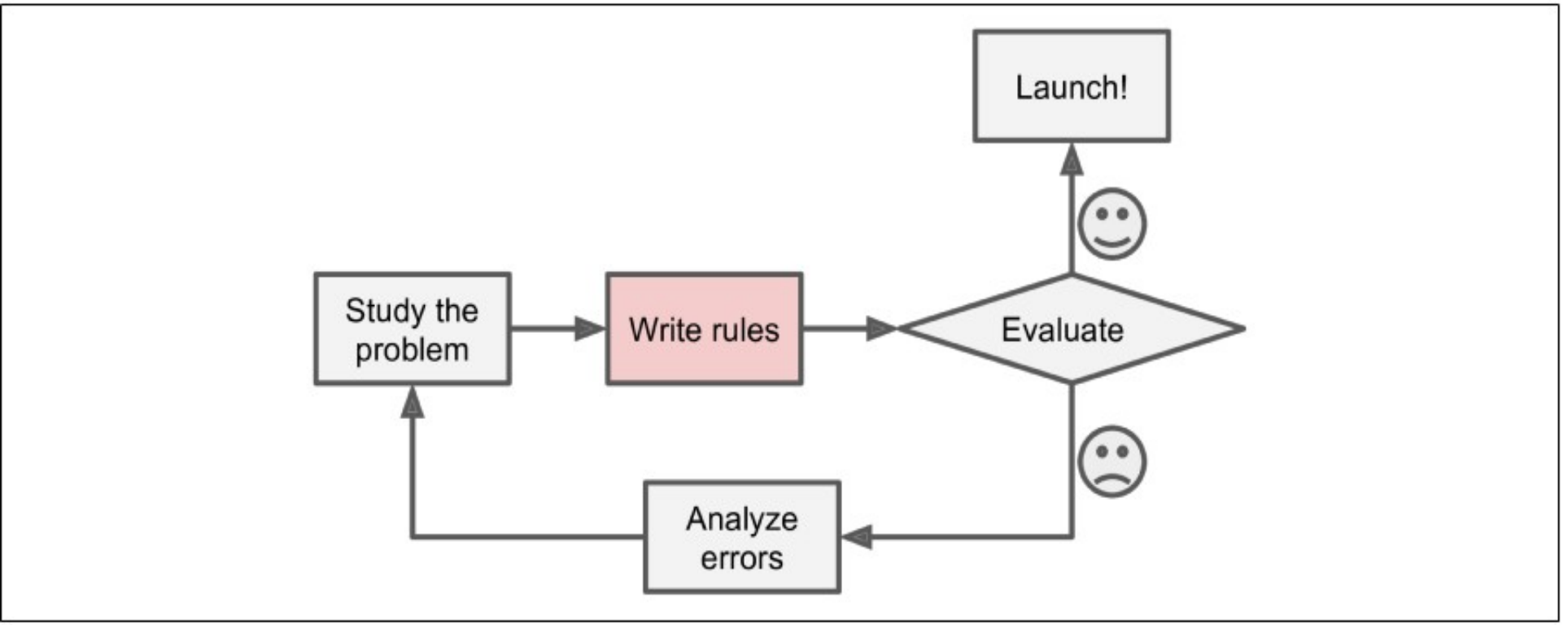
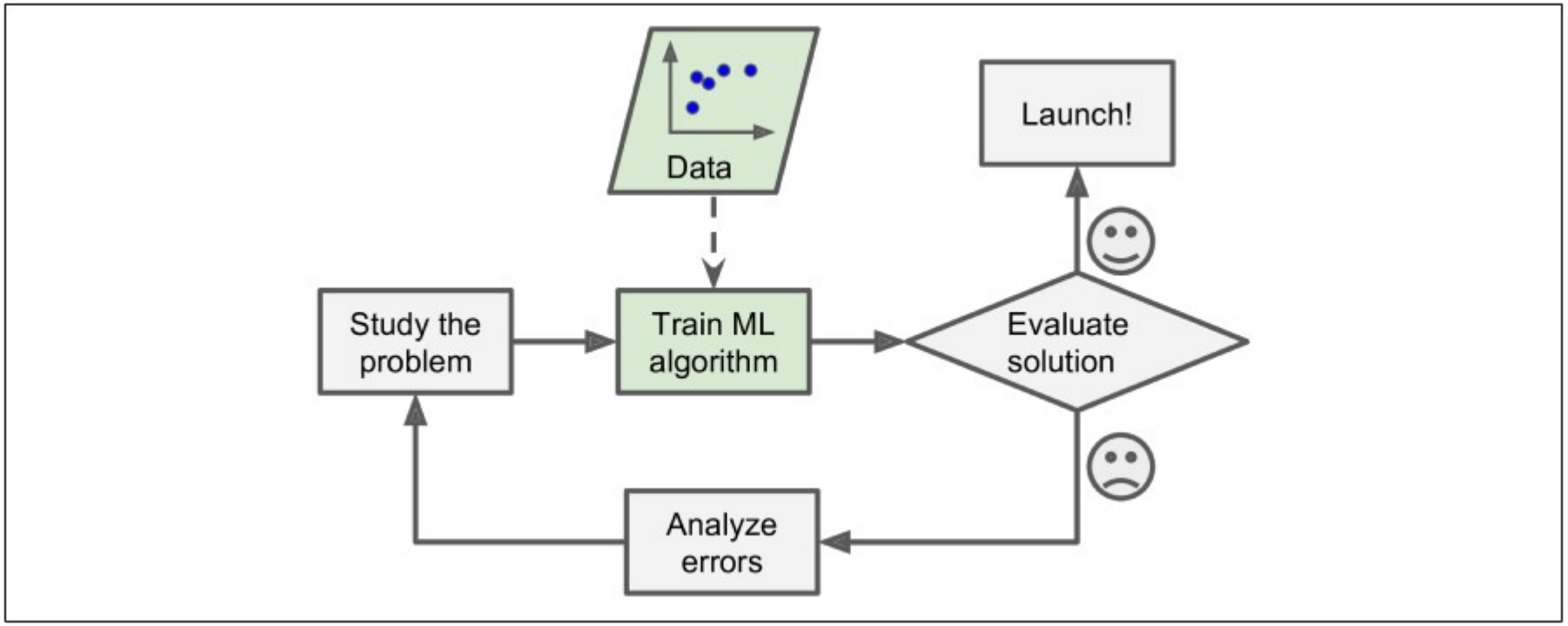
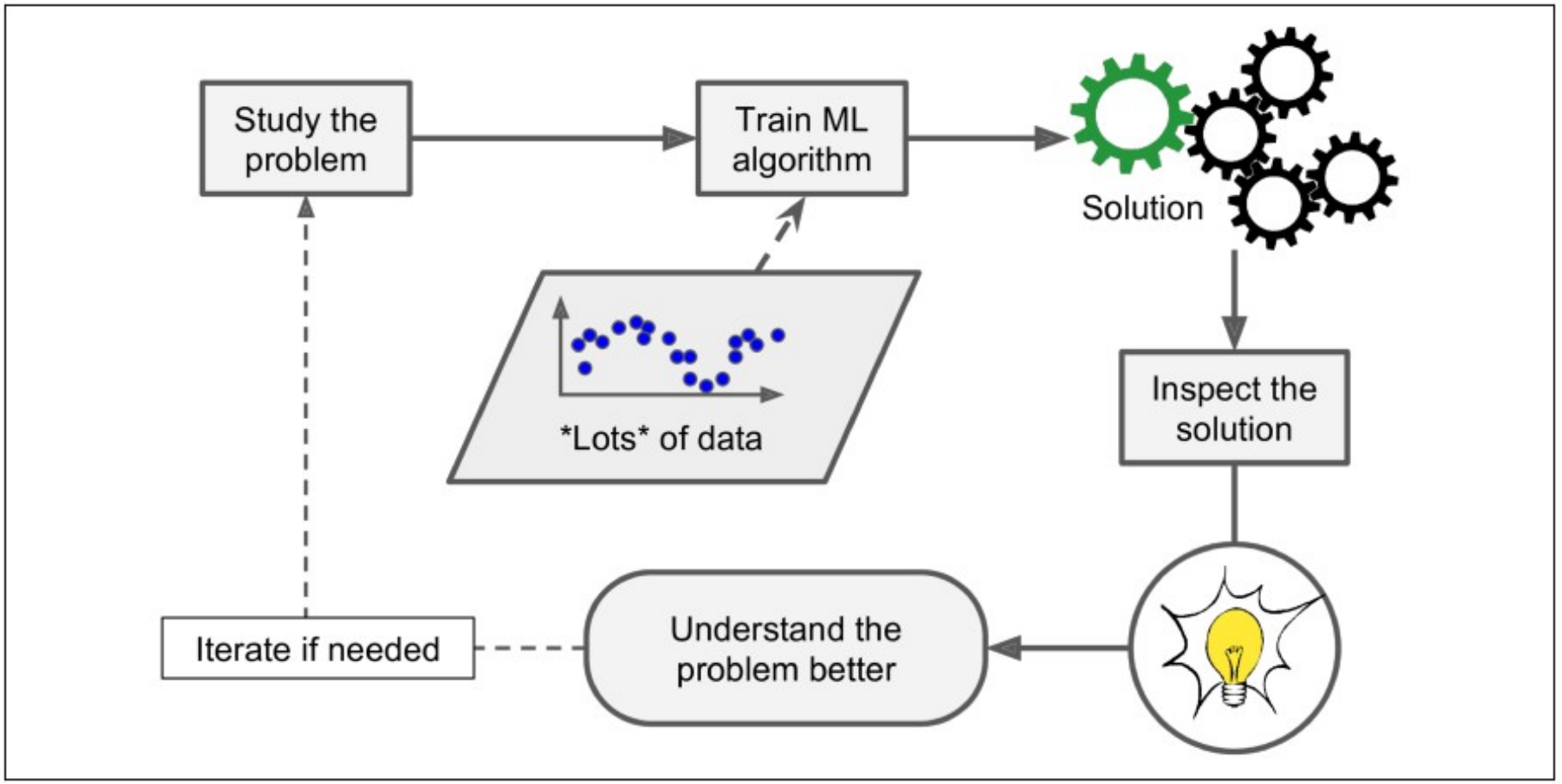
Machine learning. Introduction

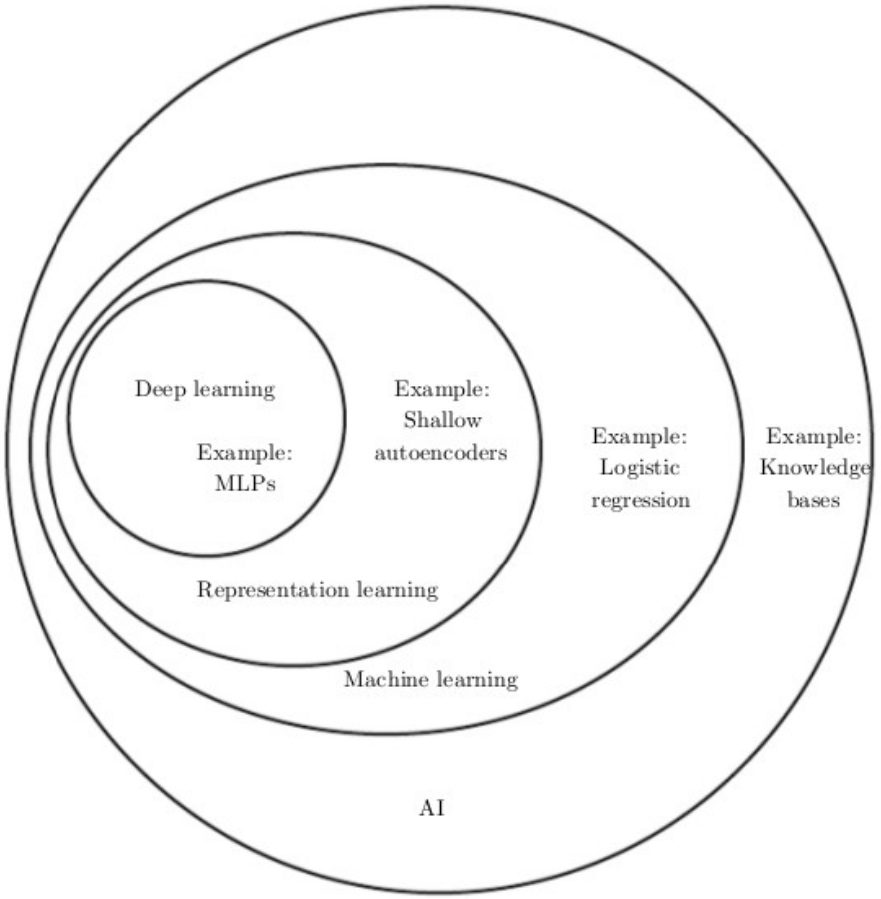

Launch!
Study the problem
Write rules
Evaluate
Analyze errors

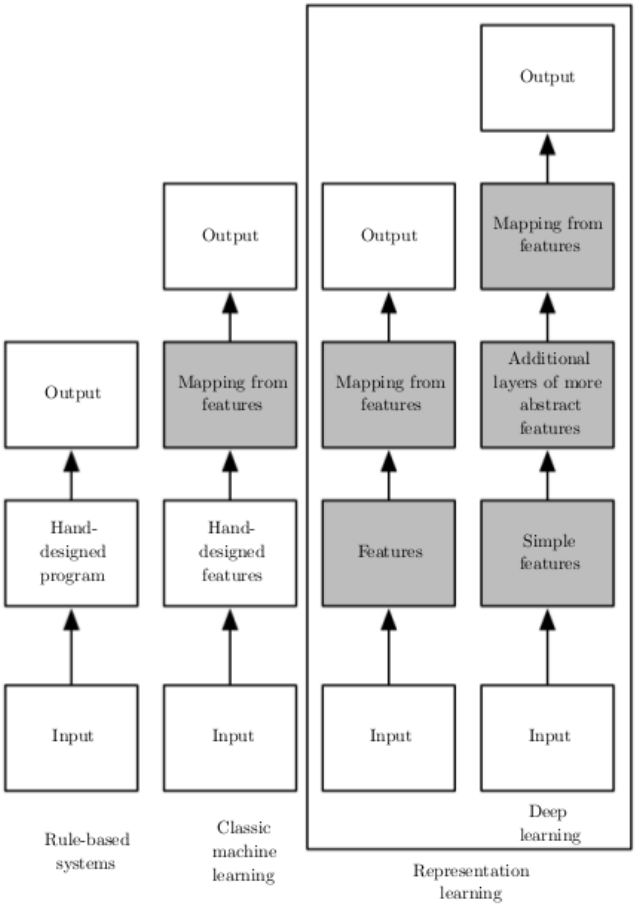

Data
Launch!
Study the problem
Train ML algorithm
Evaluate solution
Analyze errors

ML helps to understand the problem



Different types of ML





Types of learning

- **Supervised learning (classification, etc.)**

  - k-Nearest Neighbors

  - Linear Regression

  - Logistic Regression

  - Support Vector Machines (SVMs)

  - Decision Trees and Random Forests

  - Some Neural Networks

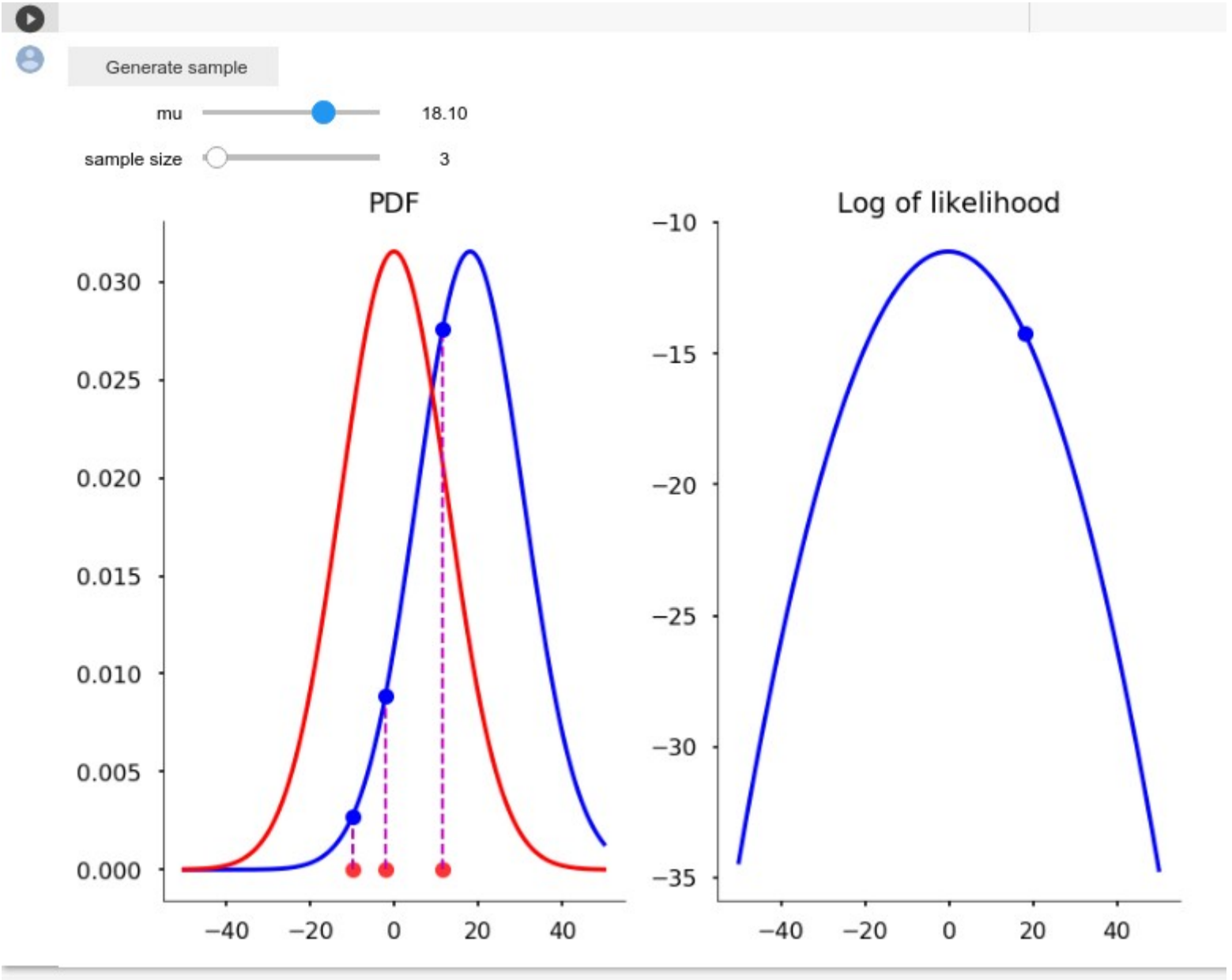
- **Semi-supervised learning (when partial labels are accessible, etc.)**

- **Reinforcement learning (play Atari game, etc.)**

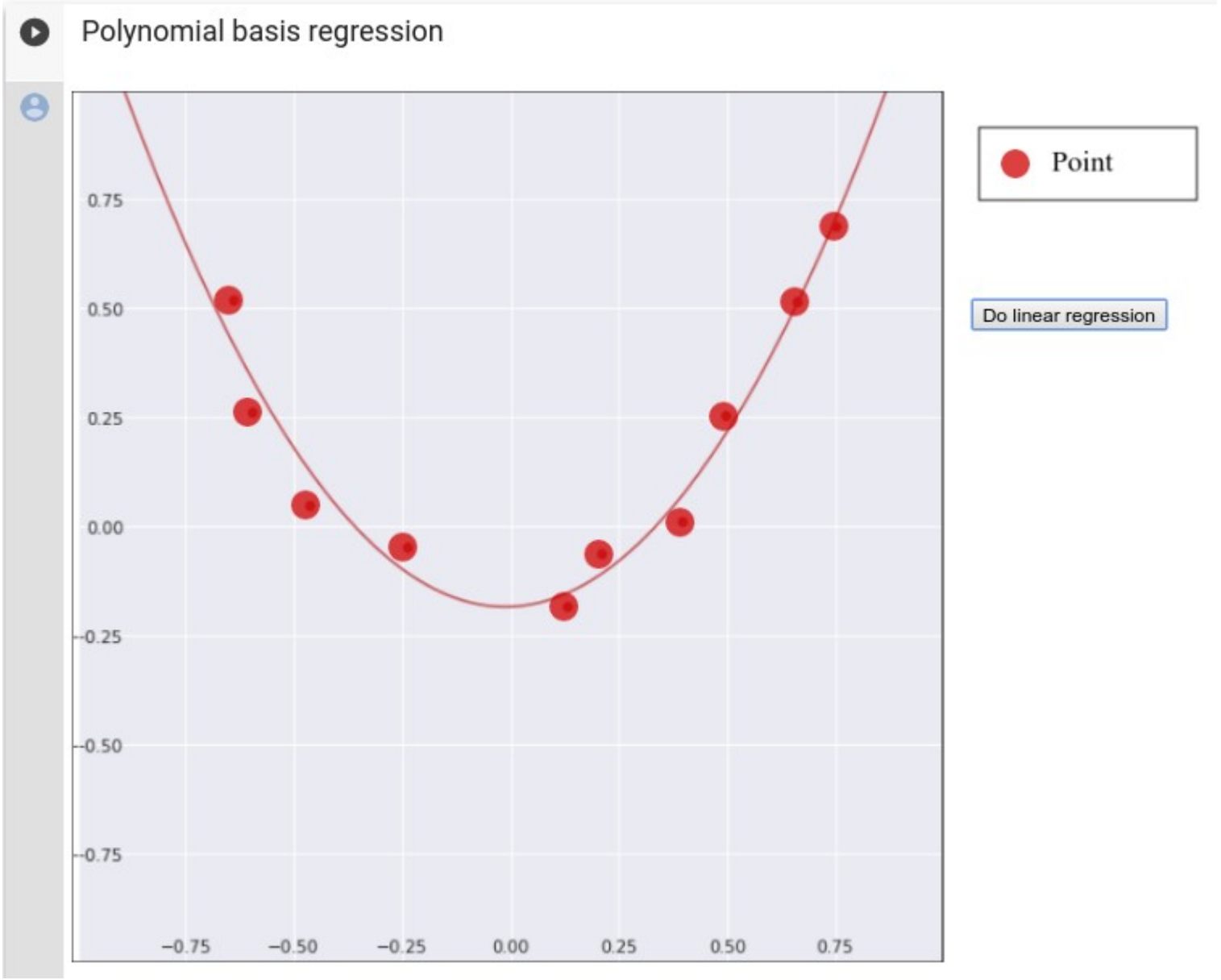
- **Unsupervised learning (clustering, etc.)**
  - Clustering
    * k-Means
    * Hierarchical Cluster Analysis (HCA)
    * Expectation Maximization
  - Visualization and dimensionality reduction
    * Principal Component Analysis (PCA)
    * Kernel PCA
    * Locally-Linear Embedding (LLE)
    * t-distributed Stochastic Neighbor Embedding (t-SNE)
  - Association rule learning
    * Apriori
    * Eclat

“Classical” machine learning

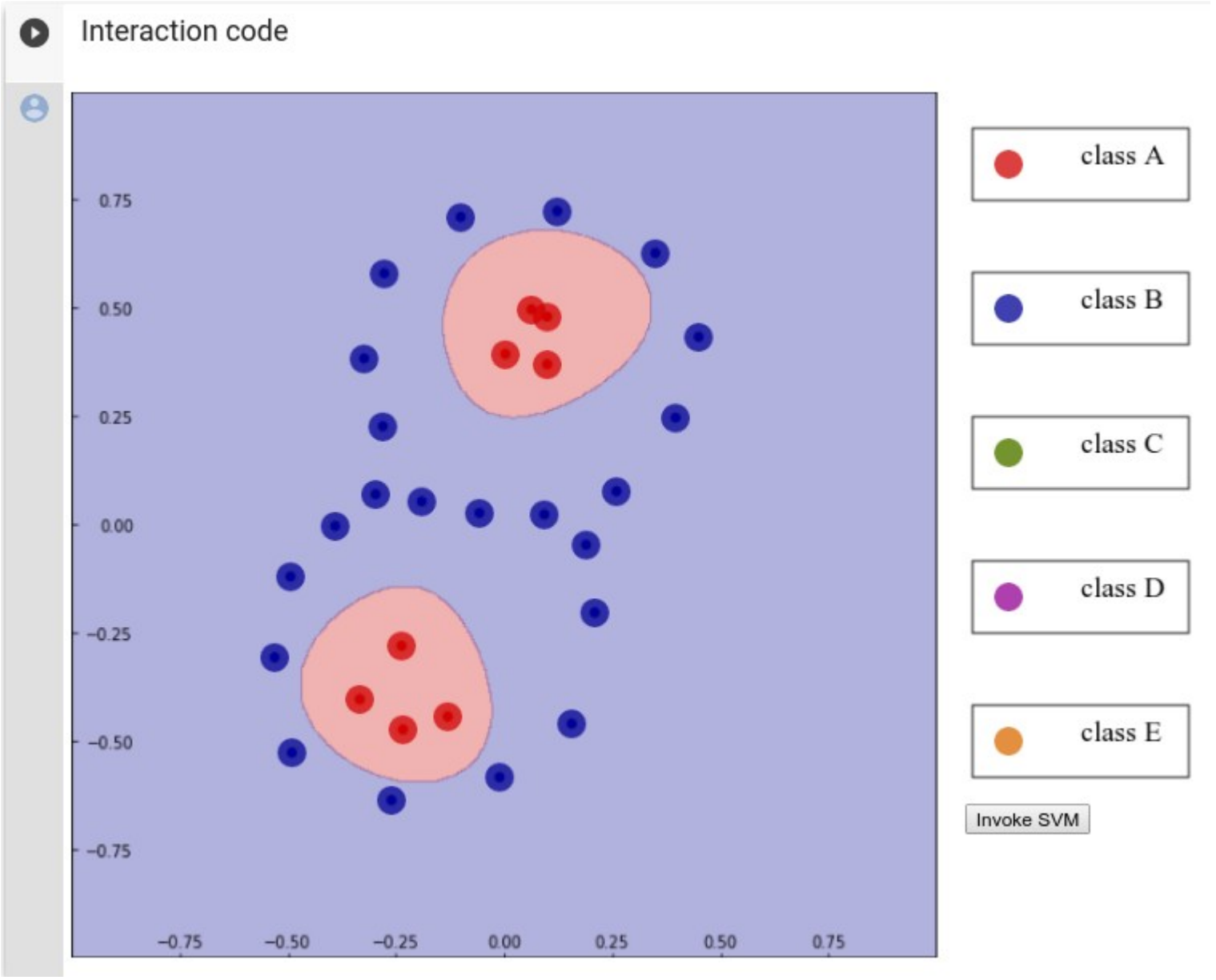
Maximum likelihood estimation



Regression

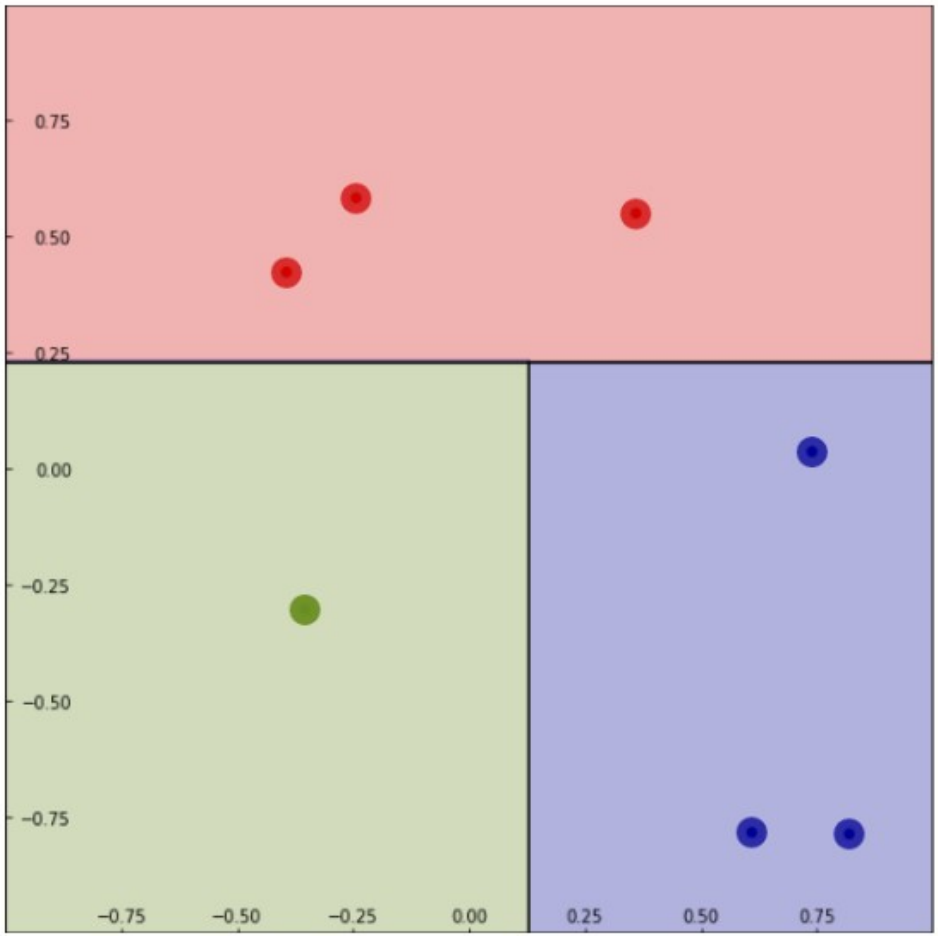

Polynomial basis regression
Point
Do linear regression
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75

Classification



Classification

Visualization code



class A

class B

class C

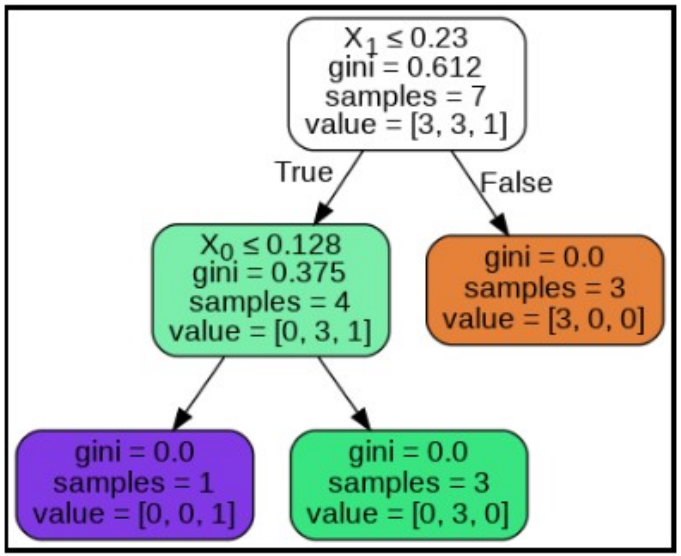
class D

class E

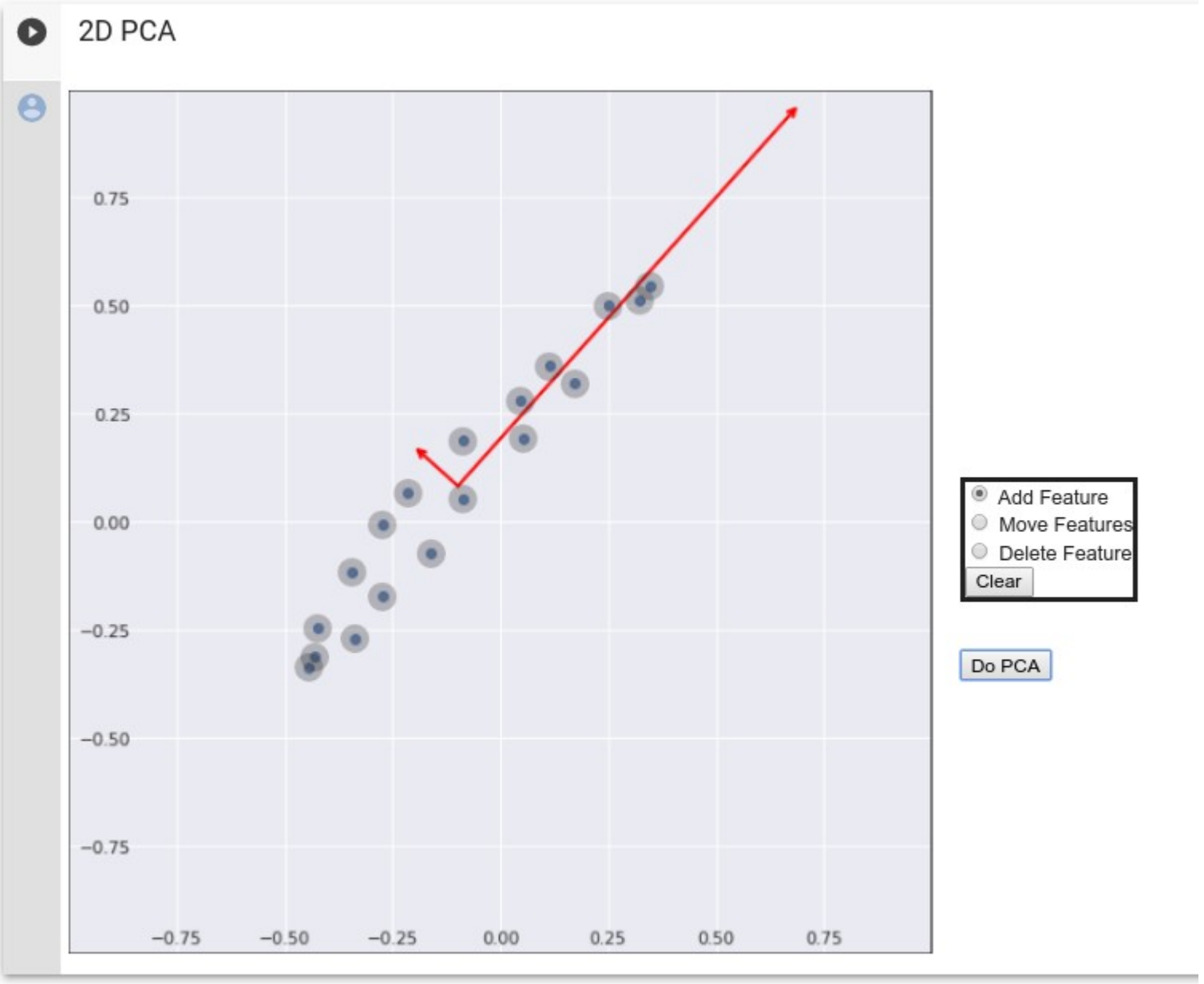
Invoke Decision Tree

Tree Depth:

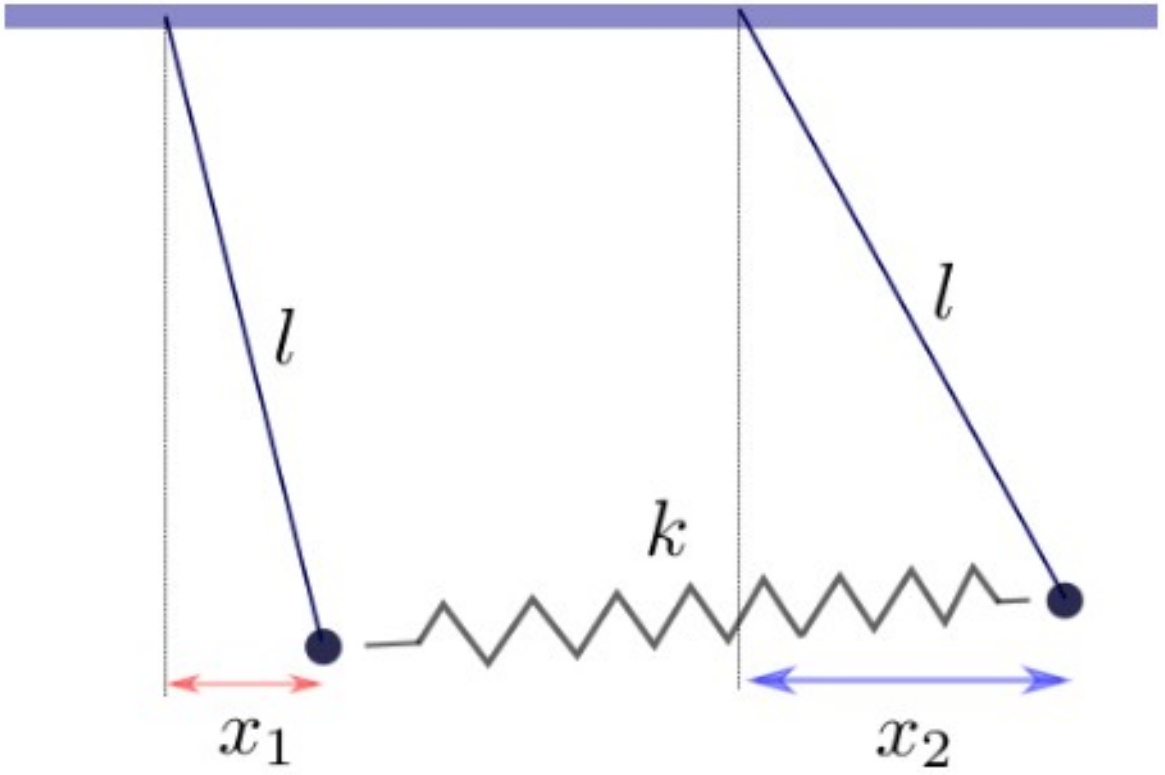
3



Dimensionality reduction



Consider the system



Equations of motion are coupled.

$$m\,\ddot{x}_1 = -\frac{mg}{l}x_1 + k(x_2 - x_1),$$
$$m\,\ddot{x}_2 = -\frac{mg}{l}x_2 - k(x_2 - x_1).$$

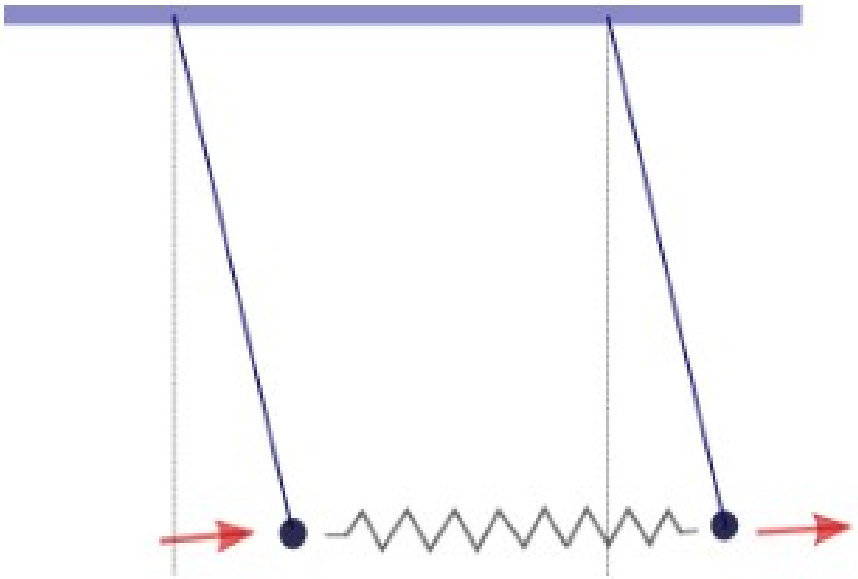
Why is that? Full energy (Hamiltonian) is the sum of two quadratic forms

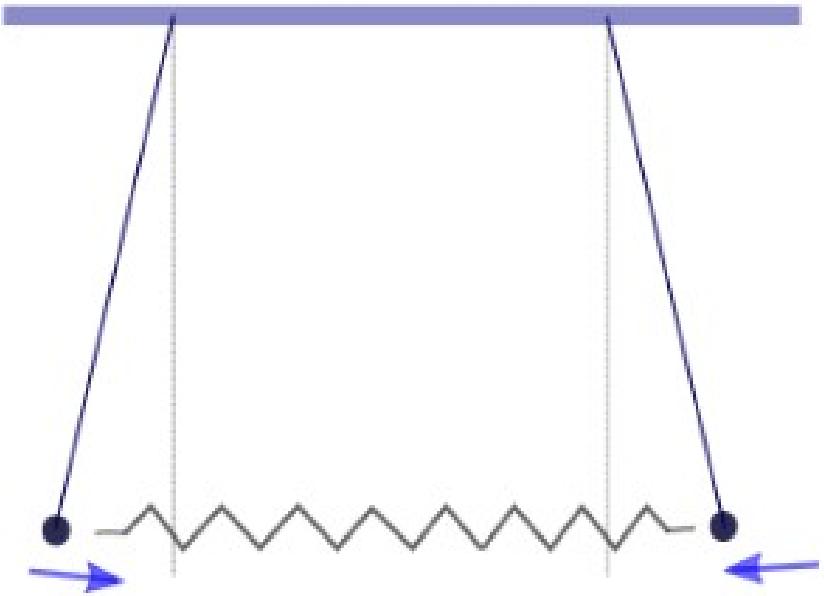
$$\mathcal{H} = \mathcal{K} + \mathcal{U},$$
$$\mathcal{K} = \frac{1}{2}m\dot{x}_1^2 + \frac{1}{2}m\dot{x}_2^2,$$
$$\mathcal{U} = \frac{1}{2}\left(\frac{g}{l} + \frac{k}{m}\right)(x_1^2 + x_2^2) - k\,x_1 x_2.$$

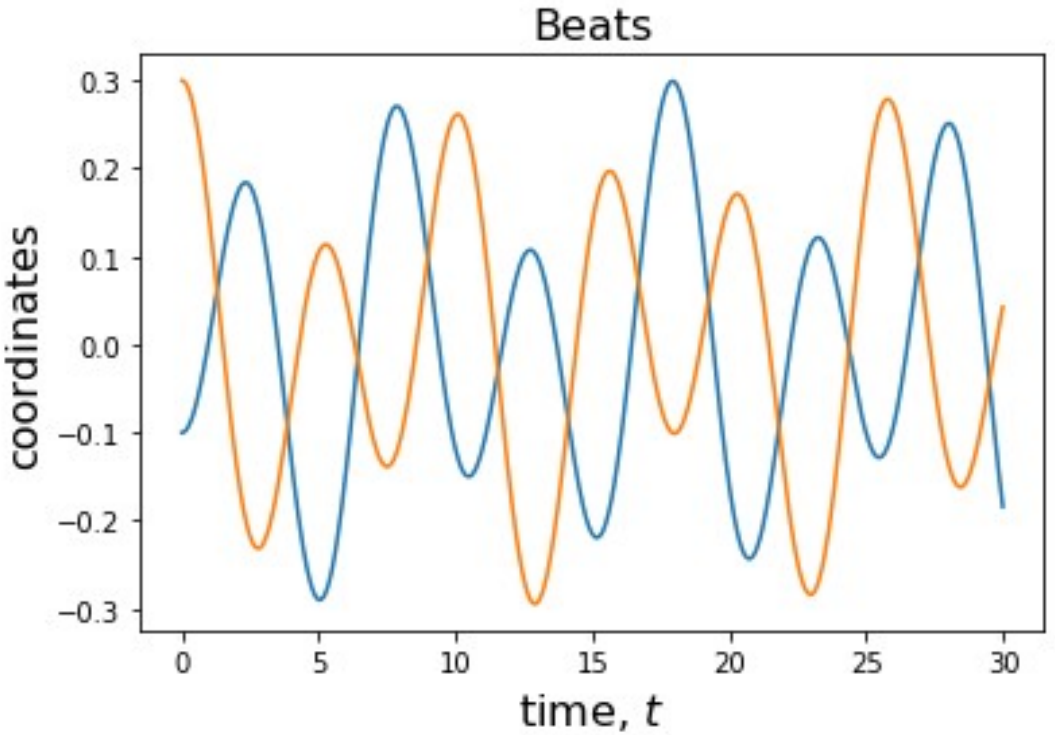
Normal modes

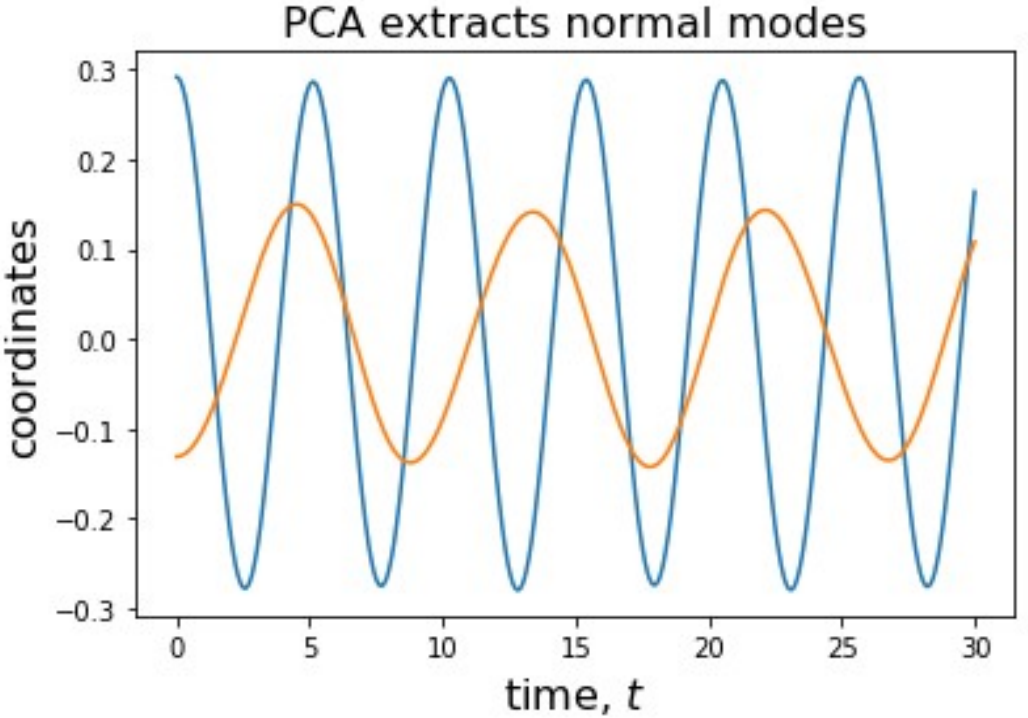
$$m\ddot{q}_1 = -\frac{mg}{l}q_1,$$

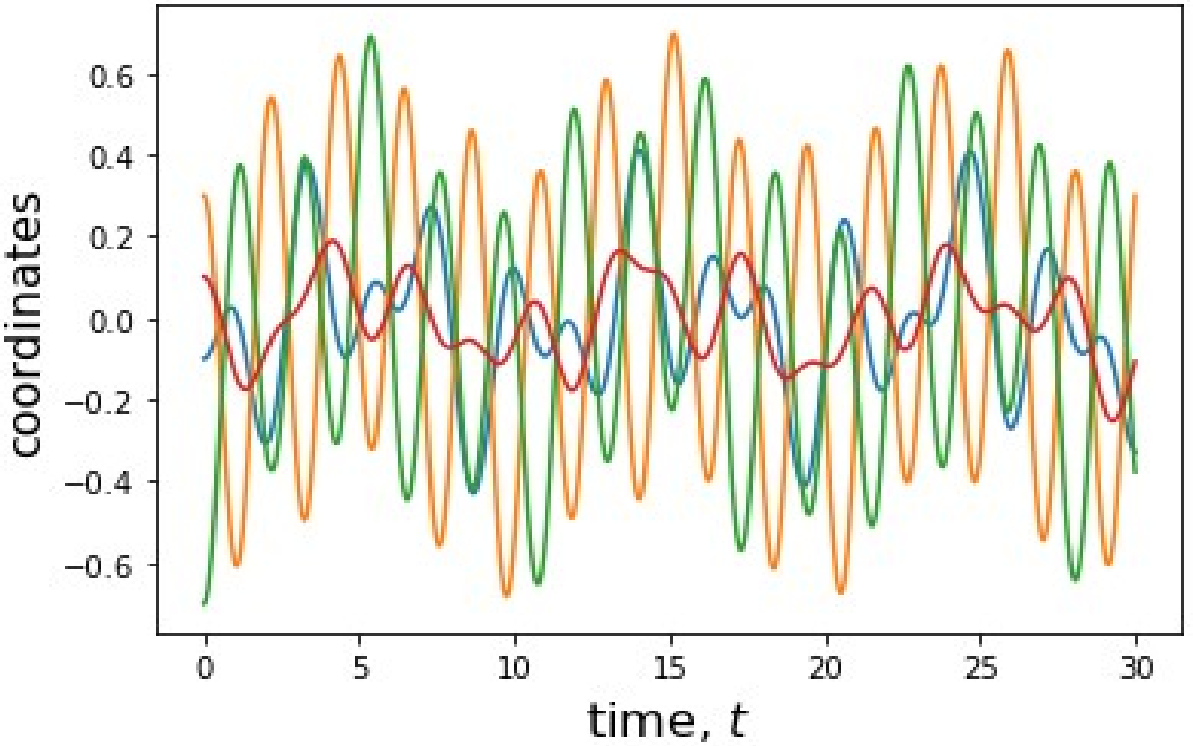
$$m\ddot{q}_2 = -m\left(\frac{g}{l} + \frac{2k}{m}\right)q_2.$$

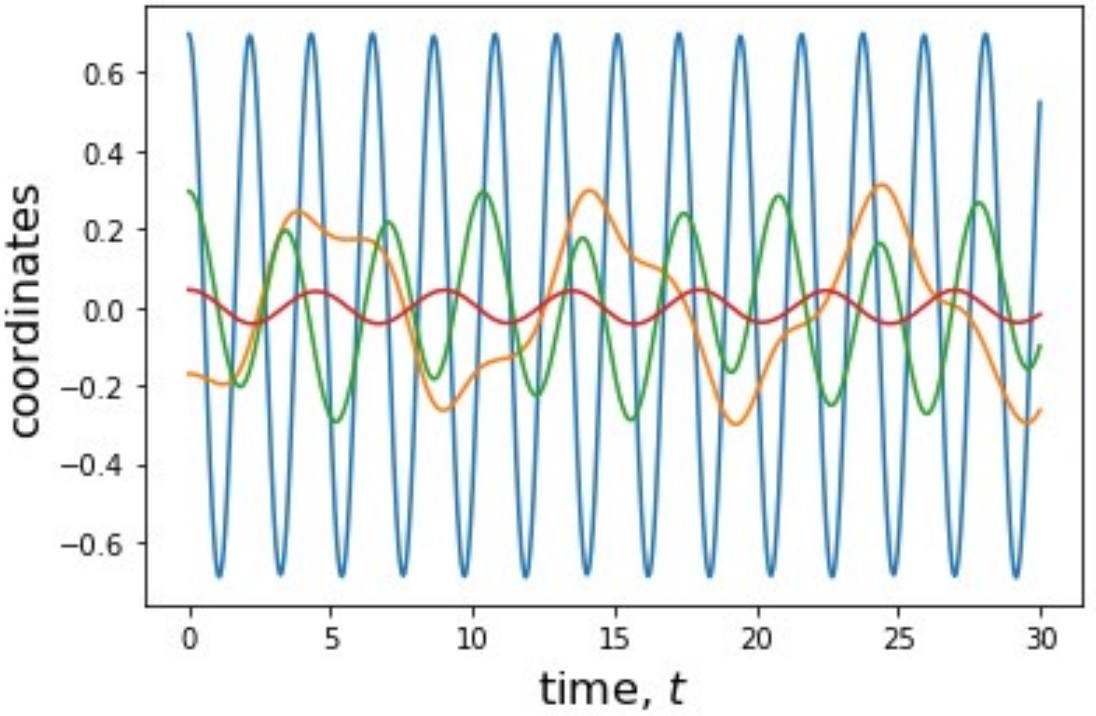




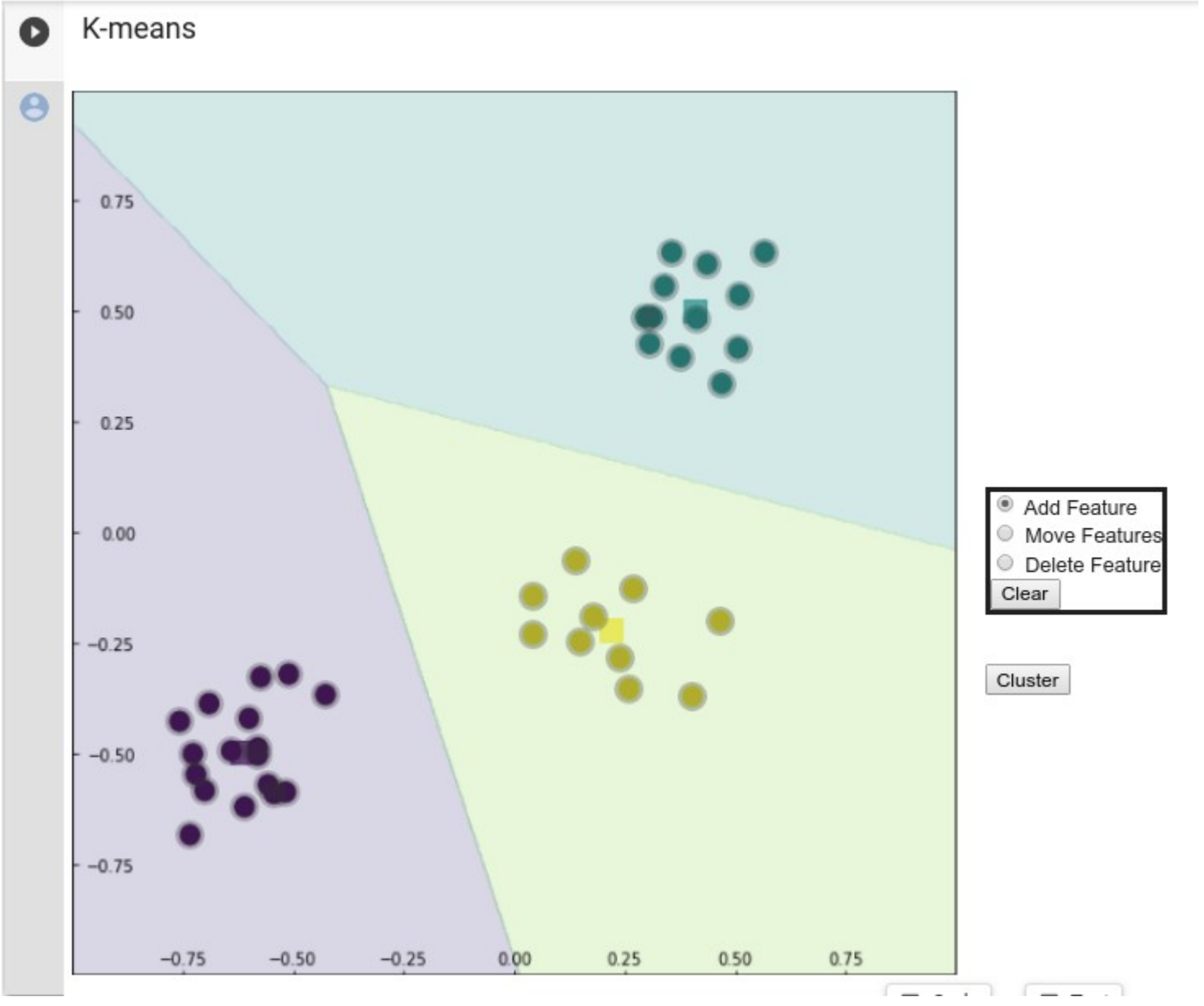

Beats
coordinates
time, t
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
0
5
10
15
20
25
30


PCA extracts normal modes
coordinates
time, t
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
0
5
10
15
20
25
30

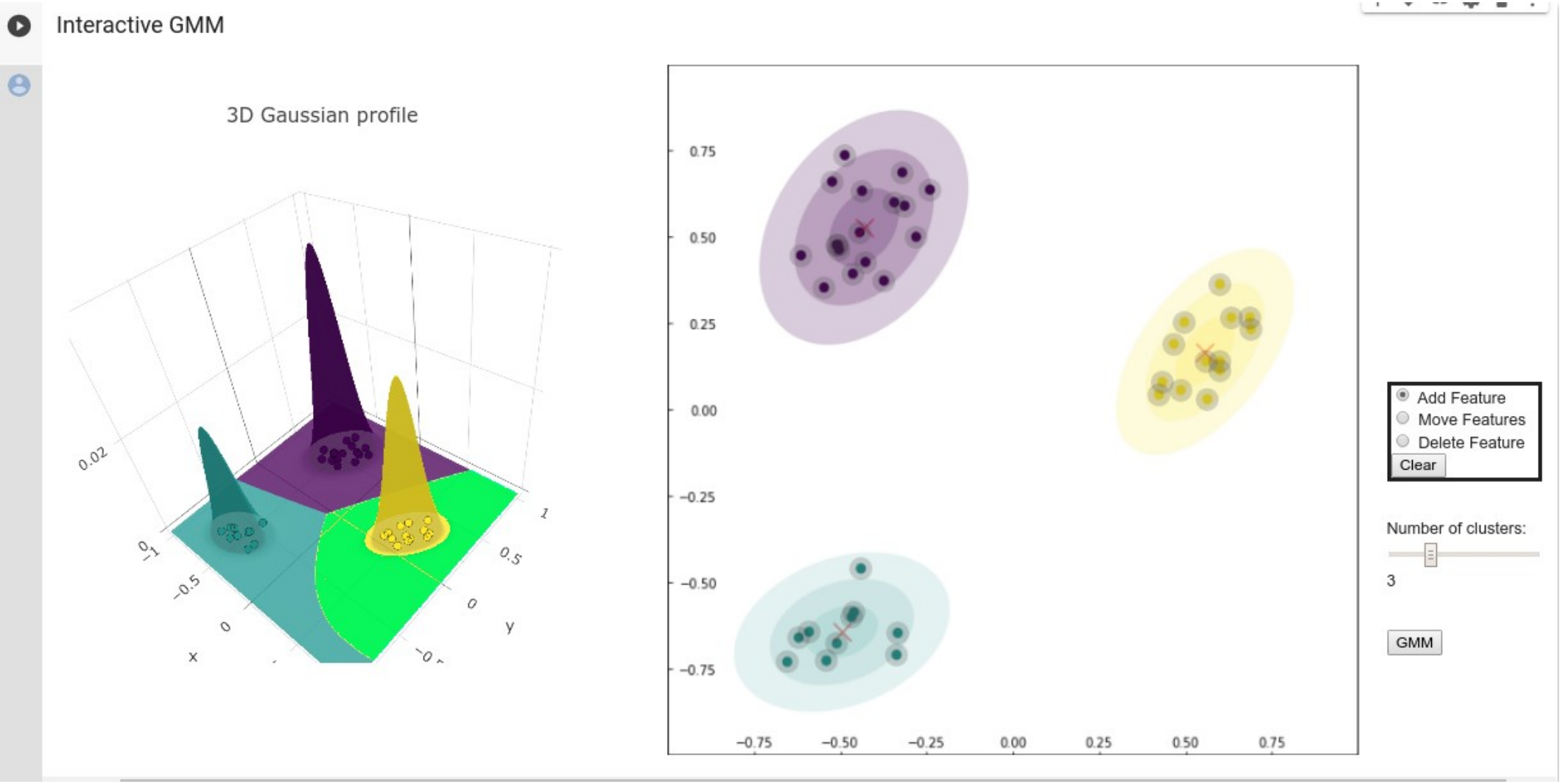

coordinates
time, t
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0
5
10
15
20
25
30


coordinates
time, t
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0
5
10
15
20
25
30

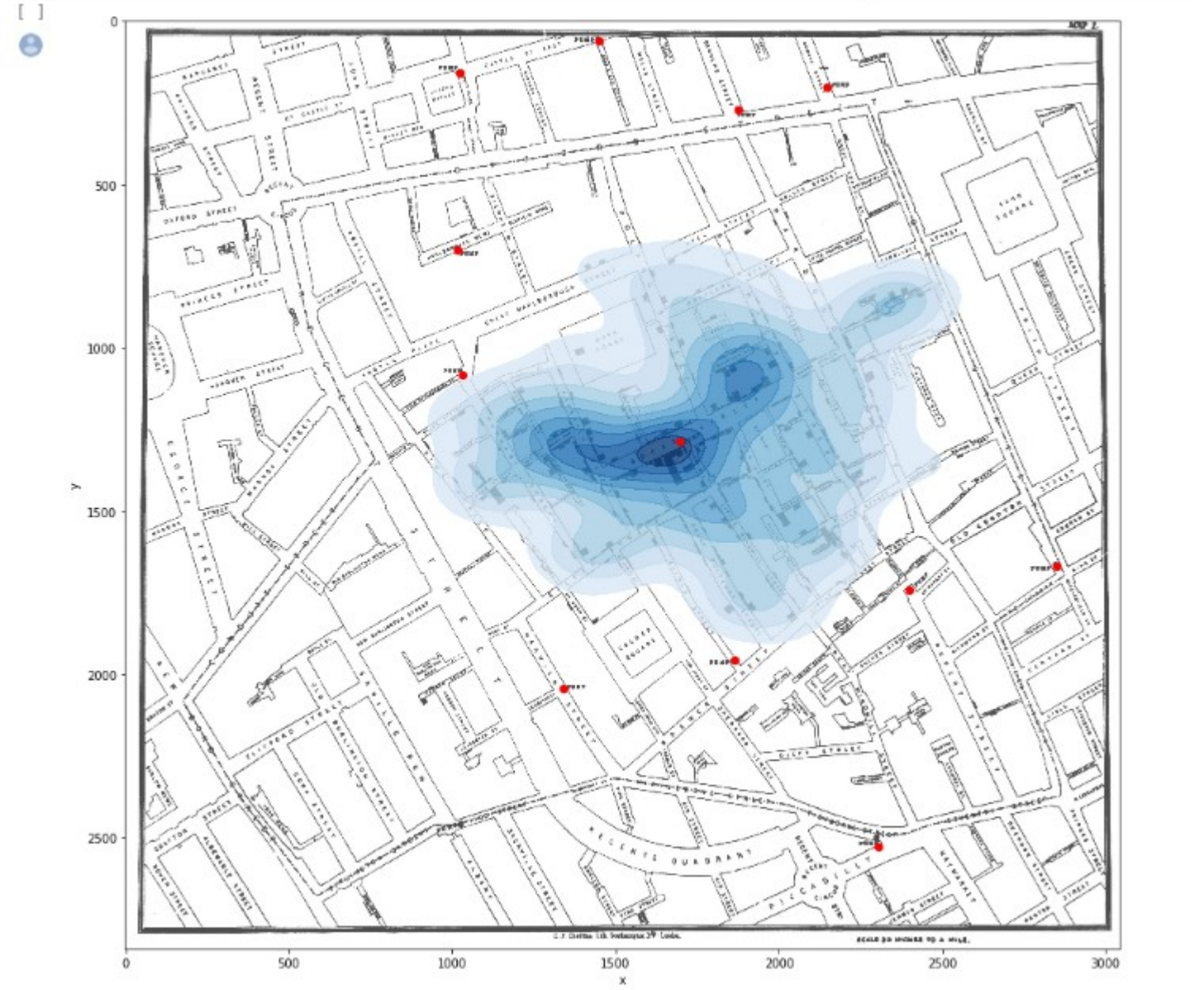
Clustering


K-means
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
Add Feature
Move Features
Delete Feature
Clear
Cluster

Gaussian Mixture Model



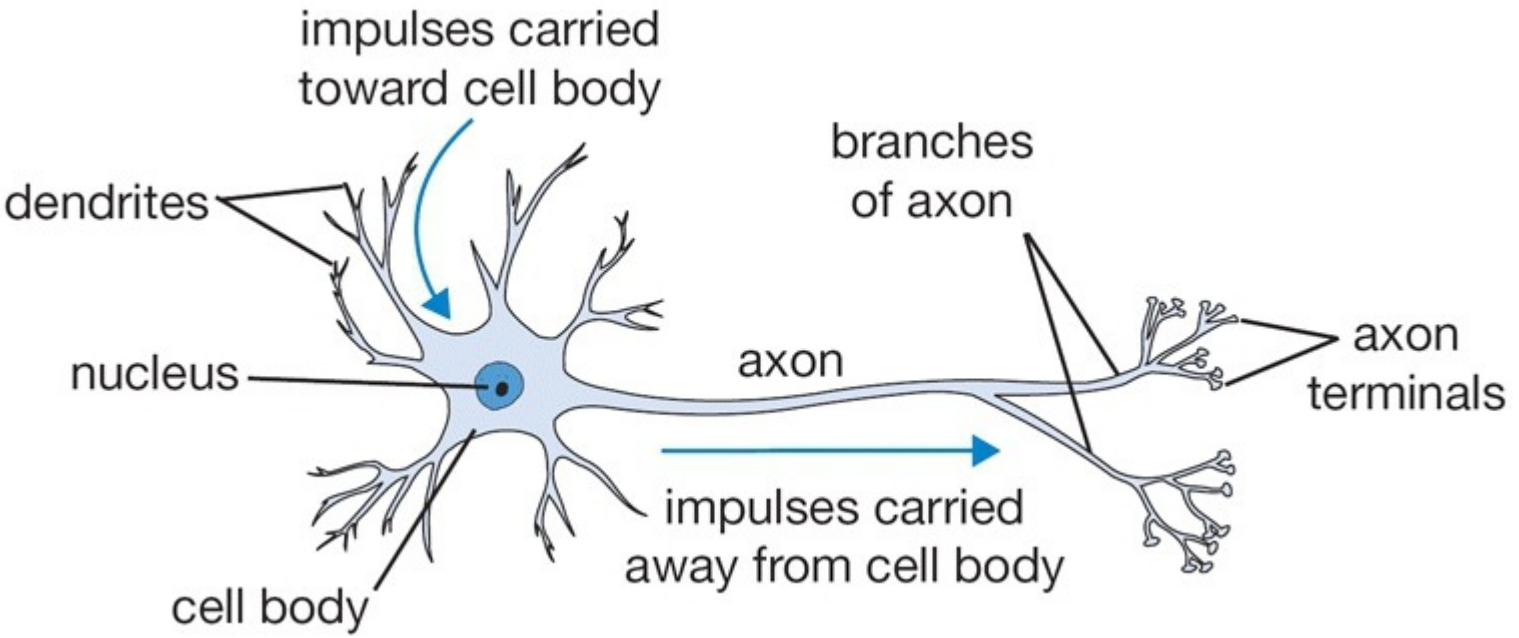
Kernel Density Estimation

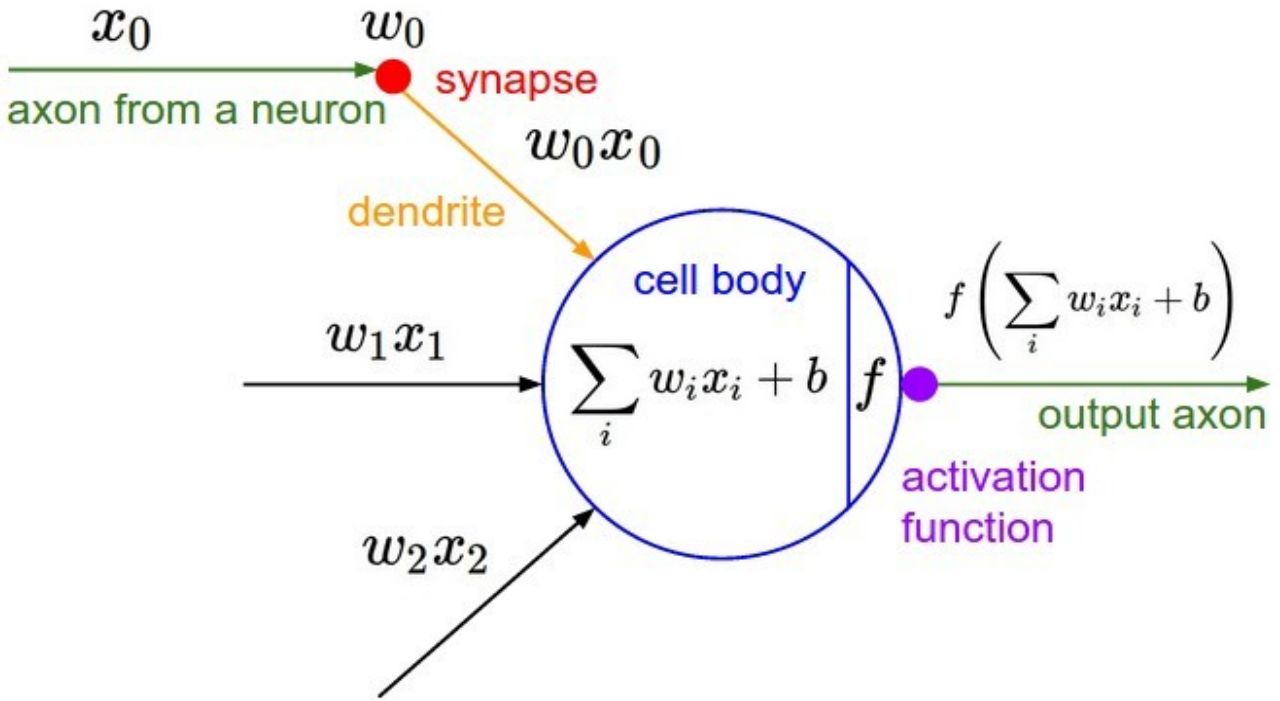


Neural Networks

Many inventions were inspired by the Nature


AIR CHINA
B-2947


impulses carried
toward cell body
dendrites
branches
of axon
nucleus
axon
axon
terminals
impulses carried
away from cell body
cell body

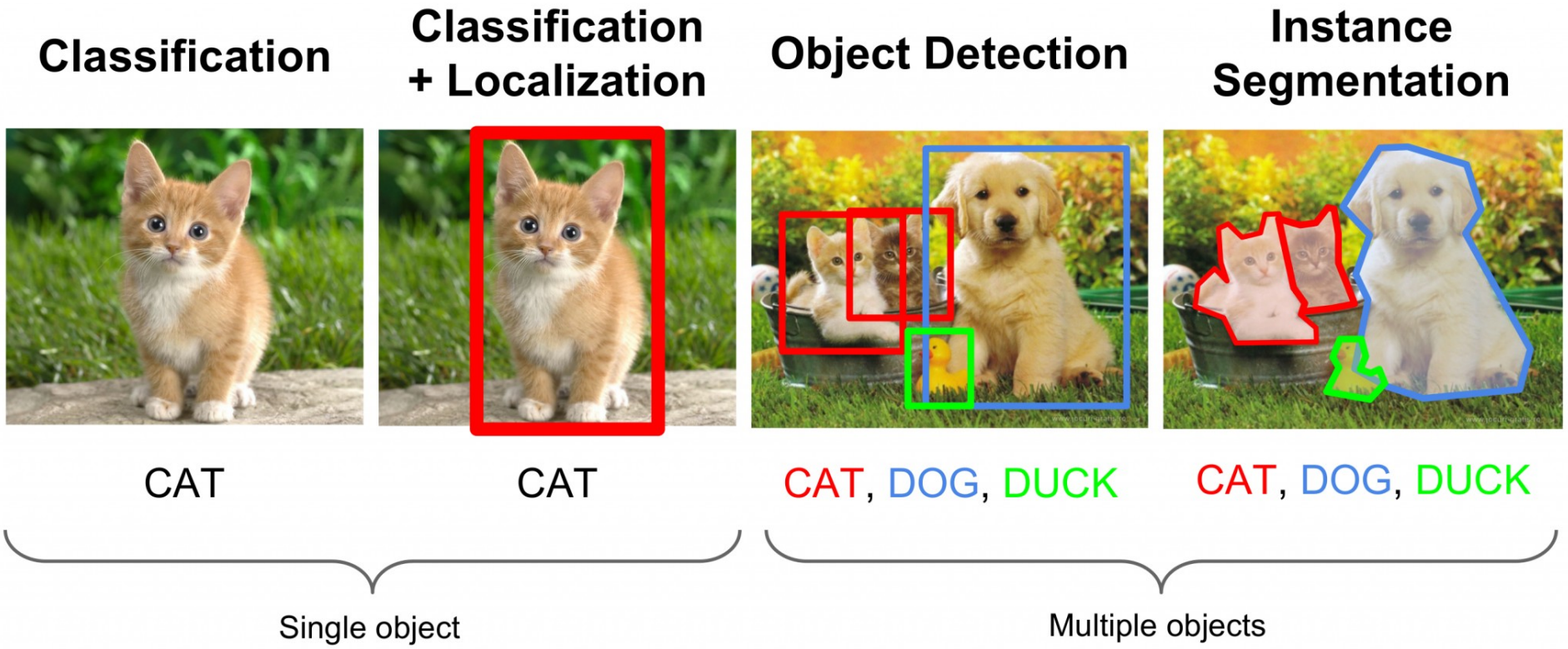

x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$w_1 x_1$
$\sum_i w_i x_i + b$
f
$f\left(\sum_i w_i x_i + b\right)$
output axon
activation
function
$w_2 x_2$

Incredibly poor analogy from biological point of view

It doesn't matter how you come up with the idea, but its utility is the only thing that matters.

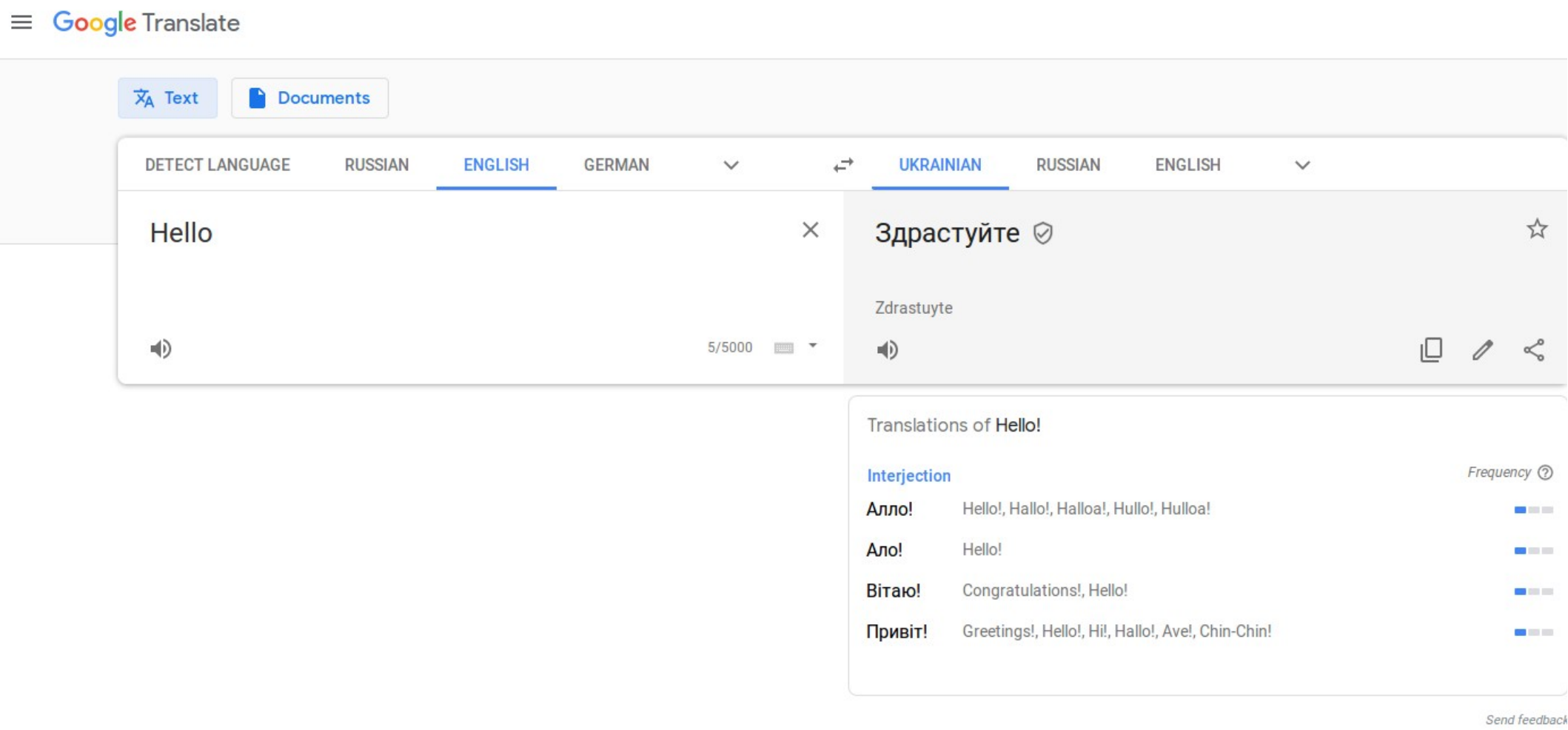
NN Tasks and Utility

- Classification
- Classification with missing inputs
- Segmentation
- Regression (predict a numerical value given some input)
- Transcription (observe a relatively unstructured representation of some kind of data and transcribe it into discrete, textual form)
- Machine translation
- Structured output (e.g. sentence to its grammar tree)
- Anomaly detection
- Synthesis and sampling
- Imputation of missing values
- Denoising
- Density estimation or probability mass function estimation
- And many-many more.....


Classification
Classification + Localization
Object Detection
Instance Segmentation
CAT
CAT
CAT, DOG, DUCK
CAT, DOG, DUCK
Single object
Multiple objects

Classification is simple?
Think once more.

Google uses NN for translation



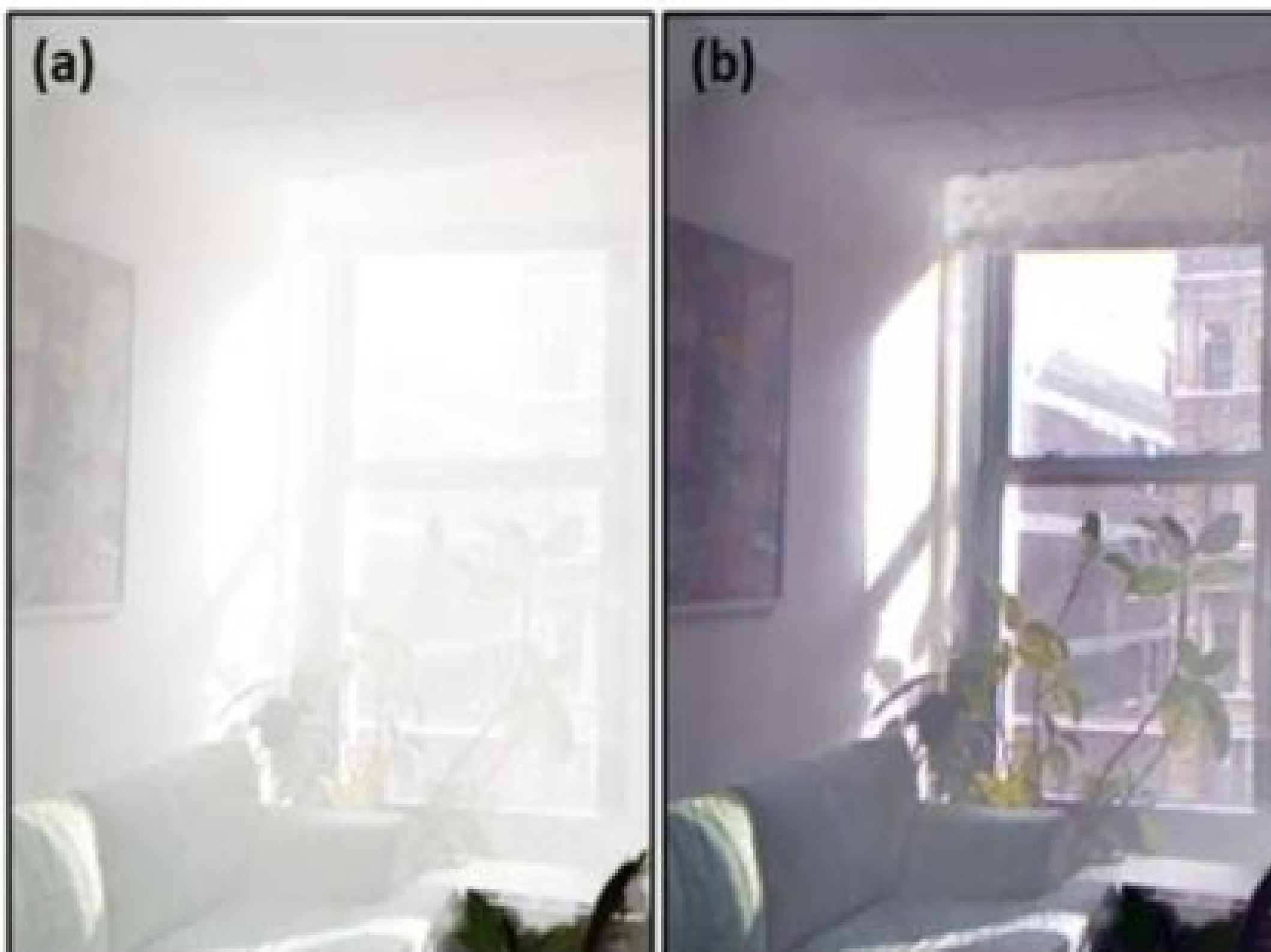
self, etc.	dynamic	ground	road	sidewalk
parking	rail track	building	wall	fence
guard rail	bridge	tunnel	pole	polegroup
traffic light	traffic sign	vegetation	terrain	sky
person	rider	car	truck	bus
caravan	trailer	train	motorcycle	bicycle

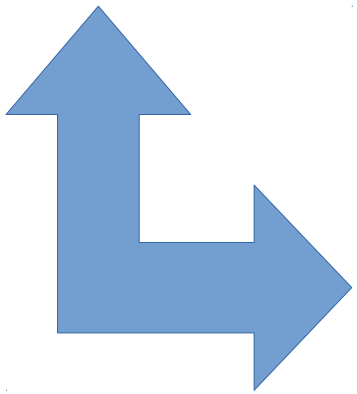

LEFT REARWARD VEHICLE CAMERA
MEDIUM RANGE VEHICLE CAMERA
RIGHT REARWARD VEHICLE CAMERA
MOTION FLOW
LANE LINES
LANE LINES
ROAD FLOW
IN-PATH OBJECTS
ROAD LIGHTS
OBJECTS
ROAD SIGNS

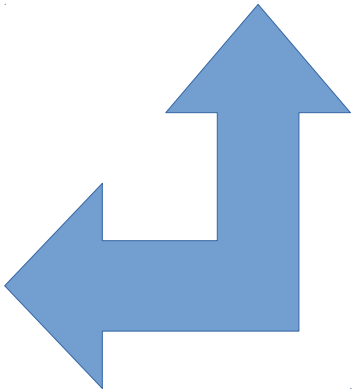
Real noisy photos

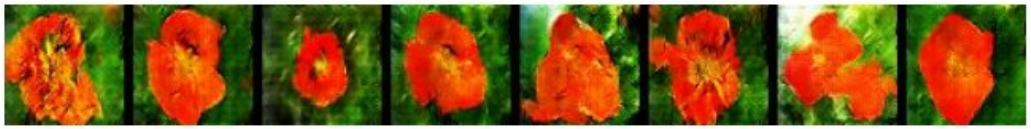
Input

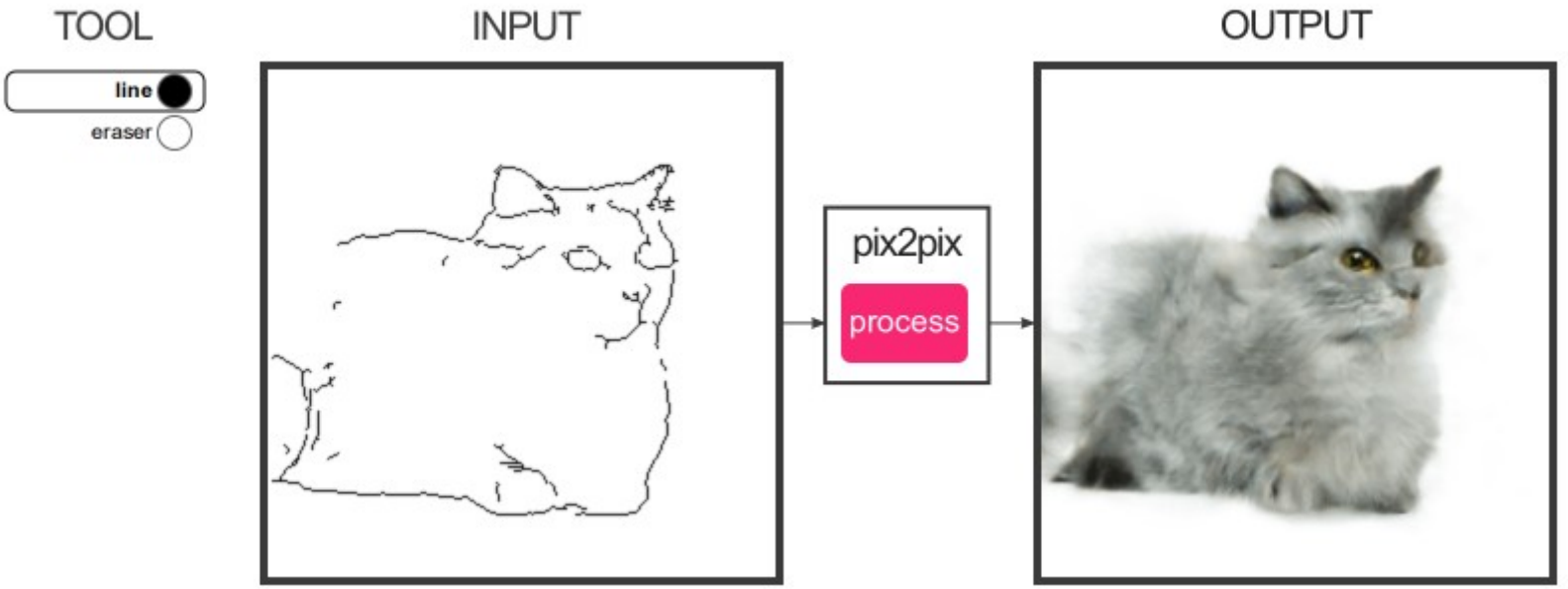
Output

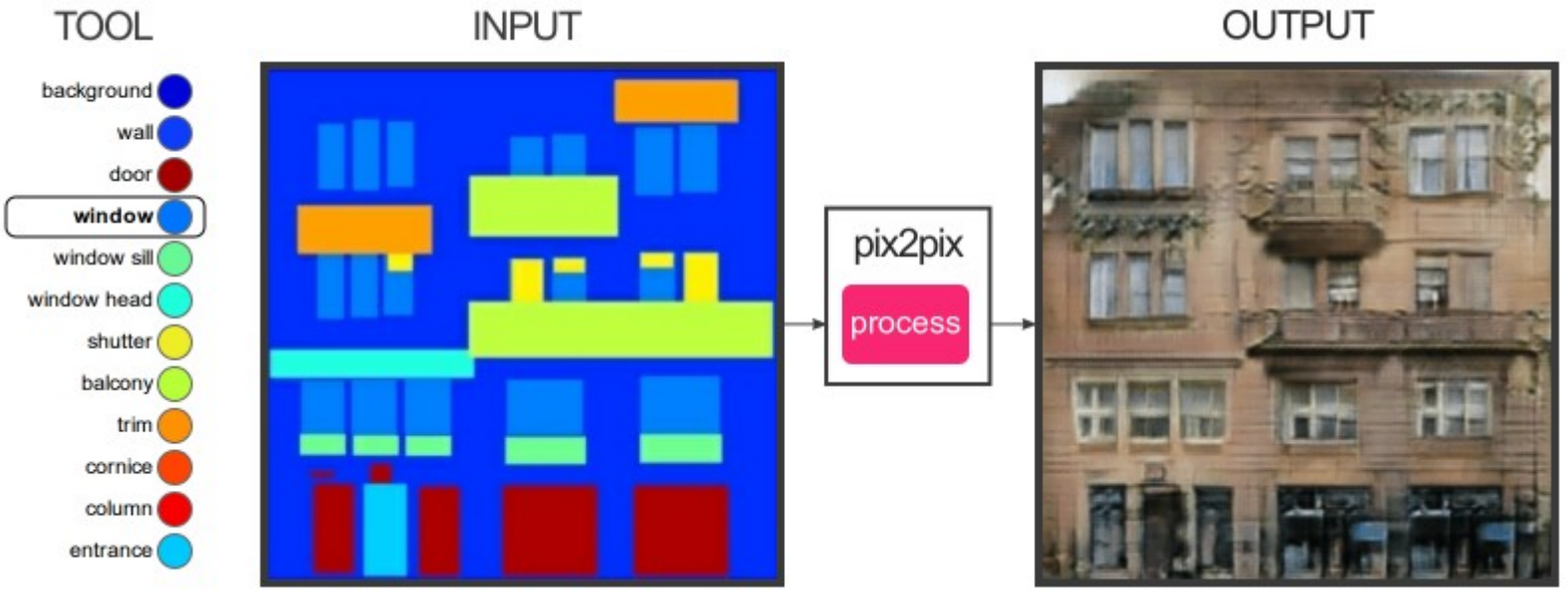

(a)
(b)





Caption	Generated Images
the flower shown has yellow anther red pistil and bright red petals	
this flower has petals that are yellow, white and purple and has dark lines	
the petals on this flower are white with a yellow center	
this flower has a lot of small round pink petals.	
this flower is orange in color, and has petals that are ruffled and rounded.	
the flower has yellow petals and the center of it is brown	

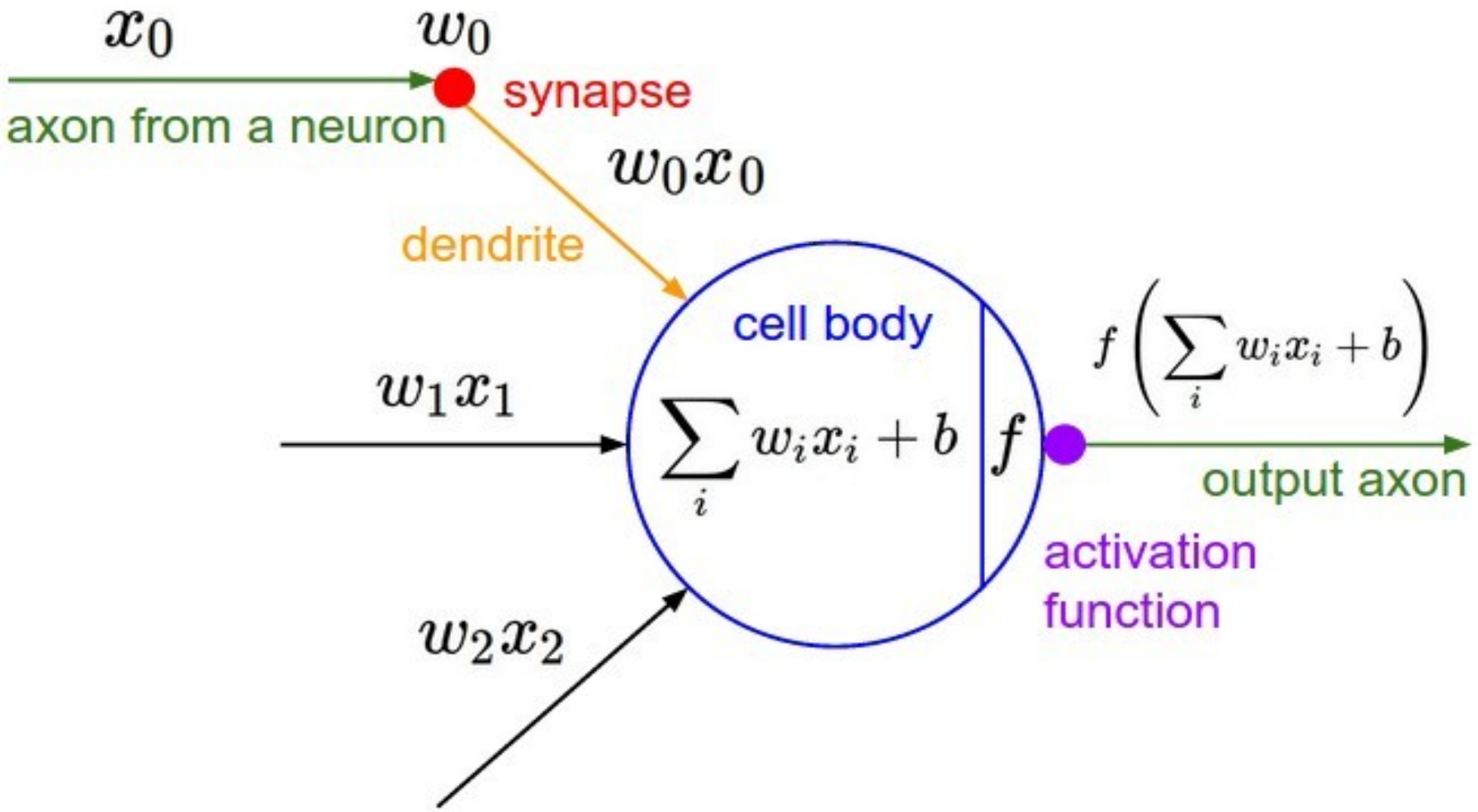

TOOL
INPUT
OUTPUT
line
eraser
pix2pix
process


TOOL
INPUT
OUTPUT
background
wall
door
window
window sill
window head
shutter
balcony
trim
cornice
column
entrance
pix2pix
process


ALPHA00
00:01:00
LEE SEDOL
00:01:00
AlphaGo
Lee Sedol

What is a neural network?

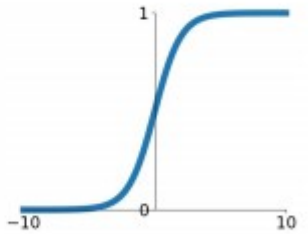
Single Neuron



Activation functions

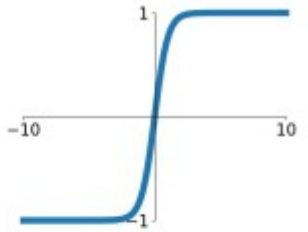
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



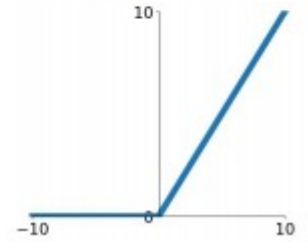
tanh

$$\tanh(x)$$



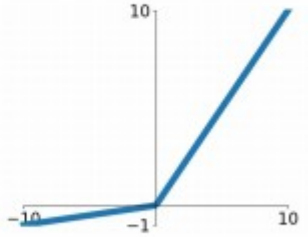
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

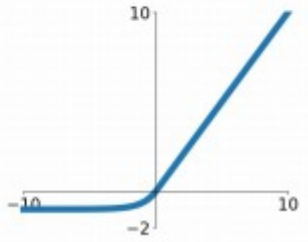


Maxout

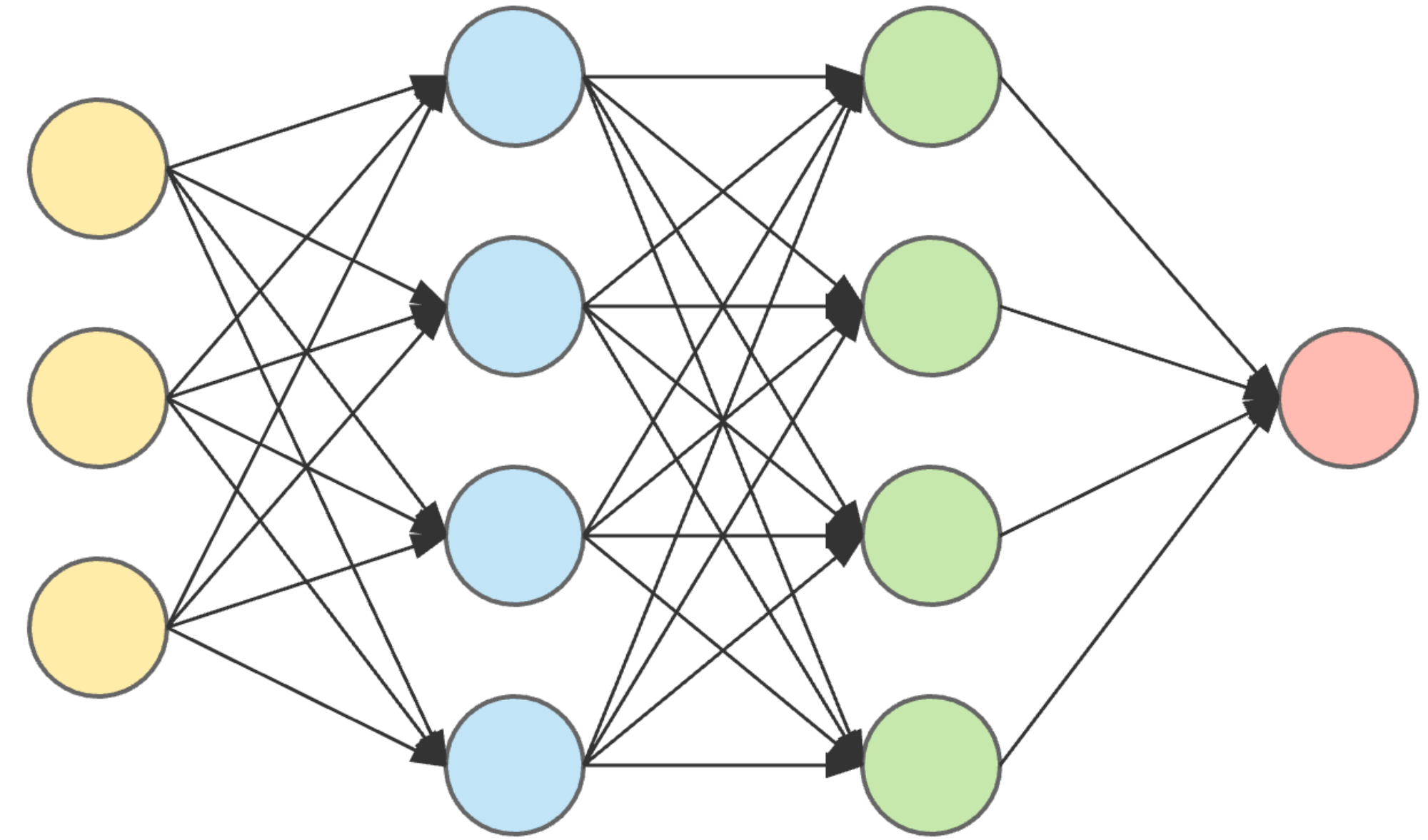
$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

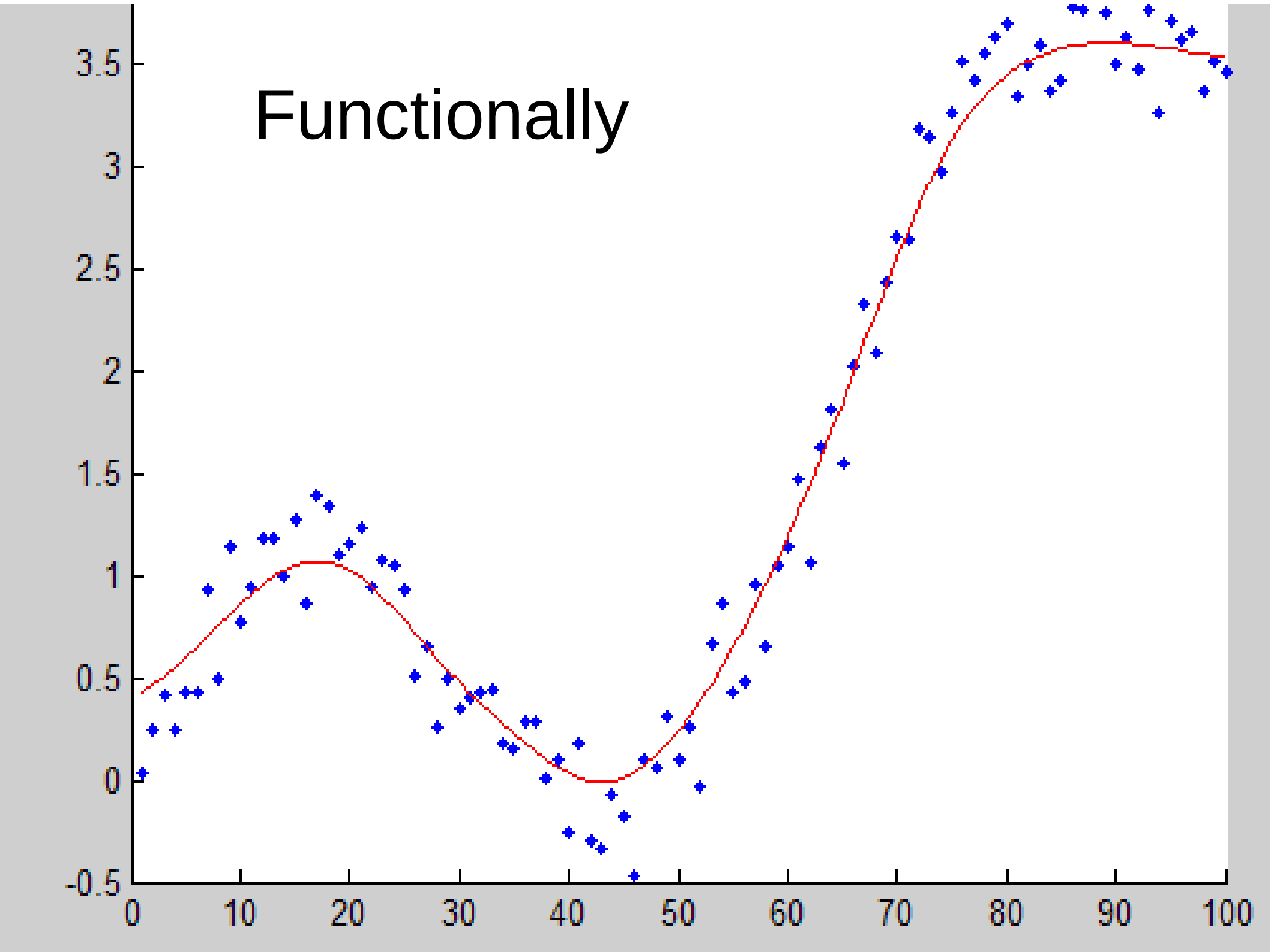
ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Structurally




Functionally
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
0
10
20
30
40
50
60
70
80
90
100

Functionally, neural network is a gargantuan interpolator-approximator with millions internal parameters

Example: AlexNet, 62,378,344 parameters

Where do we get the points to approximate?

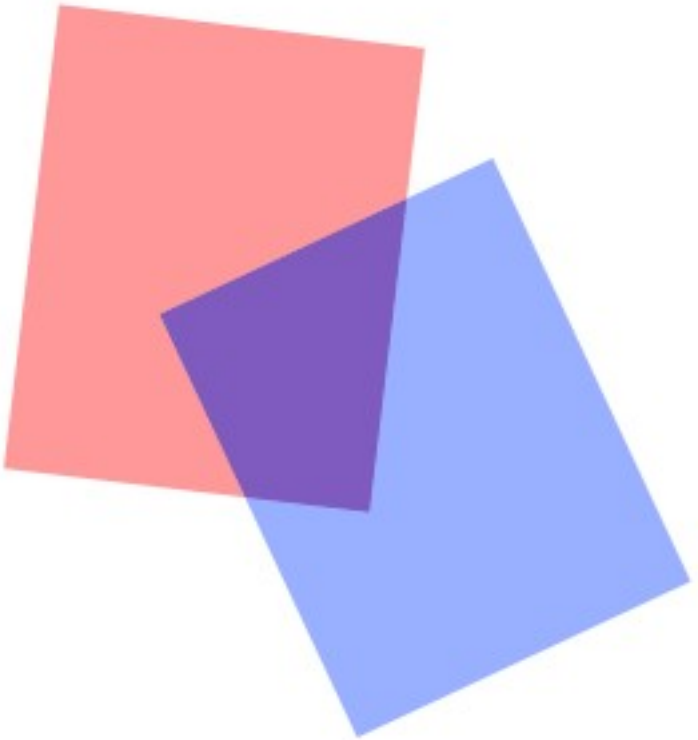
Three ways

- We define the desired output ourselves – Supervised learning (classification, etc.)
- We cannot define the desired output, but we can tell how bad is the given output – Unsupervised learning (clustering, etc.)
- We allow NN to interact with the environment and assess the consequences – Reinforcement learning (play Atari game, etc.)

How do we know, neural network does its job good?

How do we know, it does what we want?

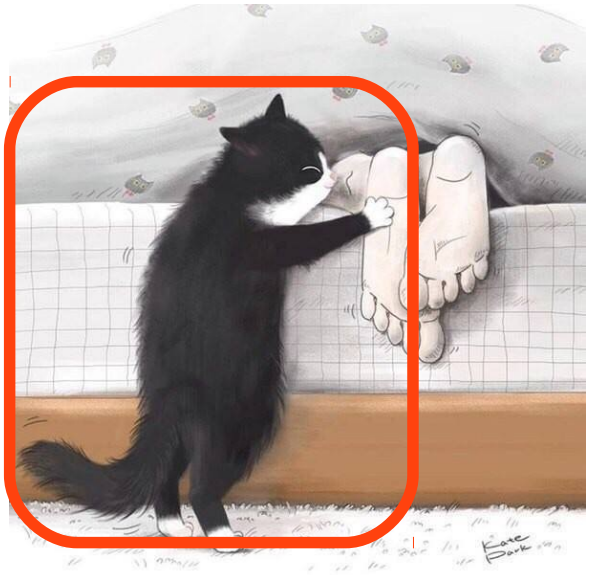
Cost Function
also known as
Loss Function

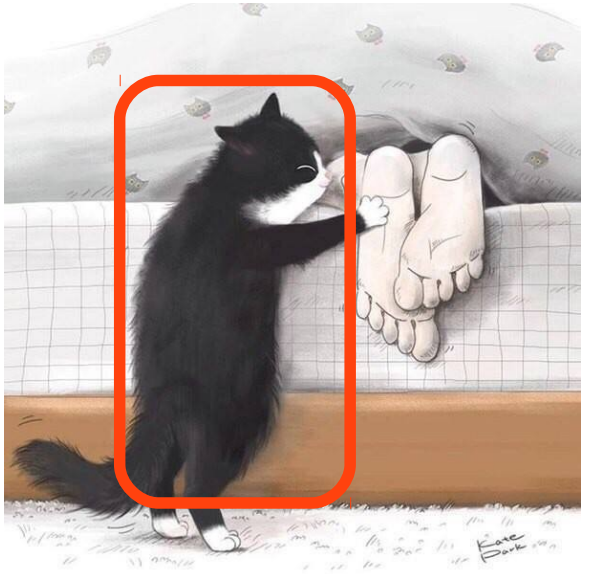


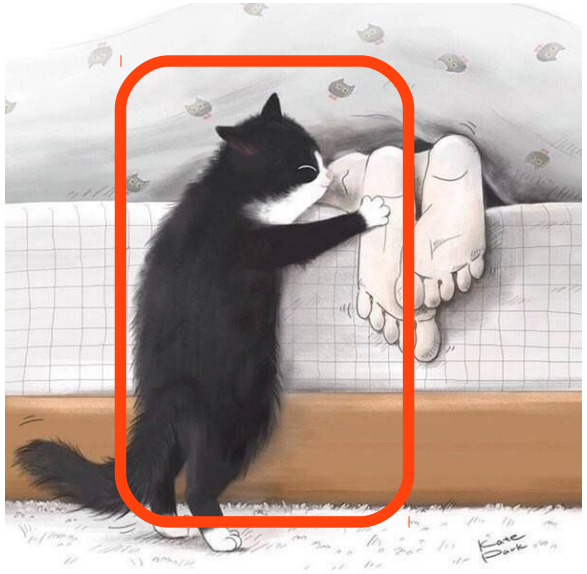
to the rescue!

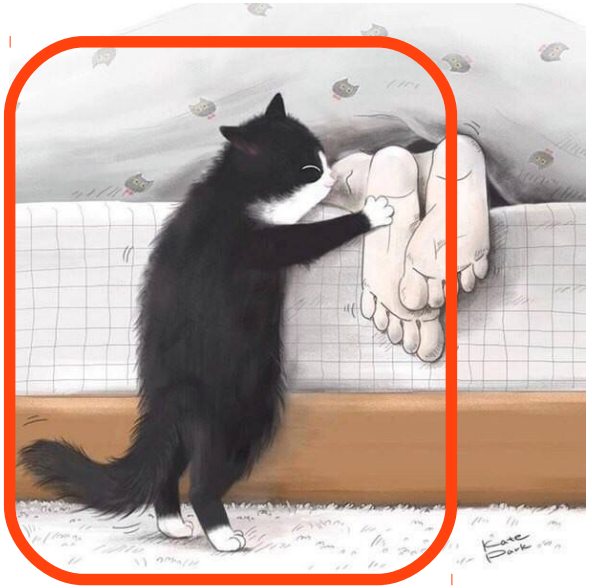
How to formalize
that two rectangles
match?

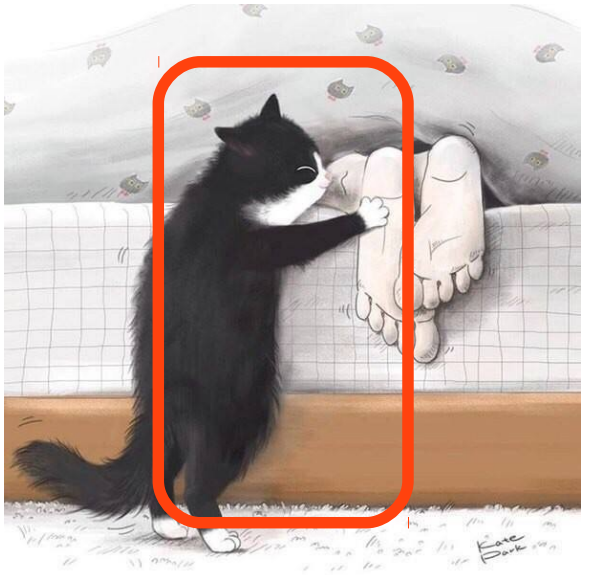
What is supposed to be “good”?



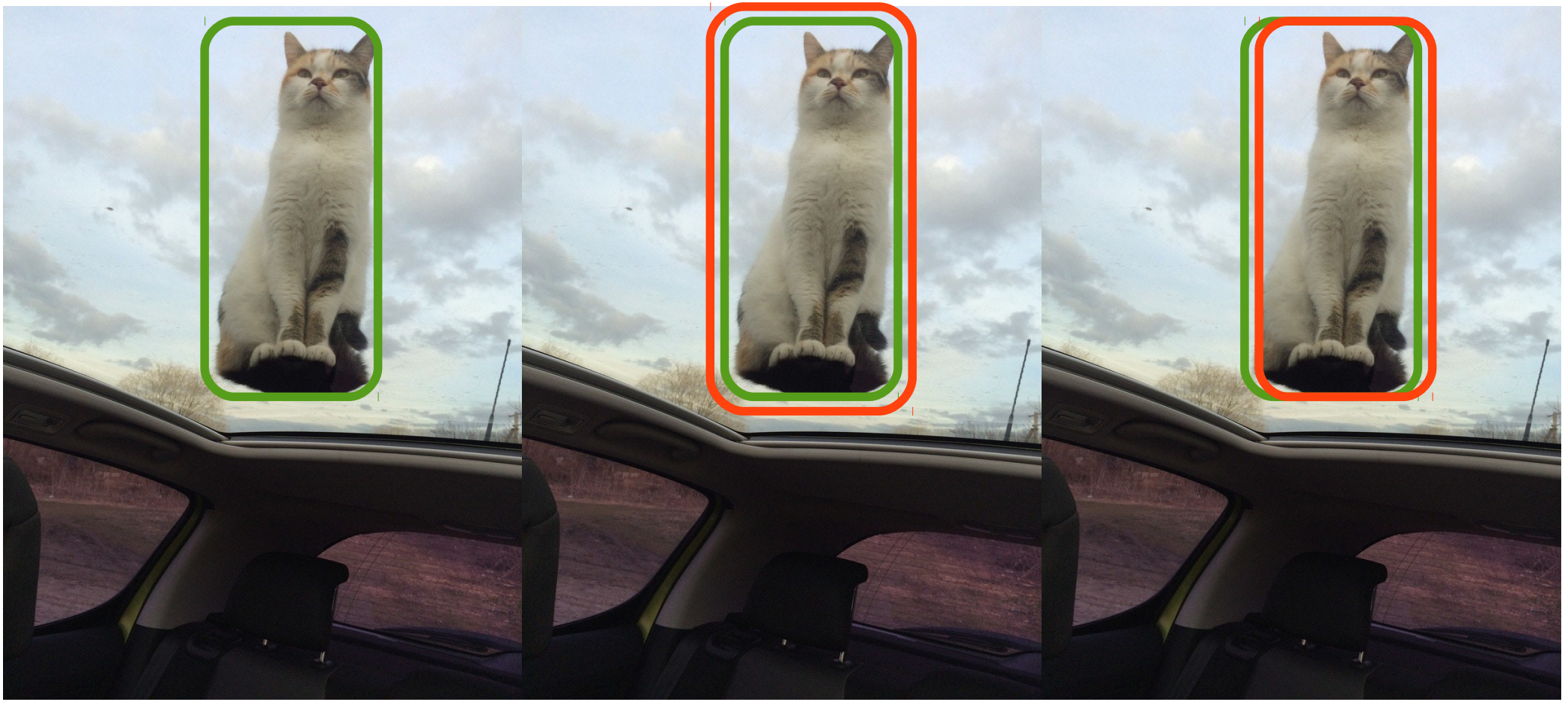








Now you have ground-truth



Which detect is better?

Where do we get internal parameters?

We train neural network

How do we train neural network?

Methodological point of view

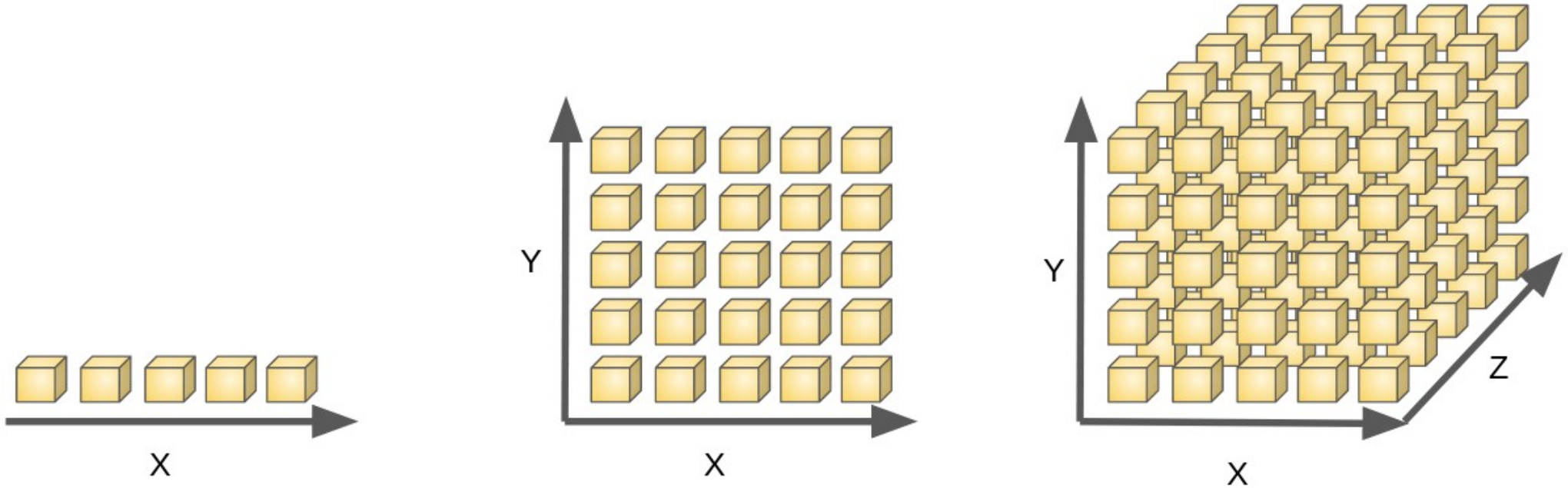
- Supervised learning (classification, etc.)
- Unsupervised learning (clustering, etc.)
- Reinforcement learning (play Atari game, etc.)

Implementation point of view

Loss function, the formalization of how good NN performs, should be minimized (we could define “gain” function and maximize it)

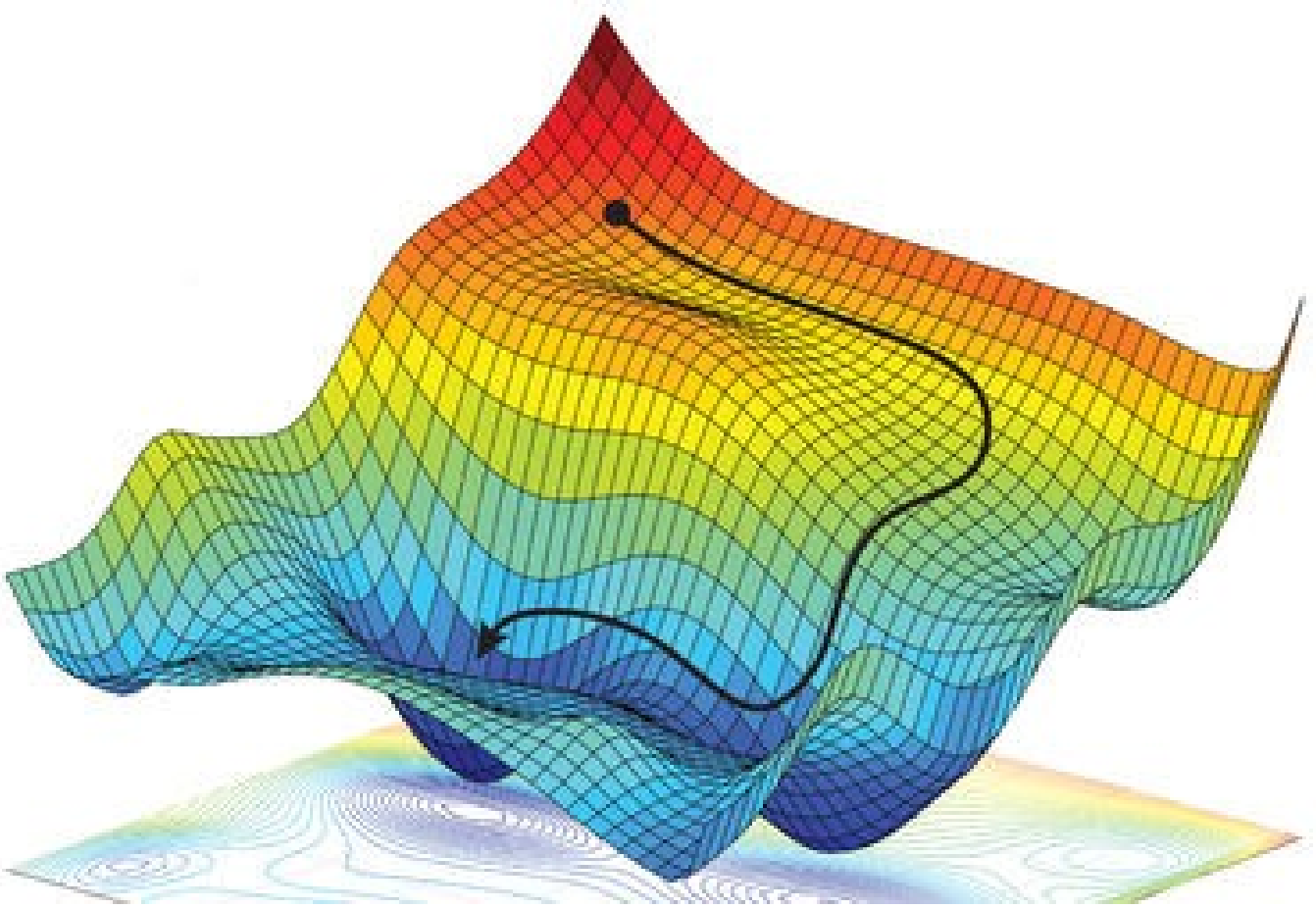
What minimization methods do we know? What is the suitable one?

Parameters... Parameters everywhere...



Curse of dimensions

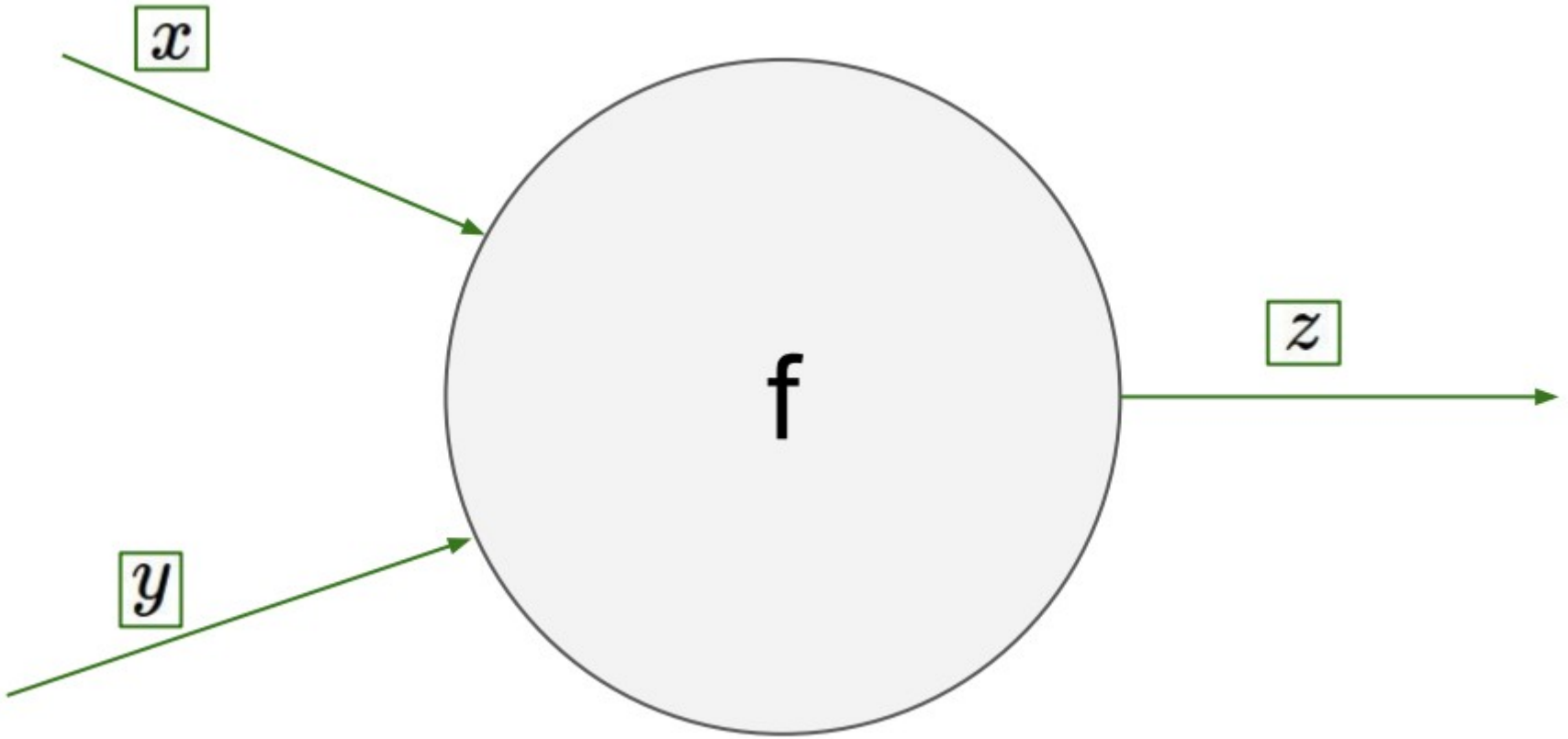
Gradient descent

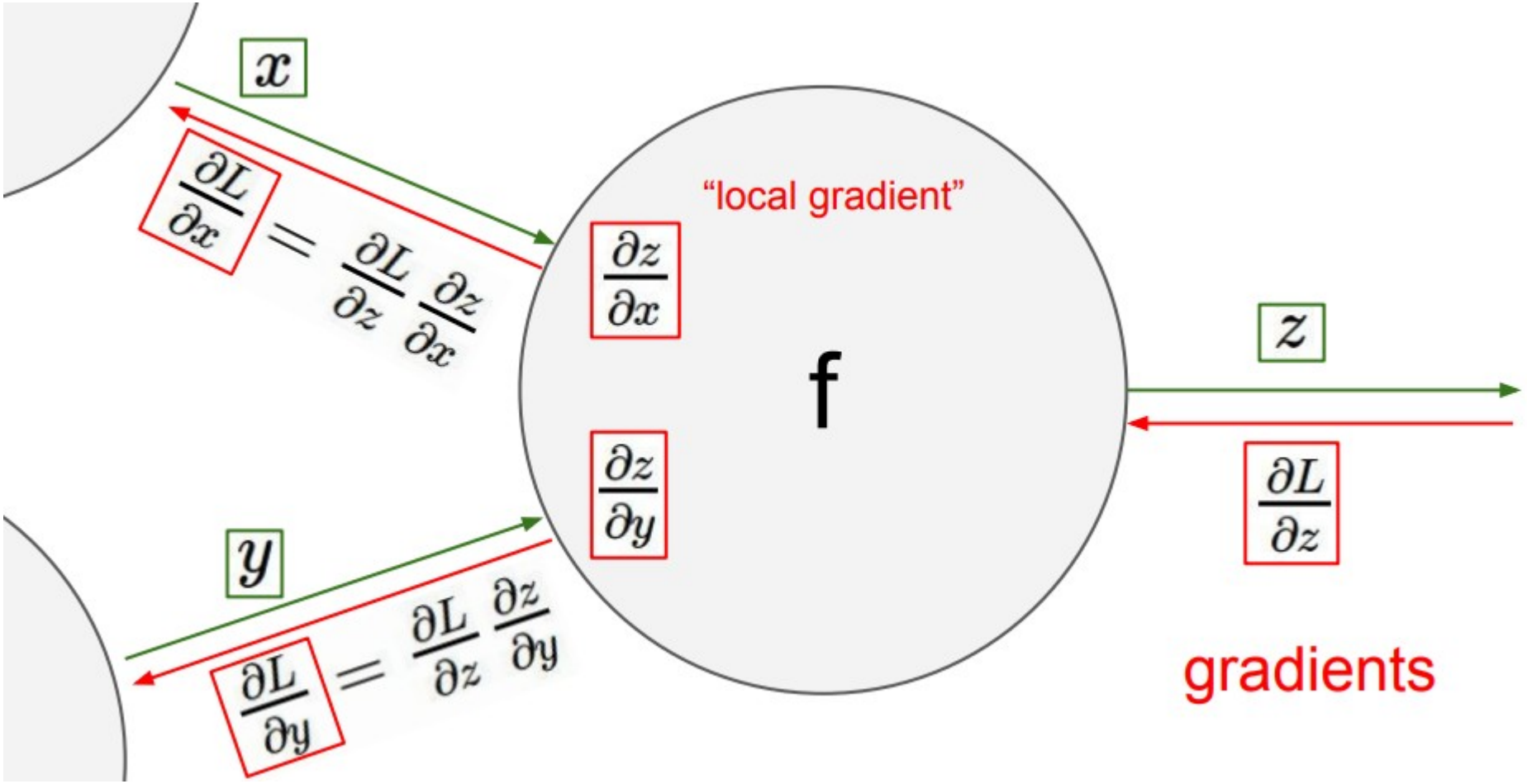


Modifications of gradient descent

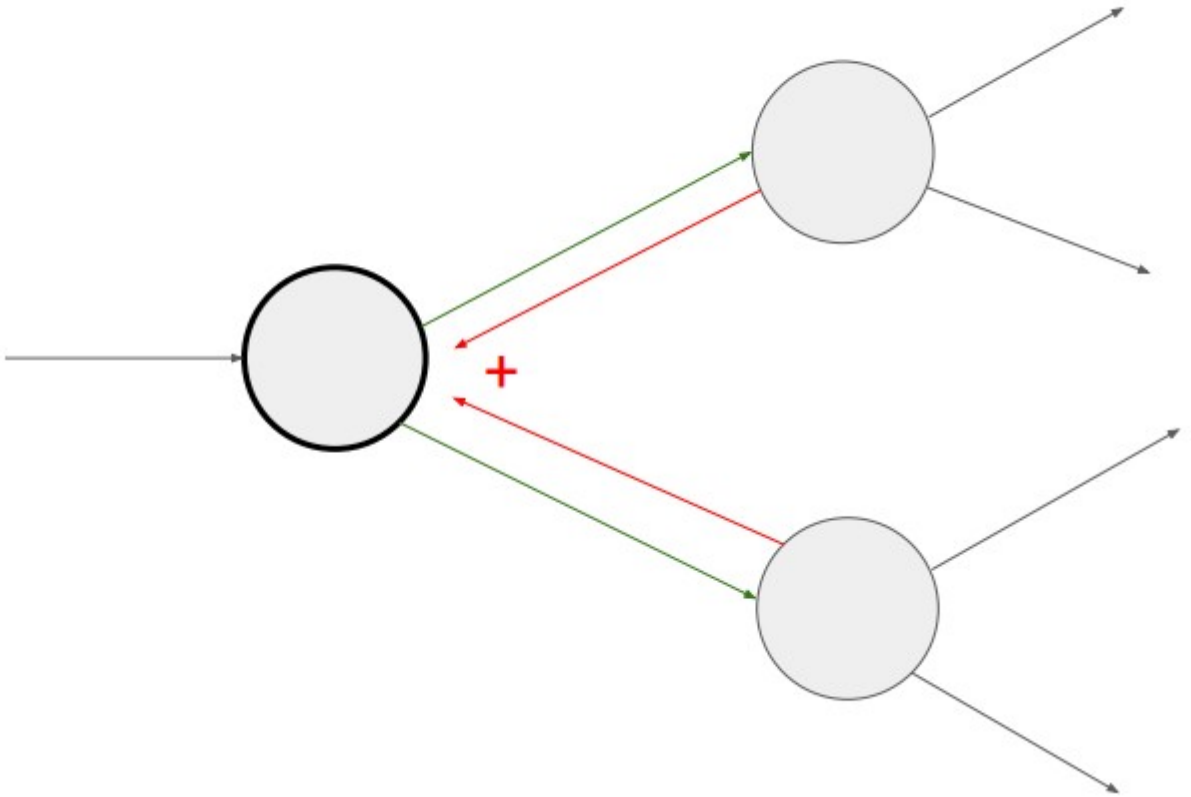
- Momentum optimization
- Nesterov Momentum optimization
- AdaGrad
- RMSProp
- Adam optimization
- Learning rate scheduling

forward
backward


x
y
f
z


x
$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial x}$
"local gradient"
$\frac{\partial z}{\partial x}$
f
$\frac{\partial z}{\partial y}$
z
$\frac{\partial L}{\partial z}$
y
$\frac{\partial L}{\partial y} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial y}$
gradients

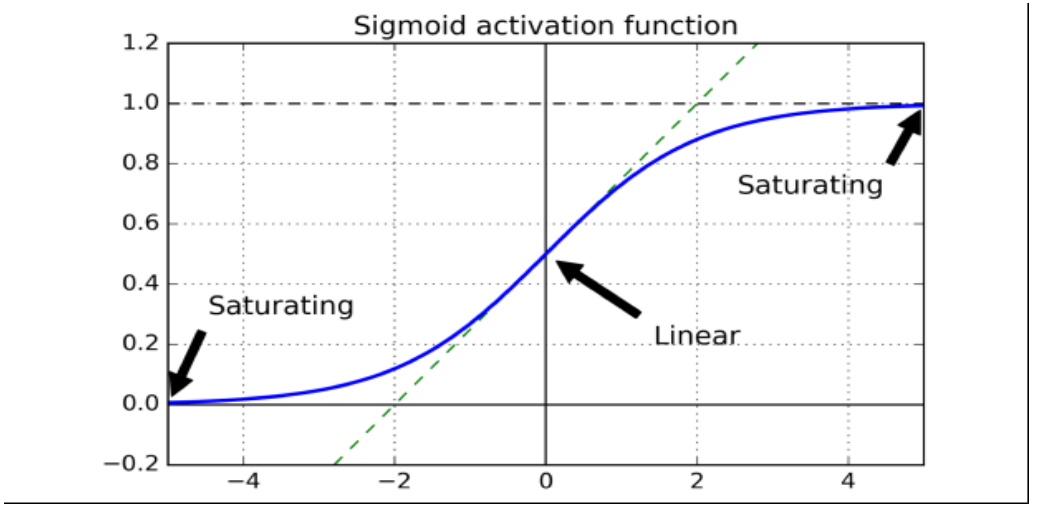
Gradients add at branches

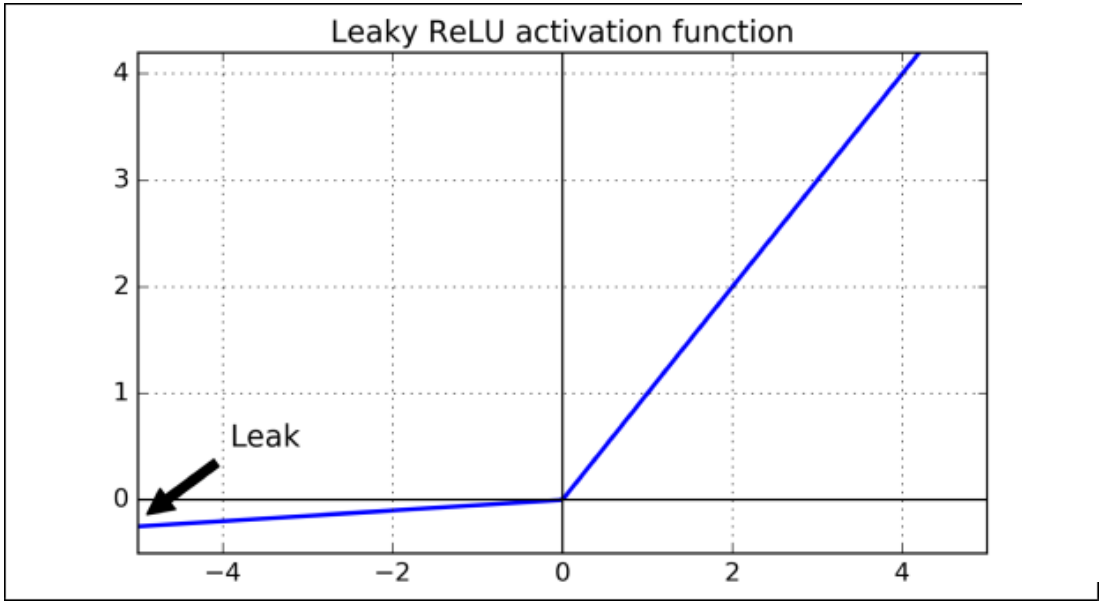


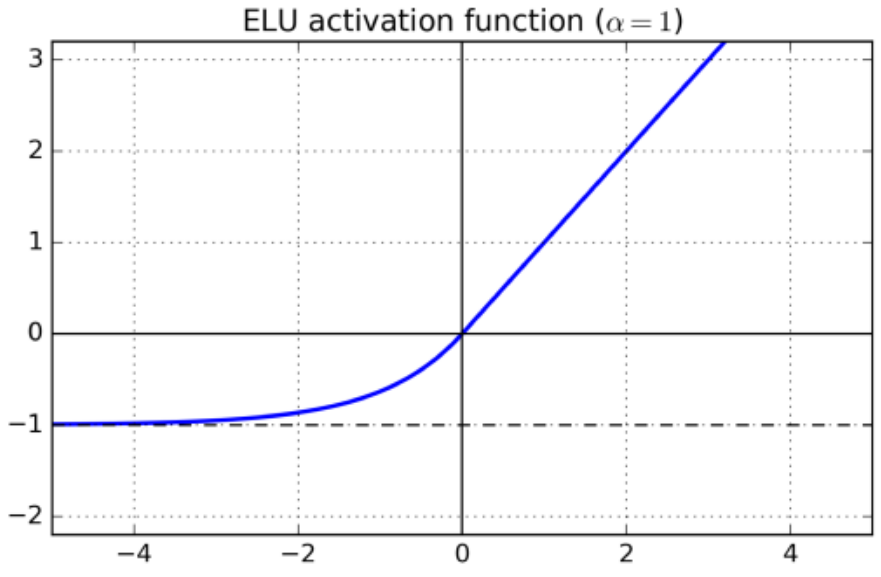
Possible issues with deep networks

Vanishing/exploding gradients problems

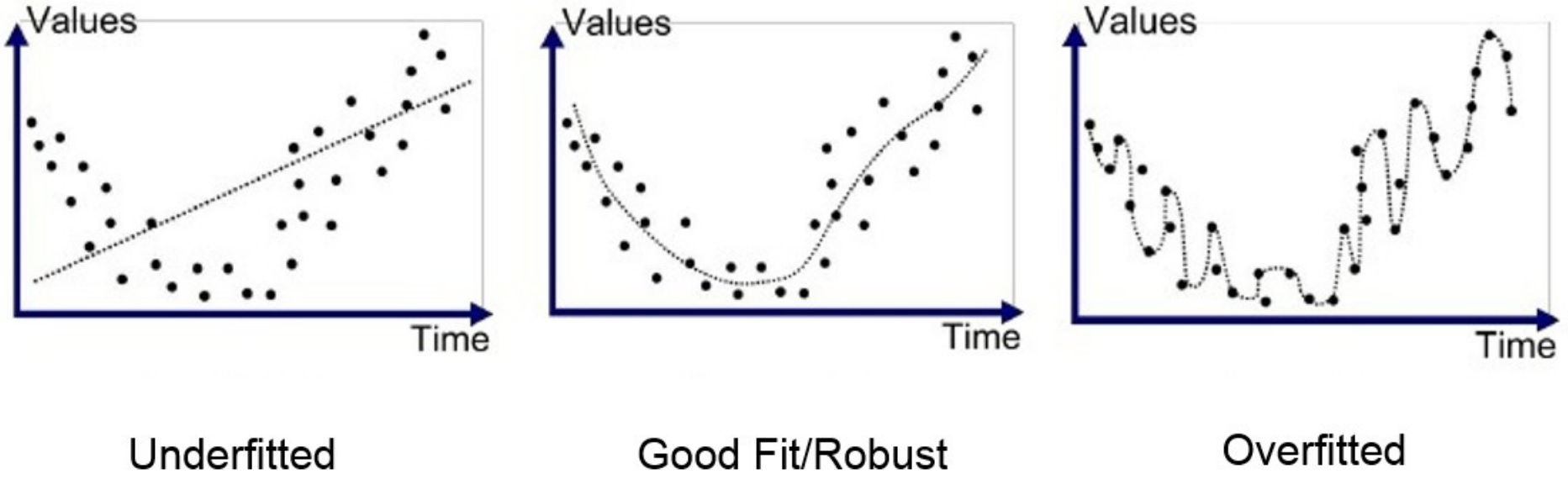
- Xavier and He initialization
- Non-saturating activation functions
- Batch Normalization
- Gradient clipping







Fitting problems



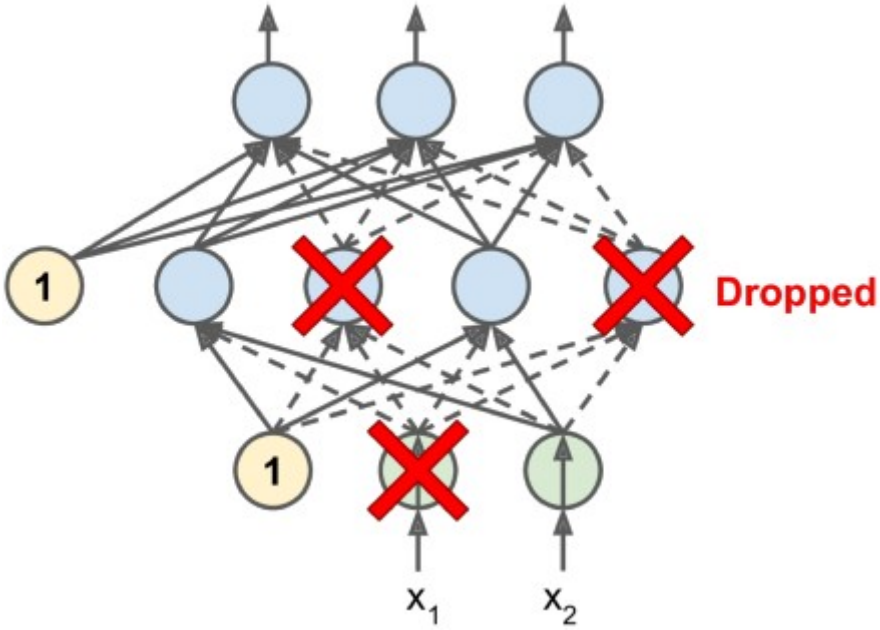
Underfitted

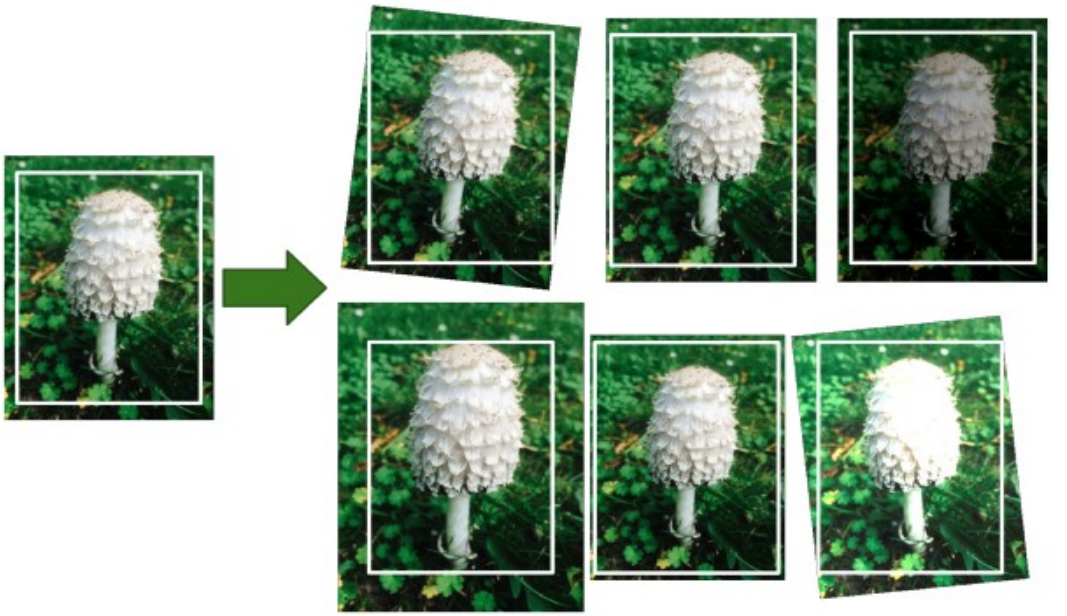
Good Fit/Robust

Overfitted

Avoid overfitting

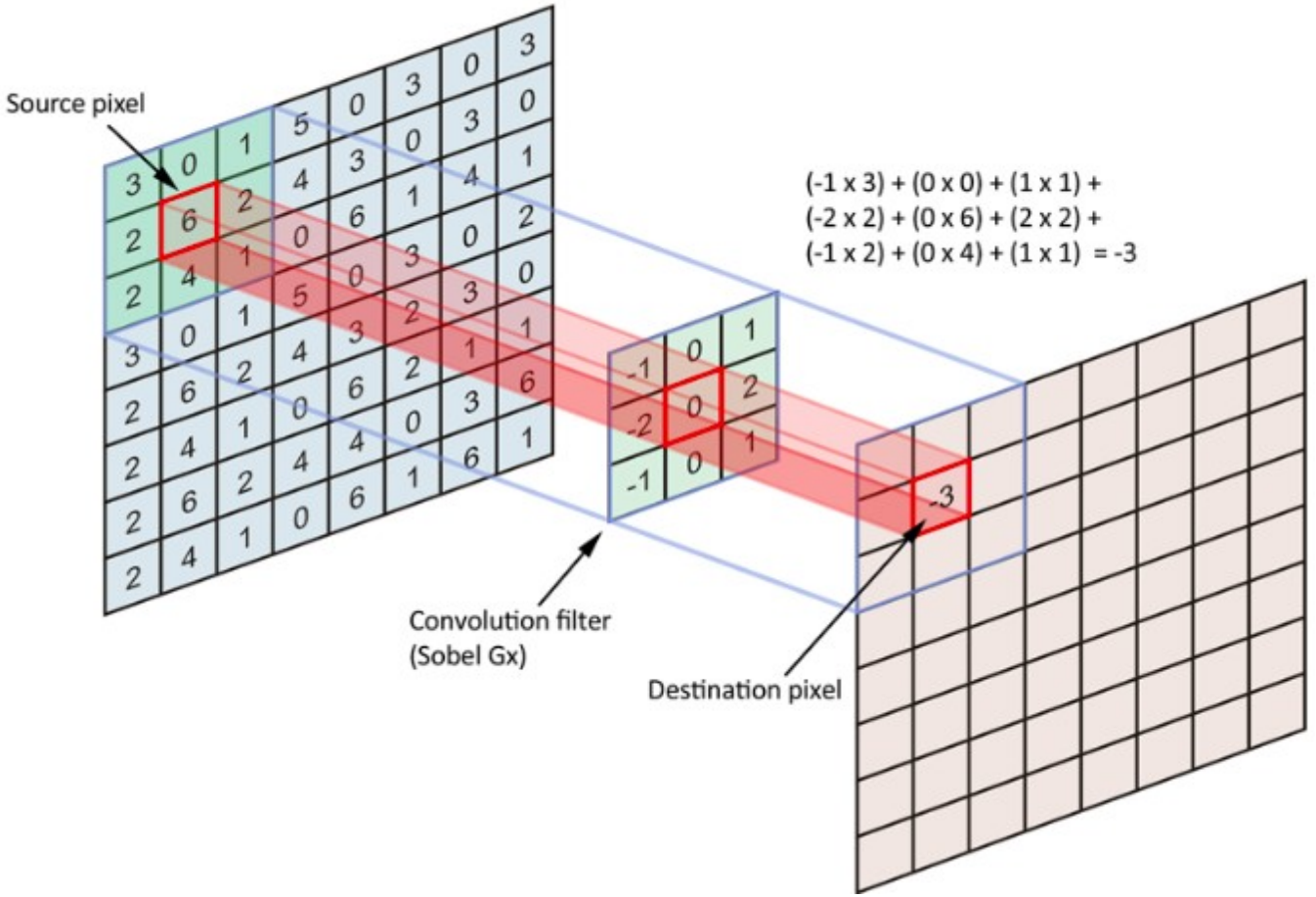
- Early Stopping
- L1 and L2 regularization
- Dropout
- Max-norm regularization
- Data augmentation





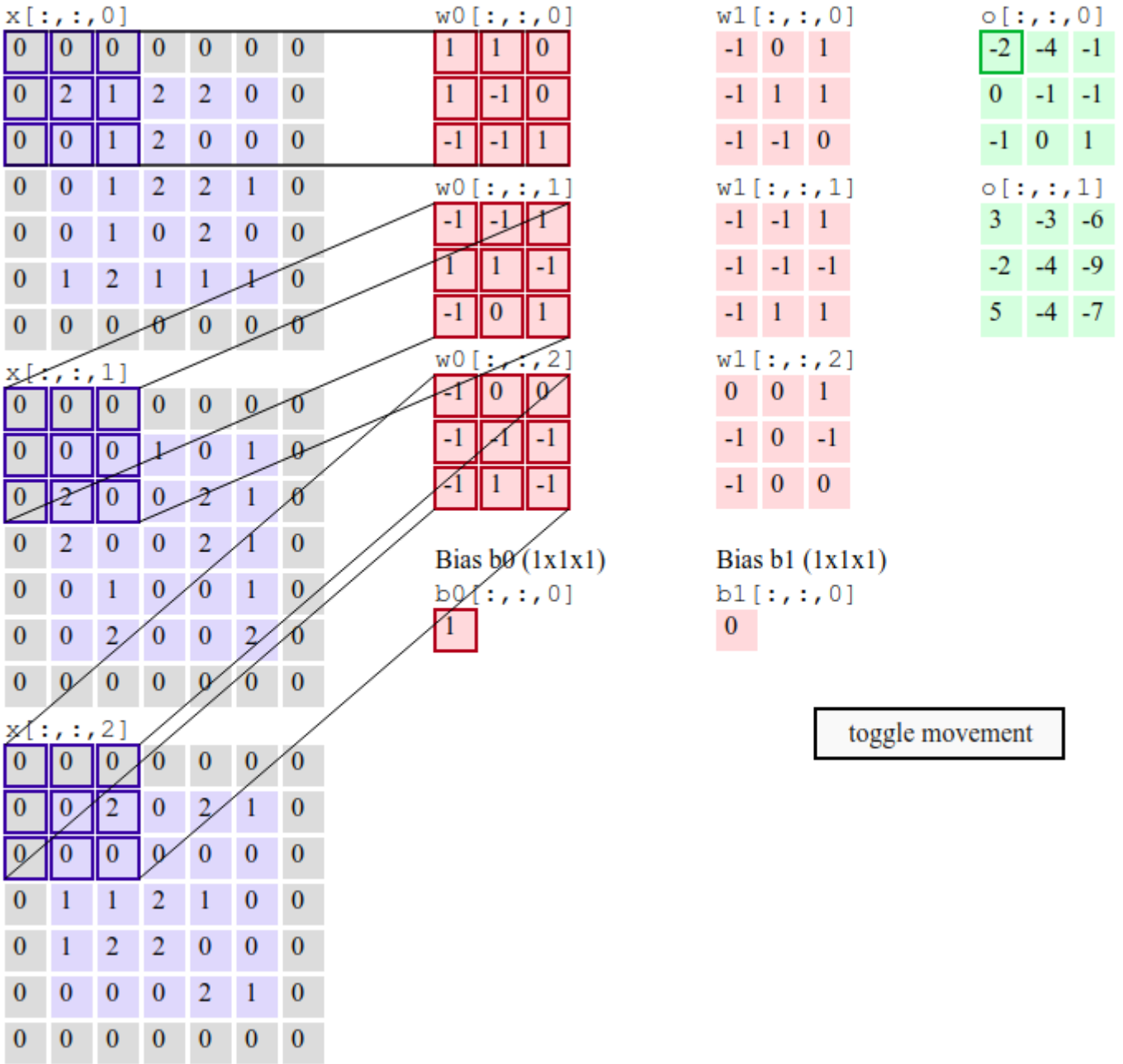
Convolution operation

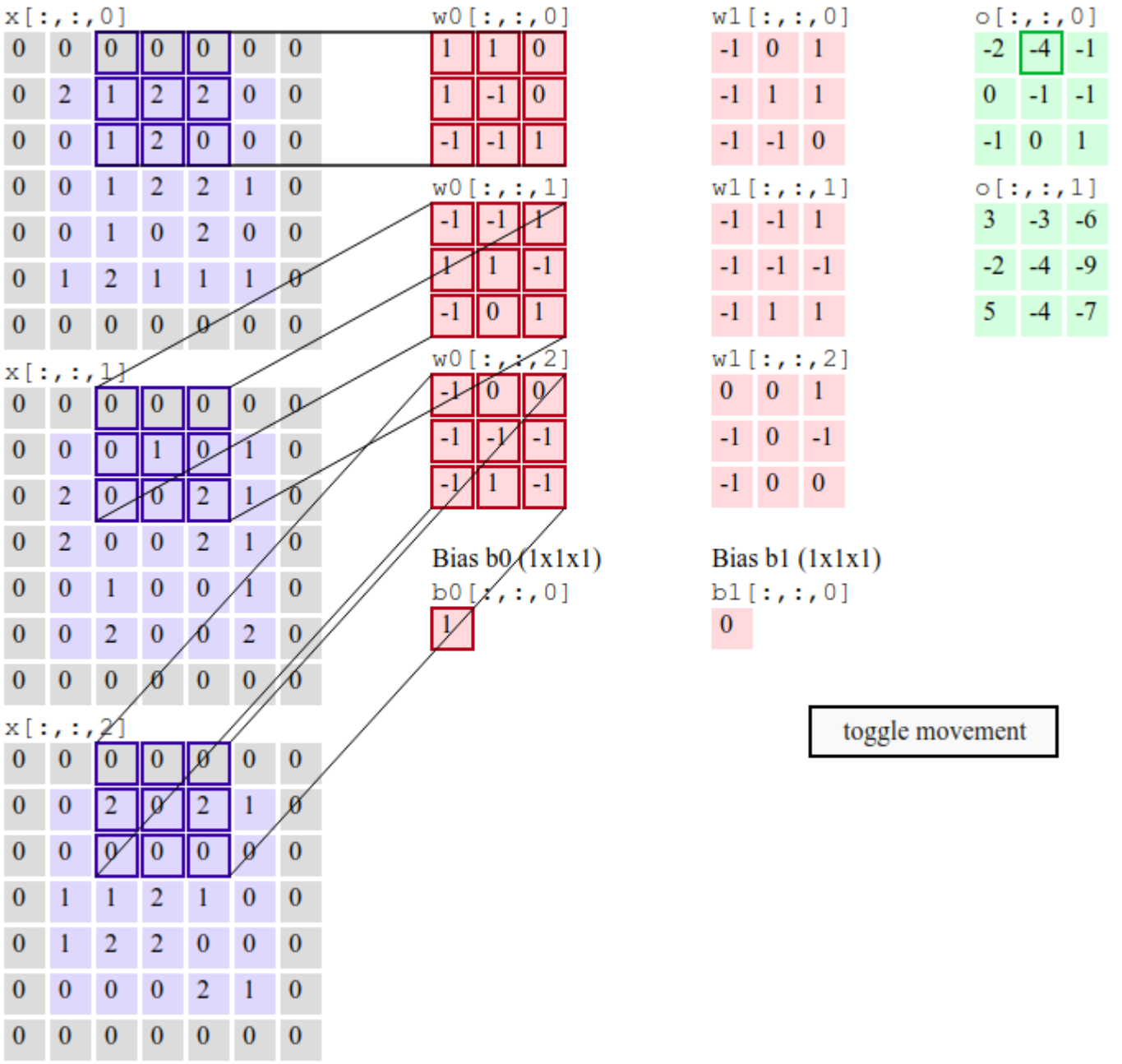
Convolution

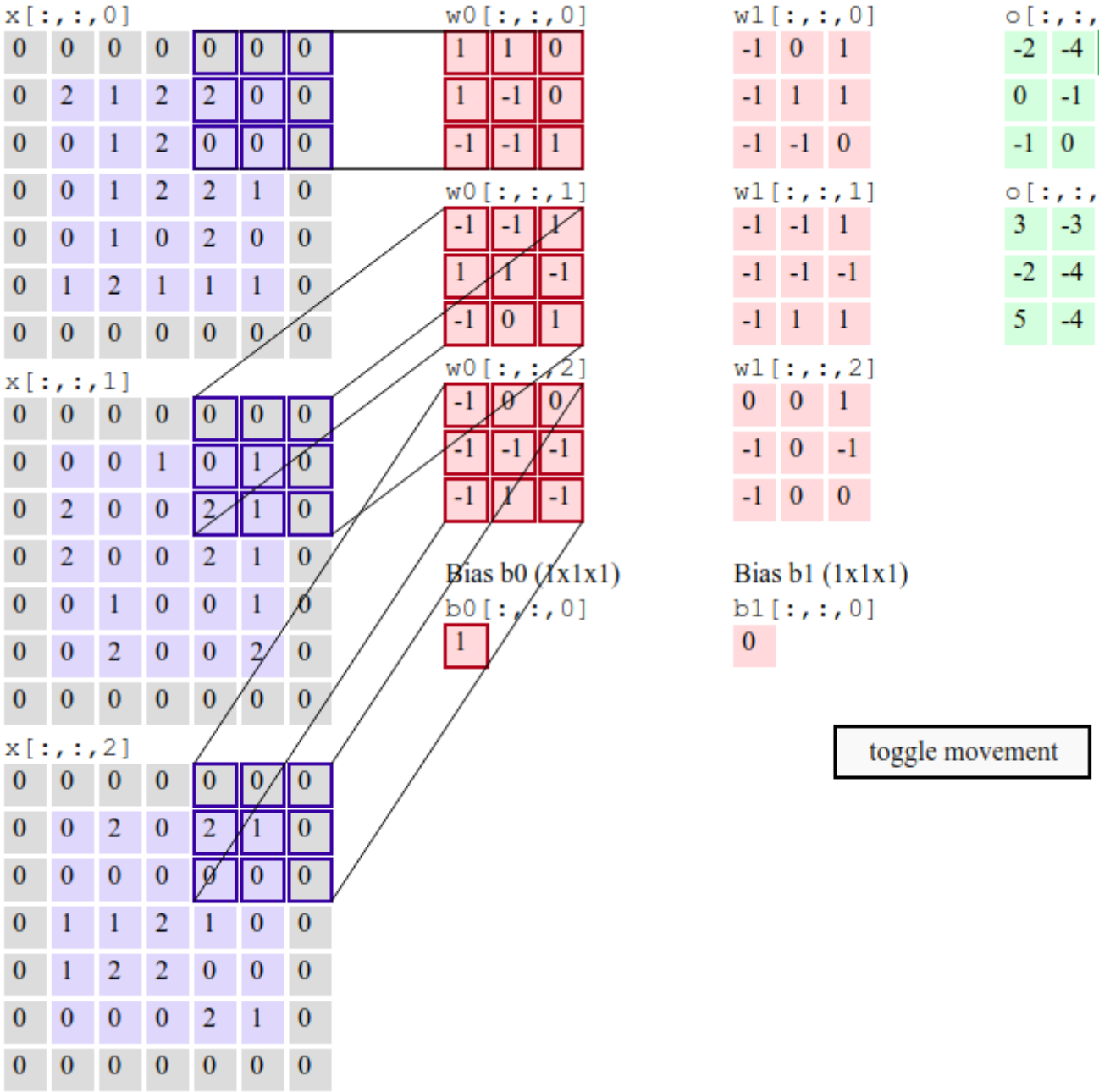


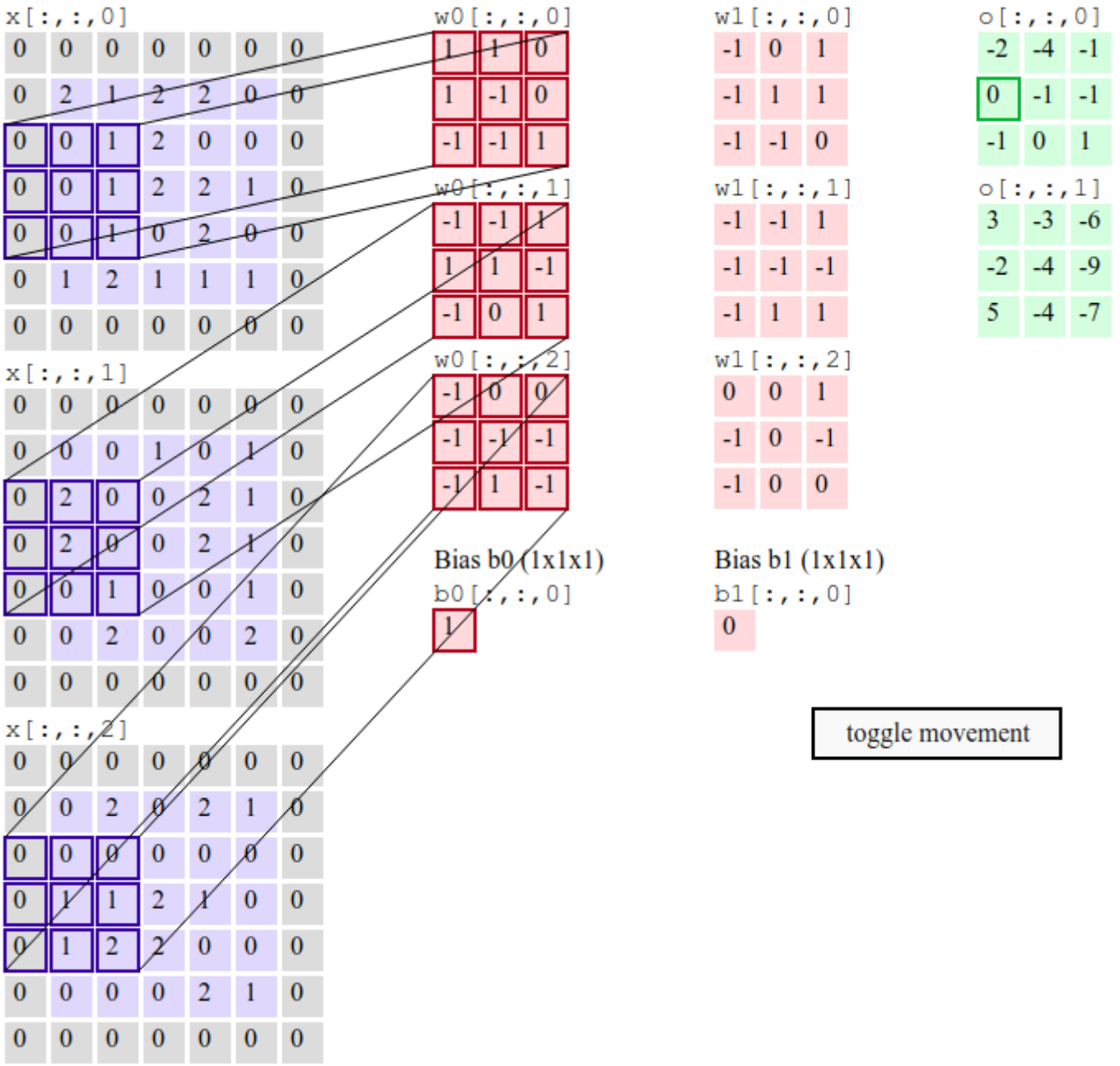
Two basic ideas:

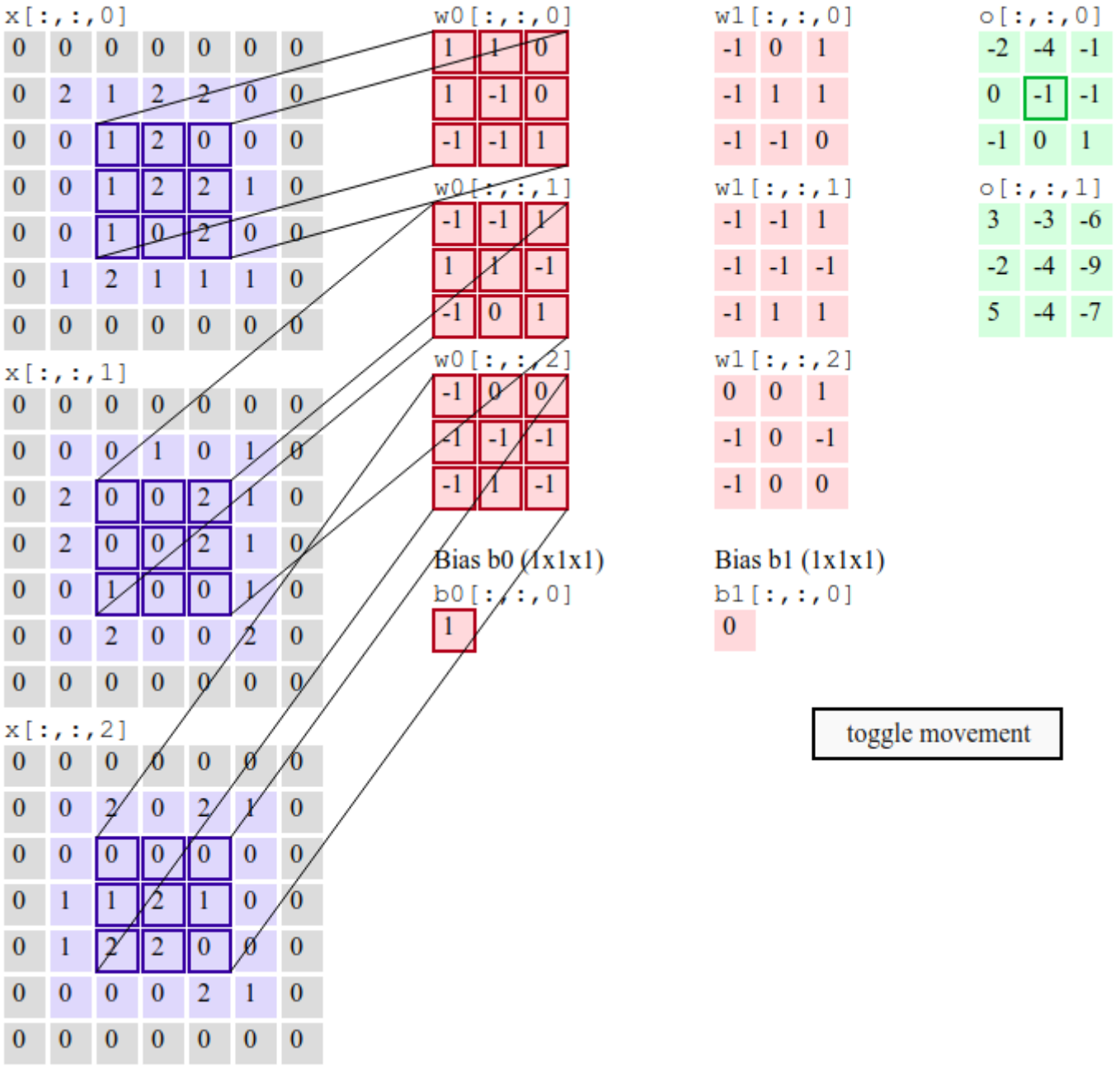
- geometrical proximity has significant meaning
- translational invariance

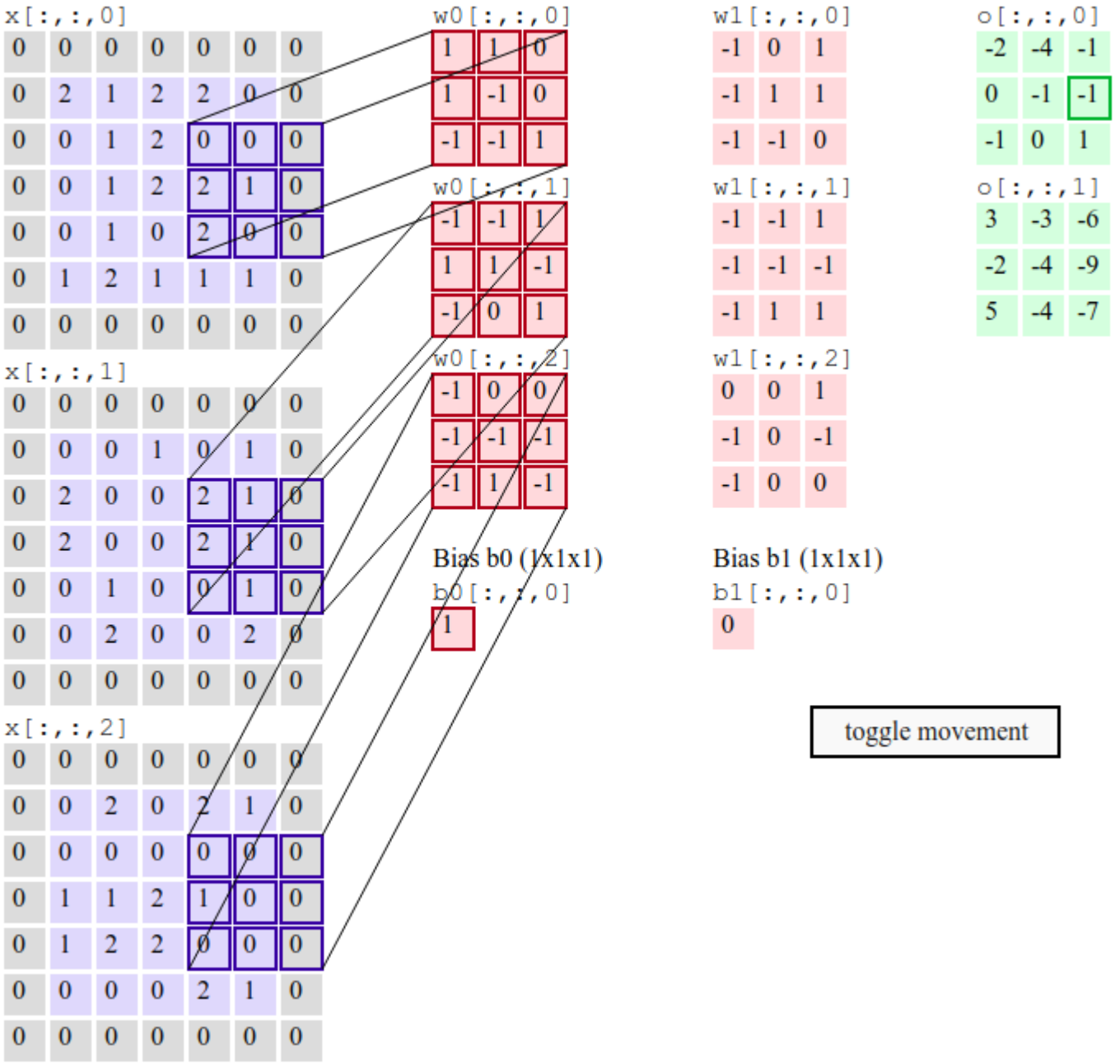

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

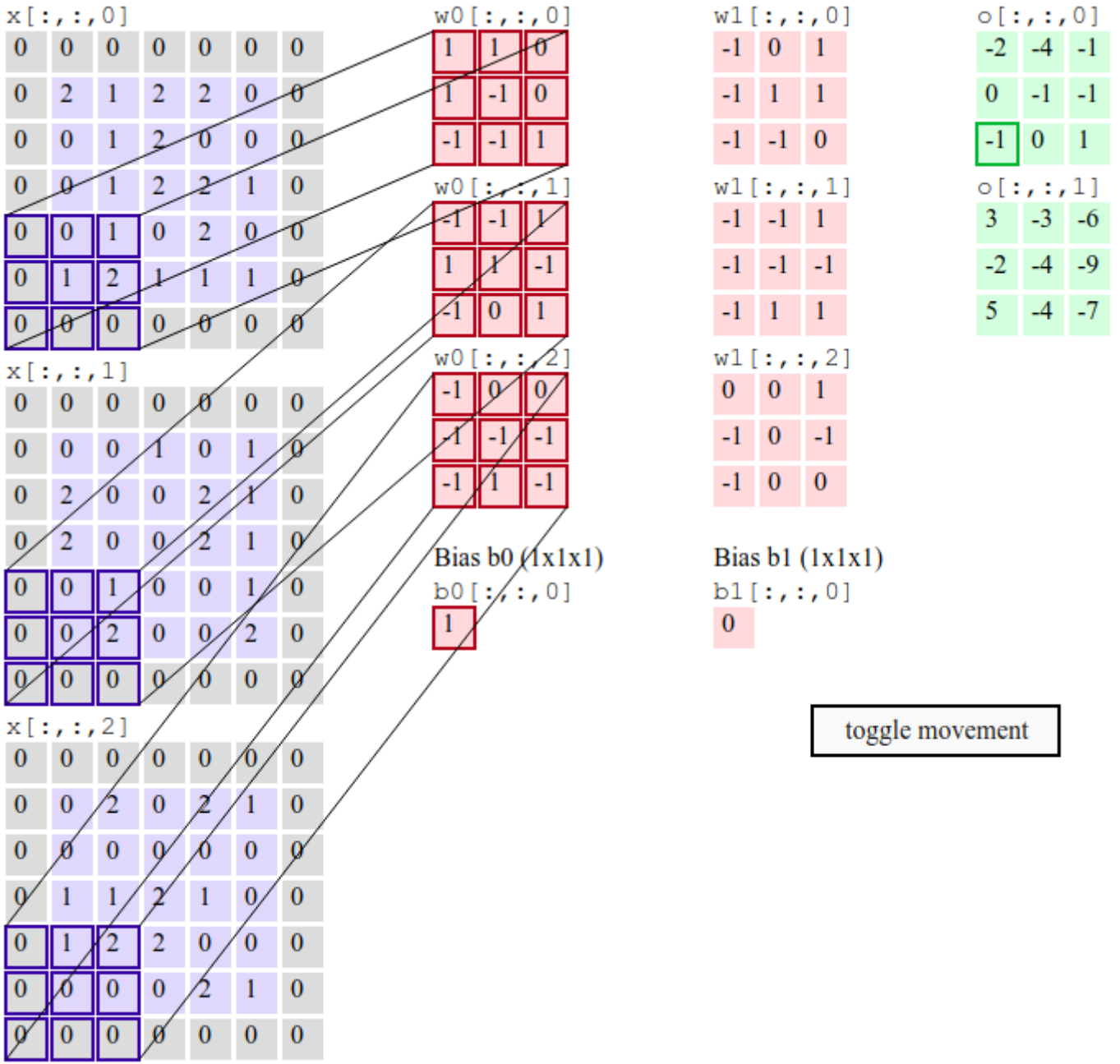

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

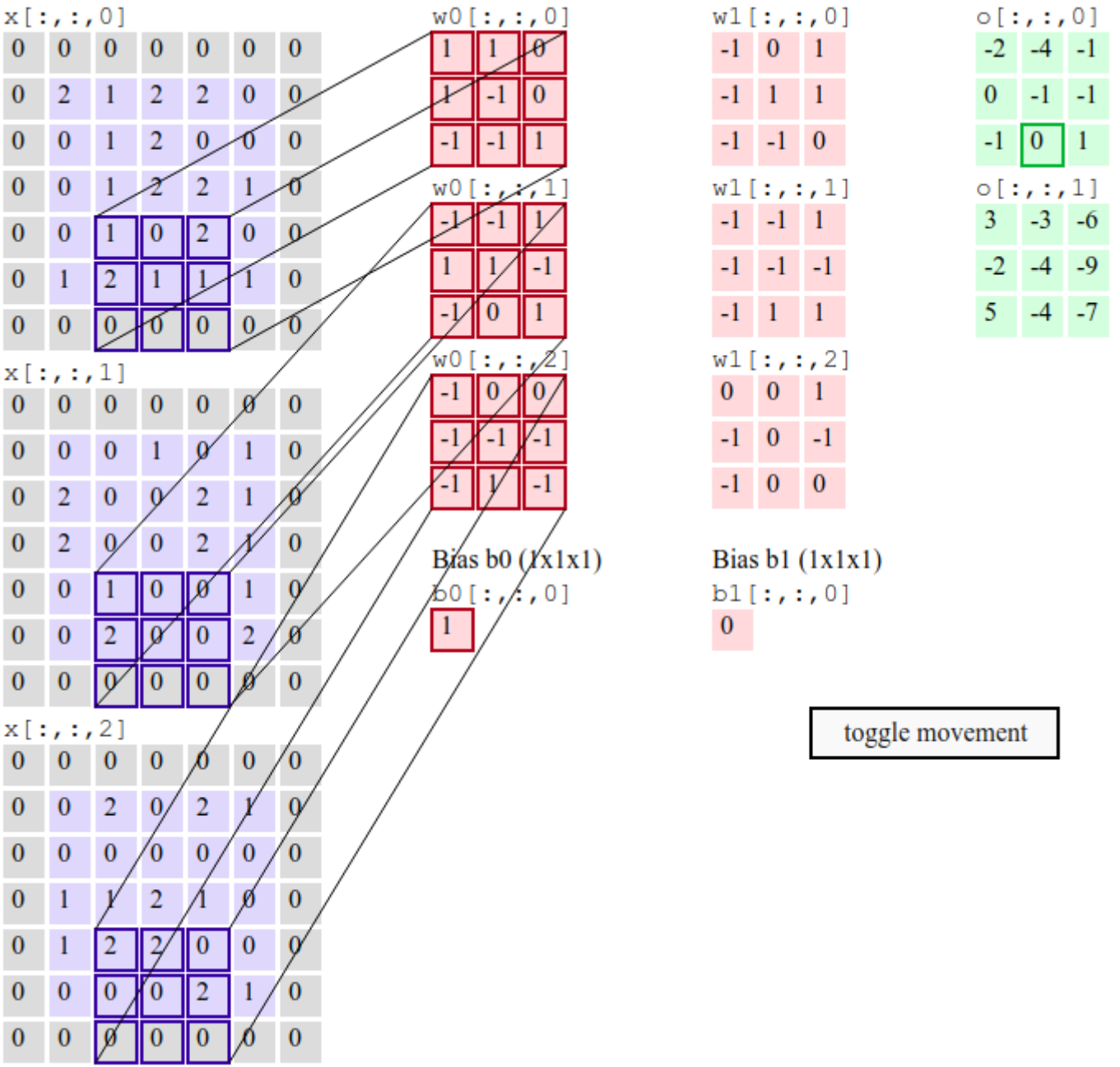

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

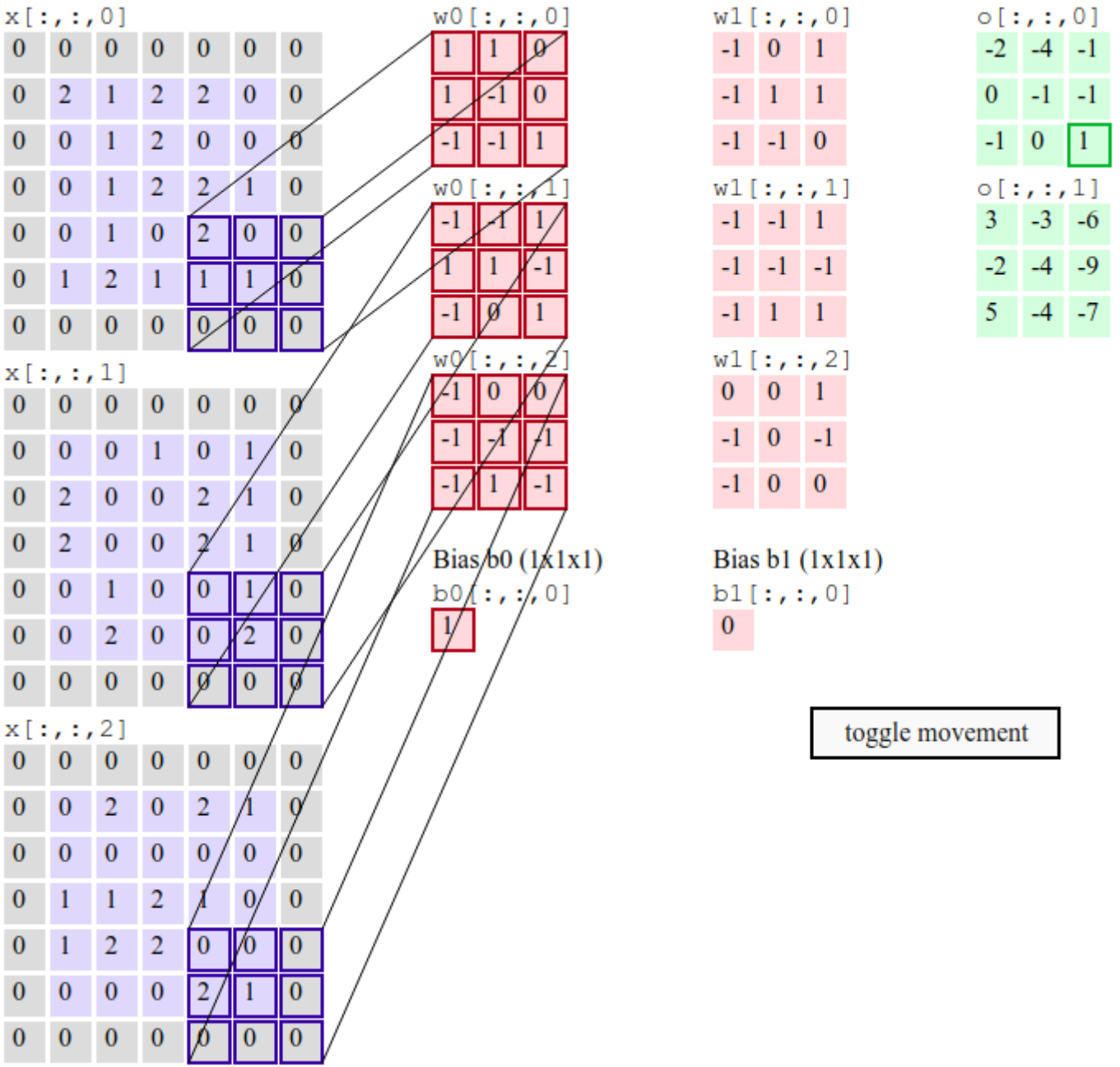

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

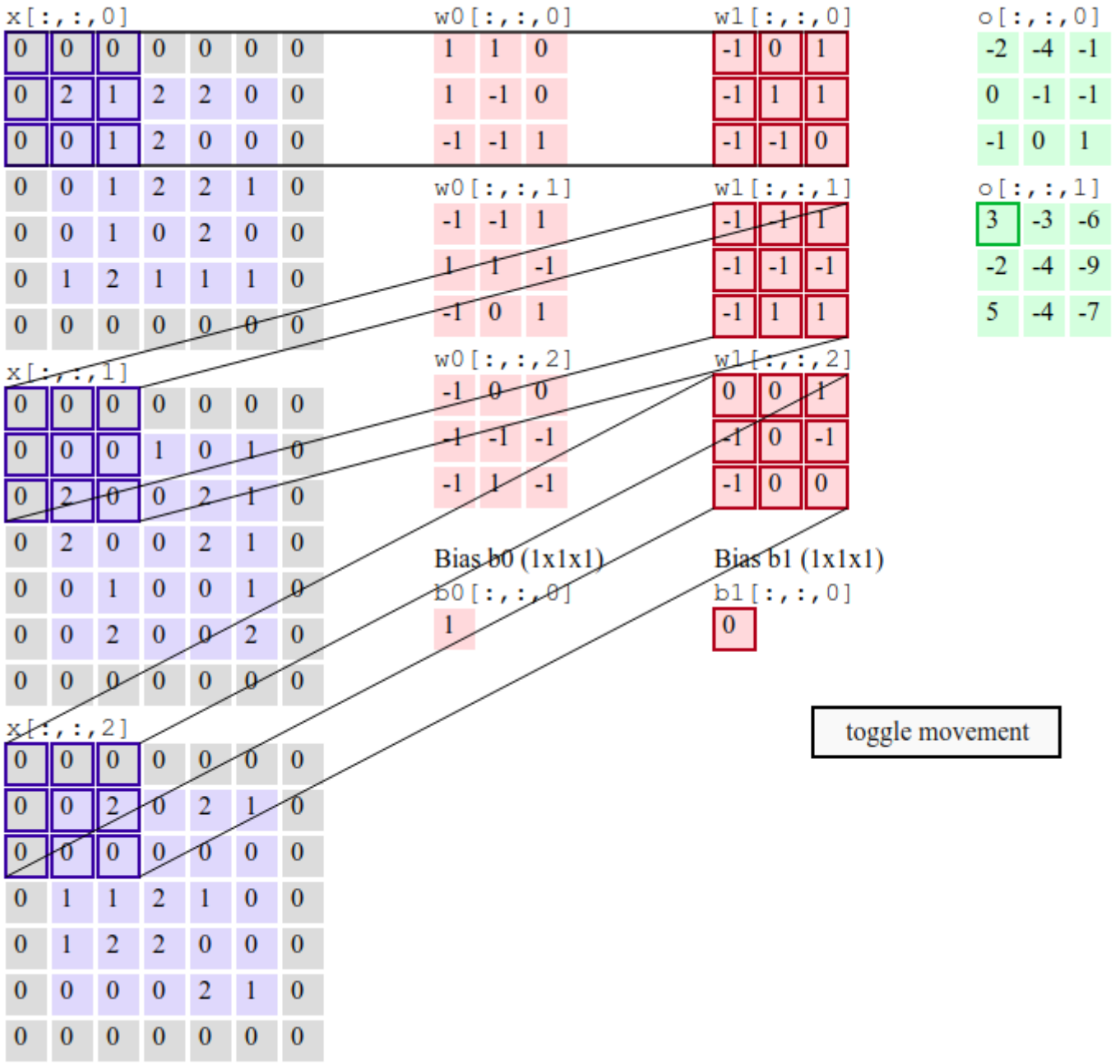

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

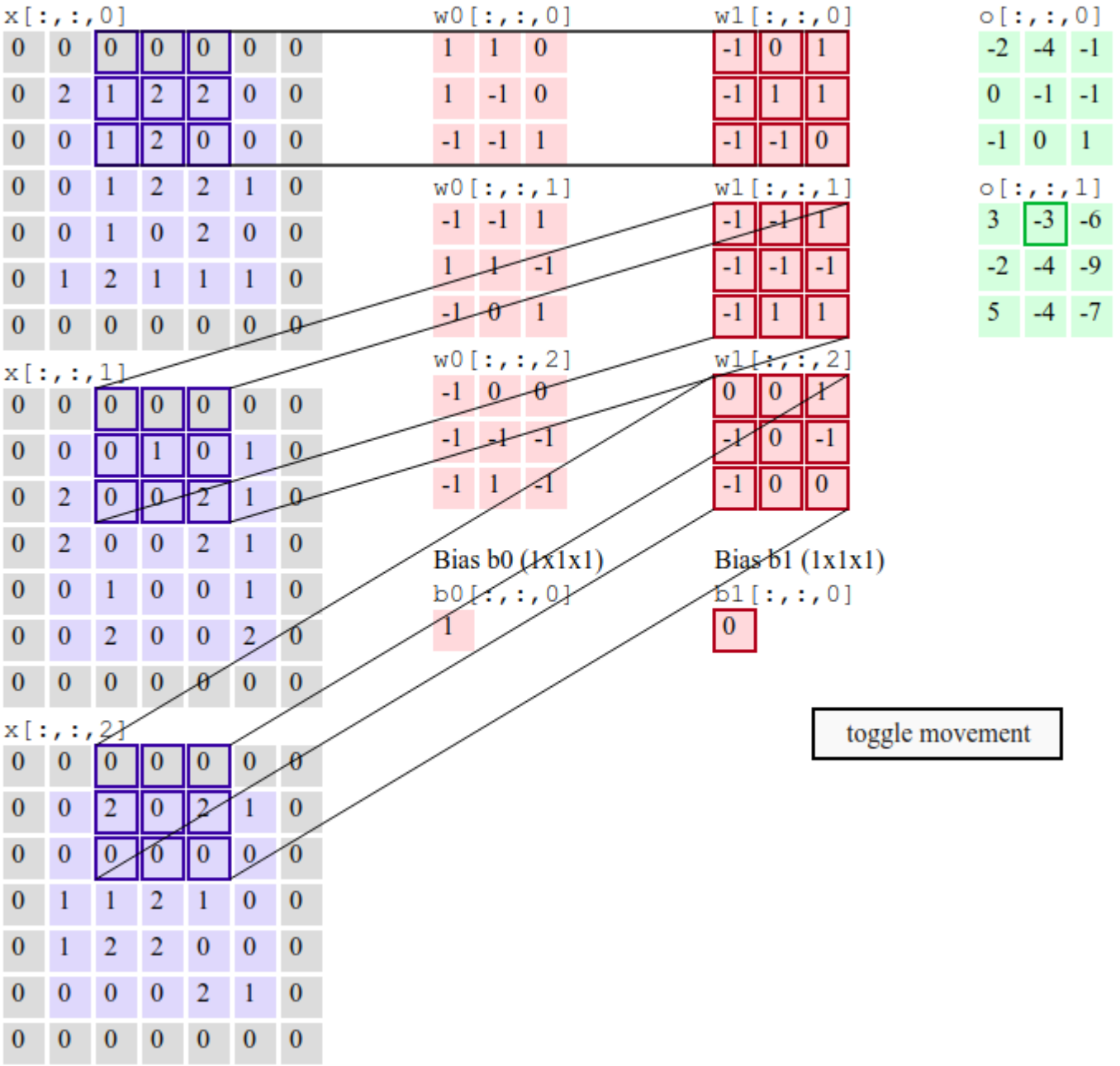

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

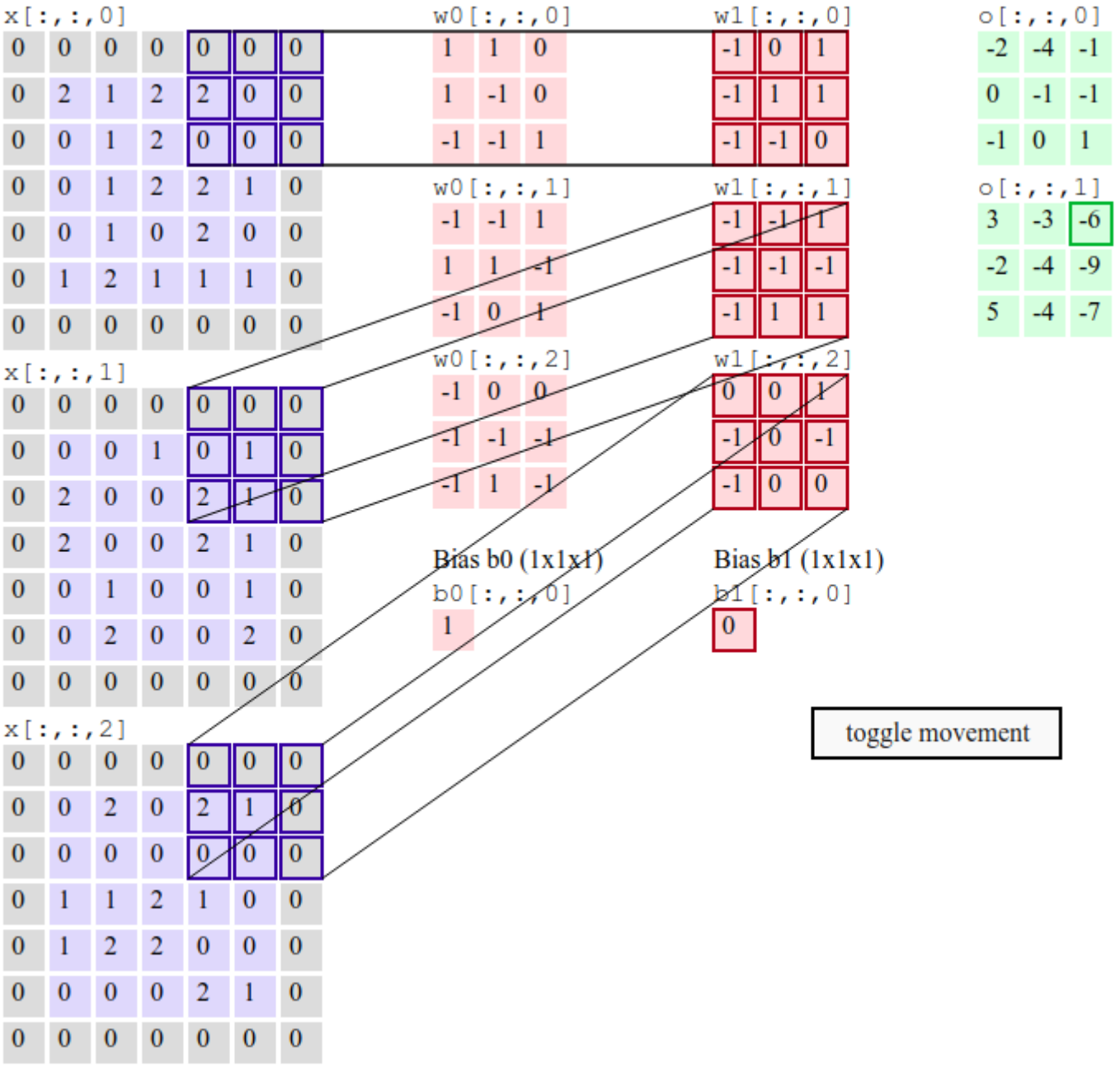

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

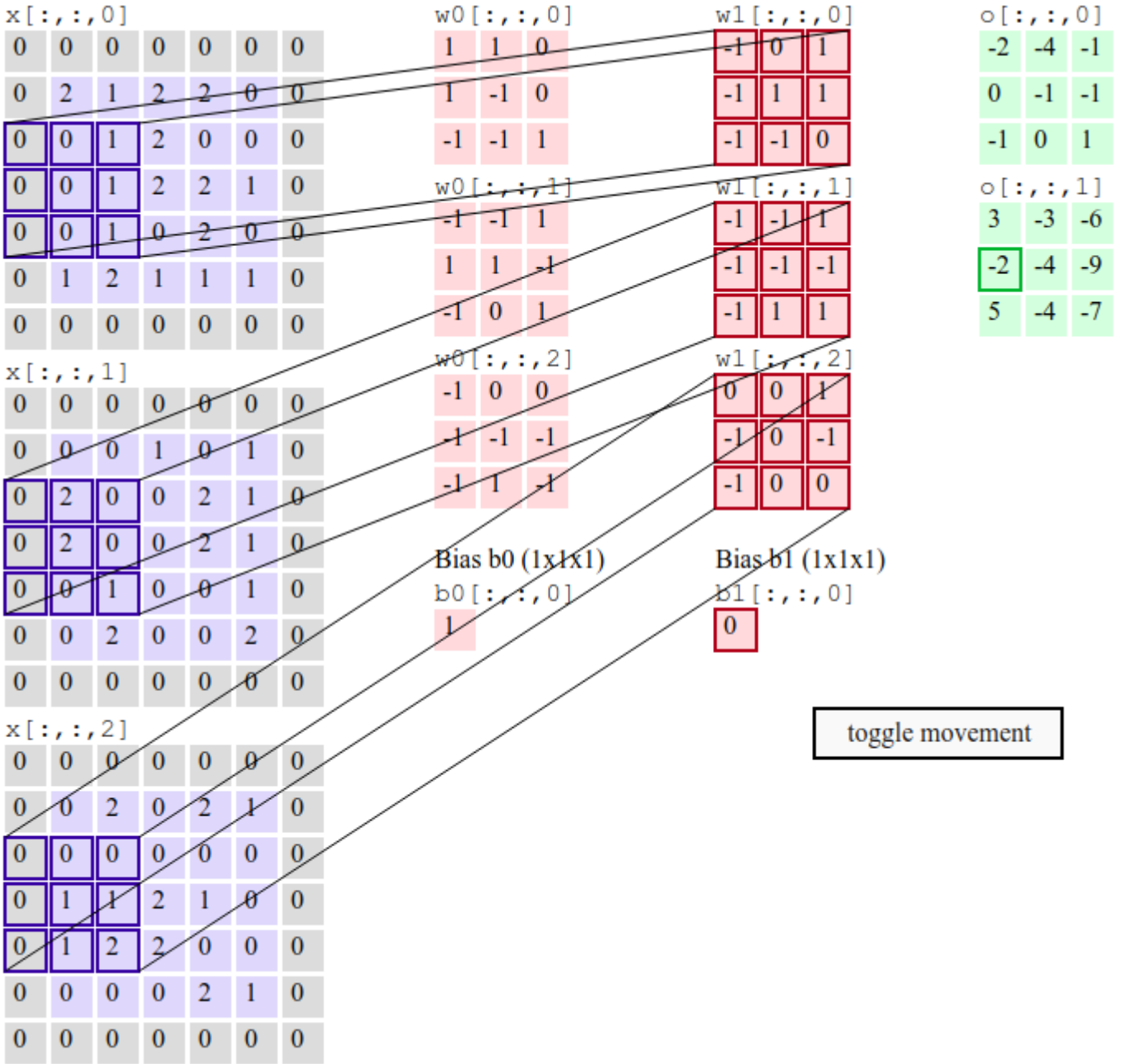

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

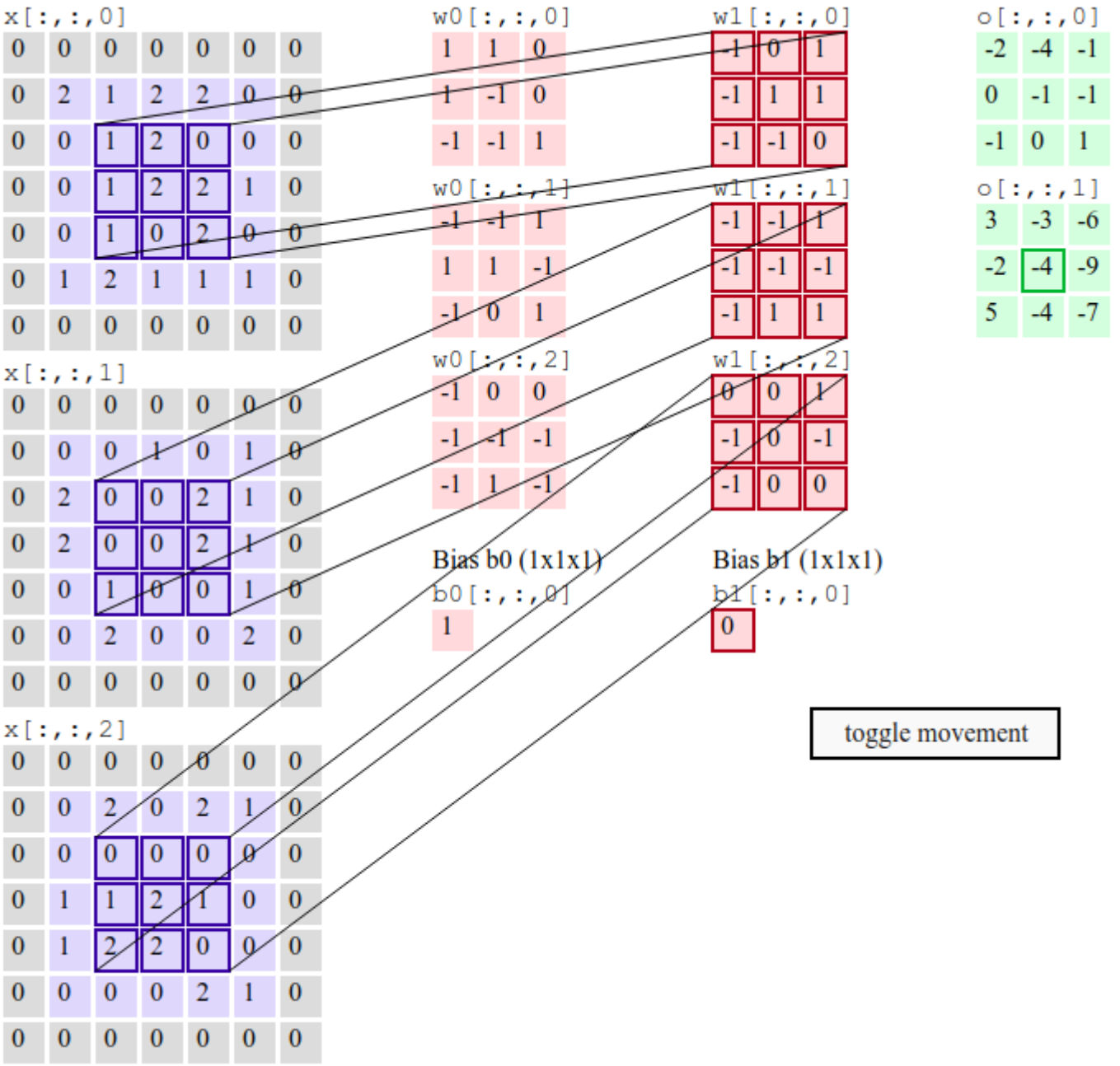

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

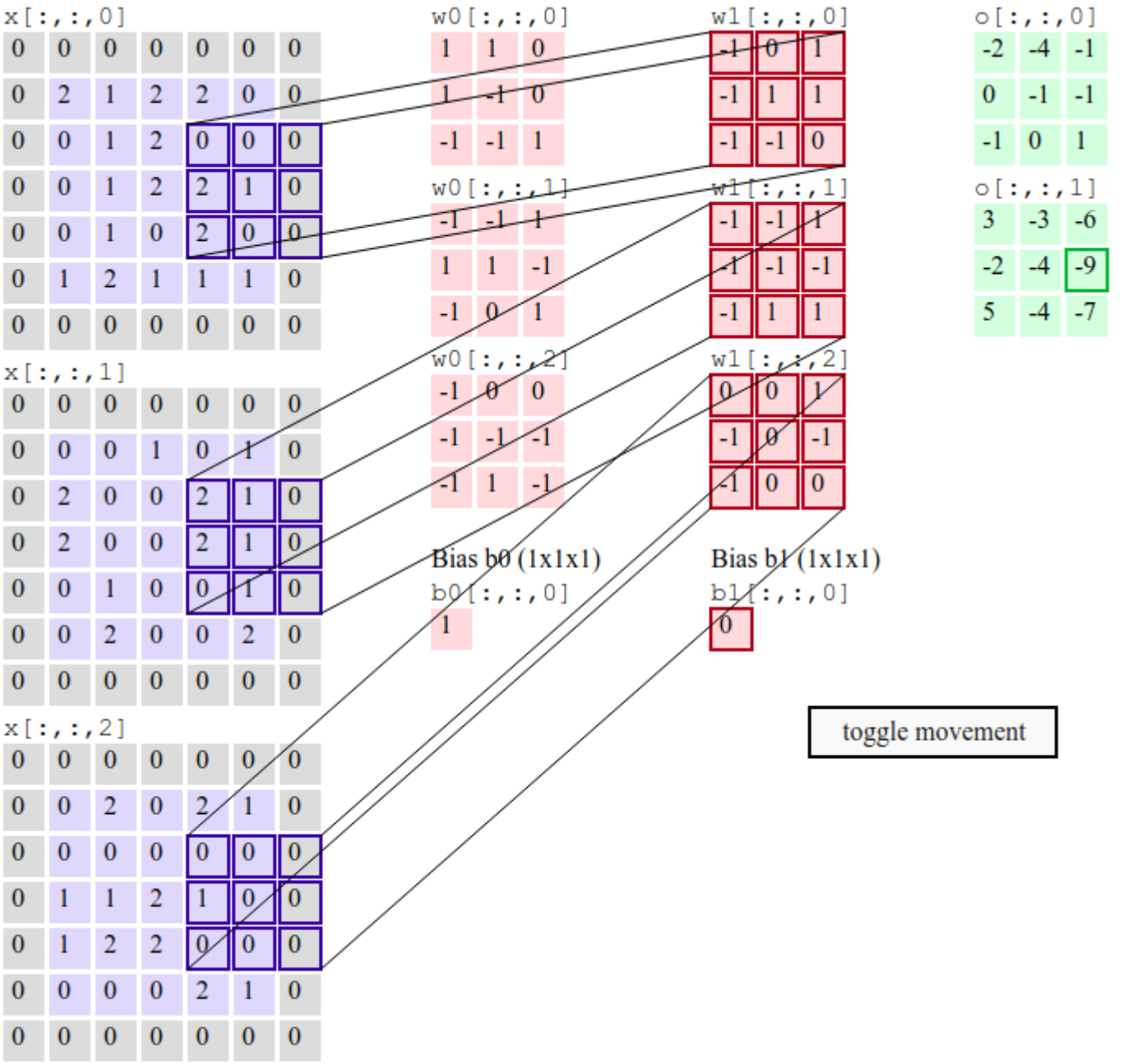

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b0 (1x1x1)
b0[:,:,0]
1
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

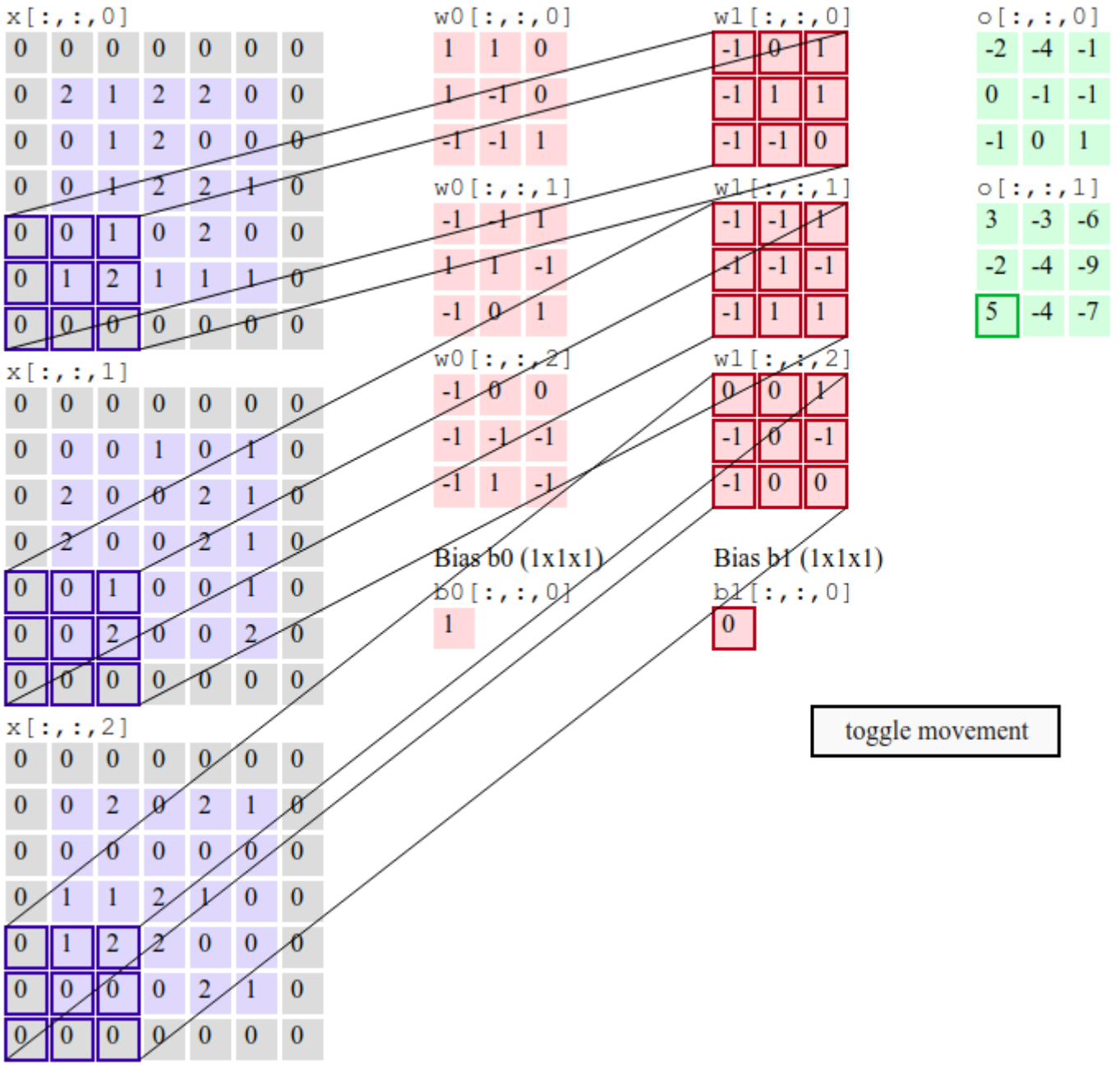

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b0 (1x1x1)
b0[:,:,0]
1
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

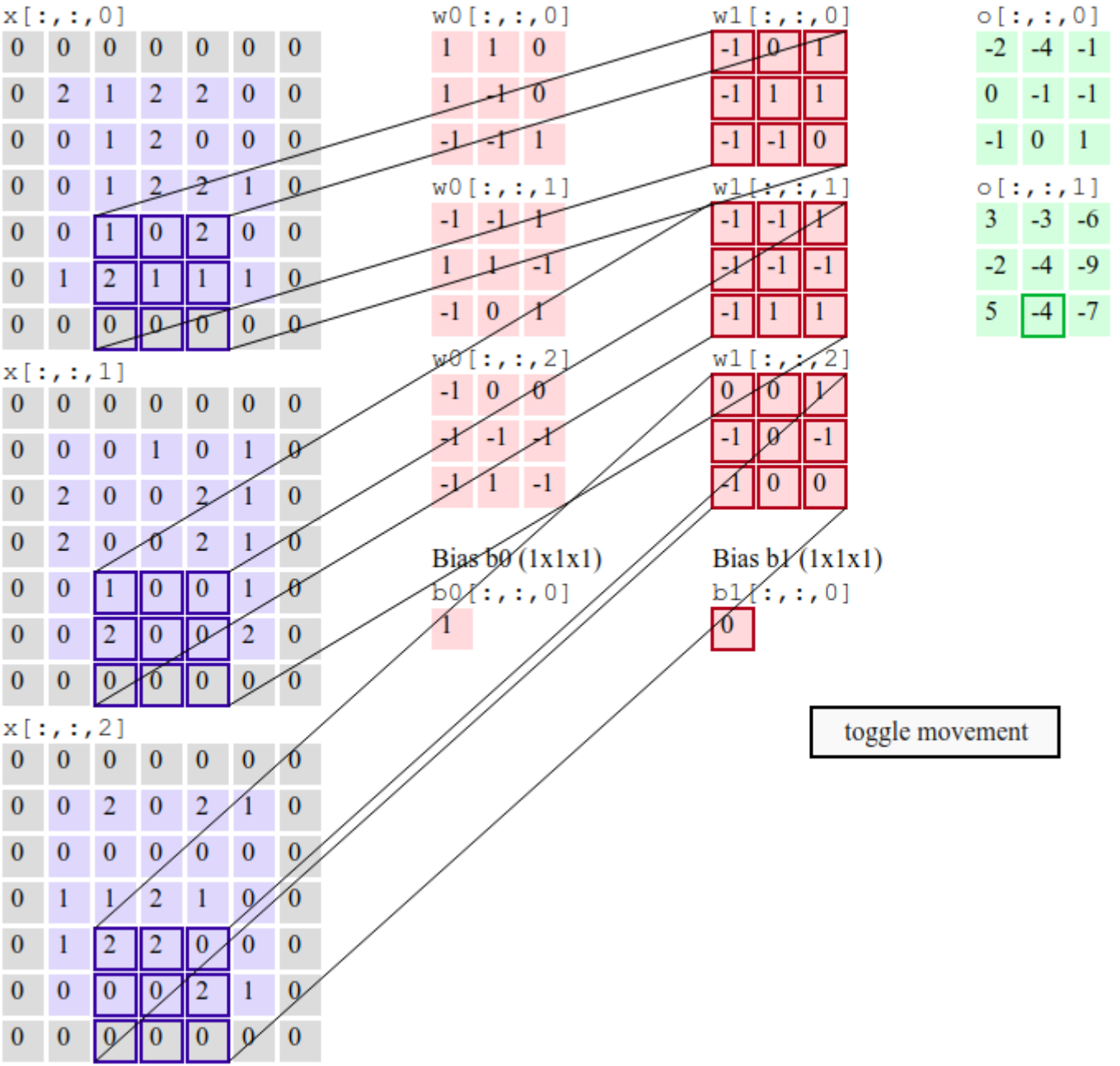

x[:,:,0]
0 0 0 0
0 2 1 2
0 0 1 2
0 0 1 2
0 0 1 0
0 1 2 1
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0
0 0 0 1
0 2 0 0
0 2 0 0
0 0 1 0
0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0
0 0 2 0
0 0 0 0
0 1 1 2
0 1 2 2
0 0 0 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
Bias b0 (1x1x1)
b0[:,:,0]
1
Bias b1 (1x1x1)
b1[:,:,0]
0
toggle movement

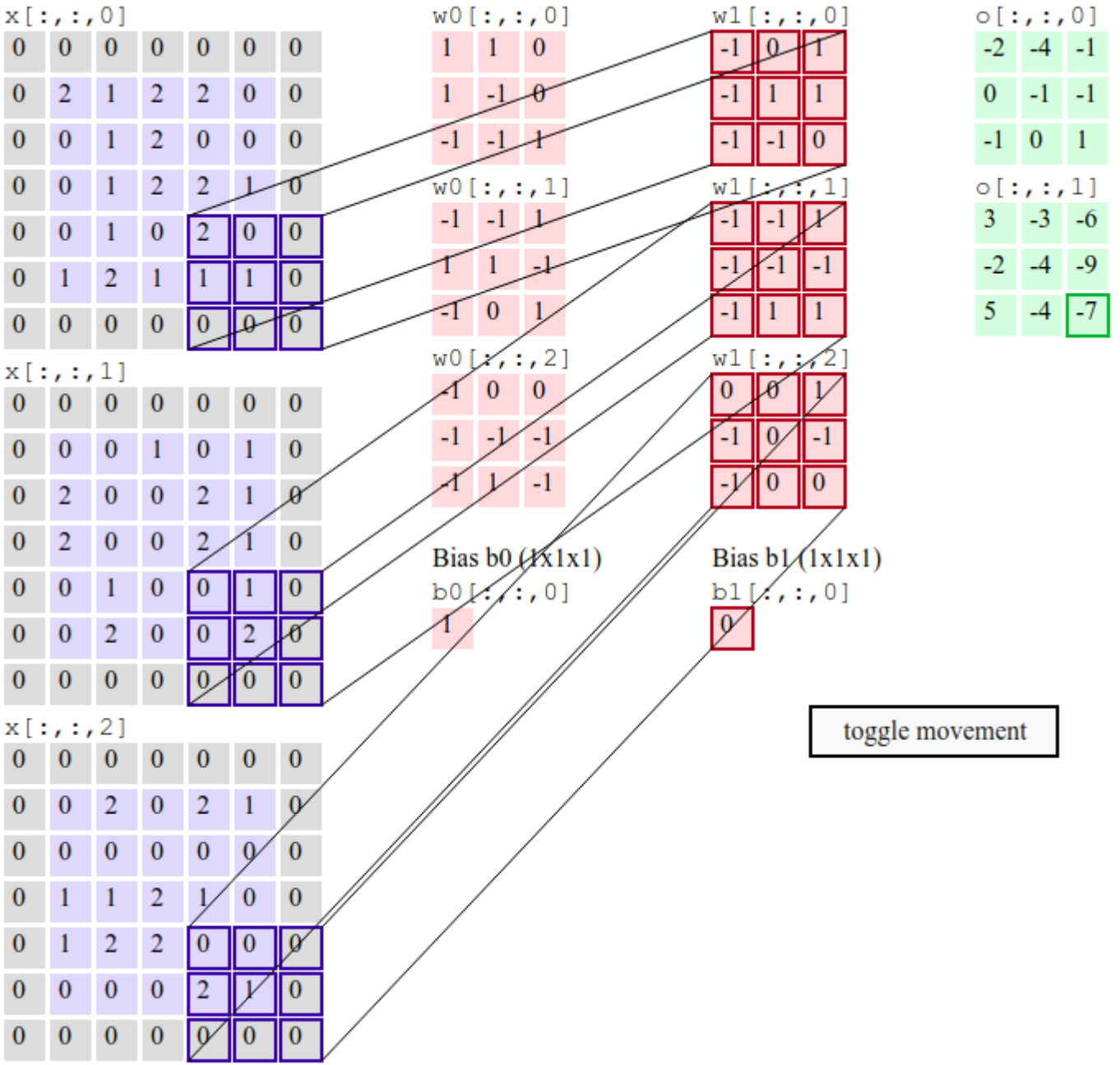

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement


x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

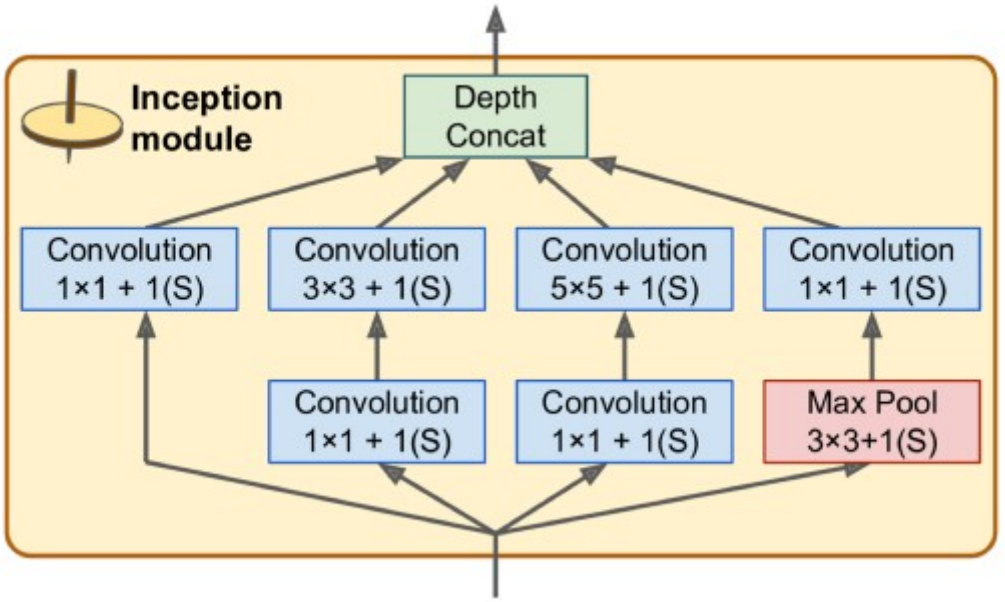

x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement


x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement


x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement


x[:,:,0]
0 0 0 0 0 0 0
0 2 1 2 2 0 0
0 0 1 2 0 0 0
0 0 1 2 2 1 0
0 0 1 0 2 0 0
0 1 2 1 1 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 1 0
0 2 0 0 2 1 0
0 2 0 0 2 1 0
0 0 1 0 0 1 0
0 0 2 0 0 2 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 2 0 2 1 0
0 0 0 0 0 0 0
0 1 1 2 1 0 0
0 1 2 2 0 0 0
0 0 0 0 2 1 0
0 0 0 0 0 0 0
w0[:,:,0]
1 1 0
1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 1 -1
-1 0 1
w0[:,:,2]
-1 0 0
-1 -1 -1
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
w1[:,:,0]
-1 0 1
-1 1 1
-1 -1 0
w1[:,:,1]
-1 -1 1
-1 -1 -1
-1 1 1
w1[:,:,2]
0 0 1
-1 0 -1
-1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
o[:,:,0]
-2 -4 -1
0 -1 -1
-1 0 1
o[:,:,1]
3 -3 -6
-2 -4 -9
5 -4 -7
toggle movement

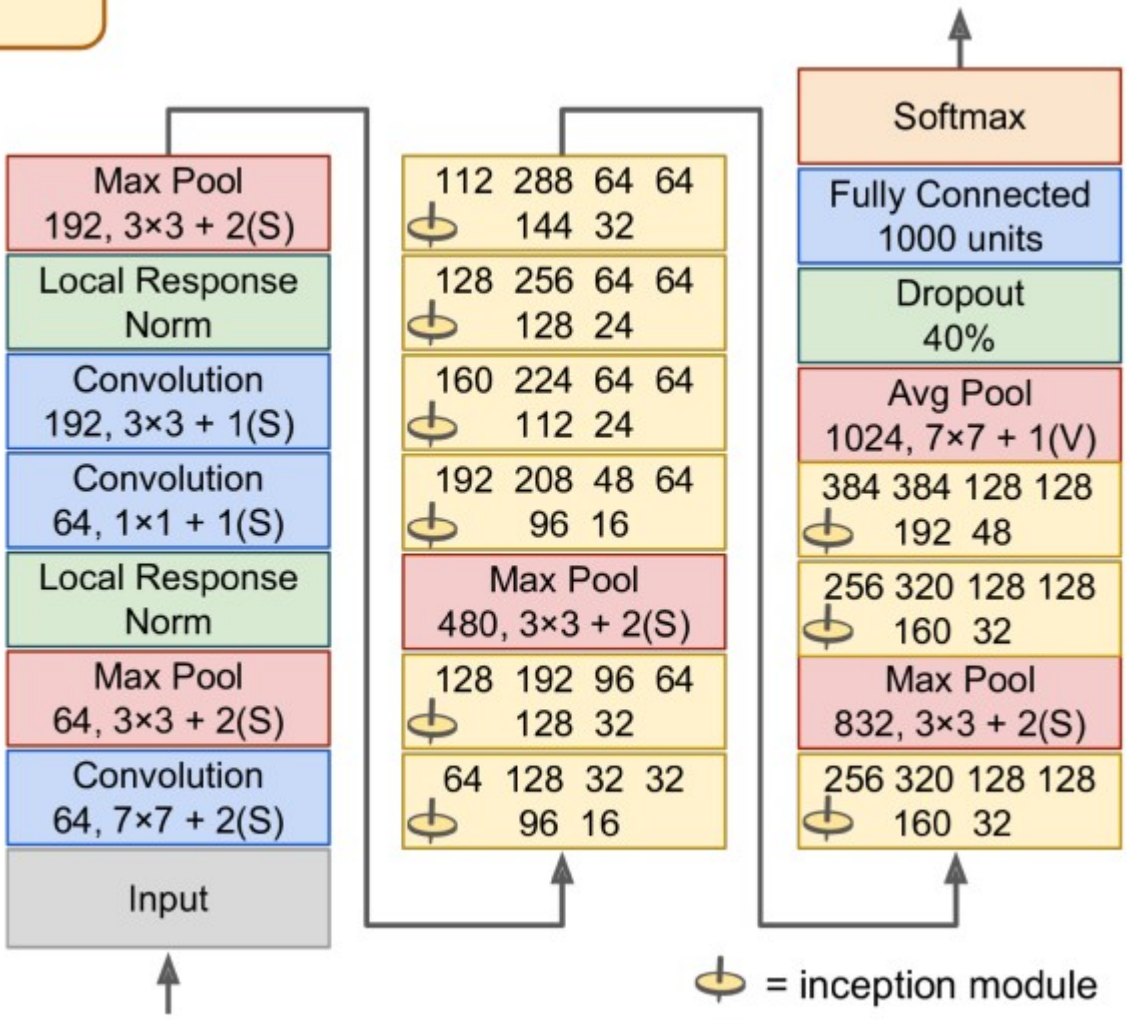
Two famous deep NN architecture



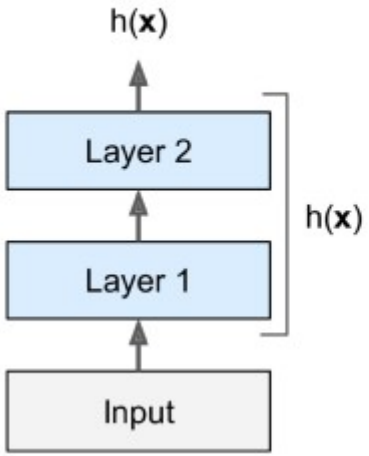
GoogleNet

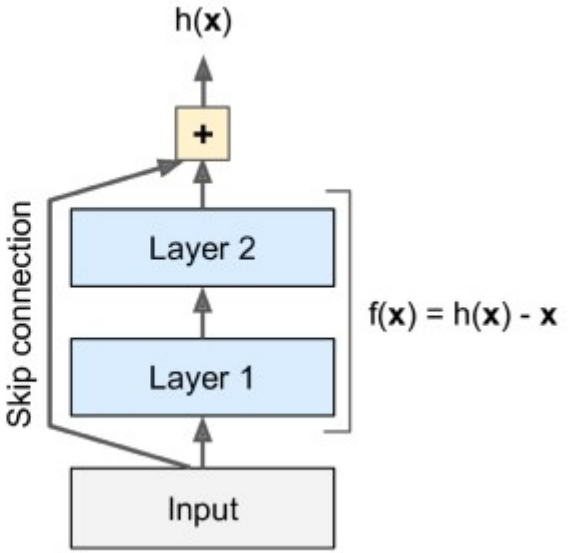
Developers – Christian Szegedy et al. from Google Research.

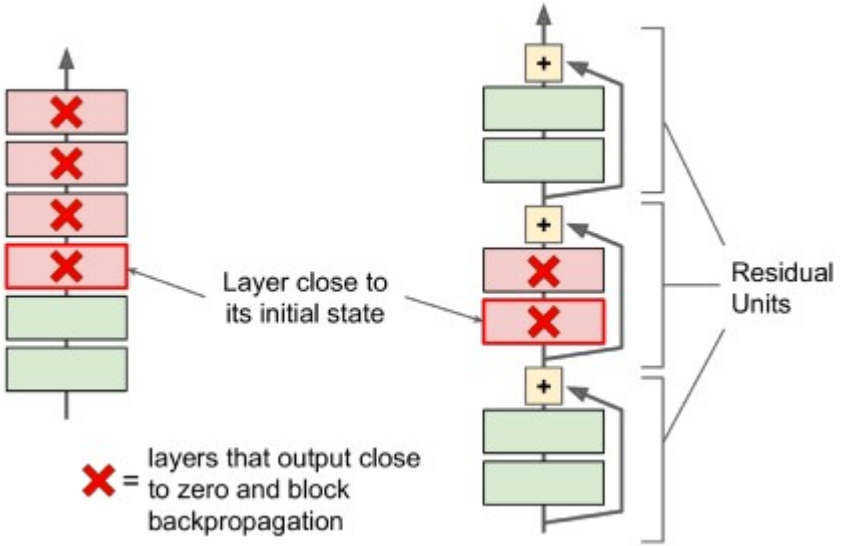
It won the ILSVRC 2014 challenge by pushing the top-5 error rate below 7%



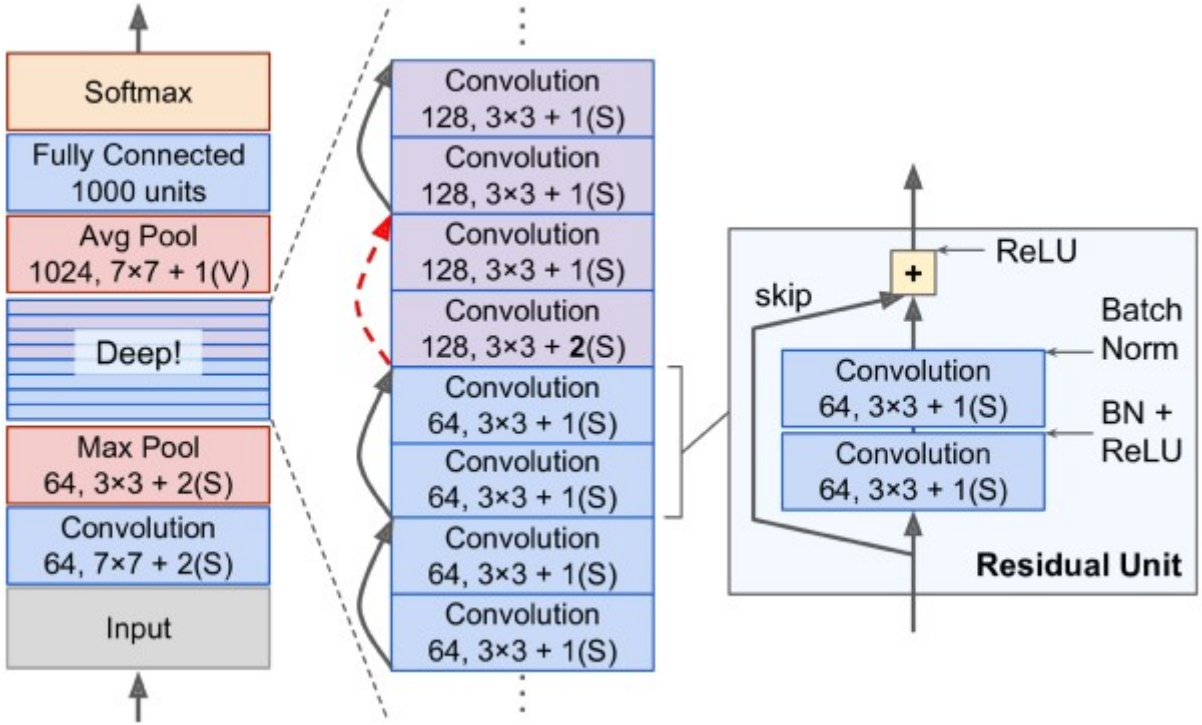
ResNet







winner of the ILSVRC 2015 challenge was the Residual Network (or ResNet), developed by Kaiming He et al.



Machine learning and physics

Quantum chemistry

- Predict binding energy of molecules, orbital energies, structure, etc.
- Success depends on input representation: “molecular fingerprints”.
- Use training data, e.g. from density functional theory (DFT) or Hartree Fock calculations.
- Errors achieved are comparable to error of QM numerics used to construct the training data.
- “Delta Learning”: Use results from less expensive methods (DFT) as input to predict results that would be obtained using more expensive methods.
- Typical errors about 0.1 eV

Ref. Garrett B. Goh, Nathan O. Hodas, Abhinav Vishnu “Deep Learning for Computational Chemistry”, https://arxiv.org/pdf/1701.04503.pdf

Computer-aided drug design

- drug discovery (activity with respect to some target molecule, toxicity)
- Databases exist containing the properties of 100 million compounds (PubChem) and 100 thousand protein structures (Protein Data Bank)
- Databases list the experimentally determined activity of millions of compounds in 100s (or more) of different experimental tests [some are proprietary, held by pharmaceutical companies]
- A typical task may be to predict whether there will be a certain kind of reaction (with some target), given an encoding of the molecule as input
- Deep neural networks recently won challenges for predicting activity and toxicity of chemical compounds
- “multi-task” nets: trained on predicted multiple properties; work better because they seem to build a more useful internal representation

Ref. Bharath Ramsundar, Steven Kearnes, Patrick Riley, Dale Webster, David Konerding, Vijay Pande, “Massively Multitask Networks for Drug Discovery”, https://arxiv.org/pdf/1502.02072.pdf
Ref. Garrett B. Goh, Nathan O. Hodas, Abhinav Vishnu “Deep Learning for Computational Chemistry”, https://arxiv.org/pdf/1701.04503.pdf

Computational structural biology

- Protein folding: given the structure, how does the final shape look like? Extremely challenging (molecular dynamics possible, but too slow for larger structures)
- One particular task: “Given two parts of the sequence that are apart, will they be close to each other in the folded structure?” (“protein contact prediction”)
- Previously, 30% was the best success rate for this task; deep neural nets have pushed this to 36%

Ref. Garrett B. Goh, Nathan O. Hodas, Abhinav Vishnu “Deep Learning for Computational Chemistry”, https://arxiv.org/pdf/1701.04503.pdf

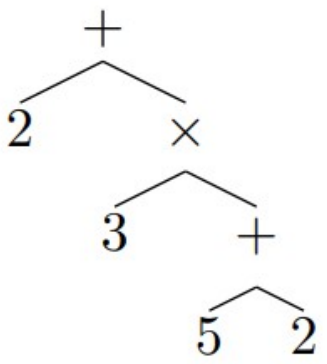
Materials research

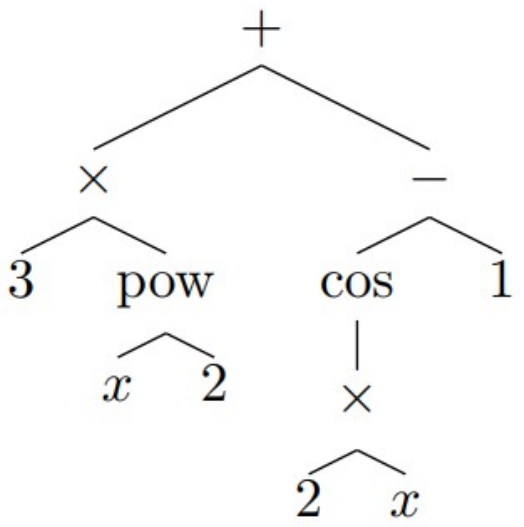
- Find better batteries, better solar cells, materials for storing molecules (like absorbing CO2)
- in general: predict crystal structures and properties (electronic, melting point, heat conductvity, magnetic, ...), of novel/untested materials, or under extreme conditions
- “Materials Genome Project” and similar large-scale efforts to collect various databases (of real materials that have been synthesized at some time, and also of computed materials, and of extrapolated/predicted properties)
- Functional materials (e.g. solar cells) vs. structural materials (e.g. steel). The latter depend on how they are processed: very difficult to model!

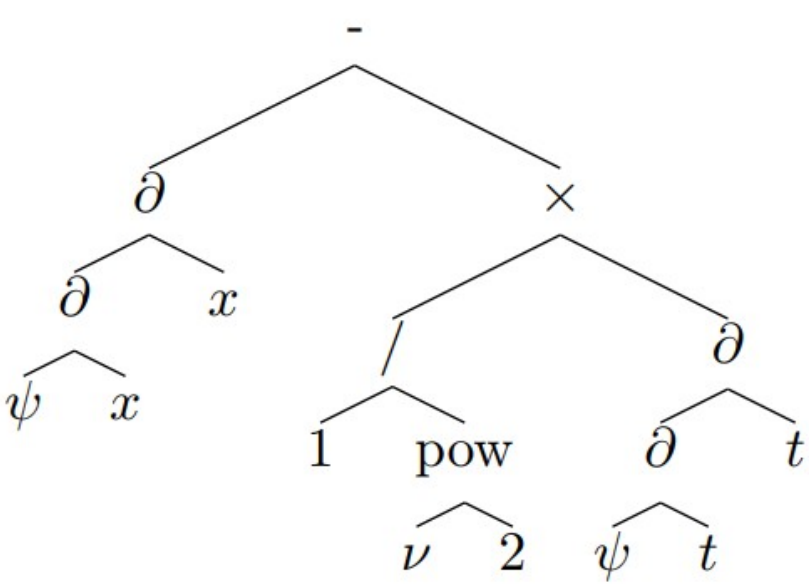
Ref. Garrett B. Goh, Nathan O. Hodas, Abhinav Vishnu “Deep Learning for Computational Chemistry”, https://arxiv.org/pdf/1701.04503.pdf

Integration

expressions $2 + 3 \times (5 + 2)$, $3x^2 + \cos(2x) - 1$, and $\frac{\partial^2 \psi}{\partial x^2} - \frac{1}{\nu^2} \frac{\partial^2 \psi}{\partial t^2}$:







Integration

Equation	Solution
$y' = \frac{16x^3 - 42x^2 + 2x}{(-16x^8 + 112x^7 - 204x^6 + 28x^5 - x^4 + 1)^{1/2}}$	$y = \sin^{-1}(4x^4 - 14x^3 + x^2)$
$3xy\cos(x) - \sqrt{9x^2\sin(x)^2 + 1}y' + 3y\sin(x) = 0$	$y = c\exp\left(\sinh^{-1}(3x\sin(x))\right)$
$4x^4yy'' - 8x^4y'^2 - 8x^3yy' - 3x^3y'' - 8x^2y^2 - 6x^2y' - 3x^2y'' - 9xy' - 3y = 0$	$y = \frac{c_1 + 3x + 3\log(x)}{x(c_2 + 4x)}$

Ref. Guillaume Lample, François Charton, “Deep Learning for Symbolic Mathematics”, https://arxiv.org/abs/1912.01412

Thank you!