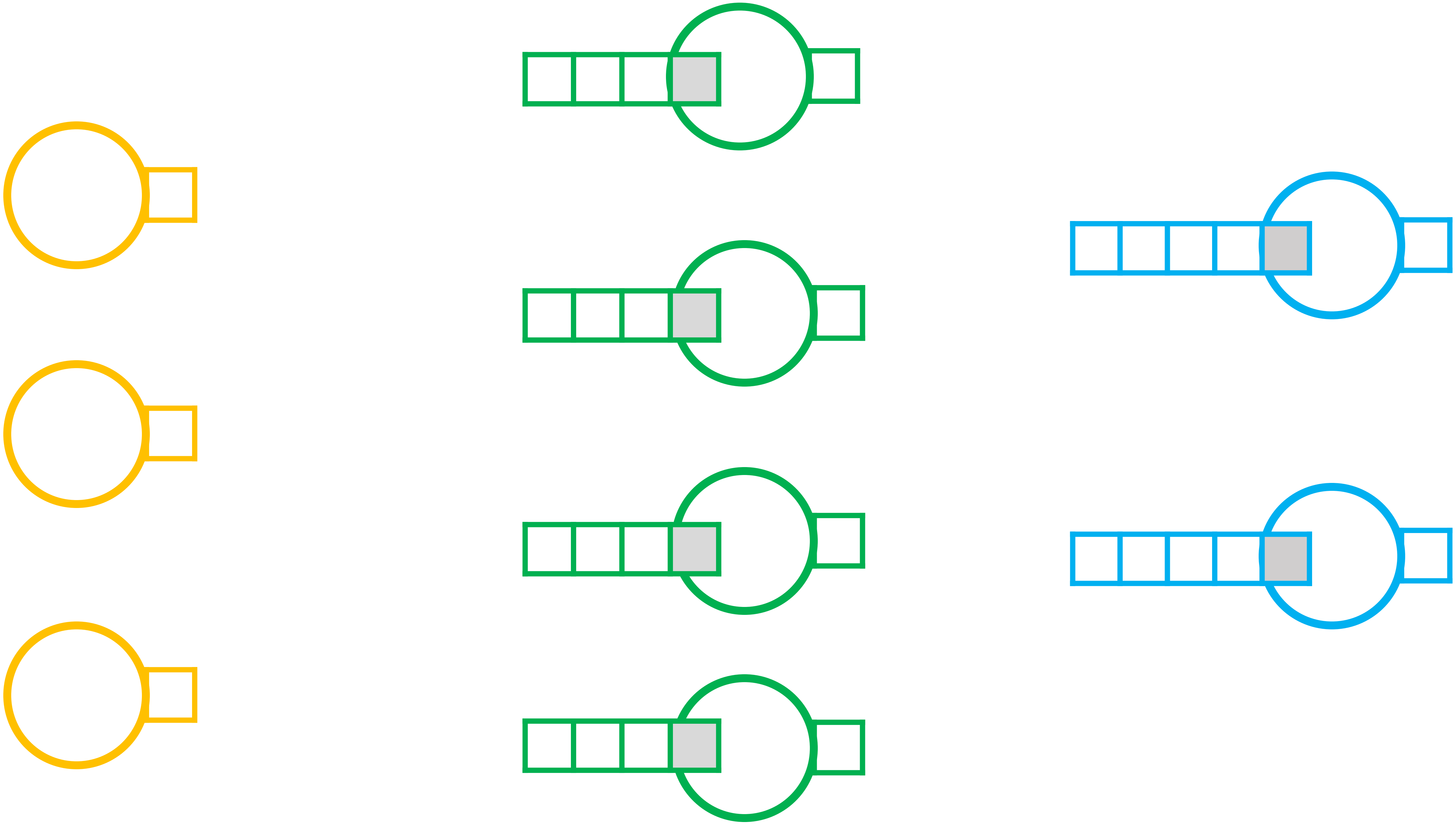


- ANN with just lines

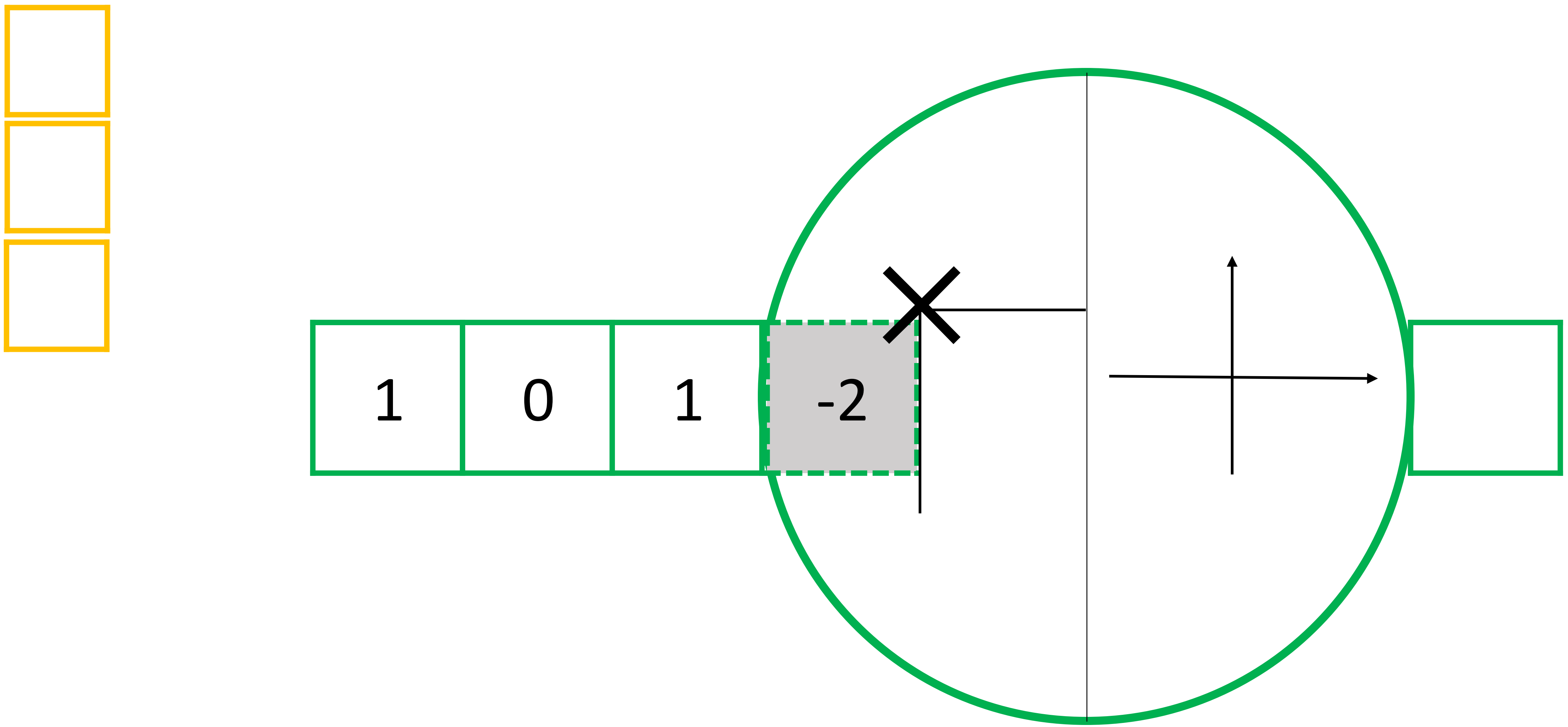


2 Layer Network

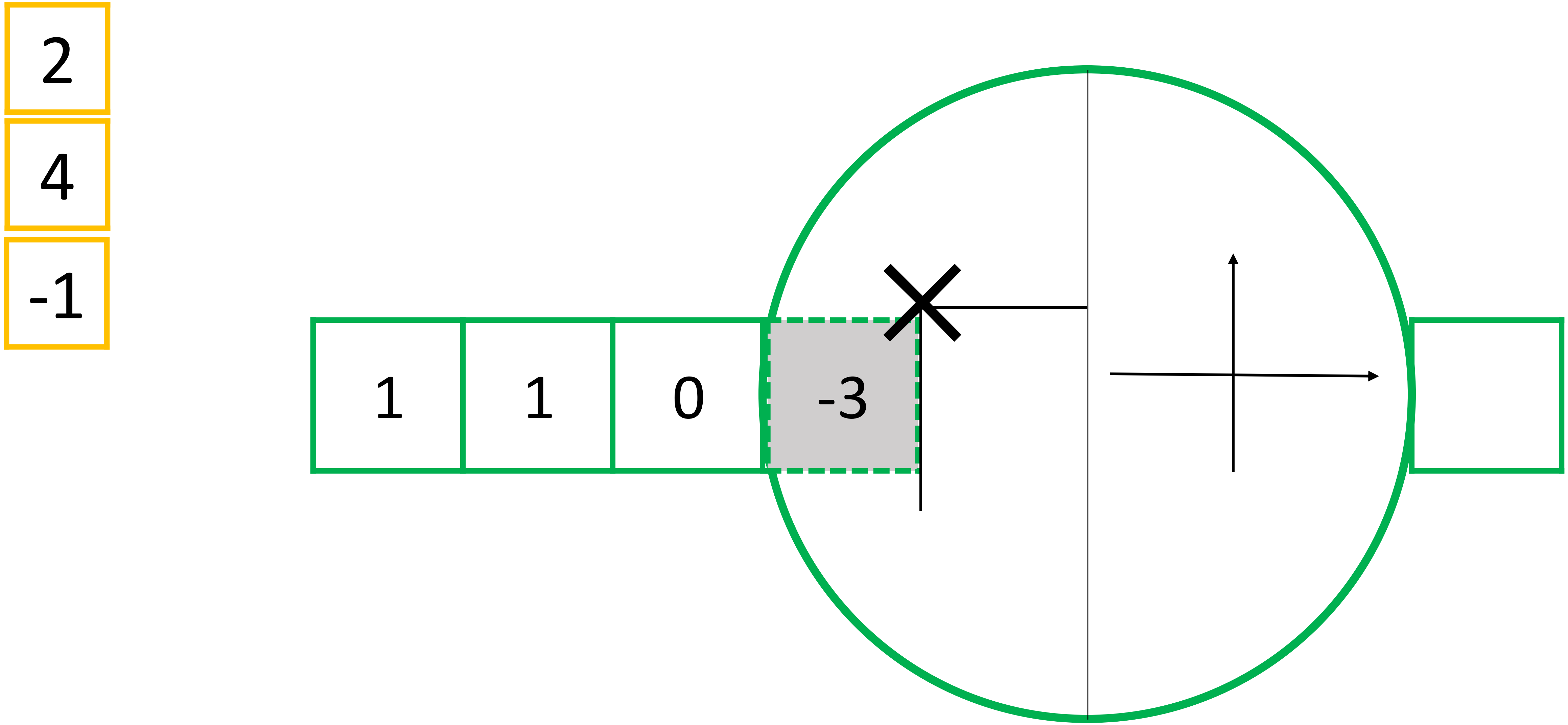




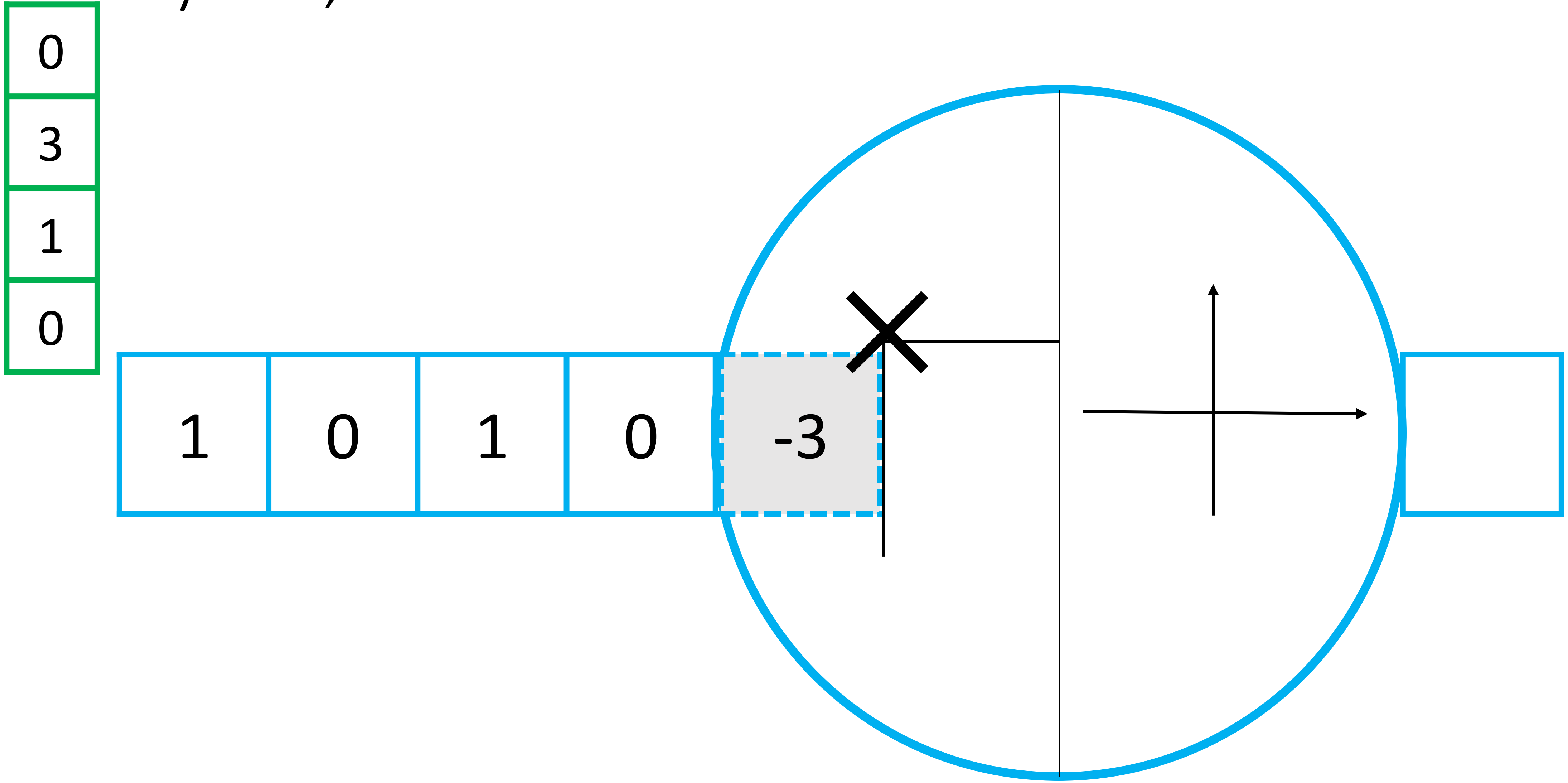
Layer 1, Node 1



Layer 1, Node 2



Layer 2, Node 1



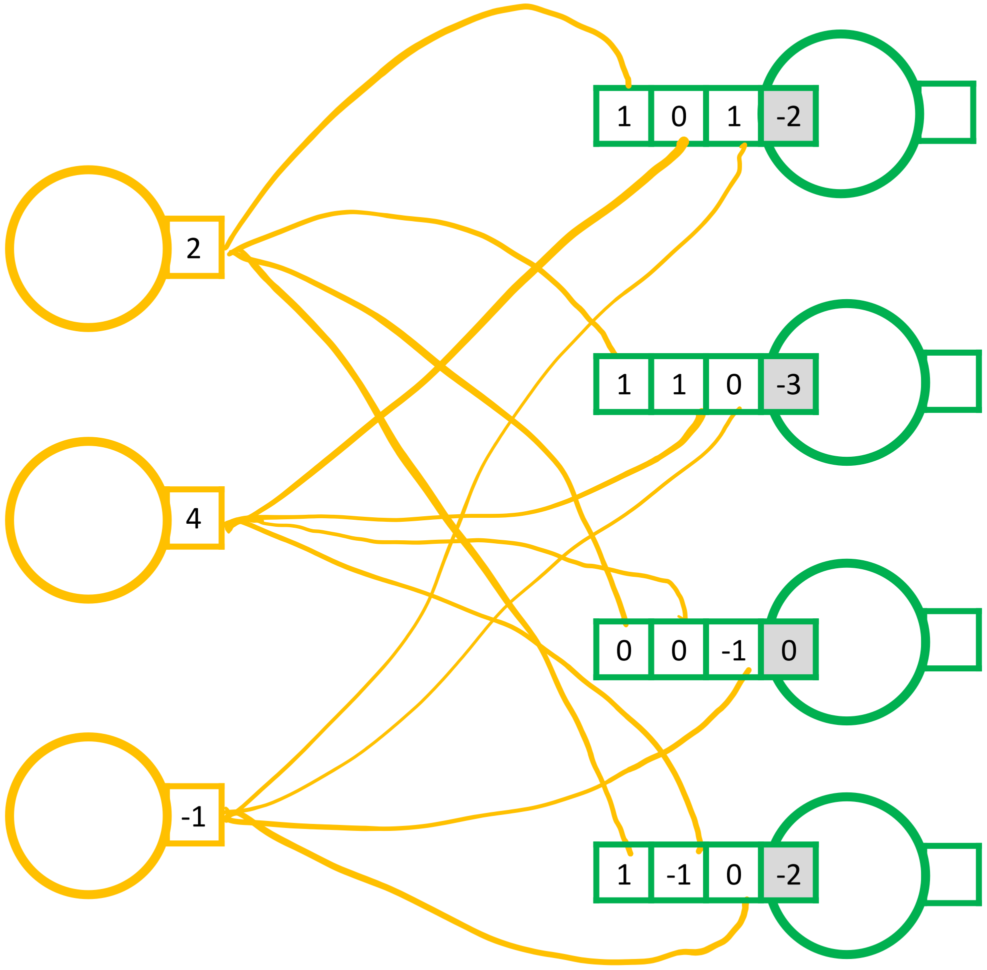
x	σ(x)
-5	~0
-4	0.02
-3	0.05
-2	0.12
-1	0.27
0	0.5
1	0.73
2	0.88
3	0.95
4	0.98
5	~1

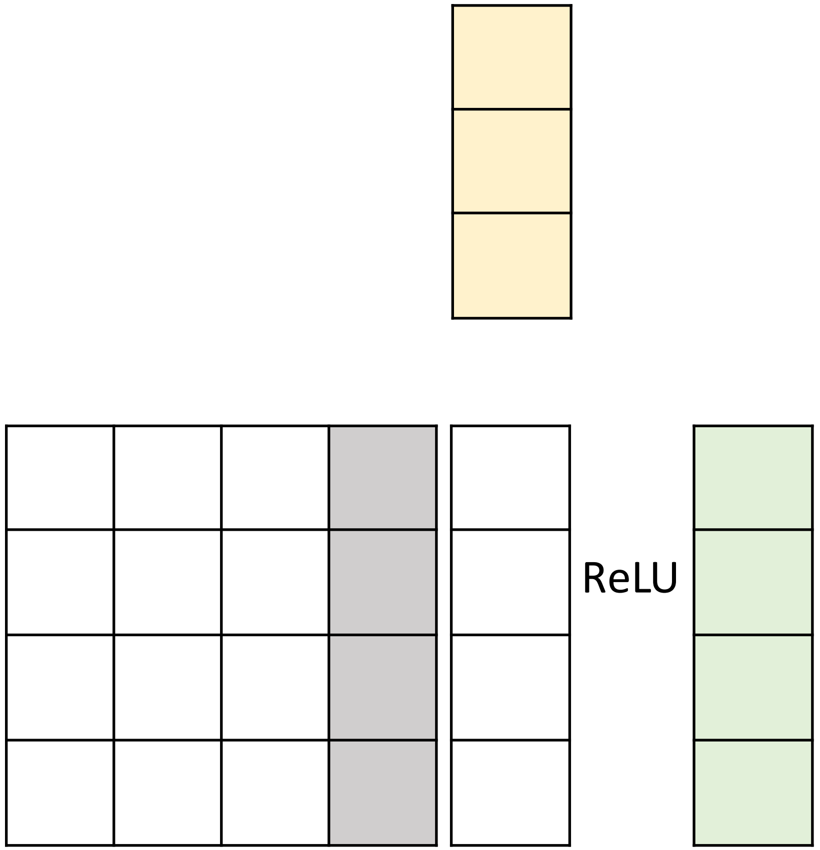
Matrix Representation

CSCI 5722/4722 Computer Vision

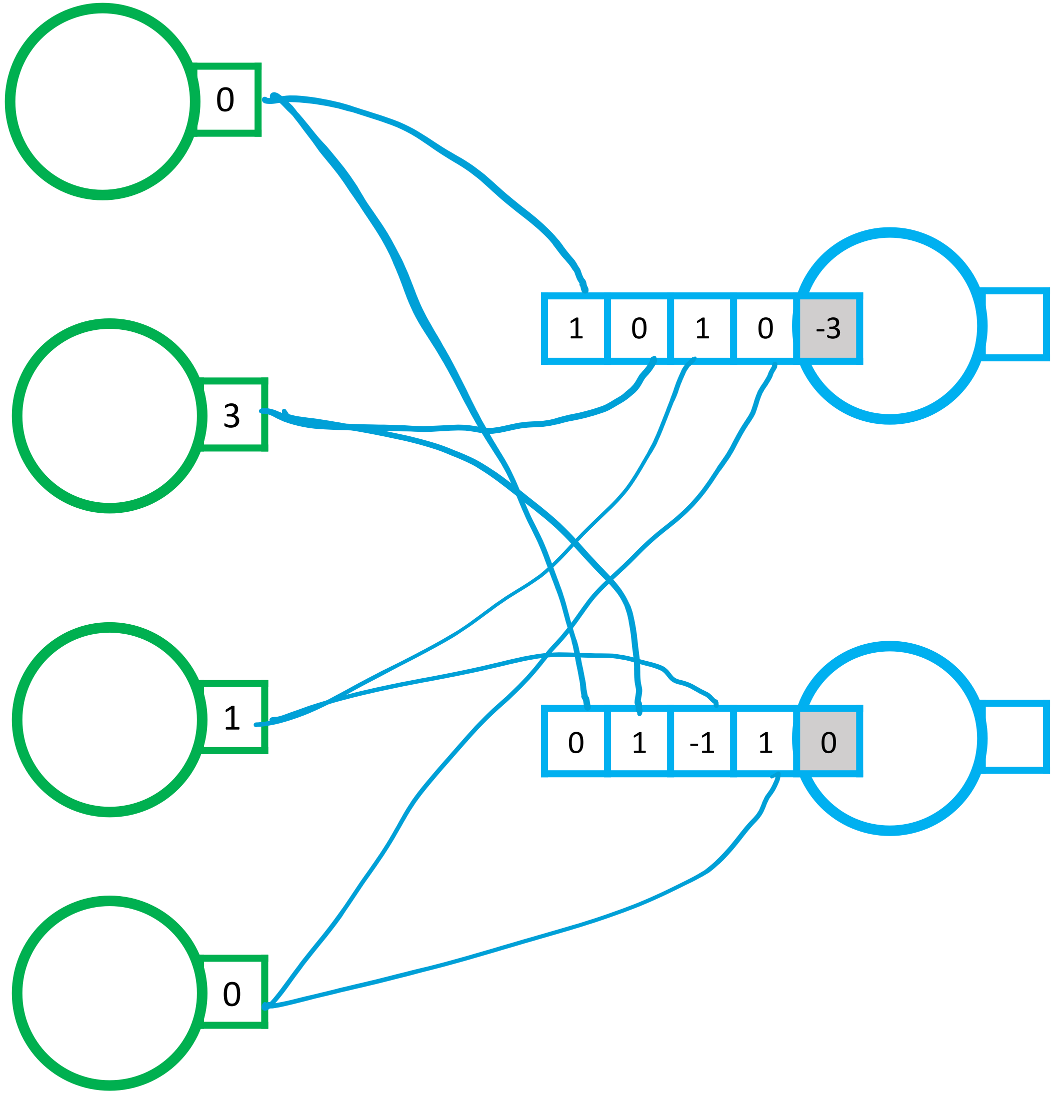
University of Colorado Boulder

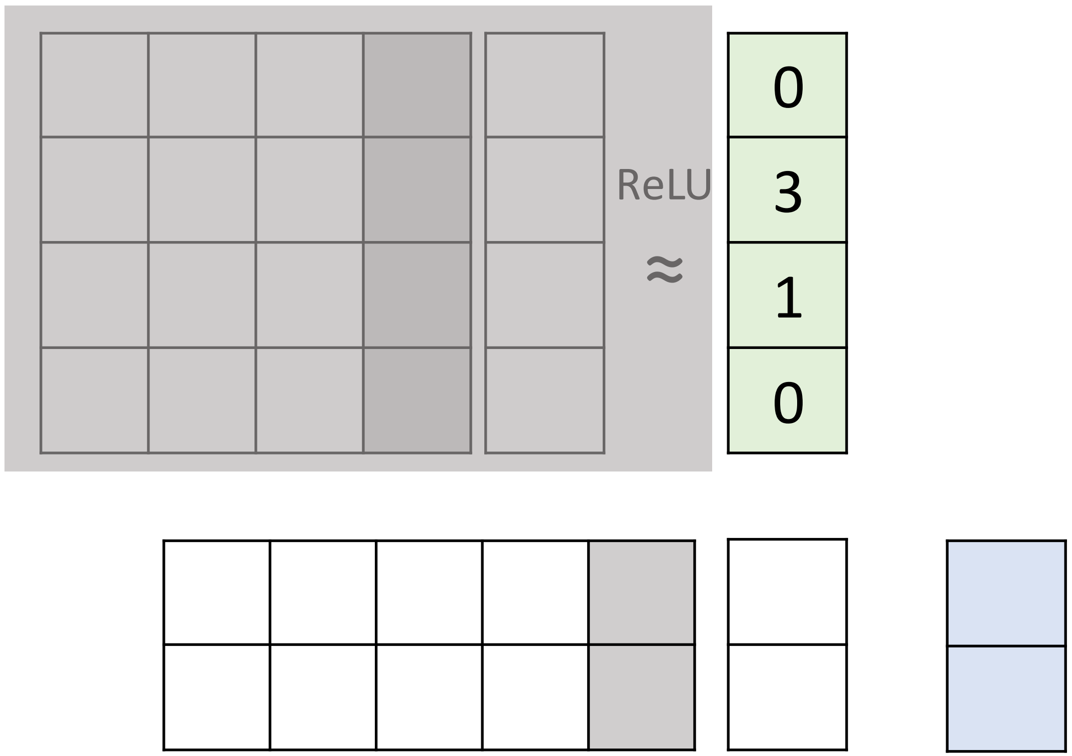
Graph → Matrices [Layer 1]





Graph → Matrices [Layer 2]





x	σ(x)
-5	~0
-4	0.02
-3	0.05
-2	0.12
-1	0.27
0	0.5
1	0.73
2	0.88
3	0.95
4	0.98
5	~1

Calculate ReLU

1	ReLU ≈	1	a
-1		0	b
3		3	c
-4		0	d
2		2	e
-1		0	f
5		5	g
0		0	h

🔑 a+b+c+d=4; e+f+g+h=7

Calculate a Linear Layer + ReLU

1
2
-3
1

1	0	1	-2	-4		0	a
1	1	0	-2	1	ReLU	1	b
0	0	-1	0	3	≈	3	c
1	-1	0	0	-1		0	d

a+b+c+d=4

Calculate a Linear Layer + ReLU

1
2
5
3
4
1

1	0	1	0	0	-3	3		3	a
0	1	0	1	0	-4	1	ReLU	1	b
0	0	1	0	1	-5	4	≈	4	c
1	0	0	0	1	-6	-1		0	d

a+b+c+d=8

Find smallest bias terms to activate (integers)

	1
	2
	-3
	1

1	0	1	3	a	1		1
1	1	0	-2	b	1	ReLU	1
0	0	-1	-2	c	1	≈	1
1	-1	0	2	d	1		1

🔑 a+b+c+d=1

Softmax

CSCI 5722/4722 Computer Vision

University of Colorado Boulder

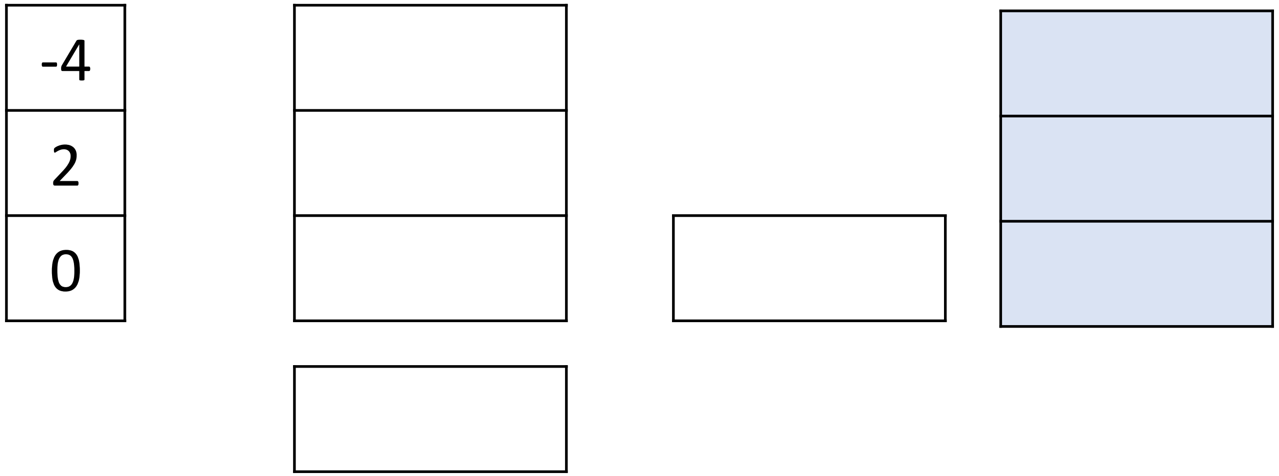
Logits → Probability Distribution

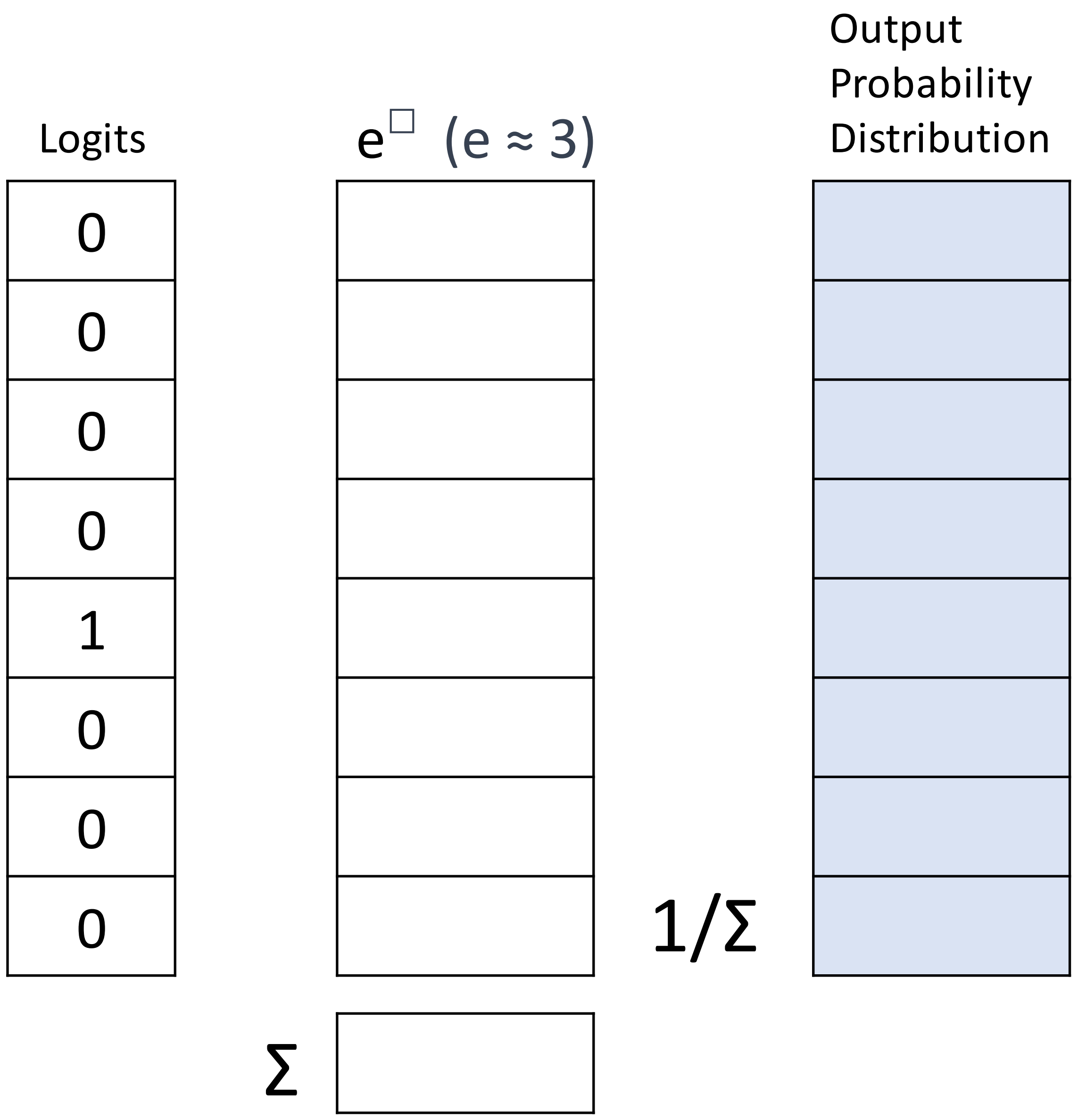
$$\begin{bmatrix} 0 \\ 3 \\ 1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 1 & 0 & -3 \\ 0 & 1 & -1 & 1 & 0 \\ 0 & 1 & 0 & -1 & -2 \end{bmatrix} \quad \begin{bmatrix} -2 \\ 2 \\ 0 \end{bmatrix} \;\; \overset{\sigma}{\approx} \;\; \begin{bmatrix} .12 \\ .88 \\ .5 \end{bmatrix}$$

$$\begin{bmatrix} -2 \\ 2 \\ 0 \end{bmatrix}$$

Calculate Softmax by hand (e ≈ 3) ✍️




Logits
$e^{\square}$ ($e \approx 3$)
Output Probability Distribution
0
0
0
0
1
0
0
0
$1/\Sigma$
Σ

Calculate Softmax

Logits	$e^{\square}$ $(e \approx 3)$	Output Probability Distribution
0	1	1/8
0	1	1/8
0	1	1/8
0	1	1/8
0	1	1/8
0	1	1/8
0	1	1/8
0	1	1/8

$1/\Sigma$

Σ 8

a

a*1000 % 13 = 8

Calculate Softmax

Logits	$e^{\square}$ $(e \approx 3)$		Output Probability Distribution	
0	1		0.05	a
2	9		0.45	b
0	1		0.05	
0	1		0.05	
1	3		0.15	c
0	1		0.05	
0	1		0.05	
1	3	$1/\Sigma$	0.15	

Σ 20

a+b+c =0.65

Calculate Softmax

Logits	$e^{\square}$ ($e \approx 3$)	Output Probability Distribution
-2	1/9	0.11 (a)
-1	1/3	0.33 (b)
-1	1/3	0.33
-2	1/9	0.11
-2	1/9	0.11
-5	0	0 (c)
-5	0	0
-5	0	0

Σ 1

$1/\Sigma$

round to 0.00

a+b+c=0.44

Multi Layer Perceptron (MLP)

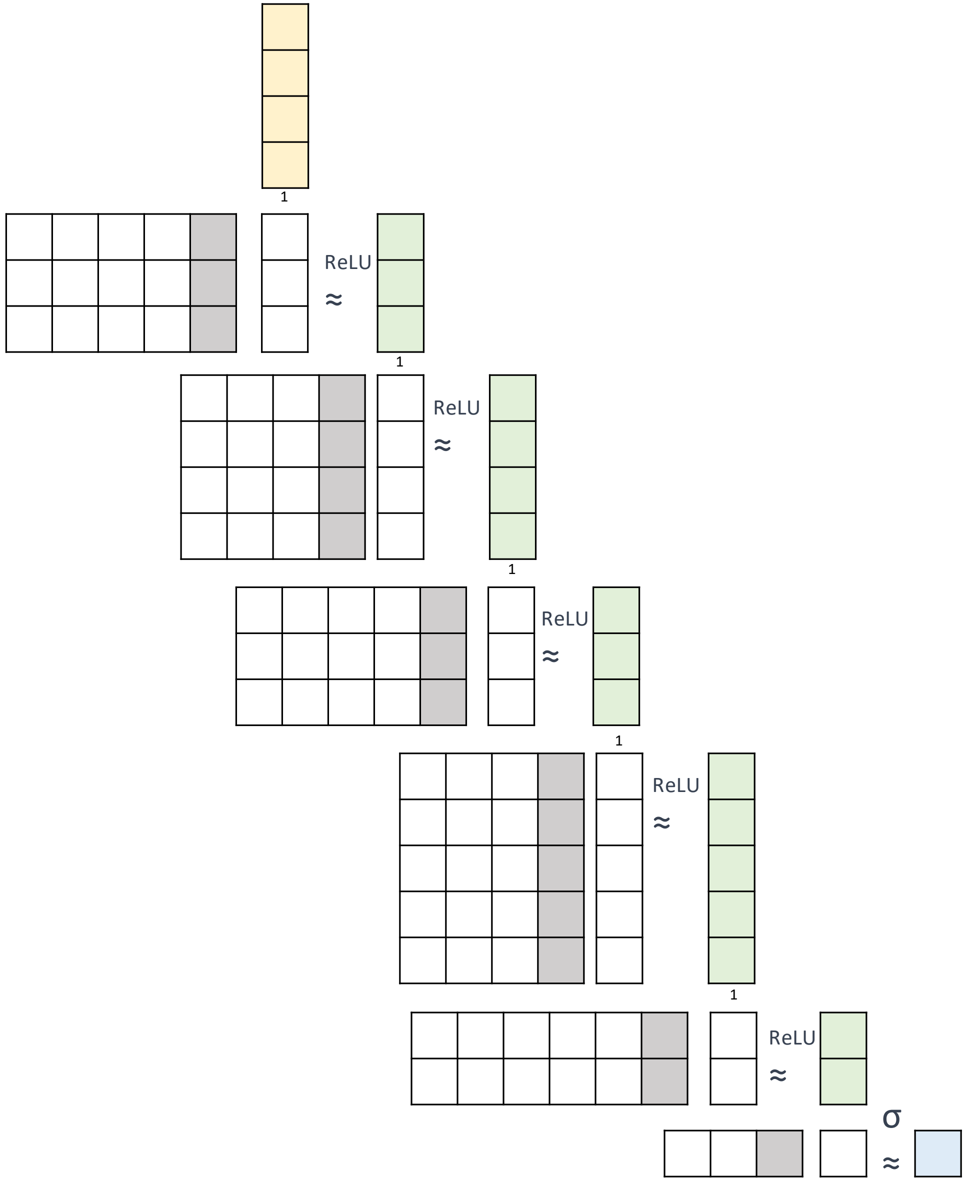
Matrices → Graphic [2 layers]

$$\begin{array}{c} 2 \\ 4 \\ -1 \\ \scriptstyle 1 \end{array}$$

$$\begin{bmatrix} 1 & 0 & 1 & -2 \\ 1 & 1 & 0 & -3 \\ 0 & 0 & -1 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ 3 \\ 1 \\ -2 \end{bmatrix} \overset{\text{ReLU}}{\approx} \begin{bmatrix} 0 \\ 3 \\ 1 \\ 0 \end{bmatrix} \begin{array}{c} \\ \\ \\ \\ \scriptstyle 1 \end{array}$$

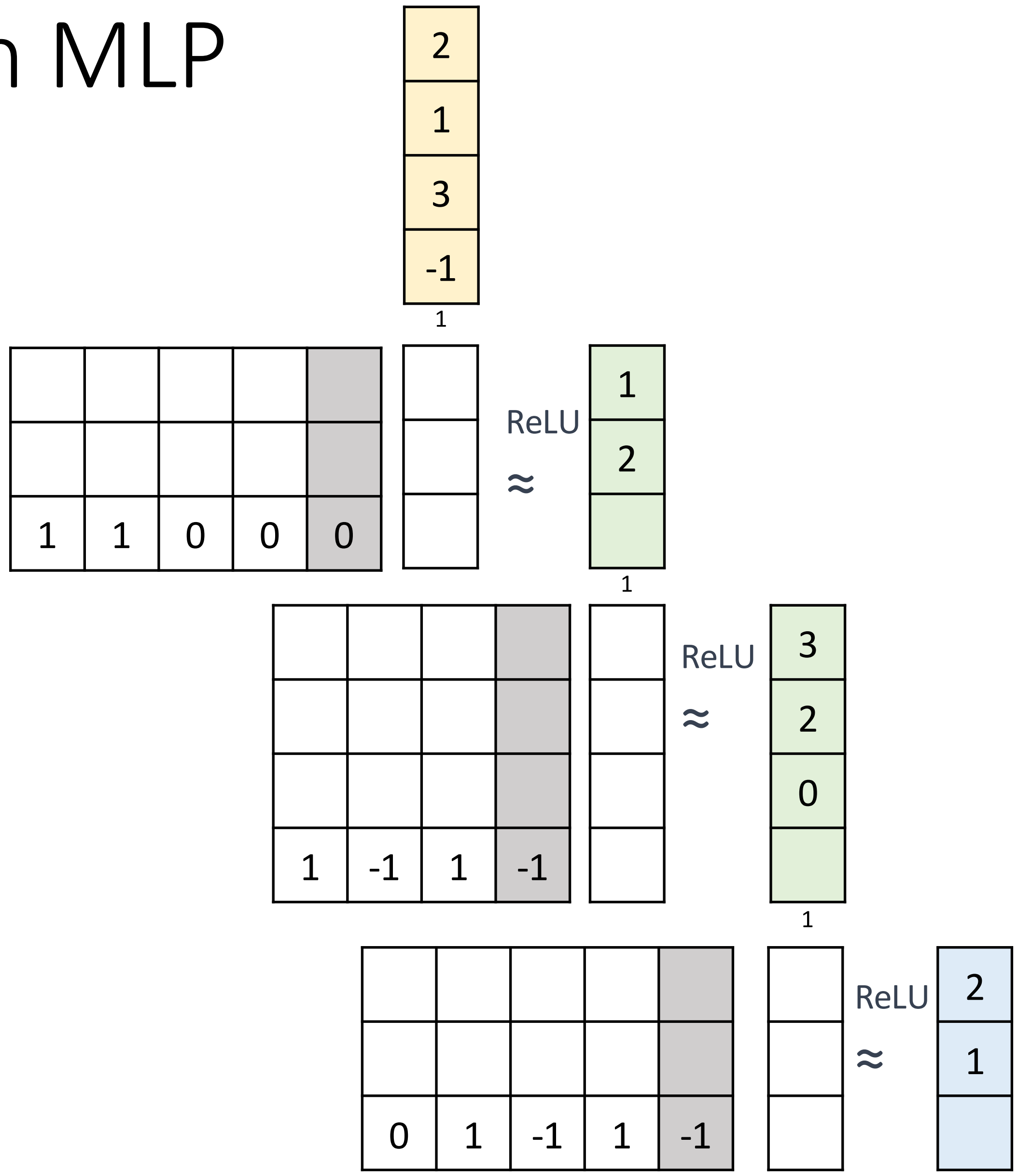
$$\begin{bmatrix} 1 & 0 & 1 & 0 & -3 \\ 0 & 1 & -1 & 1 & 0 \\ 0 & 1 & 0 & -1 & -2 \end{bmatrix} \begin{bmatrix} -2 \\ 2 \\ 0 \end{bmatrix} \overset{\sigma}{\approx} \begin{bmatrix} .12 \\ .88 \\ .5 \end{bmatrix}$$

Matrices → Graphic [6 layers]



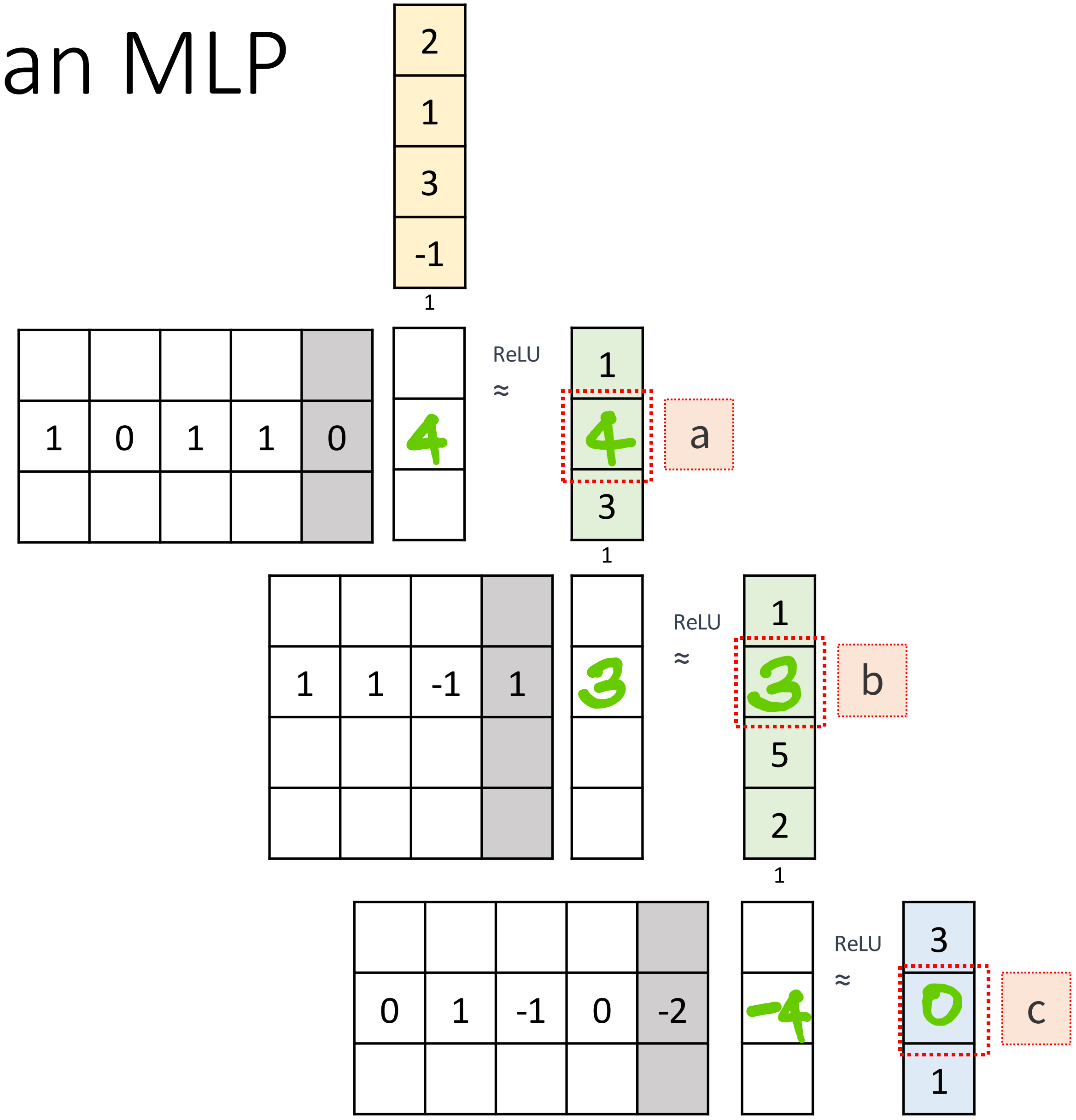
Calculate an MLP

Calculate the last feature / node / row only.



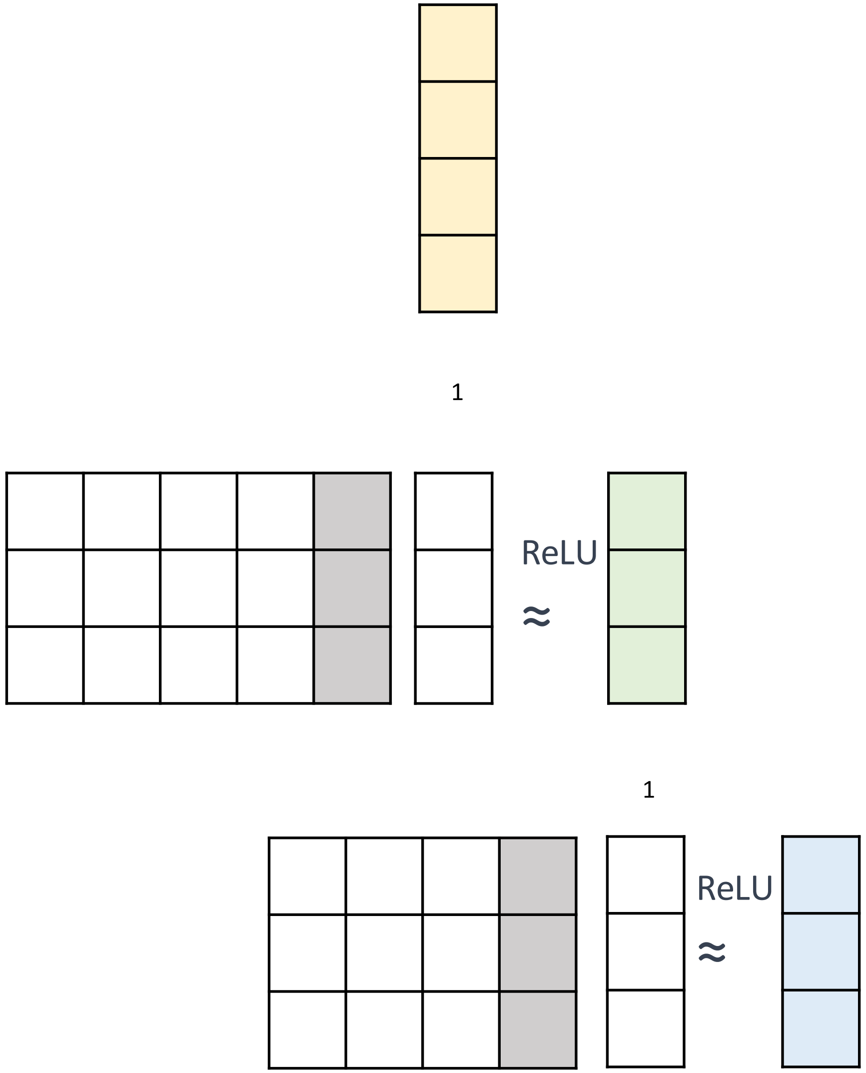
Calculate an MLP

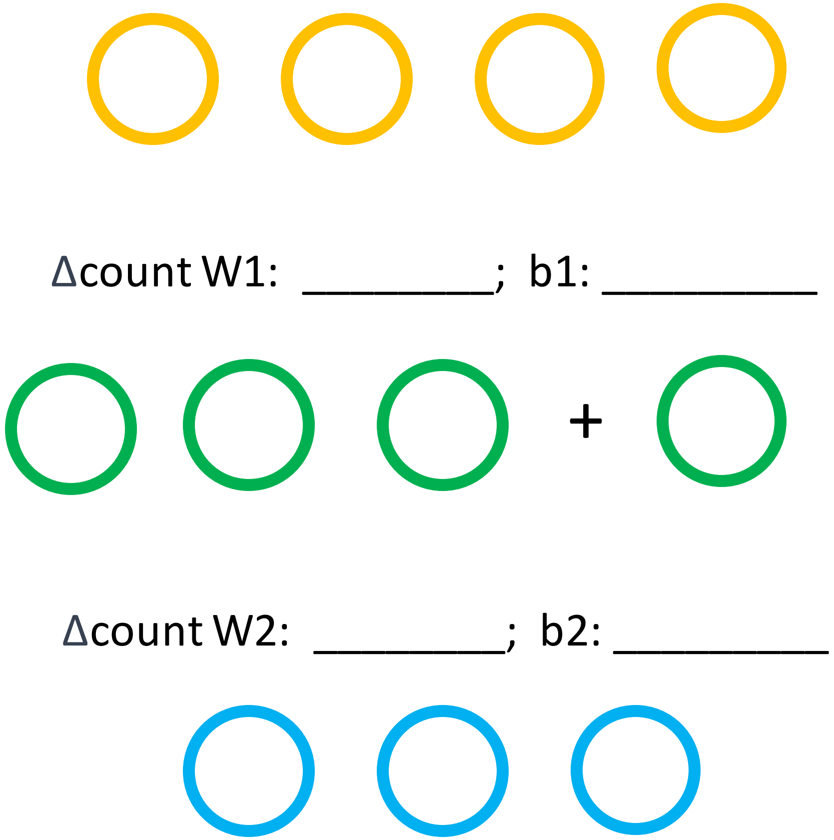
Calculate the second feature / node / row only.



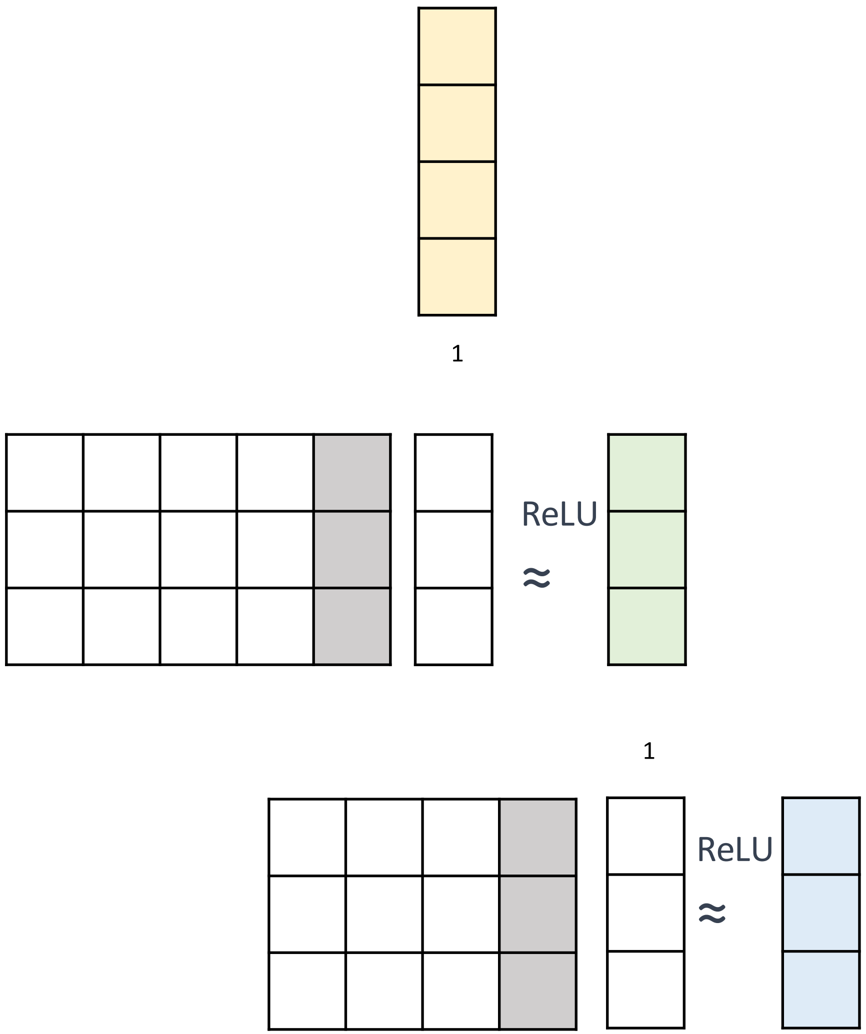
a+b+c=7

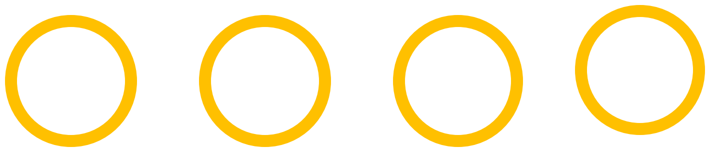
Add a hidden node (green)



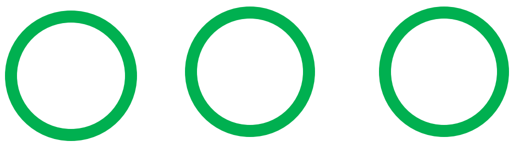


Remove a hidden node (green)

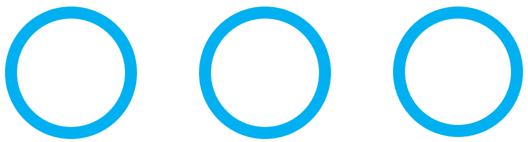




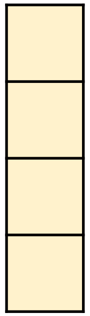
Δcount W1: __________; b1: ___________



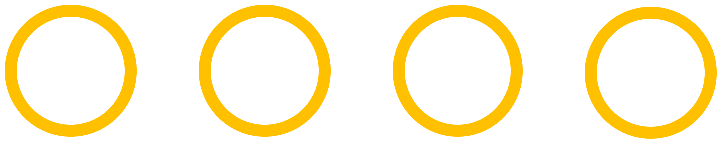
Δcount W2: __________; b2: ___________



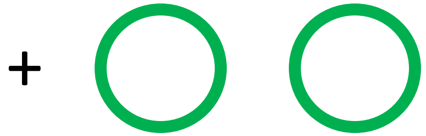
Insert a layer



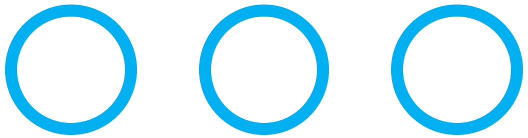
1

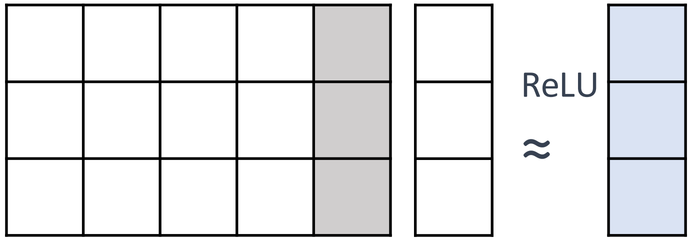


Δcount W1: __________; b1: ___________



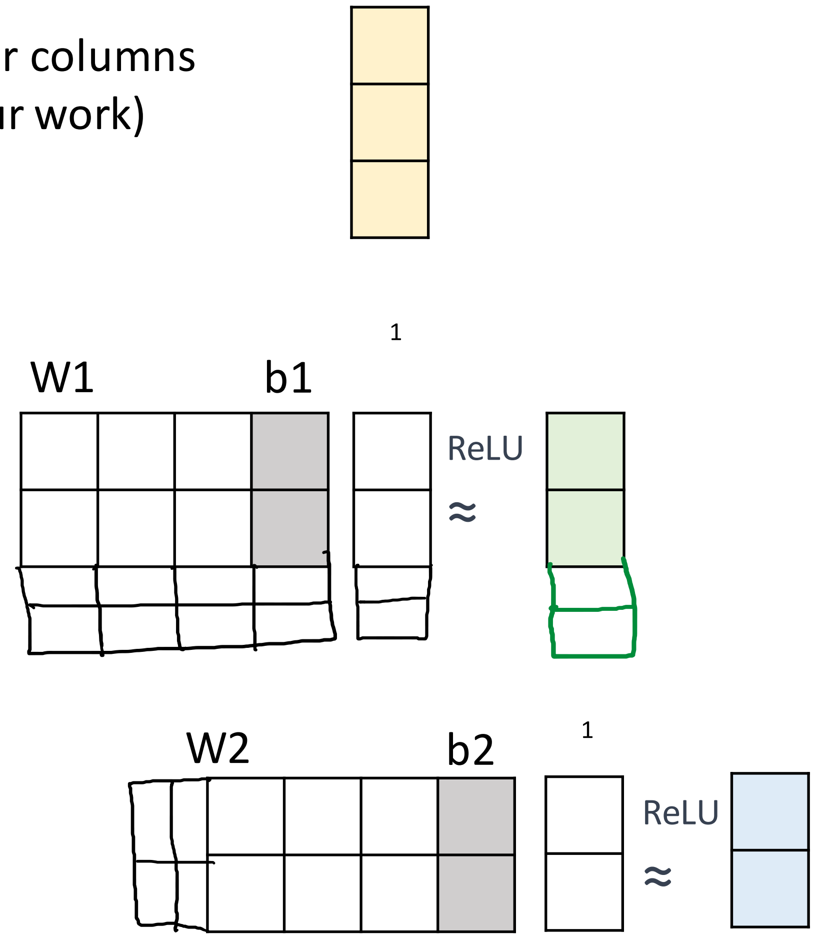
Δcount W2: __________; b2: ___________

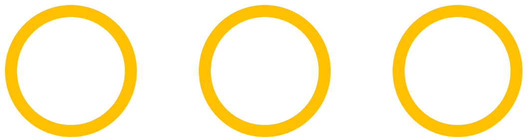




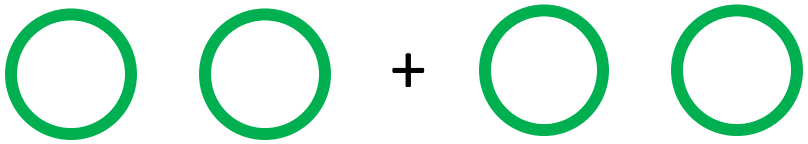
Add two hidden nodes (green)

(Add rows or columns to show your work)

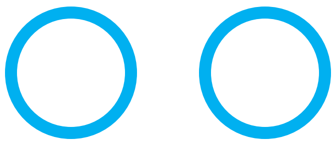




Δcount W1: +6; b1: +2

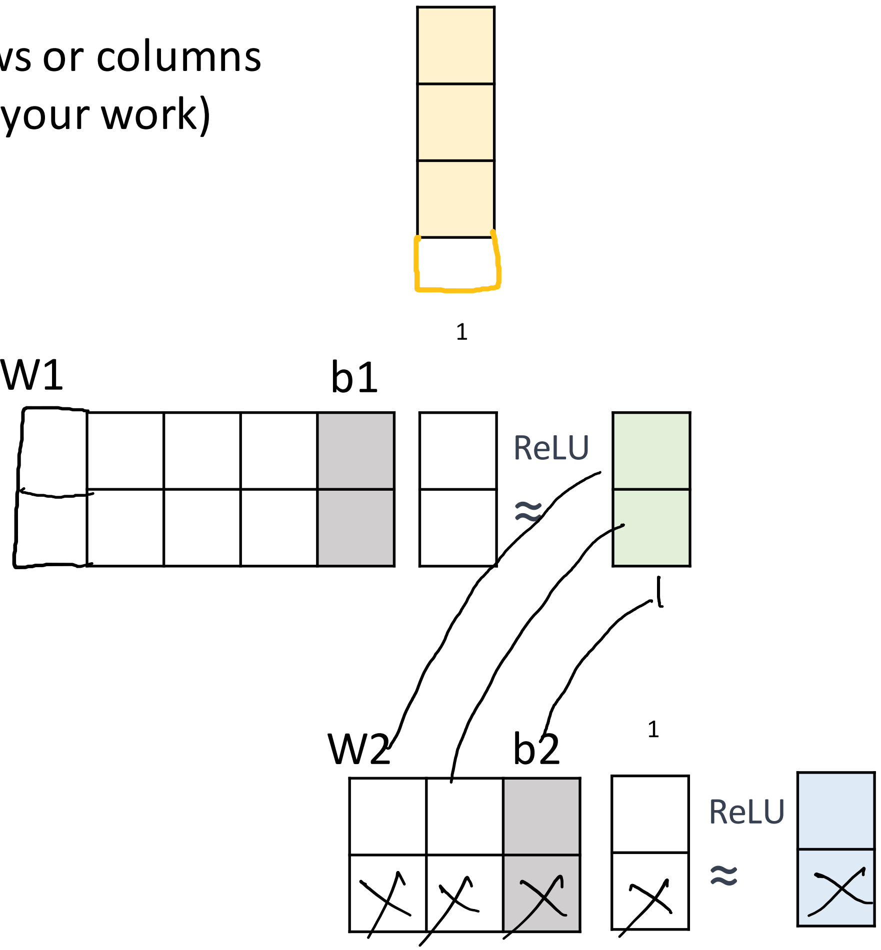


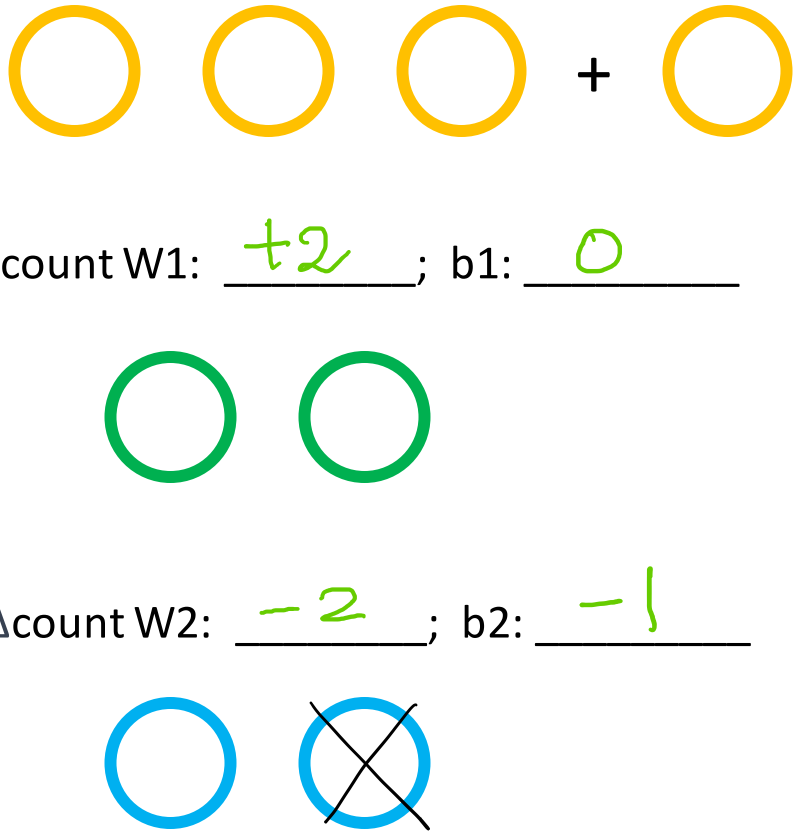
Δcount W2: +4; b2: 0



Add one input node & remove one output node

(Add rows or columns to show your work)





Δcount W1: __+2__; b1: __0__

Δcount W2: __-2__; b2: __-1__

Batch / Mini-Batch

CSCI 5722/4722 Computer Vision

University of Colorado Boulder

Batch

1	-2	1	-2	0	-2	1	2	-1	0	2	-1
2	6	-1	4	1	0	4	1	-2	2	0	3
0	1	5	0	4	0	-1	0	3	0	-1	0
1	1	1	1	1	1	1	1	1	1	1	1

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

Mini Batch

1	-2	1	-2	0	-2	1	2	-1	0	2	-1
2	6	-1	4	1	0	4	1	-2	2	0	3
0	1	5	0	4	0	-1	0	3	0	-1	0

1 1 1 1 1 1 1 1 1 1 1 1

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

2 Layers

Mini-batch size = 2
Calculate 2nd batch

ReLU: ✗

1	-2	1	-2	0	-2	1	2	-1	0	2	-1
2	6	-1	4	1	0	4	1	-2	2	0	3
0	1	5	0	4	0	-1	0	3	0	-1	0
1	1	1	1	1	1	1	1	1	1	1	1

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

1	1	1	1	1	1	1	1	1	1	1	1

1	1	1	1	0
1	0	0	-1	2

2 Layers

Mini-batch size = 2
Calculate 4th batch

ReLU: ✗

4th

1	-2	1	-2	0	-2	1	2	-1	0	2	-1
2	6	-1	4	1	0	4	1	-2	2	0	3
0	1	5	0	4	0	-1	0	3	0	-1	0
1	1	1	1	1	1	1	1	1	1	1	1

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

						~~-2~~	0				
						3	1				
						1	0				
						~~-3~~	1				
1	1	1	1	1	1	1	1	1	1	1	1

1	1	1	1	0
1	0	0	-1	2

						4	2				
						2	1				

a c
b d

🔑 a+b=6; c+d=3

Cross Entropy Loss

CSCI 5722/4722 Computer Vision

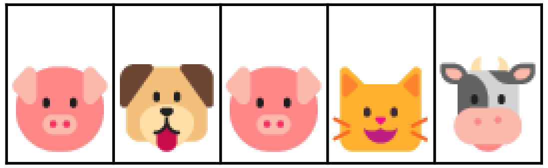
University of Colorado
Boulder

Softmax for a mini batch

1 1 1 1 1

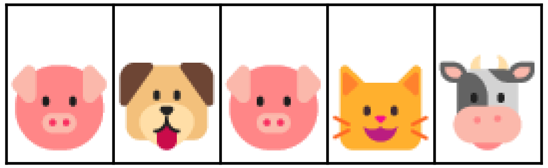
-6	-6	2	-8	-6
-5	2	0	2	2
0	-8	-6	-7	0
2	0	-7	-6	-7

Predictions vs. Targets



0	0	.9	0	0
0	.9	.1	1	.9
.1	0	0	0	.1
.9	.1	0	0	0

High vs. Low Loss

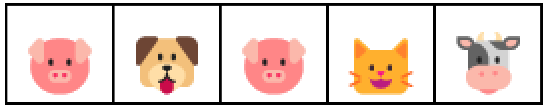


dog	0	0	.9	0	0
cat	0	.9	.1	1	.9
cow	.1	0	0	0	.1
pig	.9	.1	0	0	0

Is loss high, medium, low?

dog		1			
cat				1	
cow					1
pig	1		1		

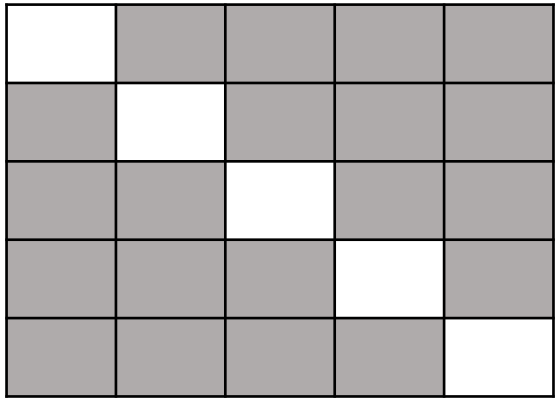
Calculate CE Loss



dog		1			
cat				1	
cow					1
pig	1		1		

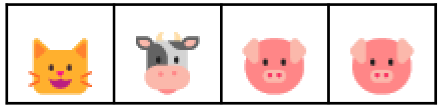
dog	0	0	.9	0	0
cat	0	.9	.1	1	.9
cow	.1	0	0	0	.1
pig	.9	.1	0	0	0

-log(0) ≈
-log(0.1) ≈
-log(0.9) ≈
-log(1) =



Calculate CE Loss

	P_1	P_2	P_3	P_4
cat	1			
cow		1		
pig			1	1



	Q_1	Q_2	Q_3	Q_4
cat	.9	0	.1	.1
cow	.1	.1	.9	0
pig	0	.9	0	.9

-log(0) ≈32
-log(0.1) ≈ 3
-log(0.9) ≈ 0.2
-log(1) = 0

T

0.2			
	3		
		32	0.2

	cat	cow	pig
P_1	1		
P_2		1	
P_3			1
P_4			1

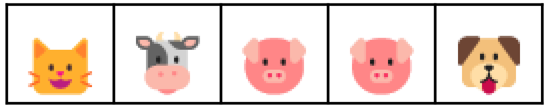
0.2				a
L	3			b
	M	32		c
		H	0.2	d

L

a+b+c+d=35.4

Calculate CE Loss

	P_1	P_2	P_3	P_4	P_5
dog					1
cat	1				
cow		1			
pig			1	1	



	Q_1	Q_2	Q_3	Q_4	Q_5
dog	0	0	.9	.1	1
cat	0	.9	.1	.9	0
cow	.1	.1	0	0	0
pig	.9	0	0	0	0

				0
32				
	3			
		32	32	

-log(0) ≈32
-log(0.1) ≈ 3
-log(0.9) ≈ 0.2
-log(1) = 0

T

	dog	cat	cow	pig
P_1		1		
P_2			1	
P_3				1
P_4				1
P_5	1			

32					a
H	3				b
	M	32			c
		H	32		d
			H	0	e

L

a+b+c+d+e=99

NumPy by Hand ✍️ [Perceptron]

CSCI 5722/4722 Computer Vision

University of Colorado Boulder

ReLU

```
1. p = z > 0
2. a = np.zeros((3,1))
3. a[p] = z[p]
```

$$z = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$$

Perceptron

```
1.  Wb = np.__stack((W,___))
2.  xh = np.__stack((W,___))
3.  z = np.matmul(Wb, Xh)
4.  p = x ___ 0
5.  a = np.zeros((___,1))
6.  a[p] = z[p]
```

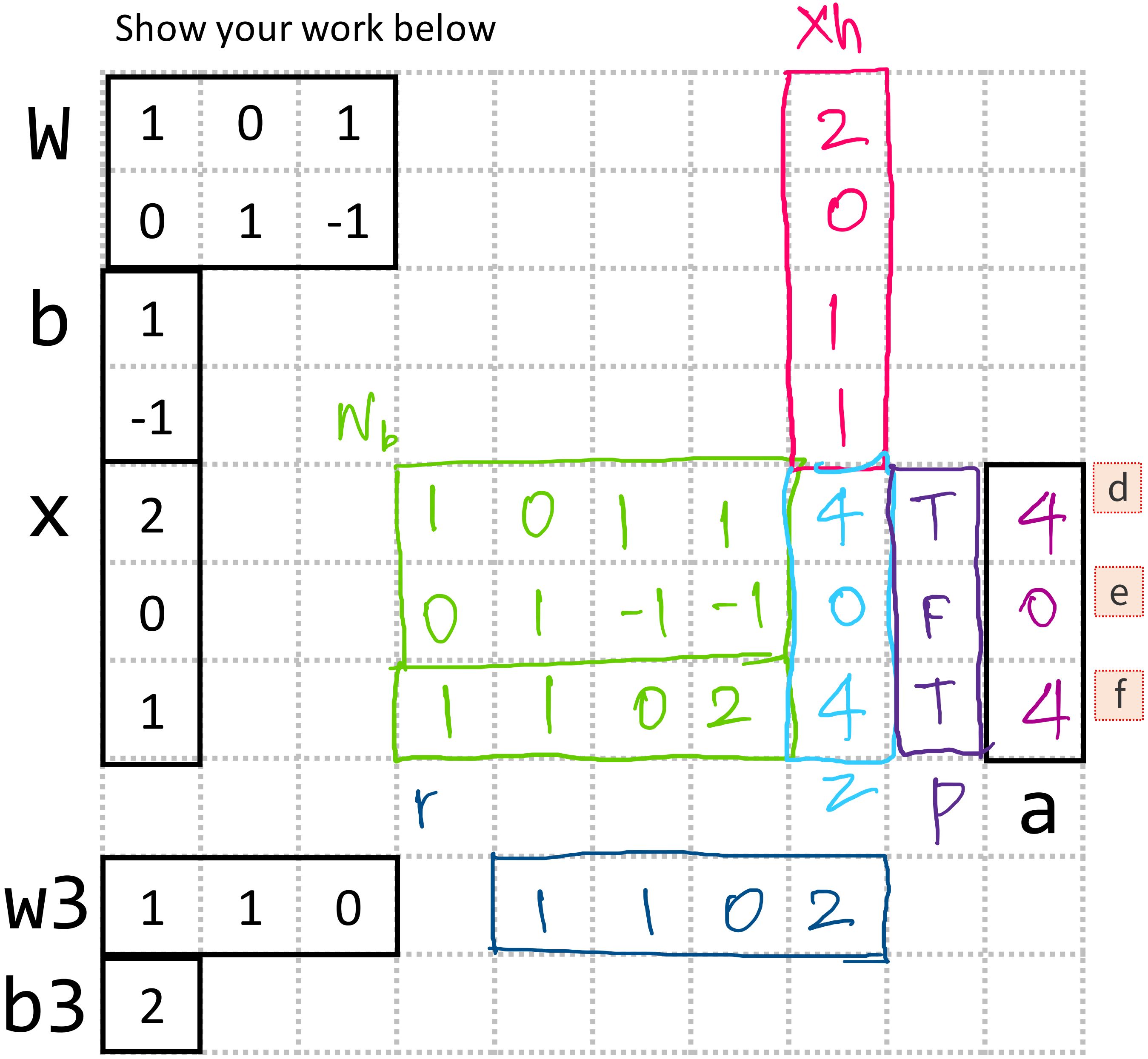
$$W = \begin{bmatrix} 1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \qquad x = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix}$$

Add a node

1. Wb = np.hstack((W,b)) a
2. xh = np.vstack((~~W~~ X,[__1__]))
3. r = np.__h__stack((w3,[b3])) b
4. Wb = np.__v__stack((Wb,r)) c
5. z = np.matmul(Wb, Xh)
6. p = x > 0
7. a = np.zeros((3,1))
8. a[p] = z[p]

a+d+e+f=9

Show your work below



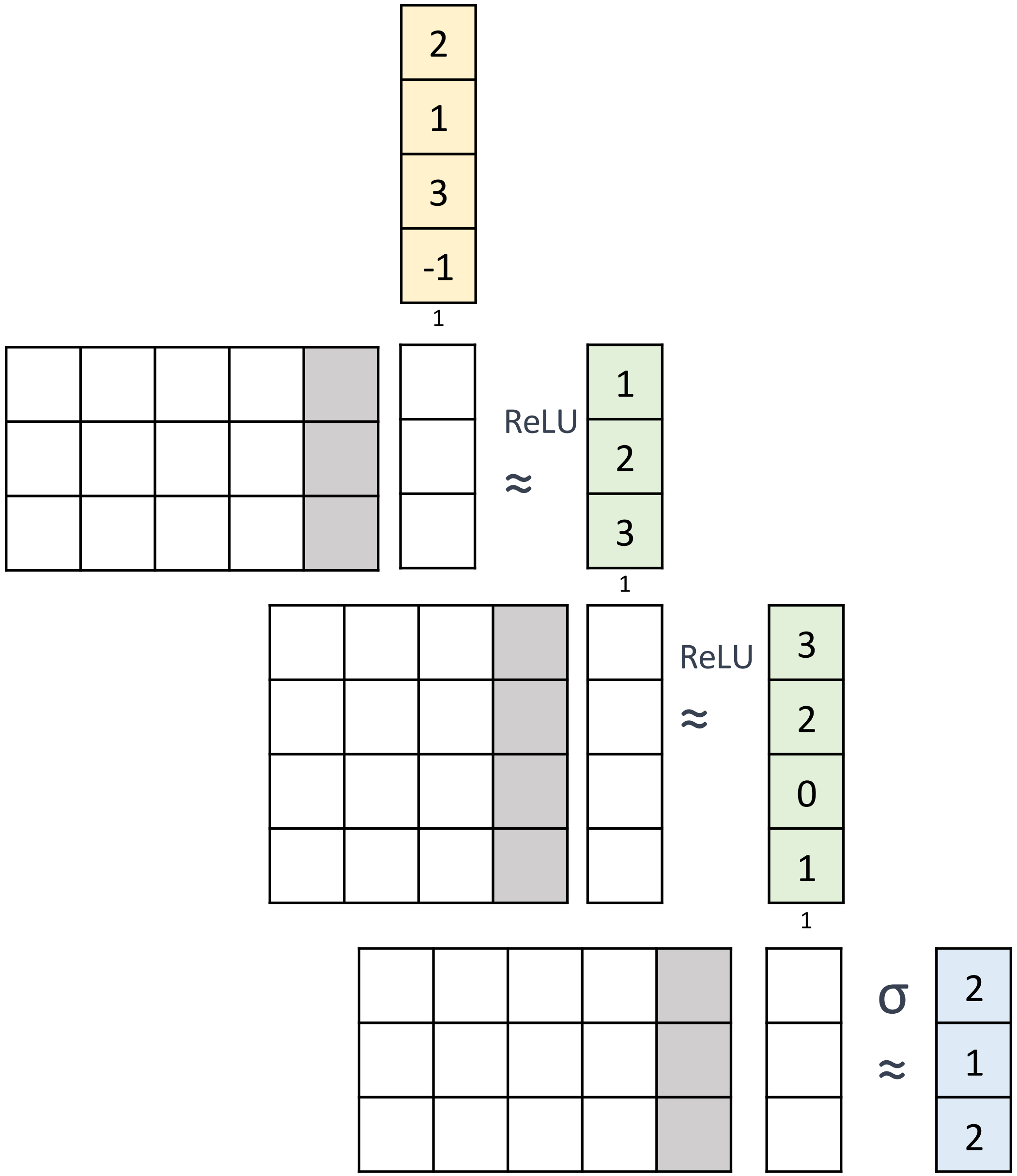
Pytorch by Hand ✍️ [MLP]

CSCI 5722/4722 Computer Vision

University of Colorado Boulder

pytorch ← Matrix

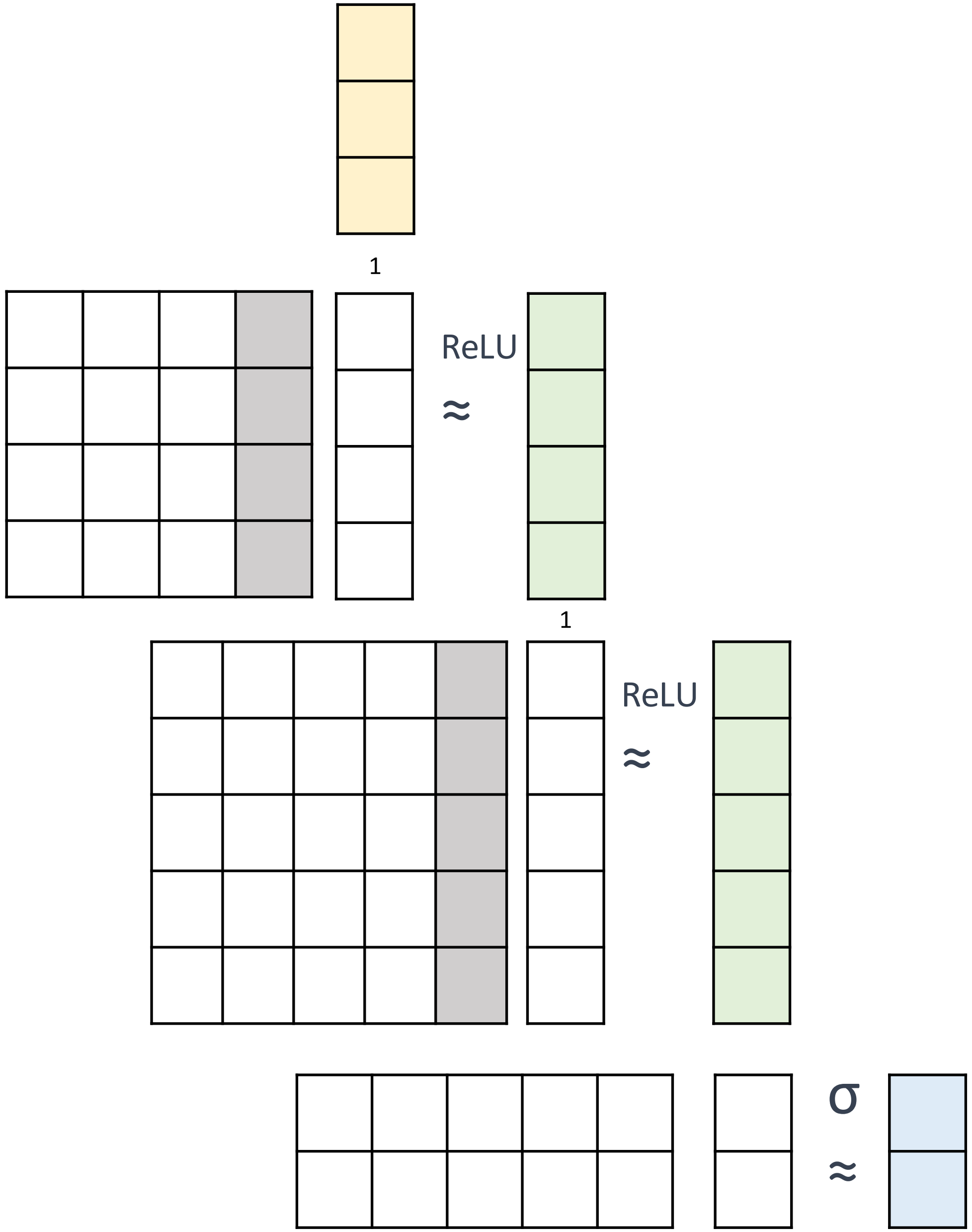
```
1.  mlp_model = nn.Sequential(
2.      nn.__________( ___, ___, bias = ___ ),
3.      nn._________(),
4.      nn._________( ___, ___, bias = ___ ),
5.      nn._________(),
6.      nn._________( ___, ___, bias = ___ ),
7.      nn._________(),
8.  )
```



pytorch ← Matrix

```
1.   mlp_model = nn.Sequential(
                      a          b          c
2.       nn.Linear( _3_, _4_, bias = _T_ ),

3.       nn.ReLU(),
                      d          e          f
4.       nn.Linear( _4_, _5_, bias = _T_ ),

5.       nn.ReLU(),
                      g          h          i
6.       nn.Linear( _5_, _2_, bias = _F_ ),

7.       nn.Sigmoid(),

8.   )
```

🔑 a+d+g=12;
b+e+h=11



Fill in the missing arguments

```
mlp_model = nn.Sequential(
    nn.Linear(128, 64),
    nn.ReLU(),
              a                b
    nn.Linear( __64__, __512__ ),
    nn.ReLU(),
    nn.Linear(512, 32),
    nn.Sigmoid(),
)
```

a+b=576

Visualize Pytorch Models
Torchview

README   MIT license

torchview

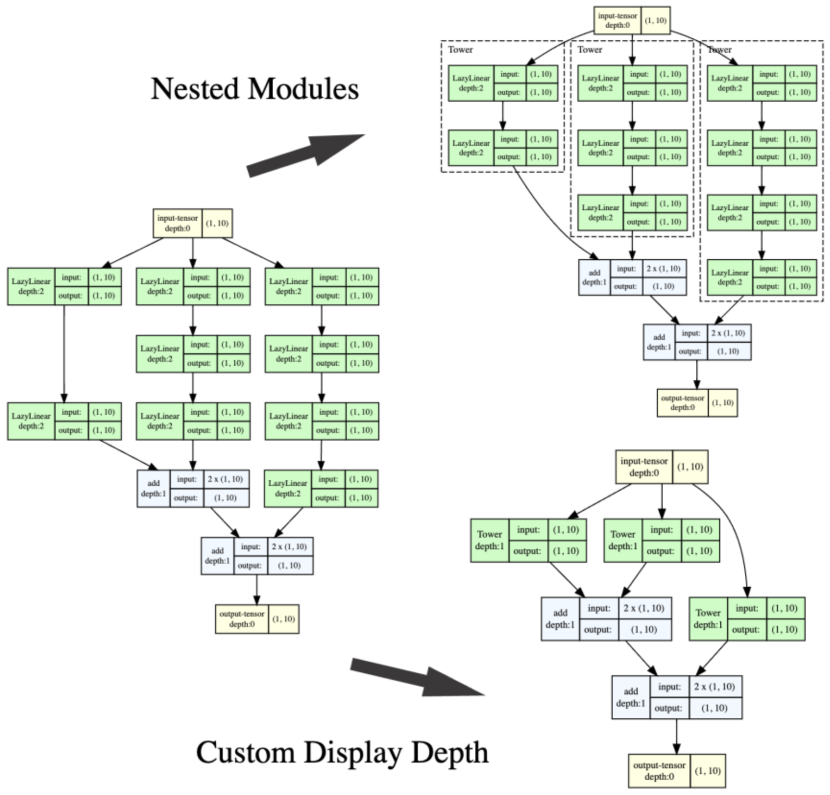
python 3.7+ | pypi package 0.2.6 | conda-forge v0.2.6 | testing passing | license MIT | codecov unknown | downloads 49k

Torchview provides visualization of pytorch models in the form of visual graphs. Visualization includes tensors, modules, torch.functions and info such as input/output shapes.

Pytorch version of `plot_model of keras` (and more)

Supports PyTorch versions $\geq$ 1.7.

Useful features



https://github.com/mert-kurttutan/torchview

Introduction Notebook
[Click to Open]

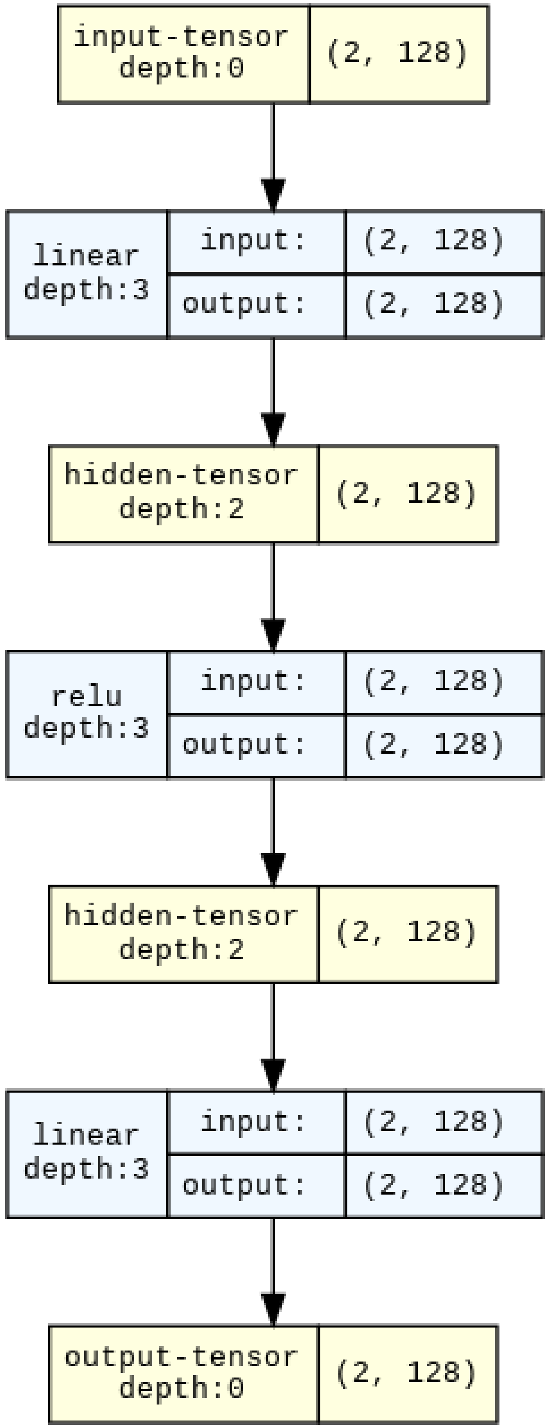
Introduction Notebook

[Link to Colab]

```
[18] class MLP(nn.Module):
         """Multi Layer Perceptron with inplace option.
         Make sure inplace=true and false has the same visual graph"""

         def __init__(self, inplace: bool = True) -> None:
             super().__init__()
             self.layers = nn.Sequential(
                 nn.Linear(128, 128),
                 nn.ReLU(inplace),
                 nn.Linear(128, 128),
             )

         def forward(self, x: torch.Tensor) -> torch.Tensor:
             x = self.layers(x)
             return x
```

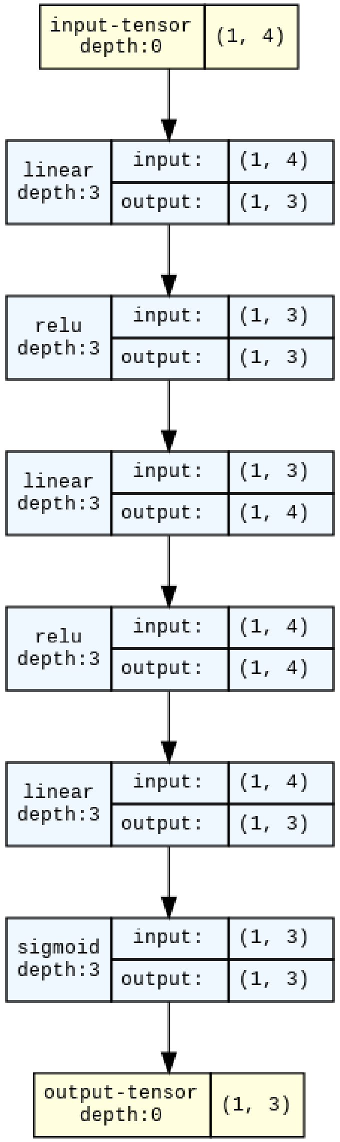
```
[22] model_graph_1 = draw_graph(
         MLP(), input_size=(2, 128),
         graph_name='MLP',
         hide_inner_tensors=False,
         hide_module_functions=False,
     )
```

```
[23] model_graph_1.visual_graph
```



```
[18] class MLP(nn.Module):
         """Multi Layer Perceptron with inplace option.
         Make sure inplace=true and false has the same visual graph"""

         def __init__(self, inplace: bool = True) -> None:
             super().__init__()
             self.layers = nn.Sequential(
                 nn.Linear(4, 3),
                 nn.ReLU(inplace),
                 nn.Linear(3, 4),
                 nn.ReLU(inplace),
                 nn.Linear(4,3),
                 nn.Sigmoid()
             )

         def forward(self, x: torch.Tensor) -> torch.Tensor:
             x = self.layers(x)
             return x
```

```
[22] model_graph_1 = draw_graph(
         MLP(), input_size=(1, 4),
         graph_name='MLP',
         hide_inner_tensors=True,
         hide_module_functions=False,
     )
```



Use Torchview to generative an MLP graph

Modify the example notebook [Link to Colab]