MACHINE LEARNING, NUCLEAR PHYSICS, AND ALGORITHM DEVELOPMENT FOR DATA ANALYSIS IN NUCLEAR RESEARCH

MICHELLE KUCHERA
DAVIDSON COLLEGE

IAEA WORKSHOP ON COMPUTATIONAL NUCLEAR SCIENCE AND ENGINEERING
16 JULY 2021

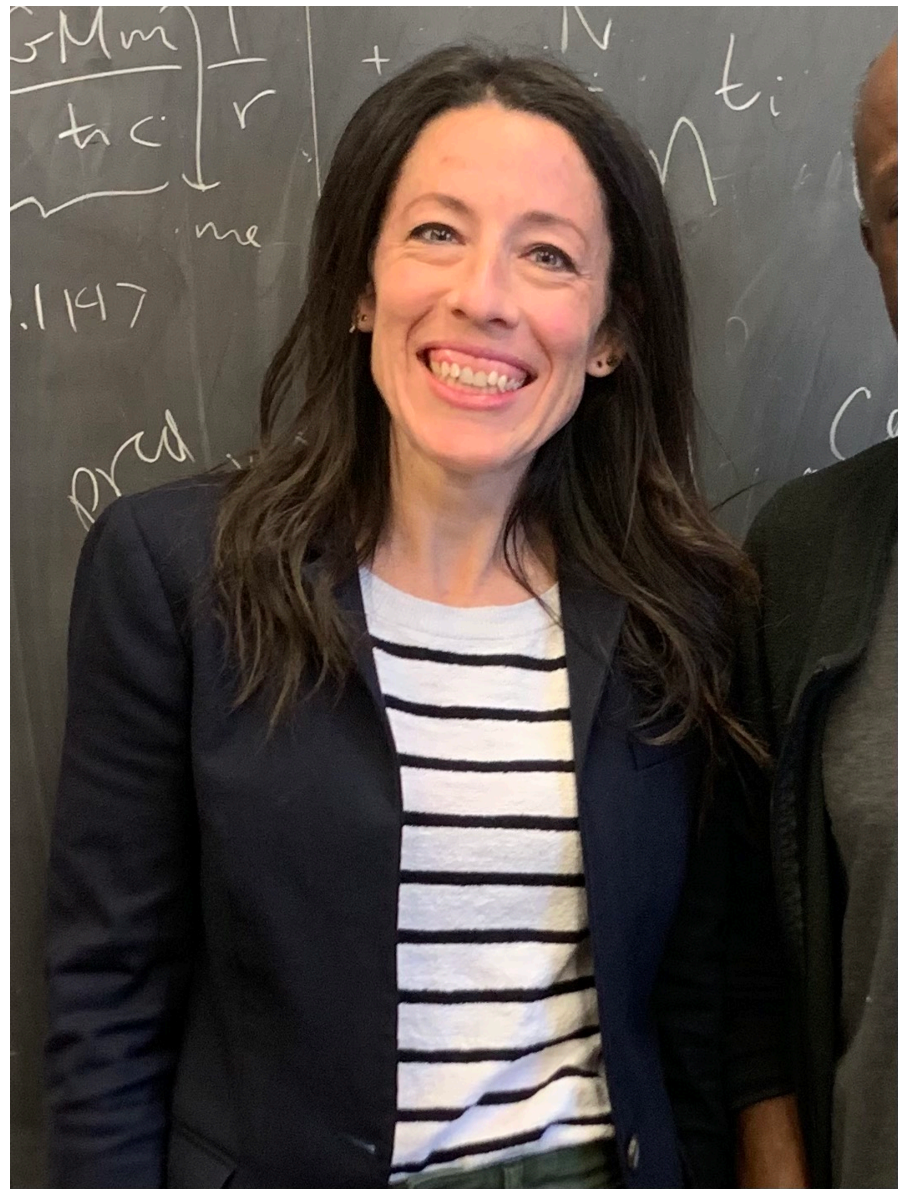
MACHINE LEARNING, NUCLEAR PHYSICS, AND ALGORITHM DEVELOPMENT FOR DATA ANALYSIS IN NUCLEAR RESEARCH

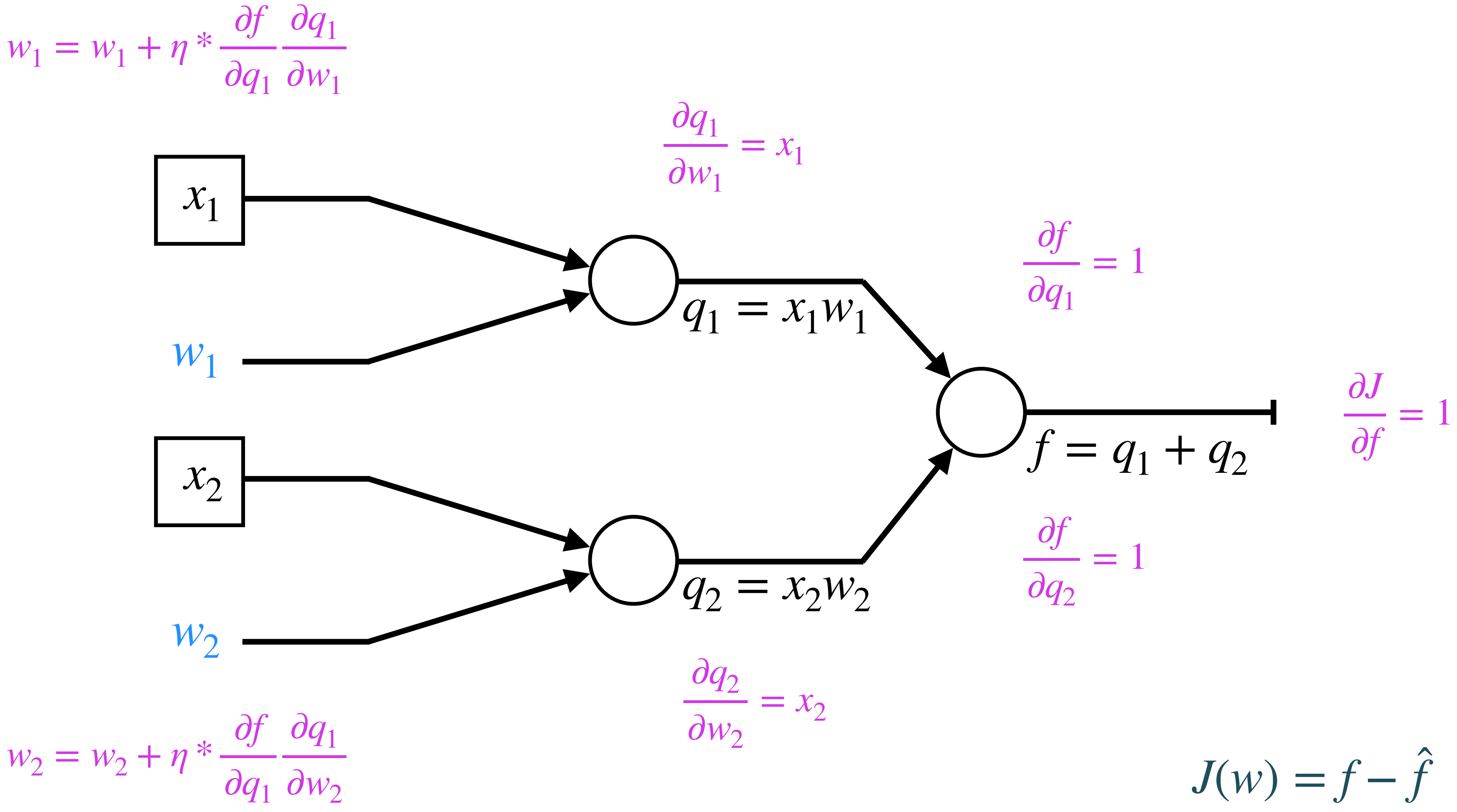
MICHELLE KUCHERA
DAVIDSON COLLEGE

IAEA WORKSHOP ON COMPUTATIONAL NUCLEAR SCIENCE AND ENGINEERING
16 JULY 2021

MICHELLE KUCHERA

B.S., M.S. PHYSICS
M.S., PH.D. COMPUTATIONAL SCIENCE





ALPhA DAVIDSON COLLEGE

CERN

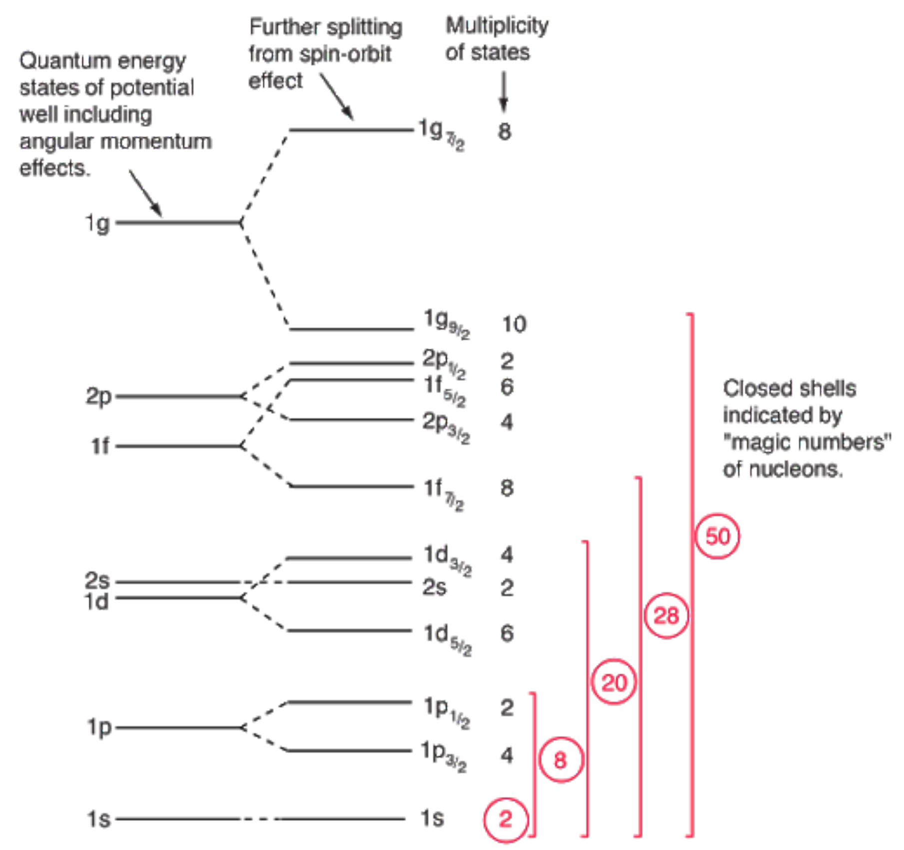
CMS Compact Muon Solenoid

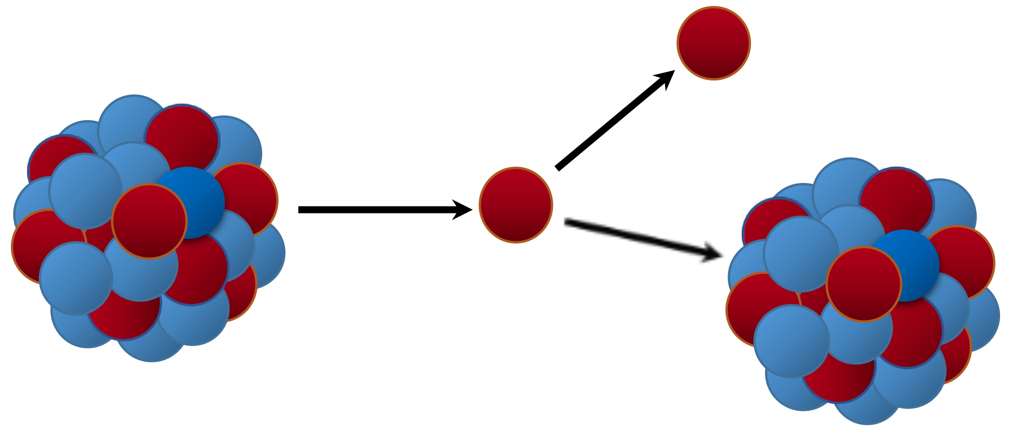
Jefferson Lab

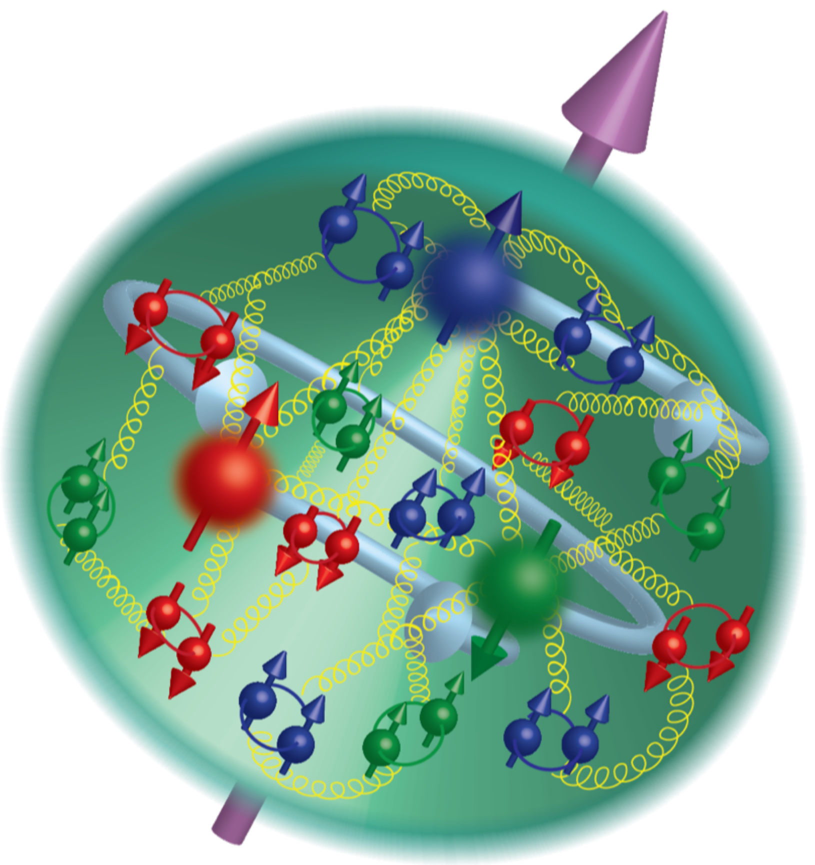
FRIB

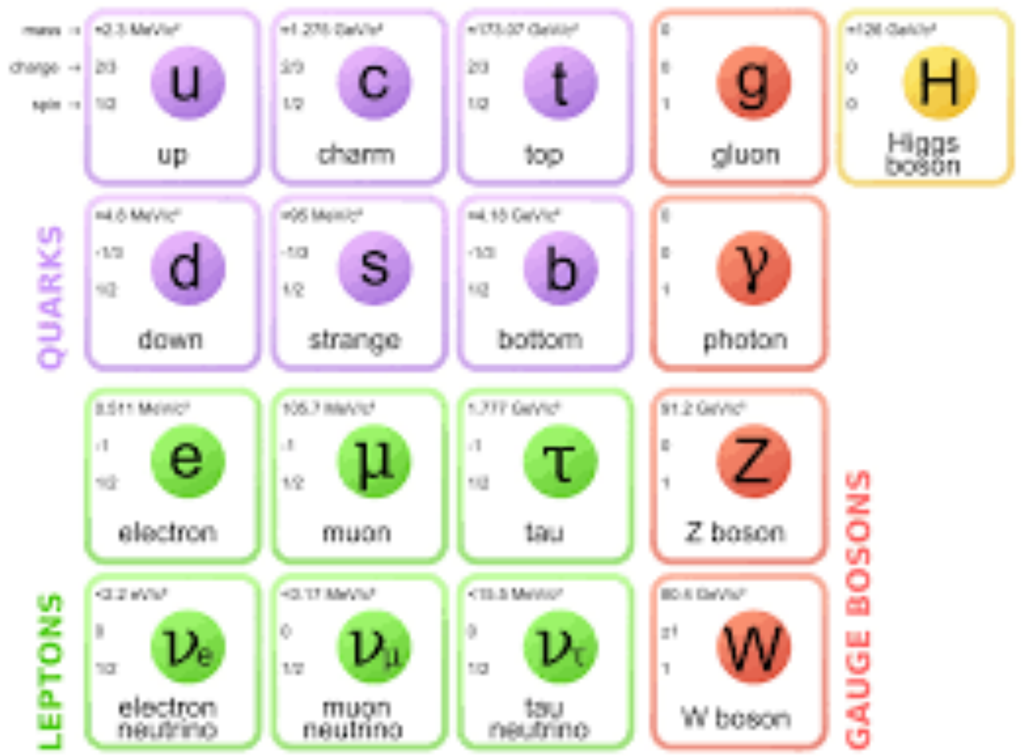
DAVIDSON

NSCL







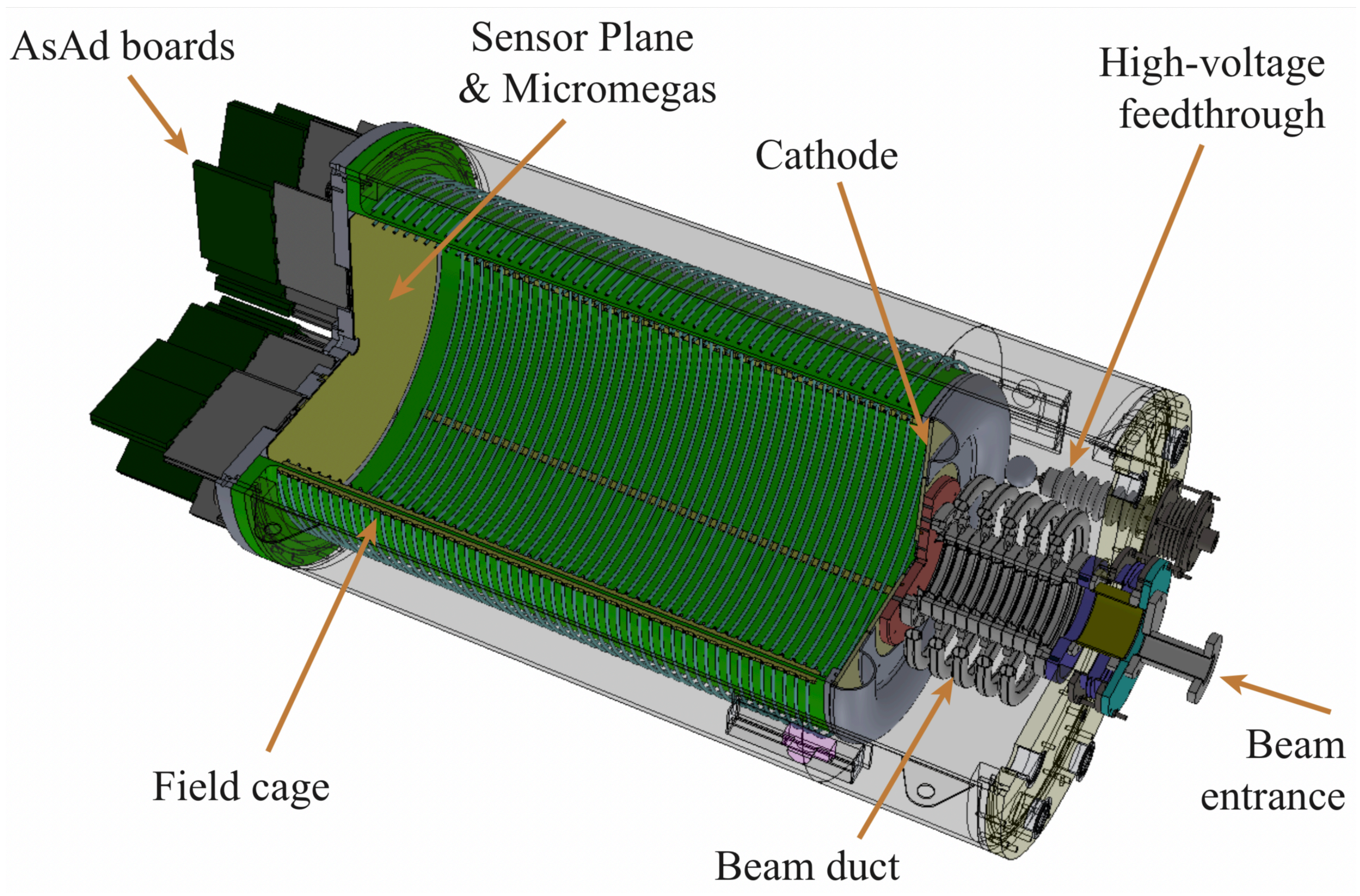


FRIB

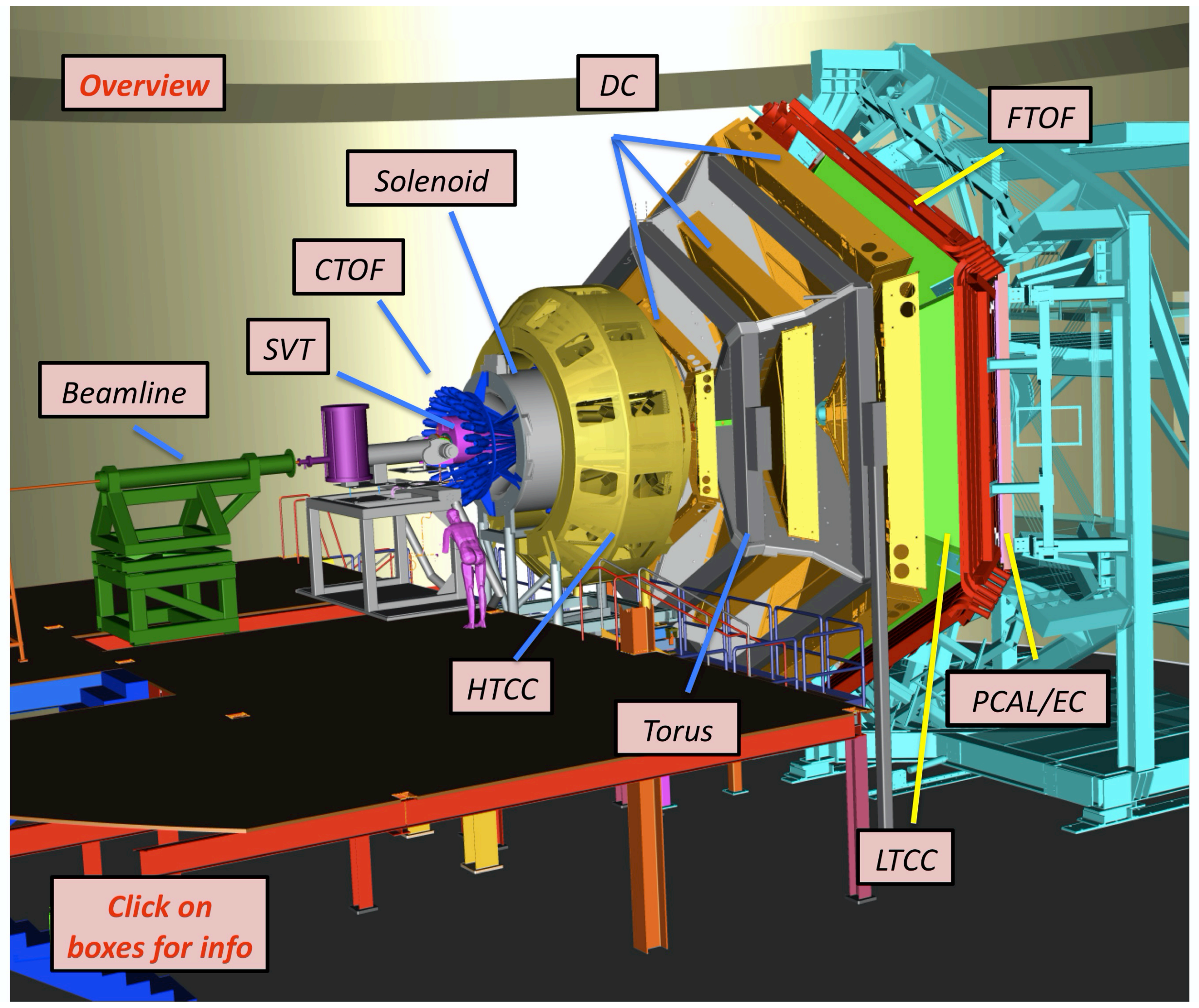
Jefferson Lab

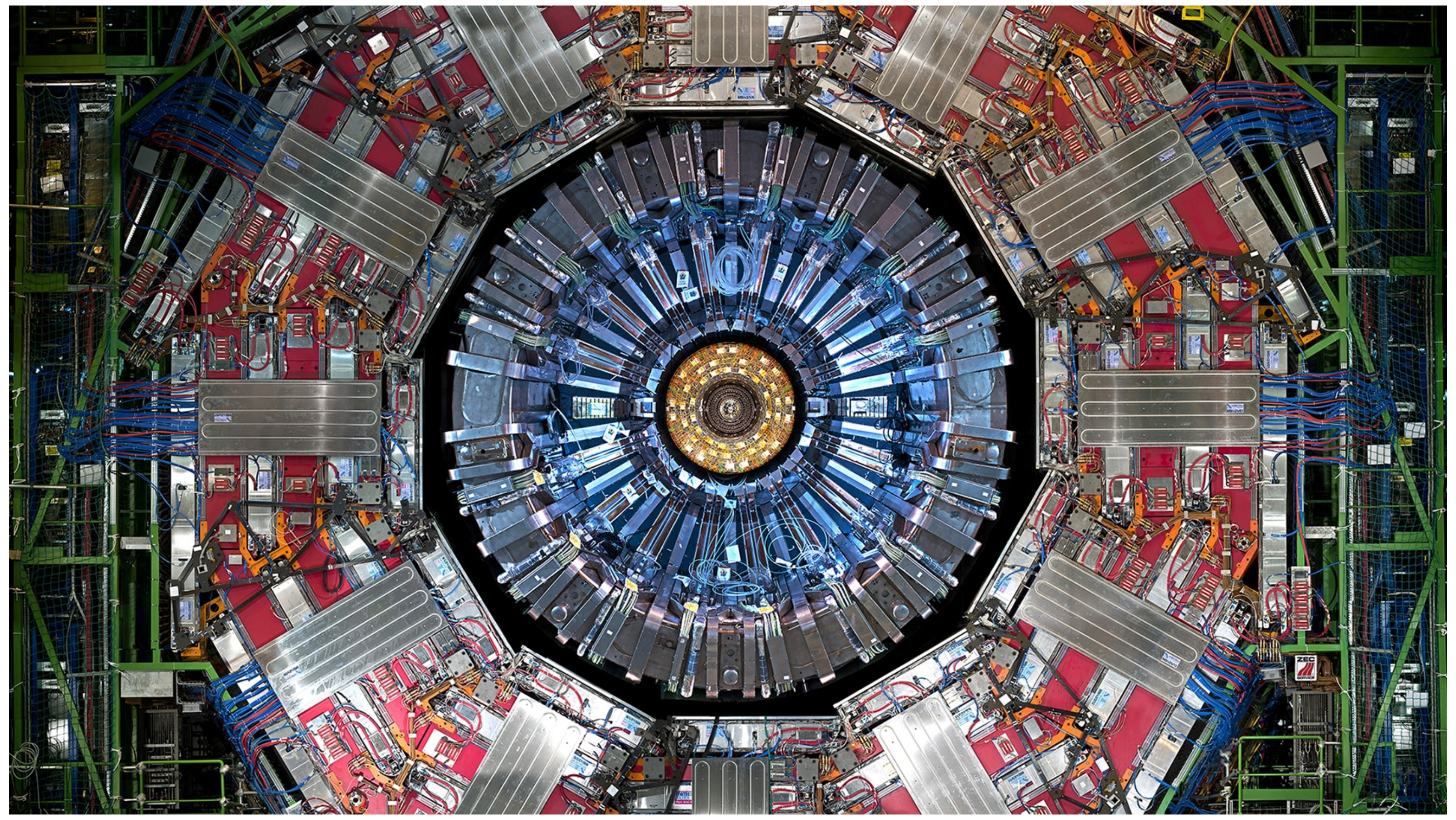
CERN

EXPERIMENTAL DATA



J. BRADT ET. AL., *NUCLEAR INSTRUMENTS AND METHODS*, 2017.





FRIB

AT-TPC

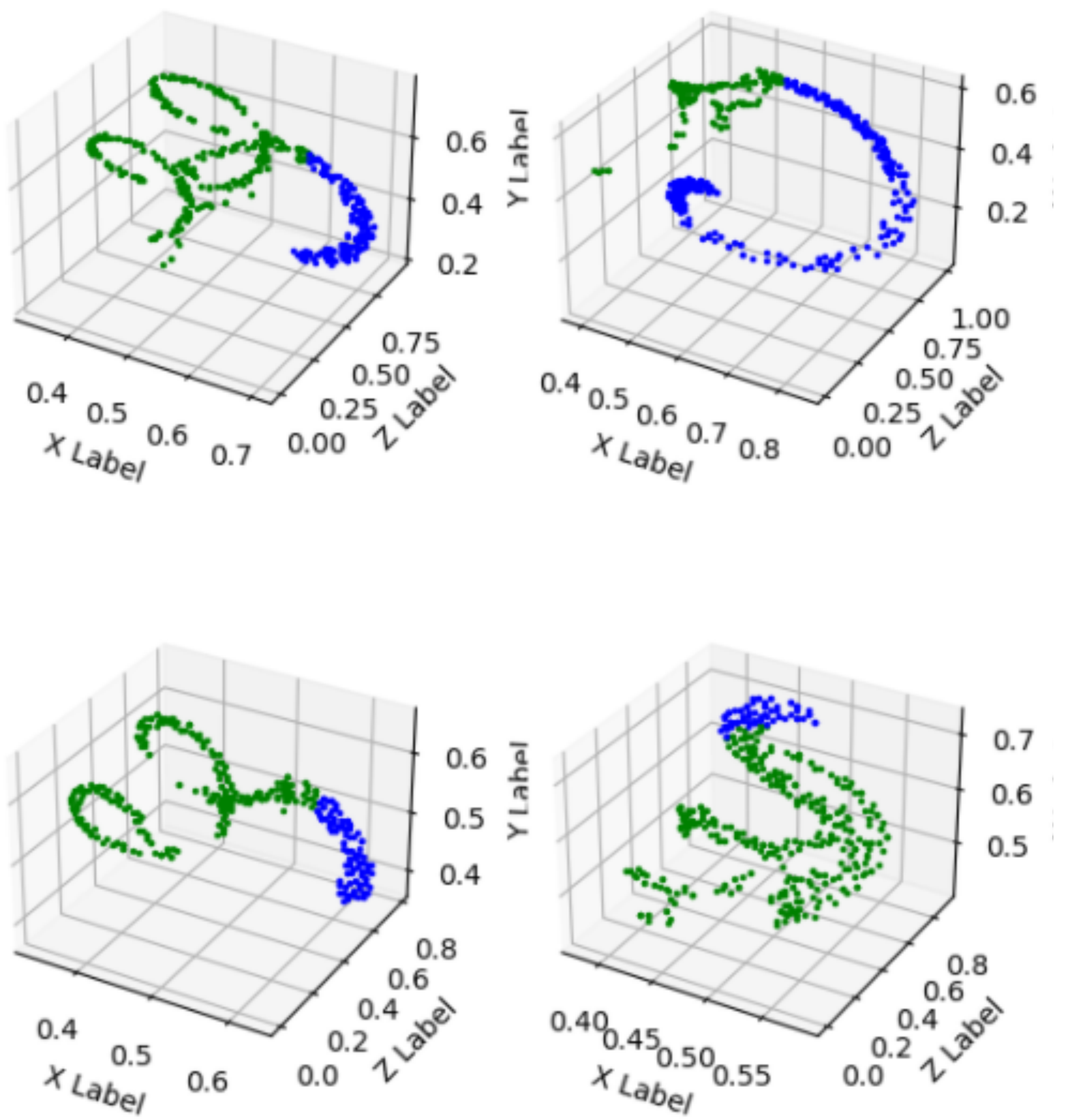
Jefferson Lab

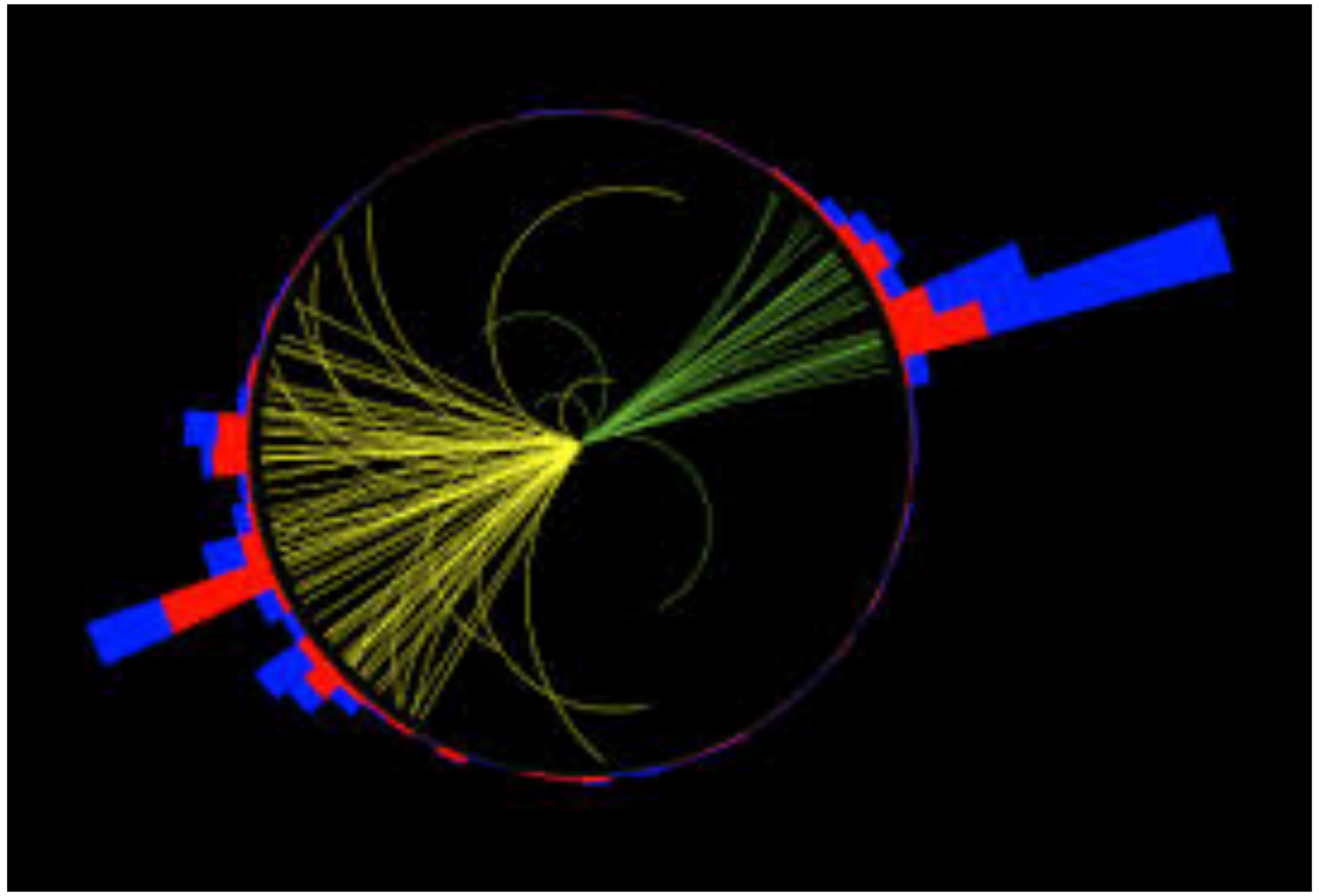
CLAS 12

CERN

CMS

EXPERIMENTAL DATA

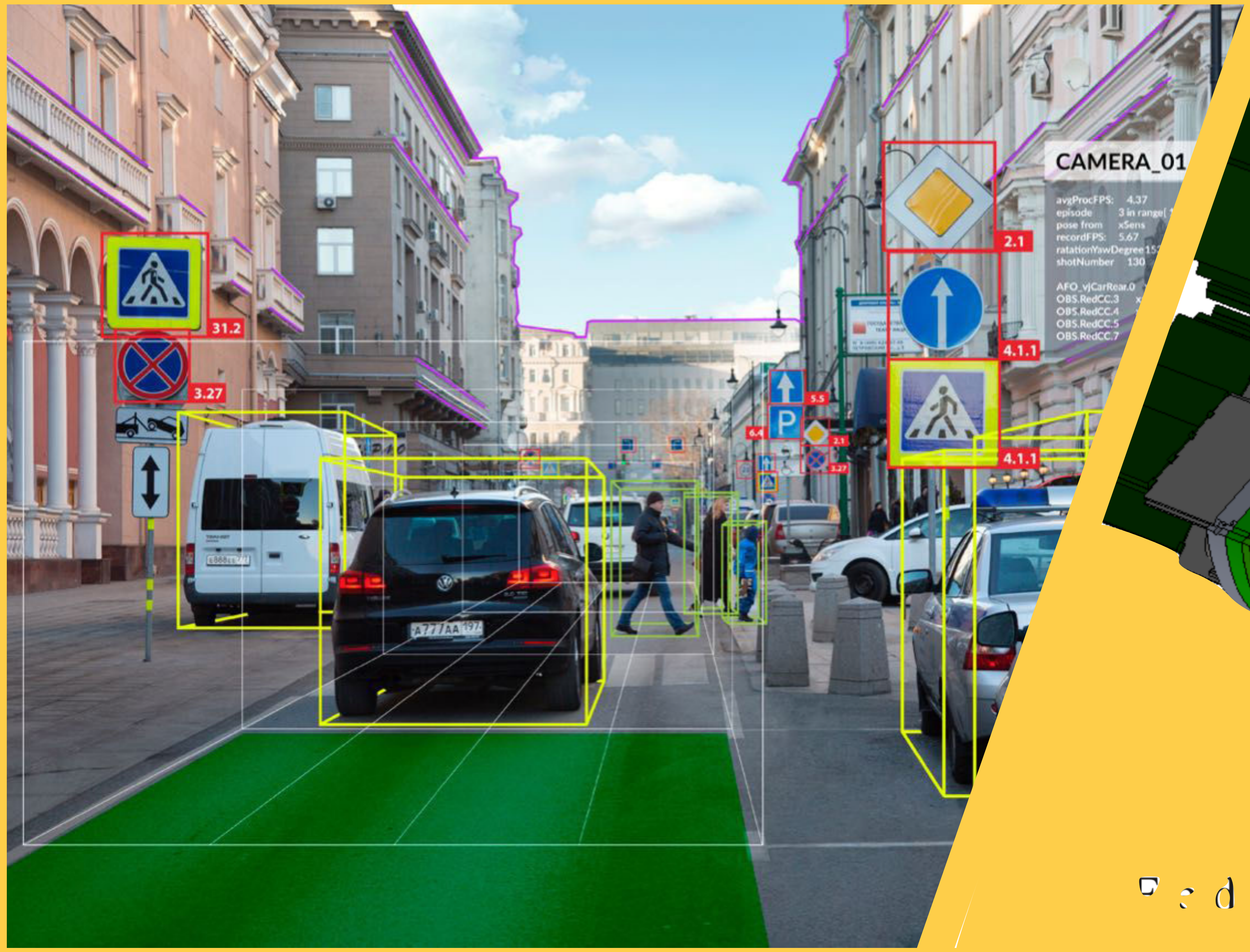


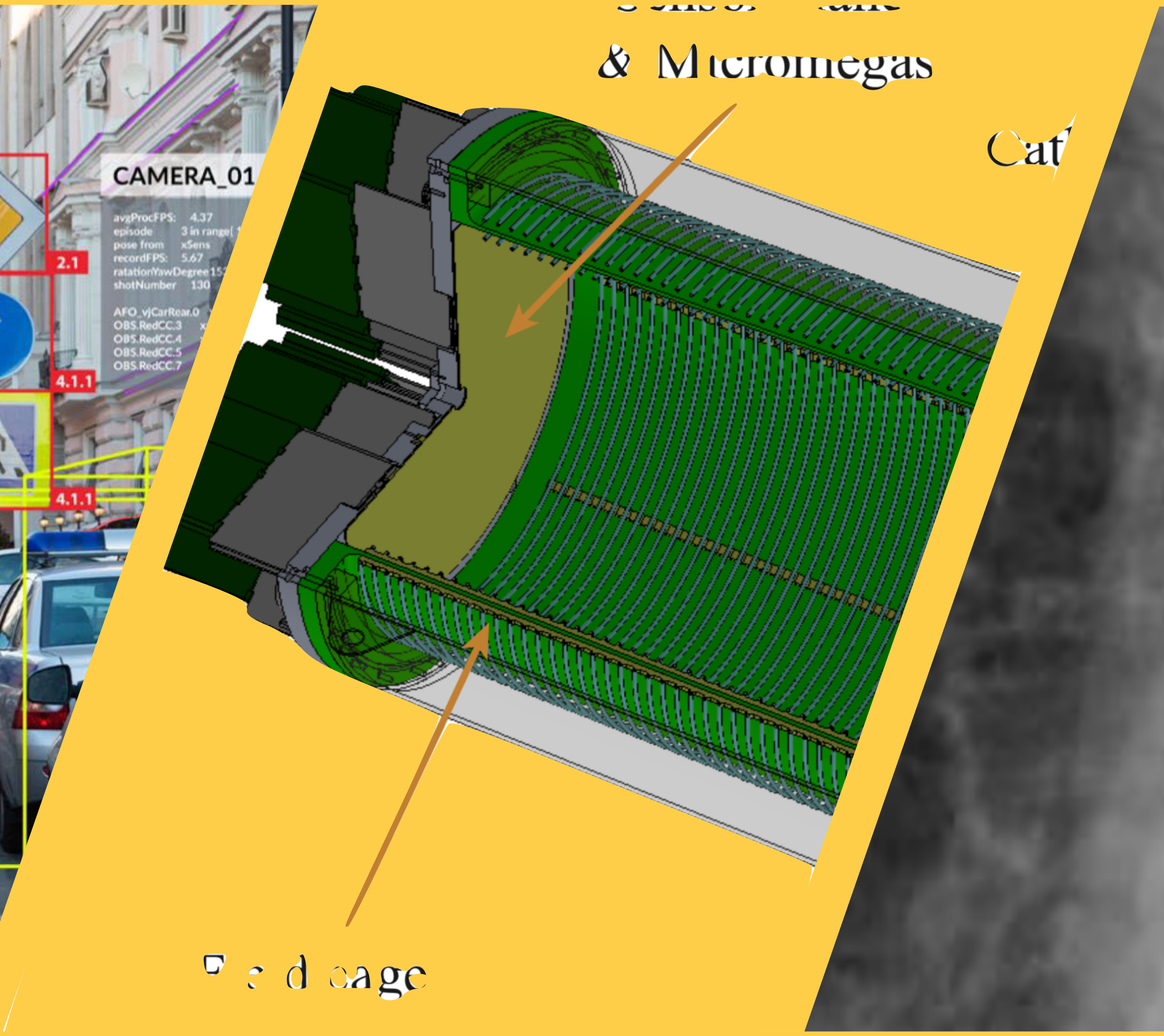


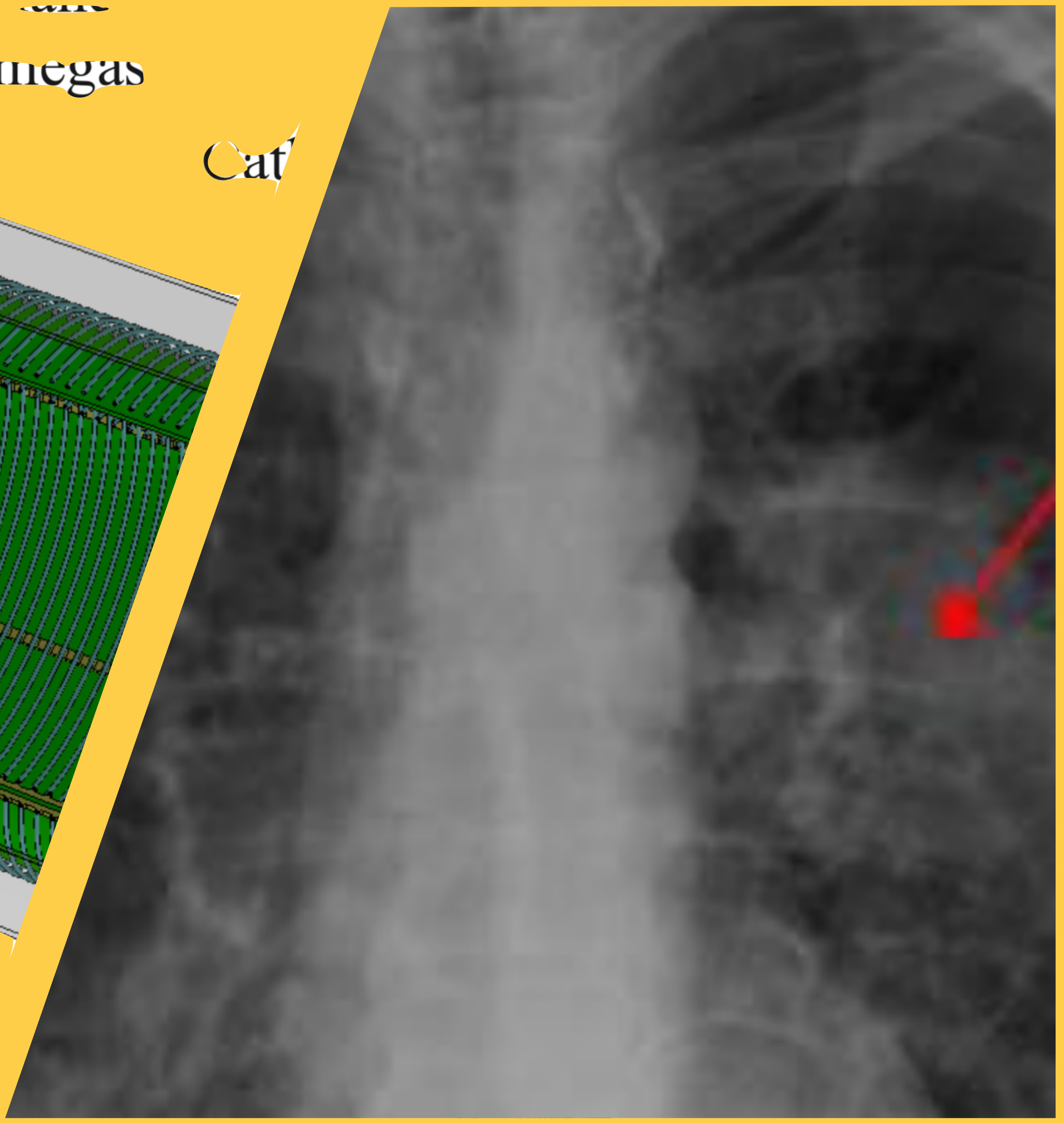
AT-TPC

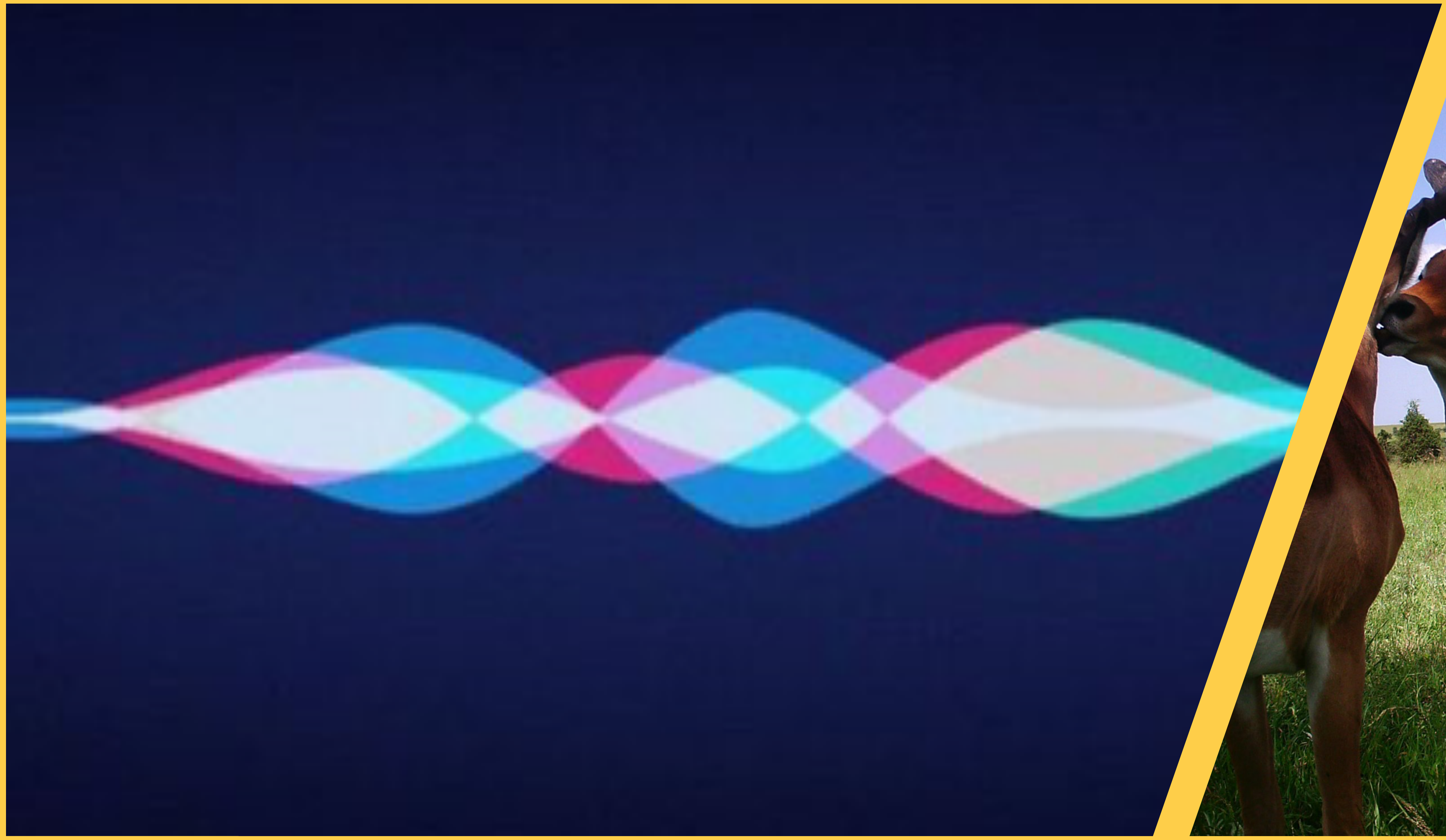
CLAS 12

CMS


CAMERA_01
31.2
3.27
2.1
4.1.1

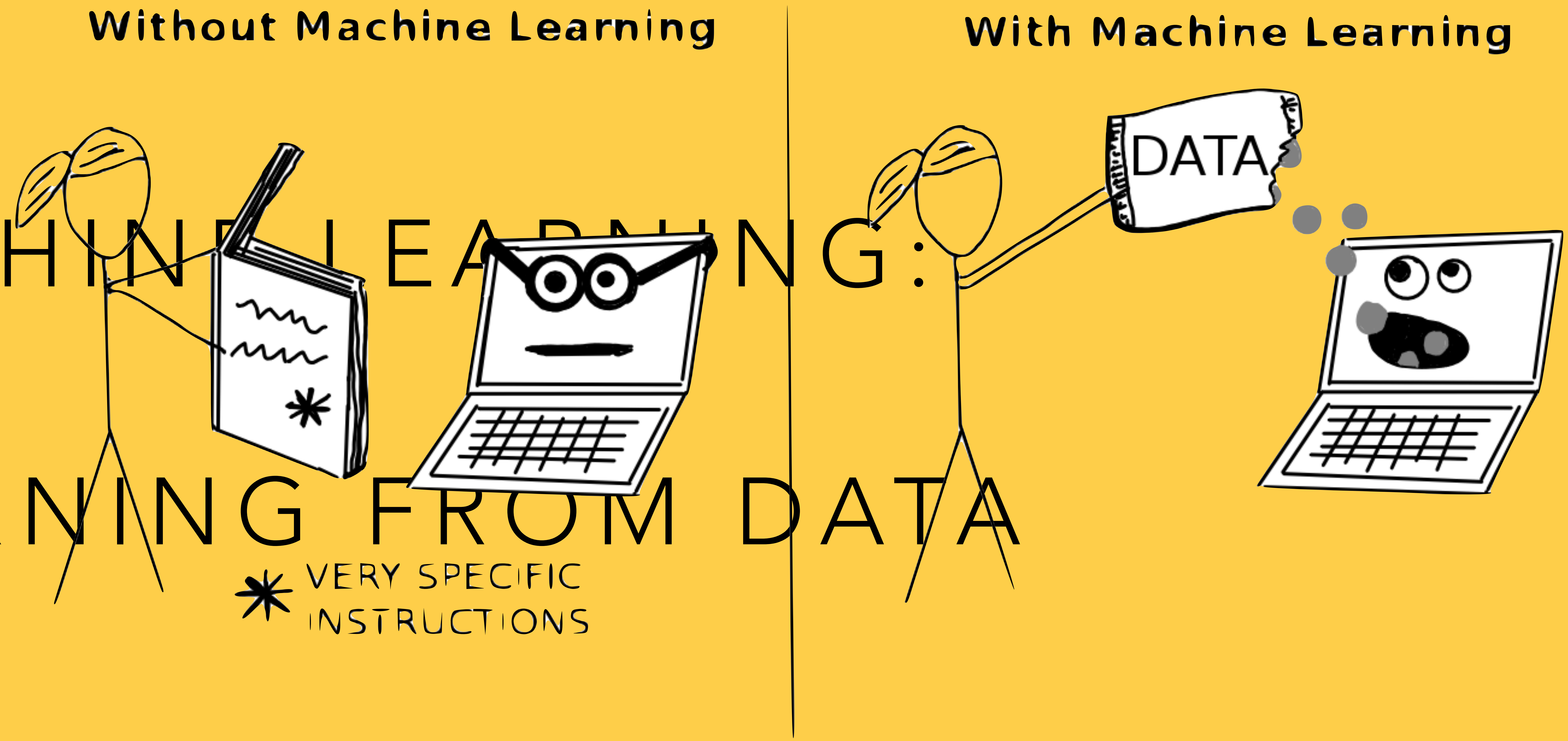




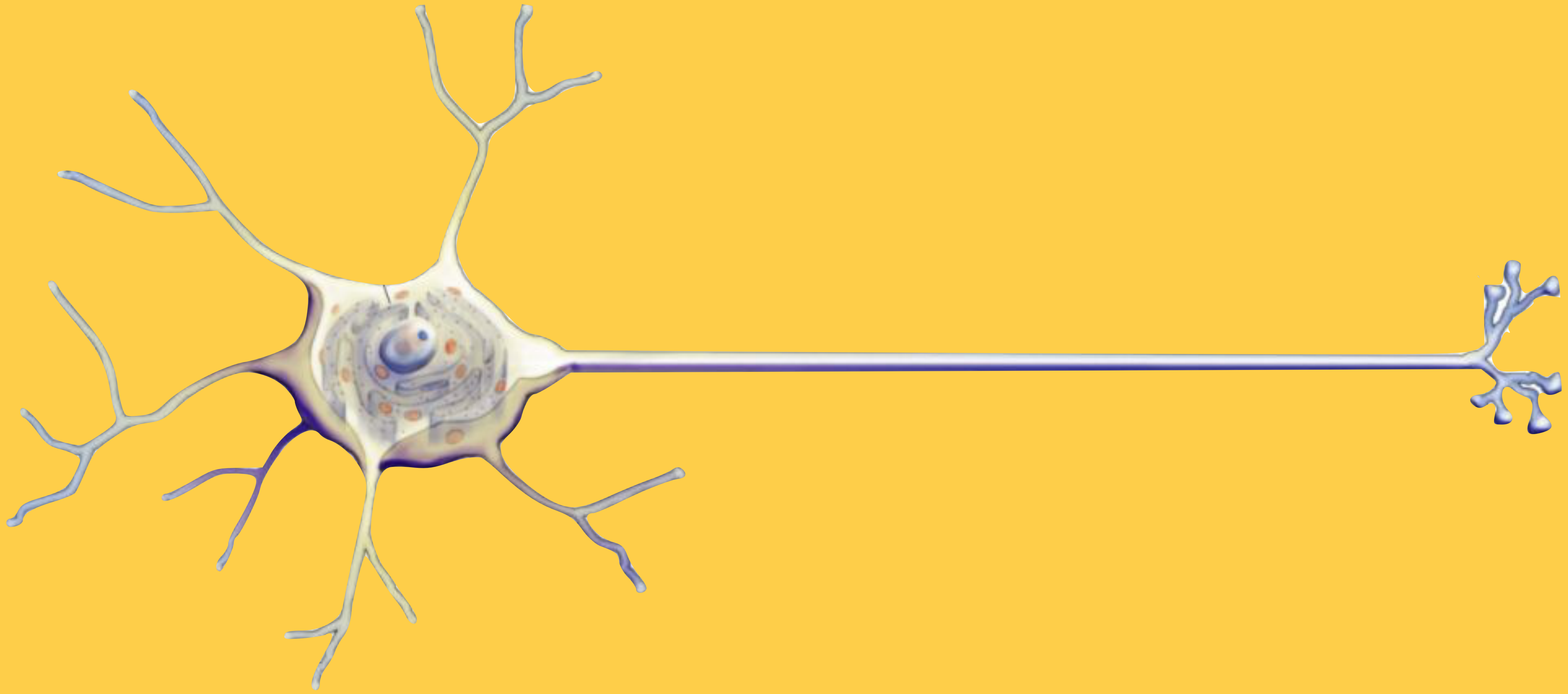


MACHINE LEARNING:

LEARNING FROM DATA



NEURON



MATHEMATICS

ELSEVIER

Neural Networks

Volume 4, Issue 2, 1991, Pages 251-257

Approximation capabilities of multilayer feedforward networks

Kurt Hornik

Show more

Share Cite

https://doi.org/10.1016/0893-6080(91)90009-T

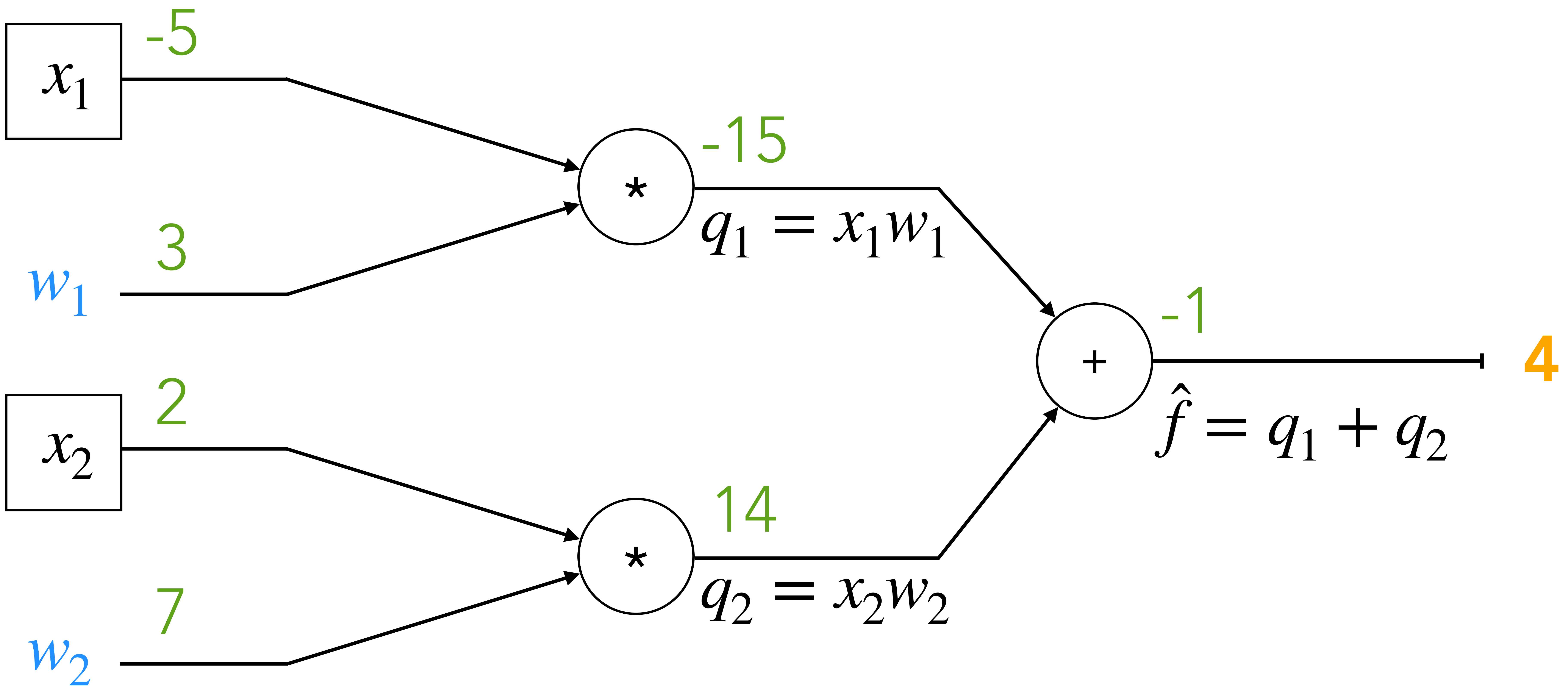
Get rights and content

Abstract

We show that standard multilayer feedforward networks with as few as a single hidden layer and arbitrary bounded and nonconstant activation function are universal approximators with respect to $L^p(\mu)$ performance criteria, for arbitrary finite input environment measures μ, provided only that sufficiently many hidden units are available. If the activation function is continuous, bounded and nonconstant, then continuous mappings can be learned uniformly over compact input sets. We also give very general conditions ensuring that networks with sufficiently smooth activation functions are capable of arbitrarily accurate approximation to a function and its derivatives.

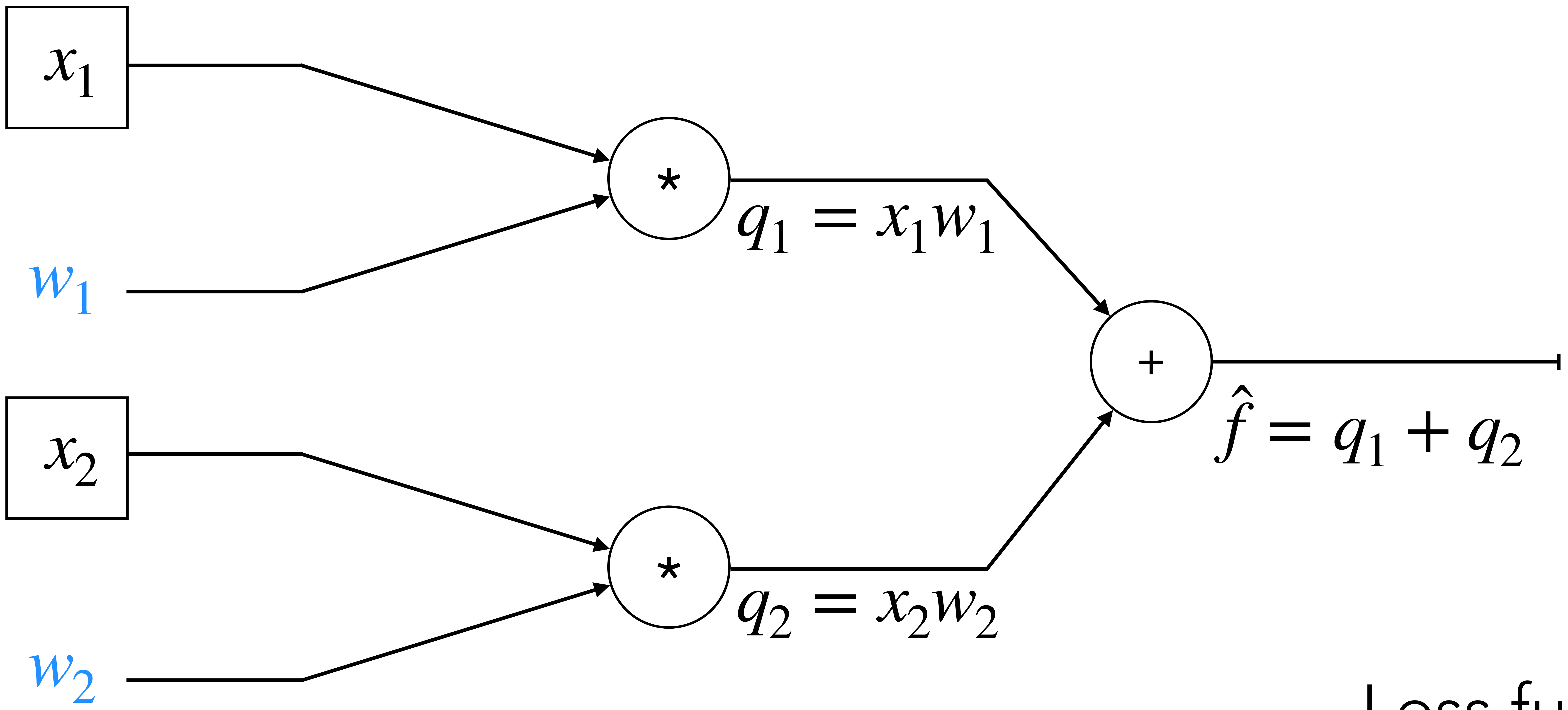
MATHEMATICS

COMPUTATIONAL GRAPH



$$\hat{f} = x_1 w_1 + x_2 w_2$$

REGRESSION

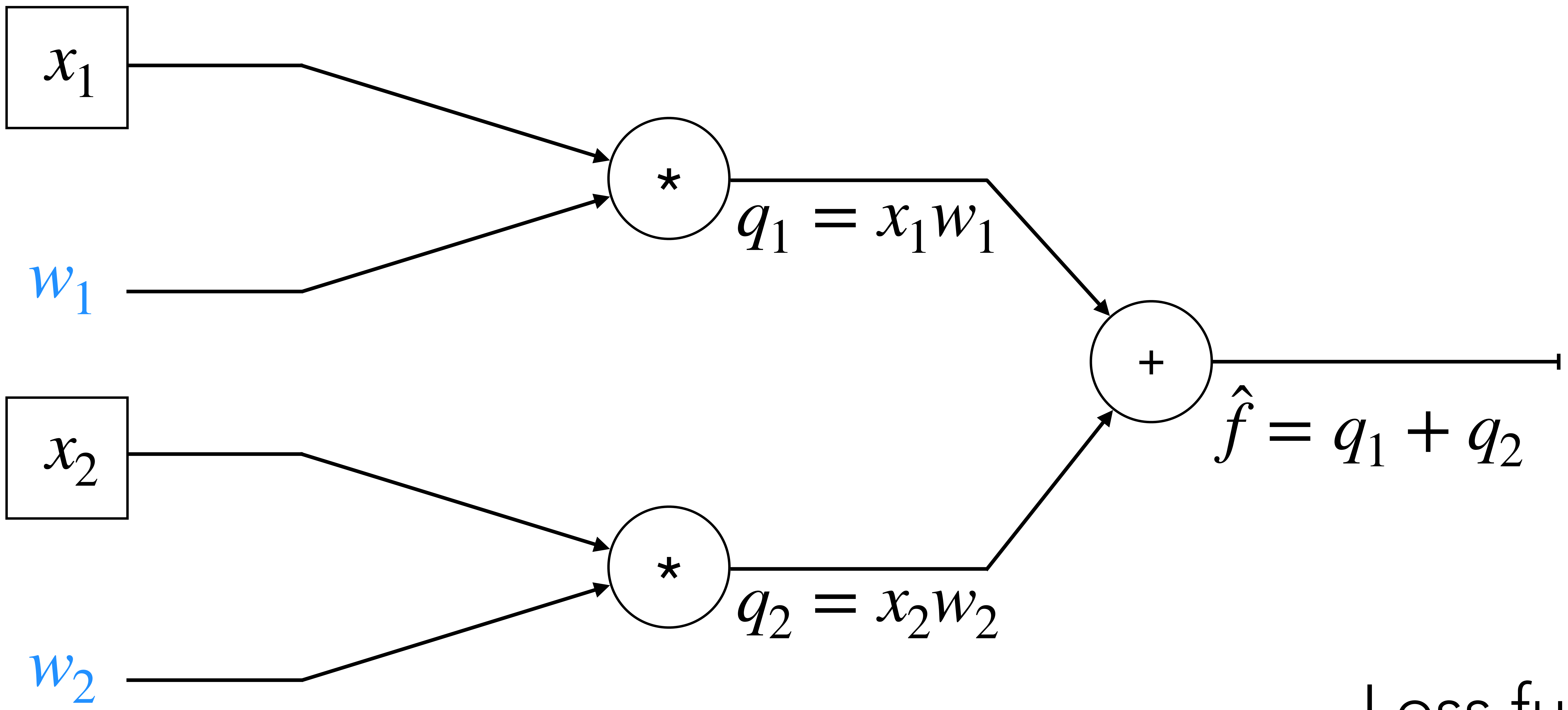


Loss function

$$\hat{f} = x_1 w_1 + x_2 w_2$$

$$J(w) = f - \hat{f}$$

SUPERVISED LEARNING



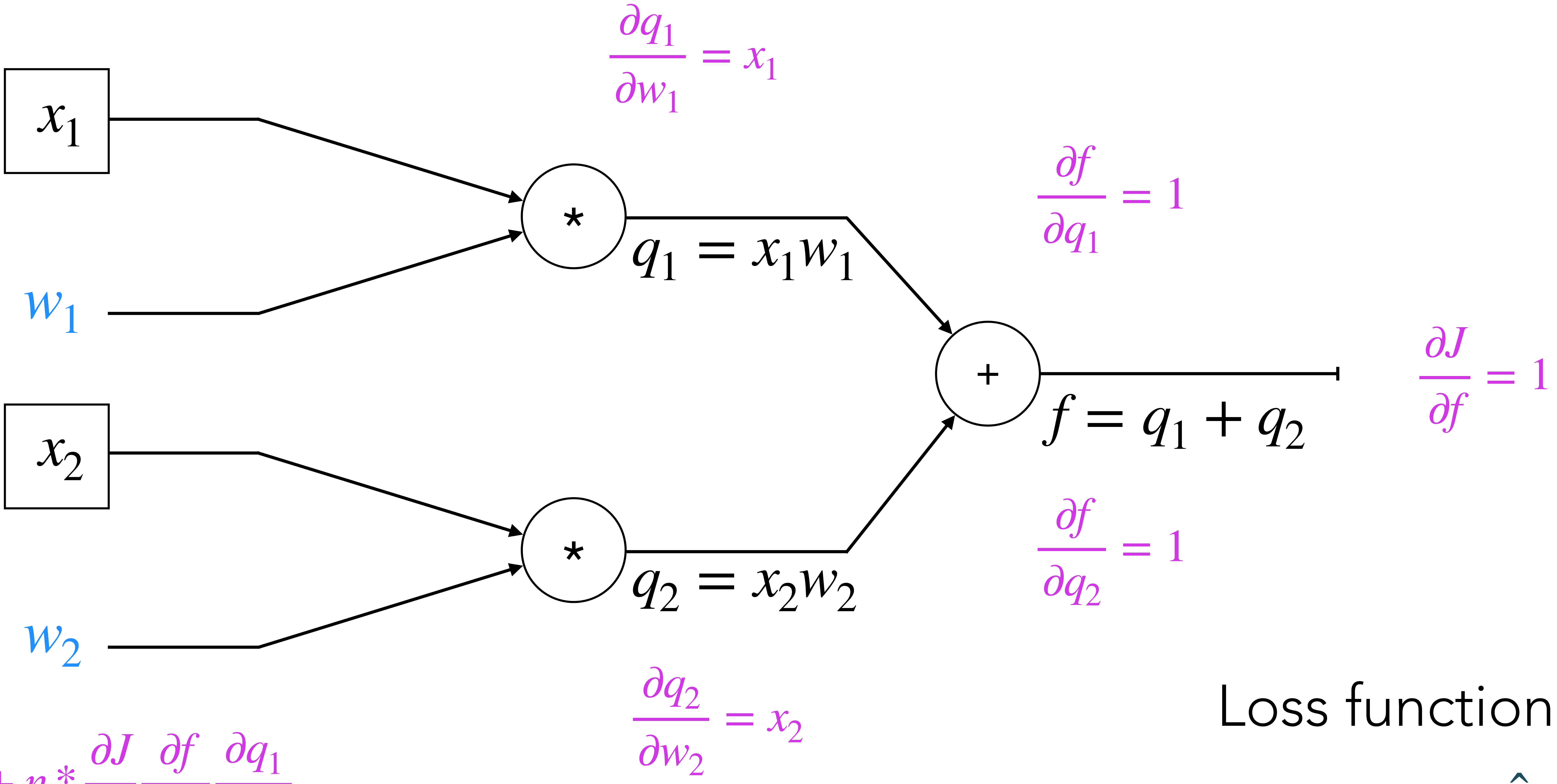
Loss function

$$\hat{f} = x_1 w_1 + x_2 w_2$$

$$J(w) = f - \hat{f}$$

BACKPROPAGATION

$$w_1 = w_1 + \eta * \frac{\partial J}{\partial f}\frac{\partial f}{\partial q_1}\frac{\partial q_1}{\partial w_1}$$

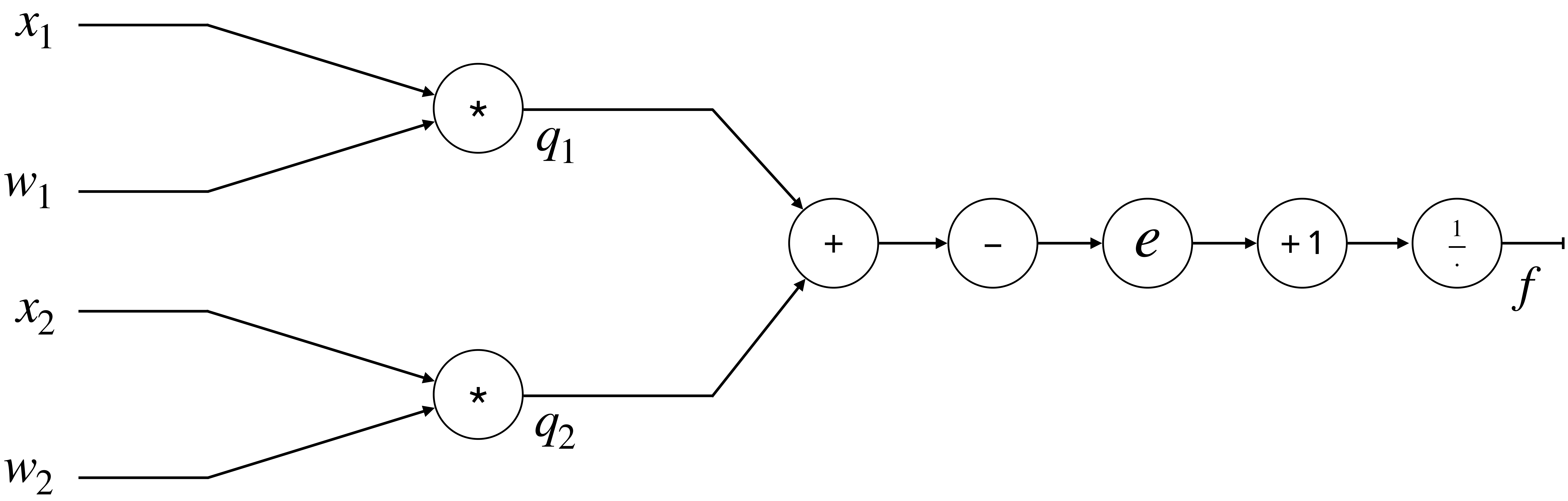


$$w_2 = w_2 + \eta * \frac{\partial J}{\partial f}\frac{\partial f}{\partial q_2}\frac{\partial q_1}{\partial w_2}$$

Loss function

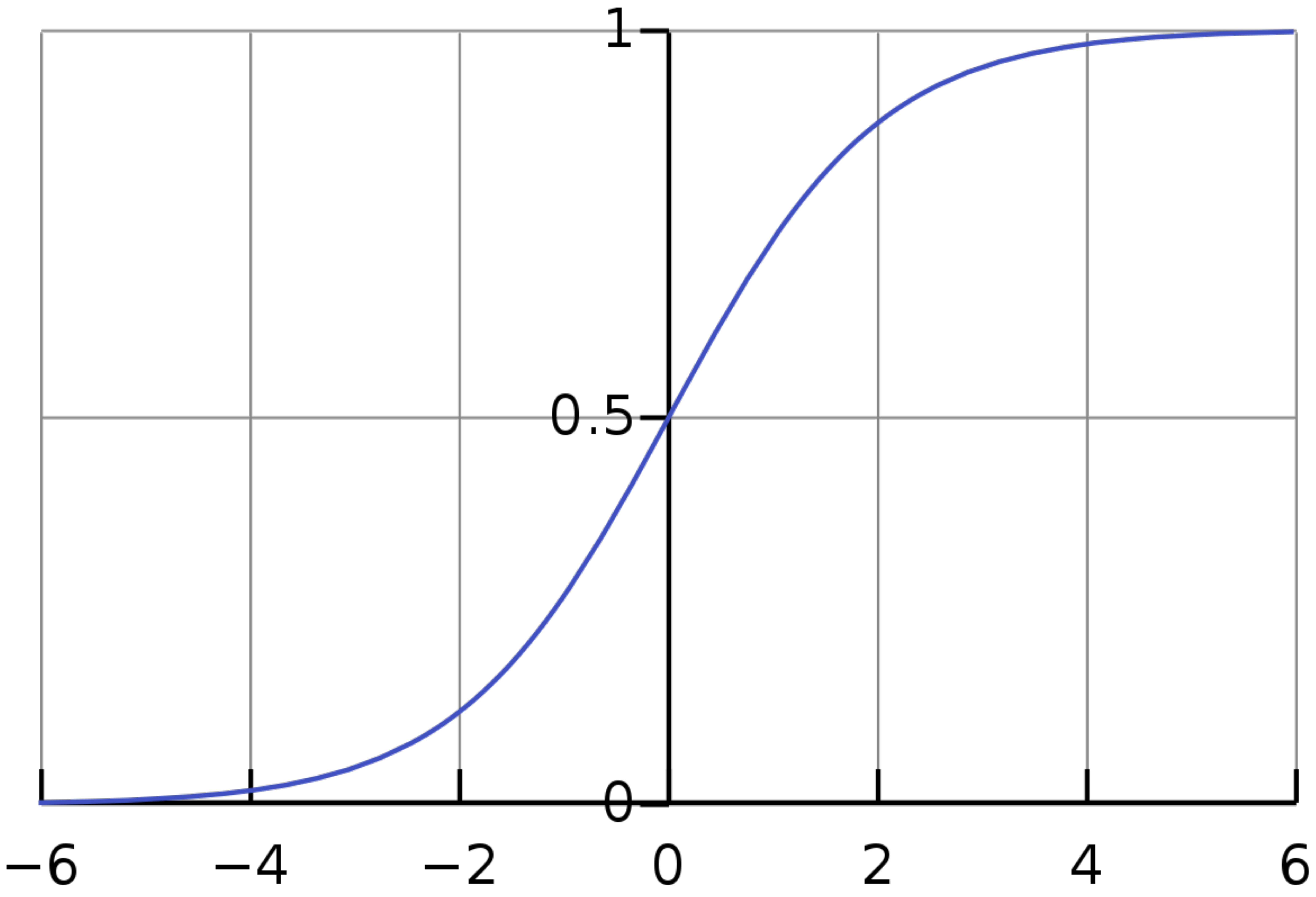
$$J(w) = f - \hat{f}$$

LOGISTIC REGRESSION

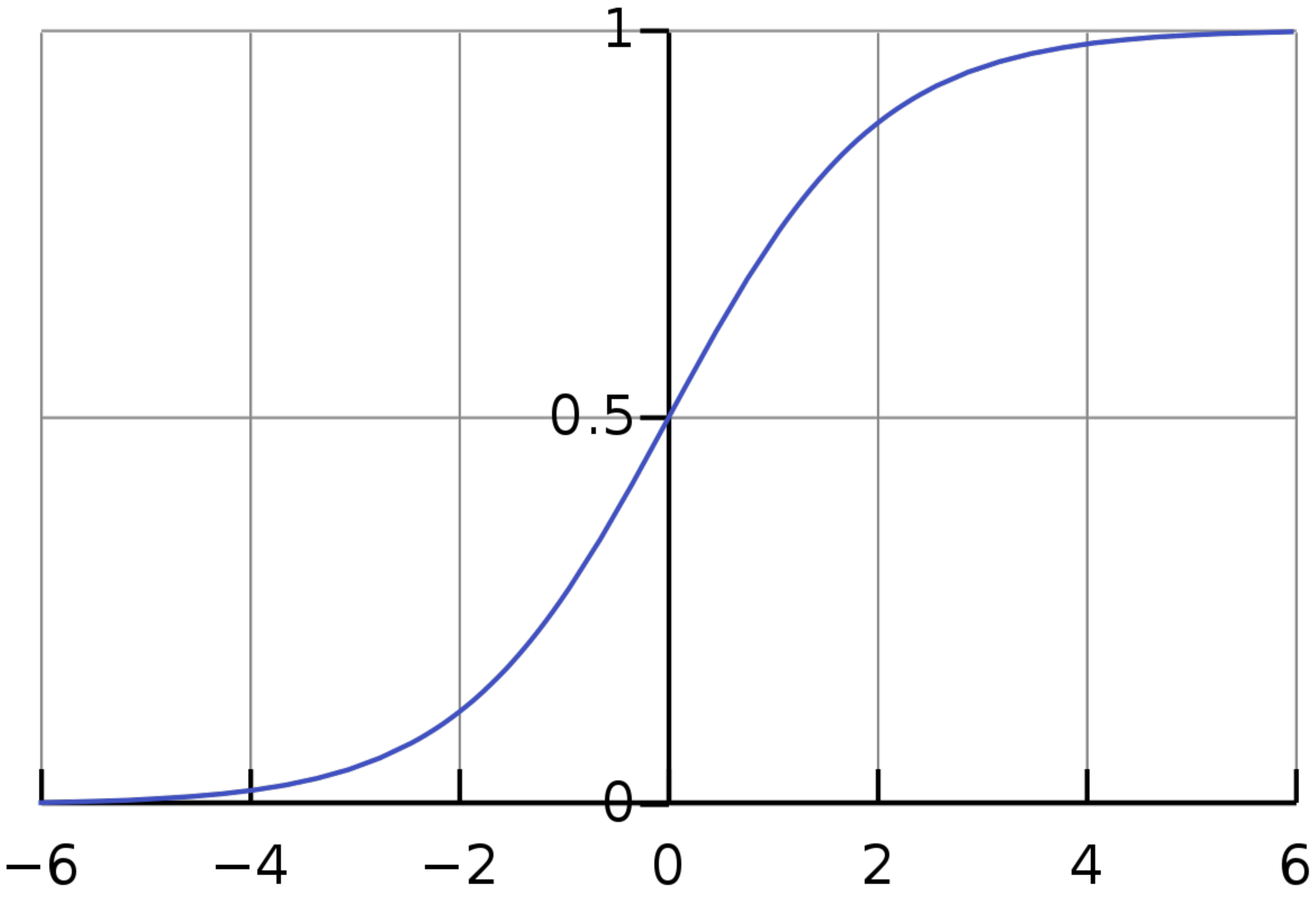


$$f = \frac{1}{1 + e^{-(x_1 w_1 + x_2 w_2)}}$$

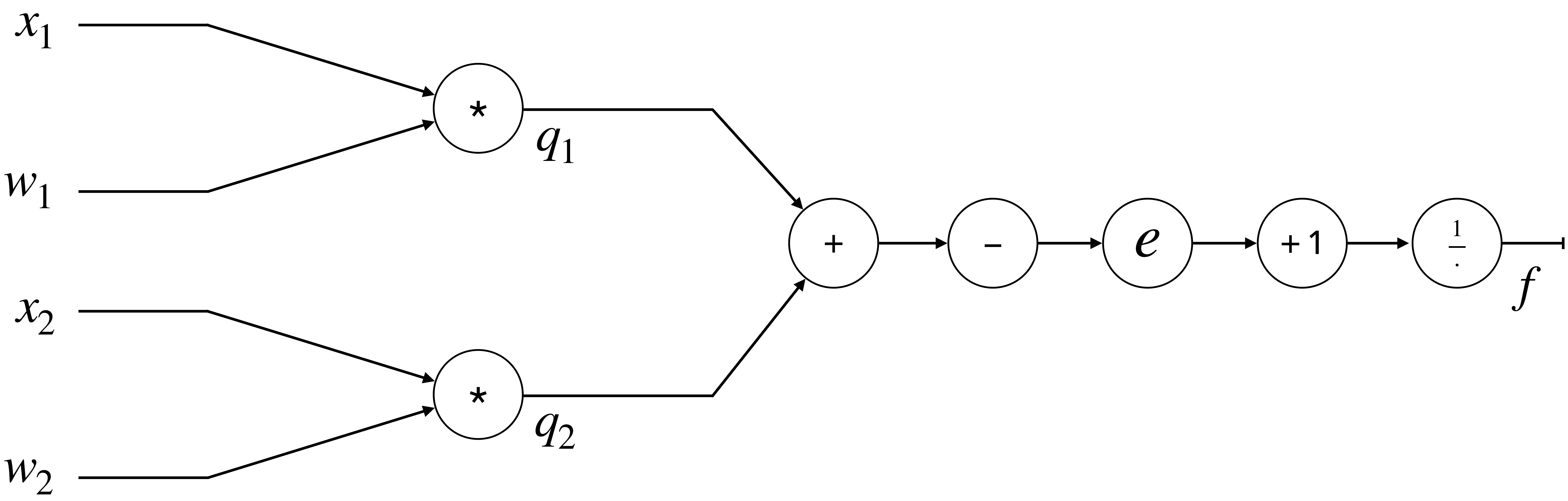
LOGISTIC REGRESSION



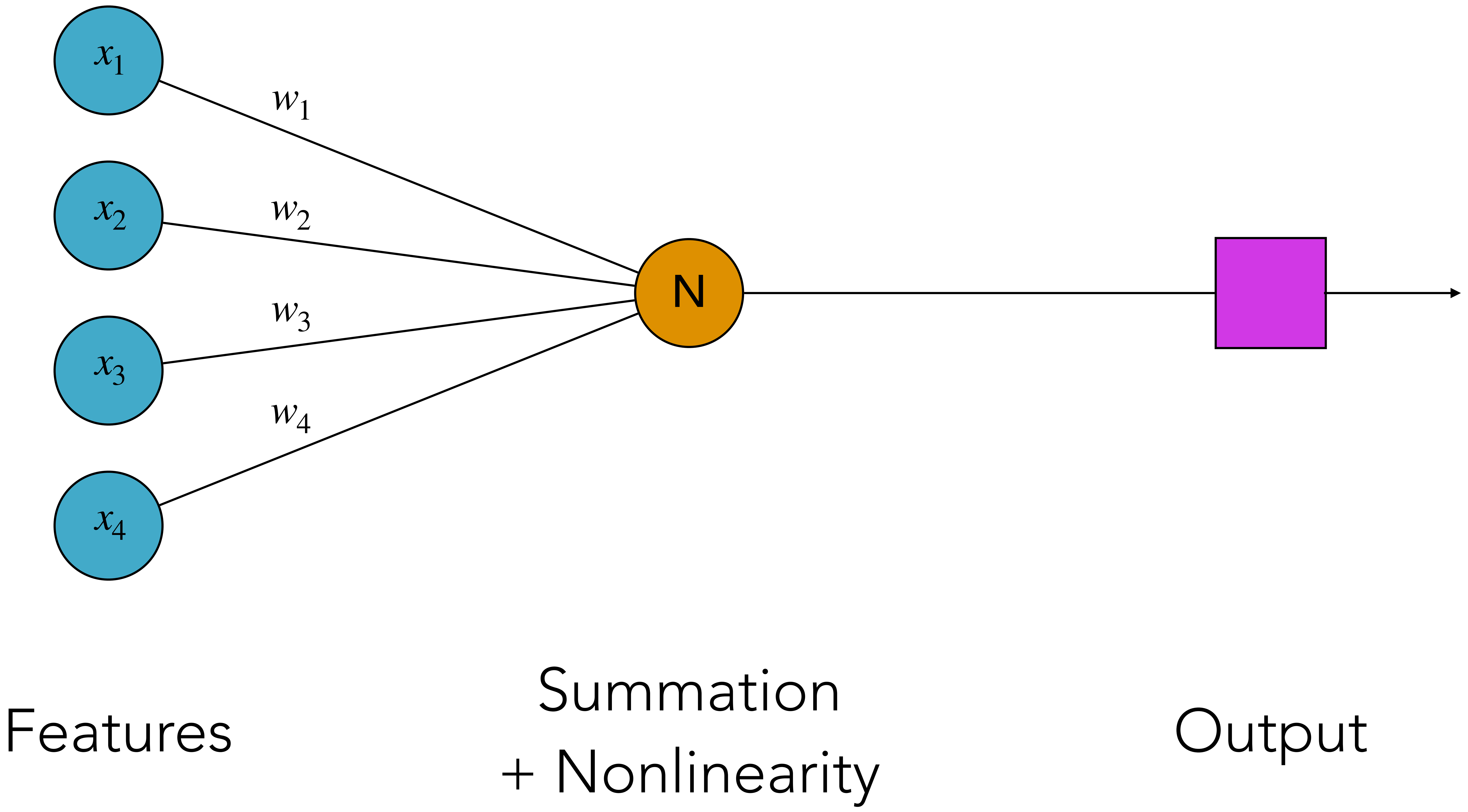
CLASSIFICATION

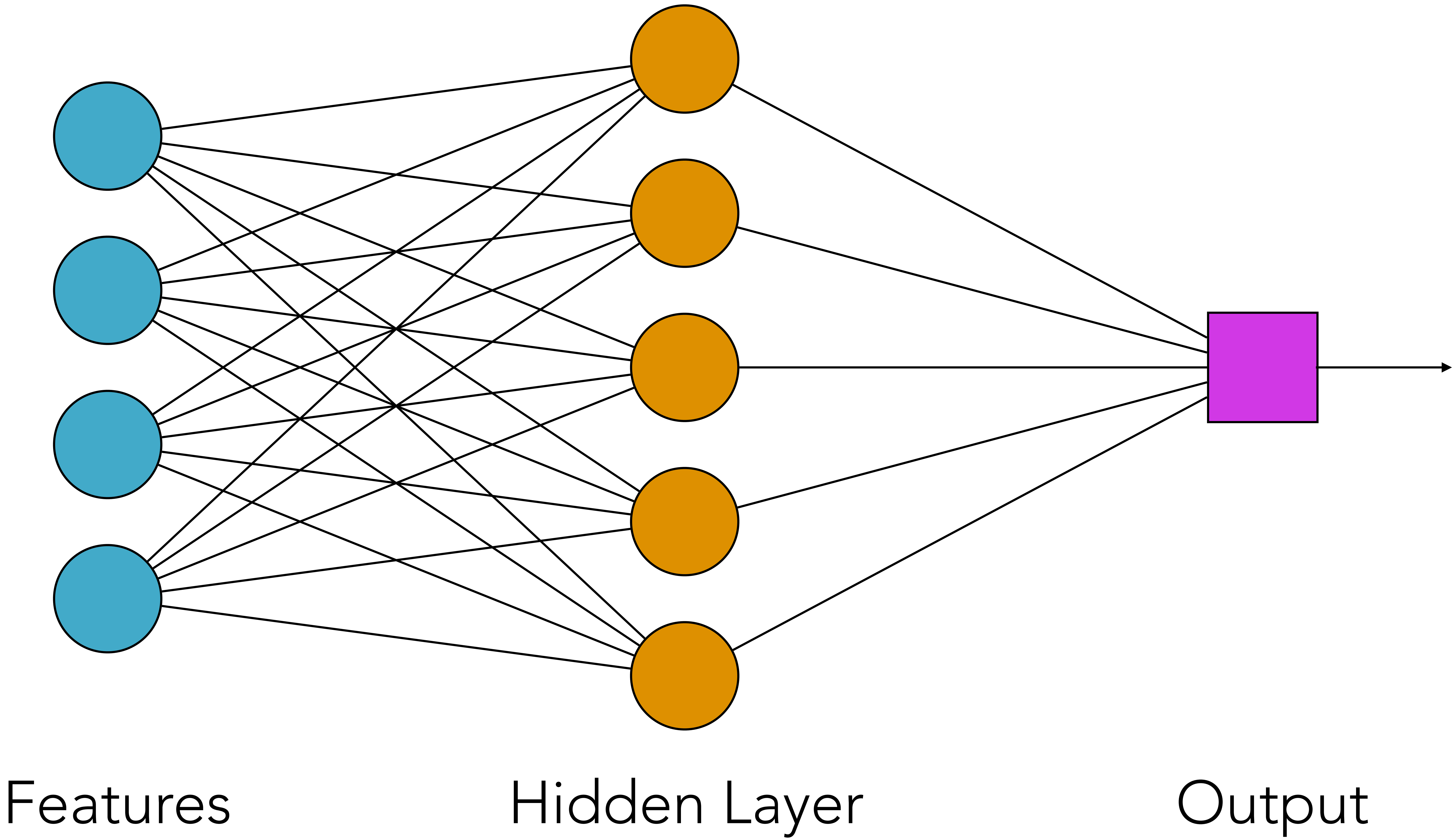


LOGISTIC REGRESSION



$$f = \frac{1}{1 + e^{-(x_1 w_1 + x_2 w_2)}}$$

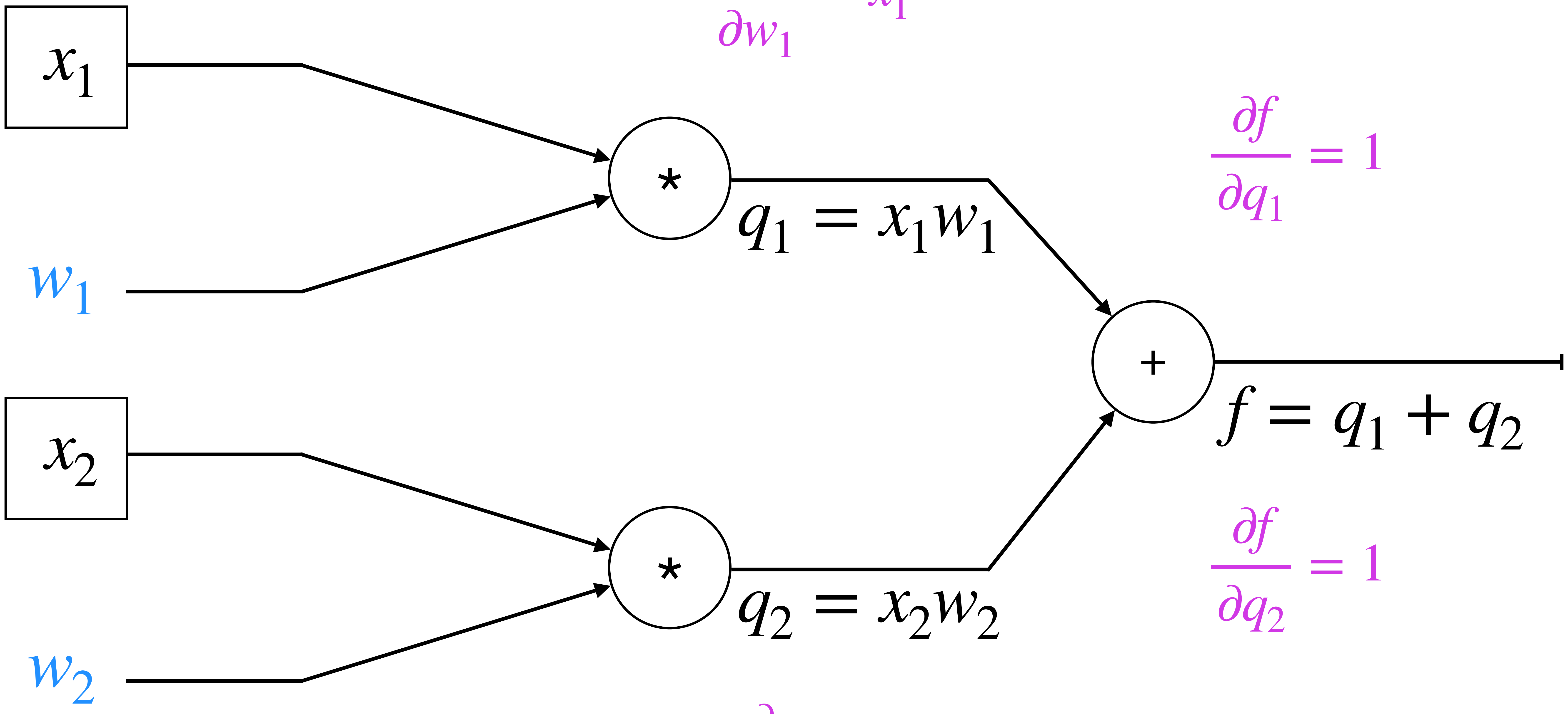

x_1
x_2
x_3
x_4
w_1
w_2
w_3
w_4
N
Features
Summation
+ Nonlinearity
Output


Features
Hidden Layer
Output

BACKPROPAGATION

$$w_1 = w_1 + \eta * \frac{\partial J}{\partial f}\frac{\partial f}{\partial q_1}\frac{\partial q_1}{\partial w_1}$$

$$\frac{\partial q_1}{\partial w_1} = x_1$$



$$\frac{\partial f}{\partial q_1} = 1$$

$$\frac{\partial J}{\partial f} = 1$$

$$\frac{\partial f}{\partial q_2} = 1$$

$$\frac{\partial q_2}{\partial w_2} = x_2$$

$$w_2 = w_2 + \eta * \frac{\partial J}{\partial f}\frac{\partial f}{\partial q_2}\frac{\partial q_1}{\partial w_2}$$

Loss function

$$J(w) = f - \hat{f}$$

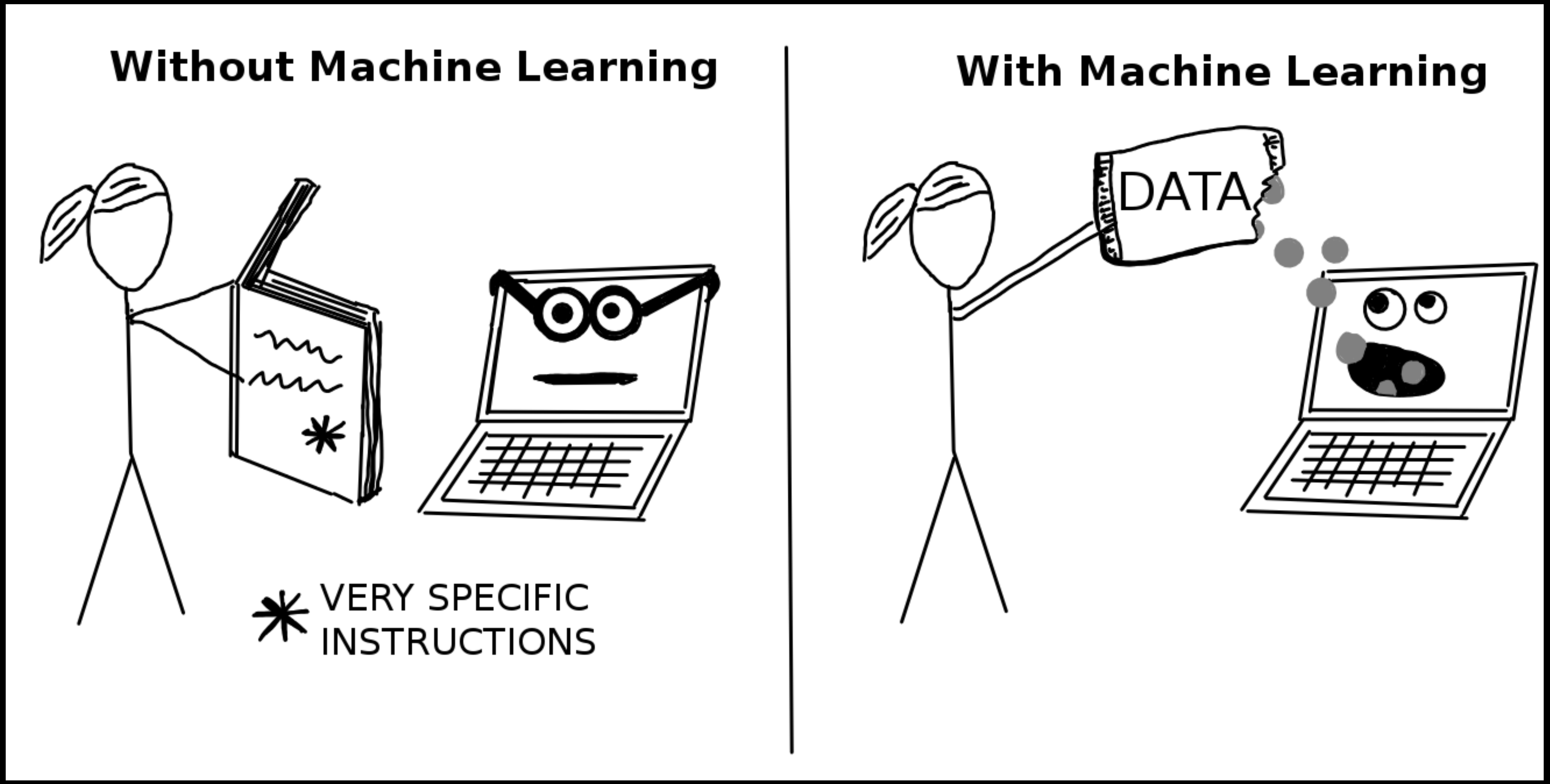
AUTOMATIC DIFFERENTIATION

TensorFlow

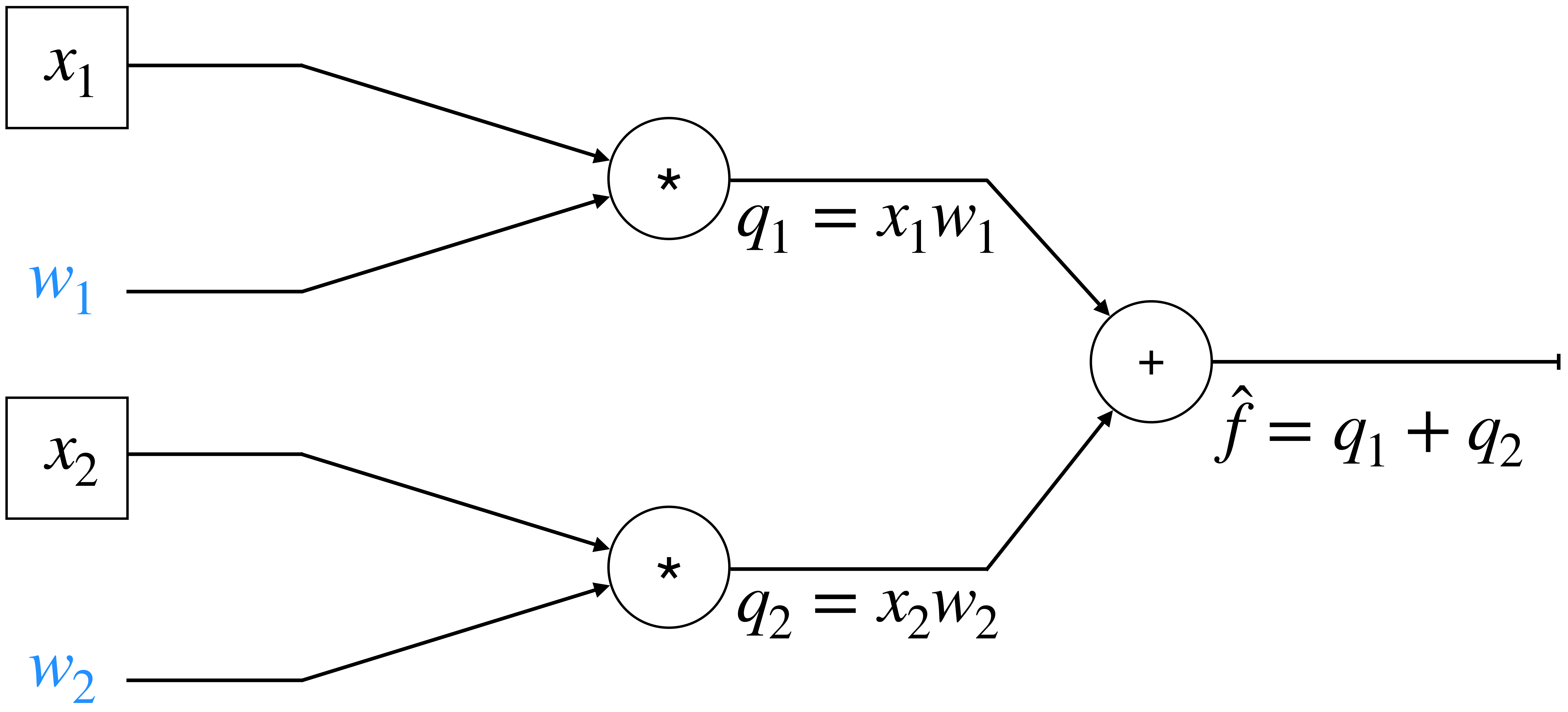
PyTorch

MACHINE LEARNING:

LEARNING FROM DATA



COMPUTATIONAL GRAPH



$$\hat{f} = x_1 w_1 + x_2 w_2$$

MACHINE LEARNING

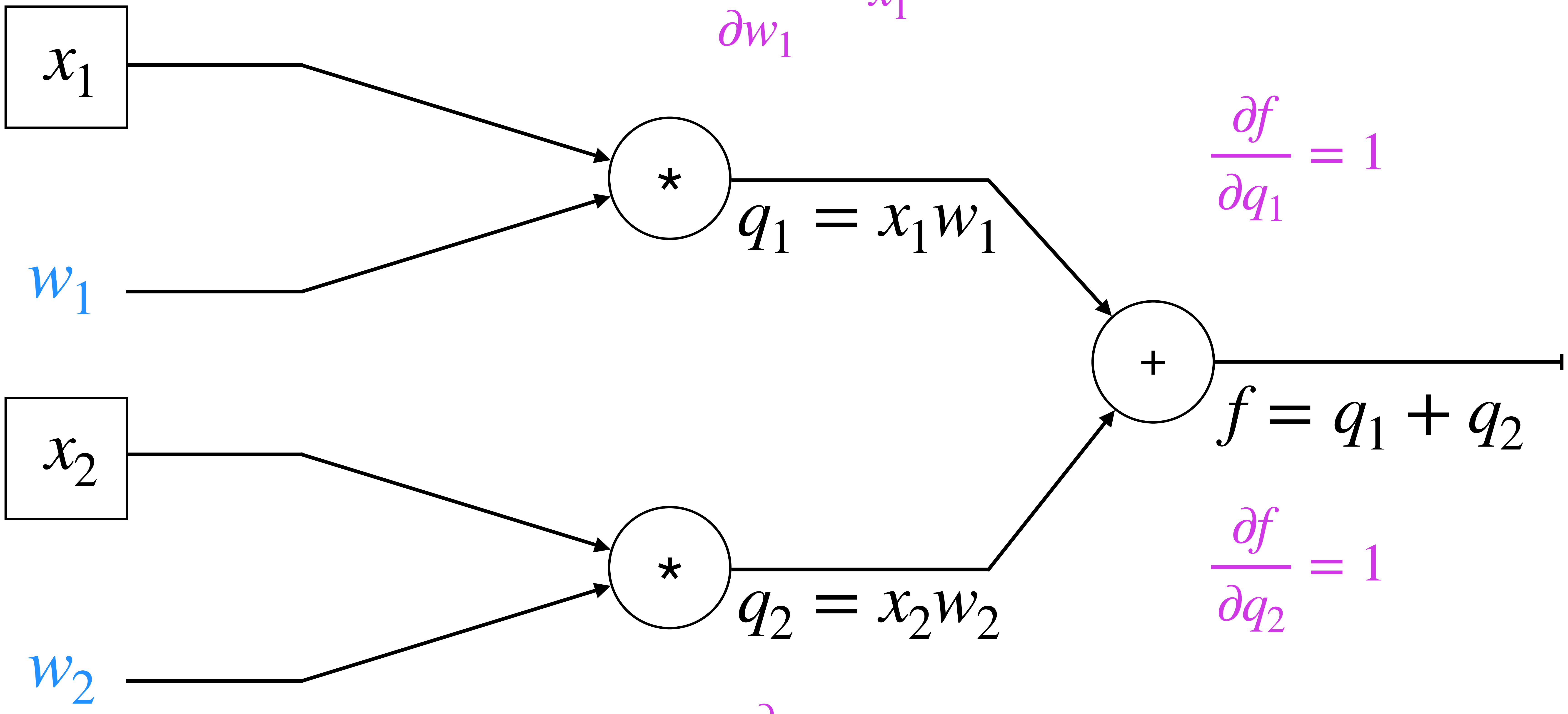
MACHINE LEARNING

SUPERVISED LEARNING

BACKPROPAGATION

$$w_1 = w_1 + \eta * \frac{\partial f}{\partial q_1}\frac{\partial q_1}{\partial w_1}$$

$$\frac{\partial q_1}{\partial w_1} = x_1$$



$$\frac{\partial f}{\partial q_1} = 1$$

$$\frac{\partial J}{\partial f} = 1$$

$$\frac{\partial f}{\partial q_2} = 1$$

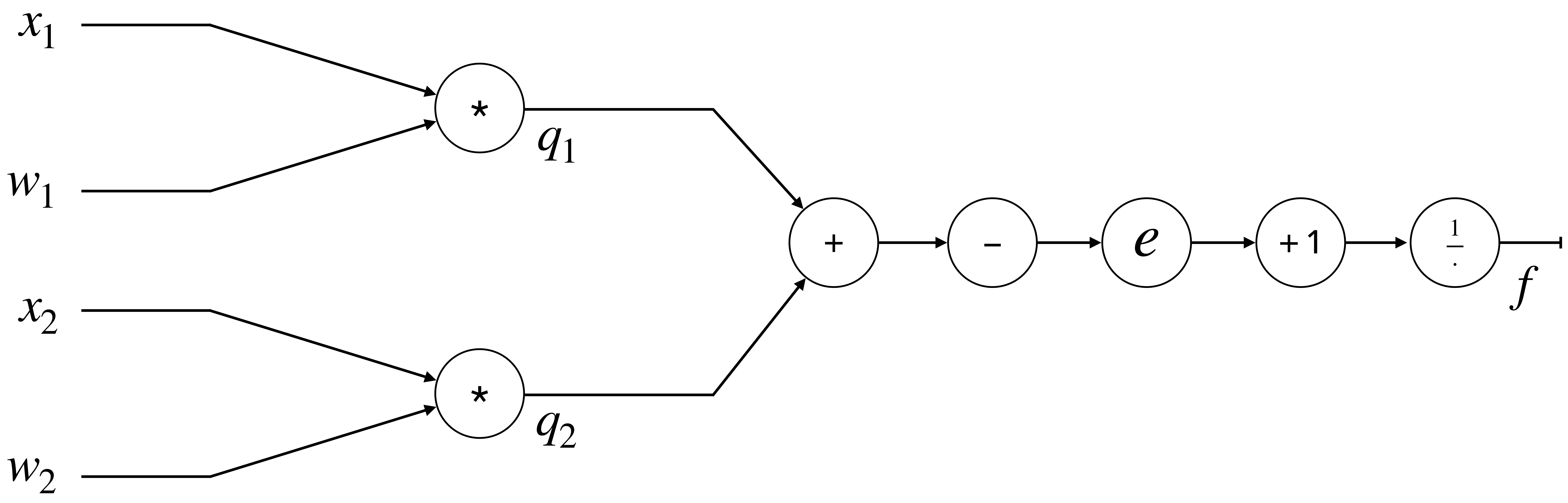
$$\frac{\partial q_2}{\partial w_2} = x_2$$

$$w_2 = w_2 + \eta * \frac{\partial f}{\partial q_1}\frac{\partial q_1}{\partial w_2}$$

Loss function

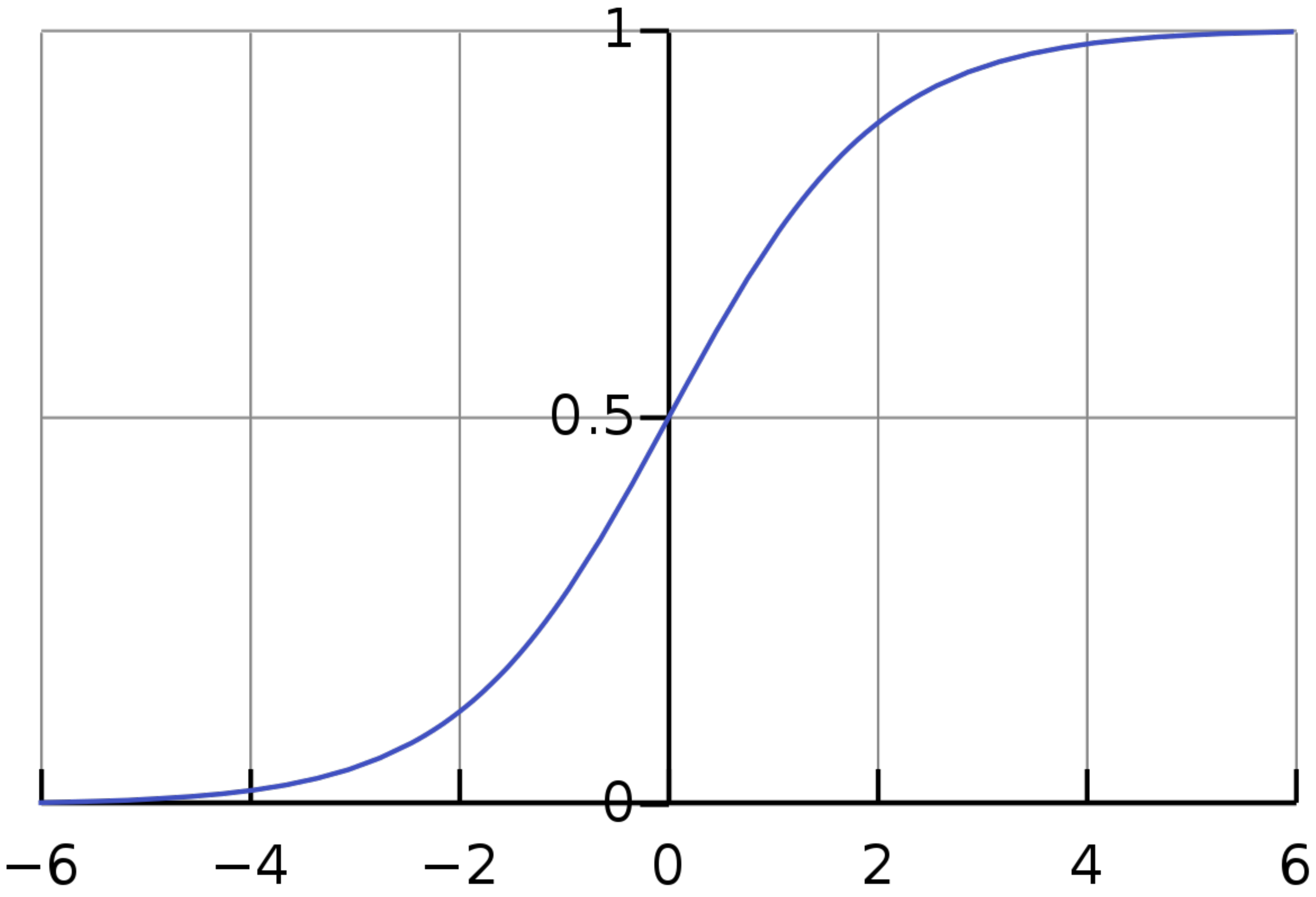
$$J(w) = f - \hat{f}$$

LOGISTIC REGRESSION

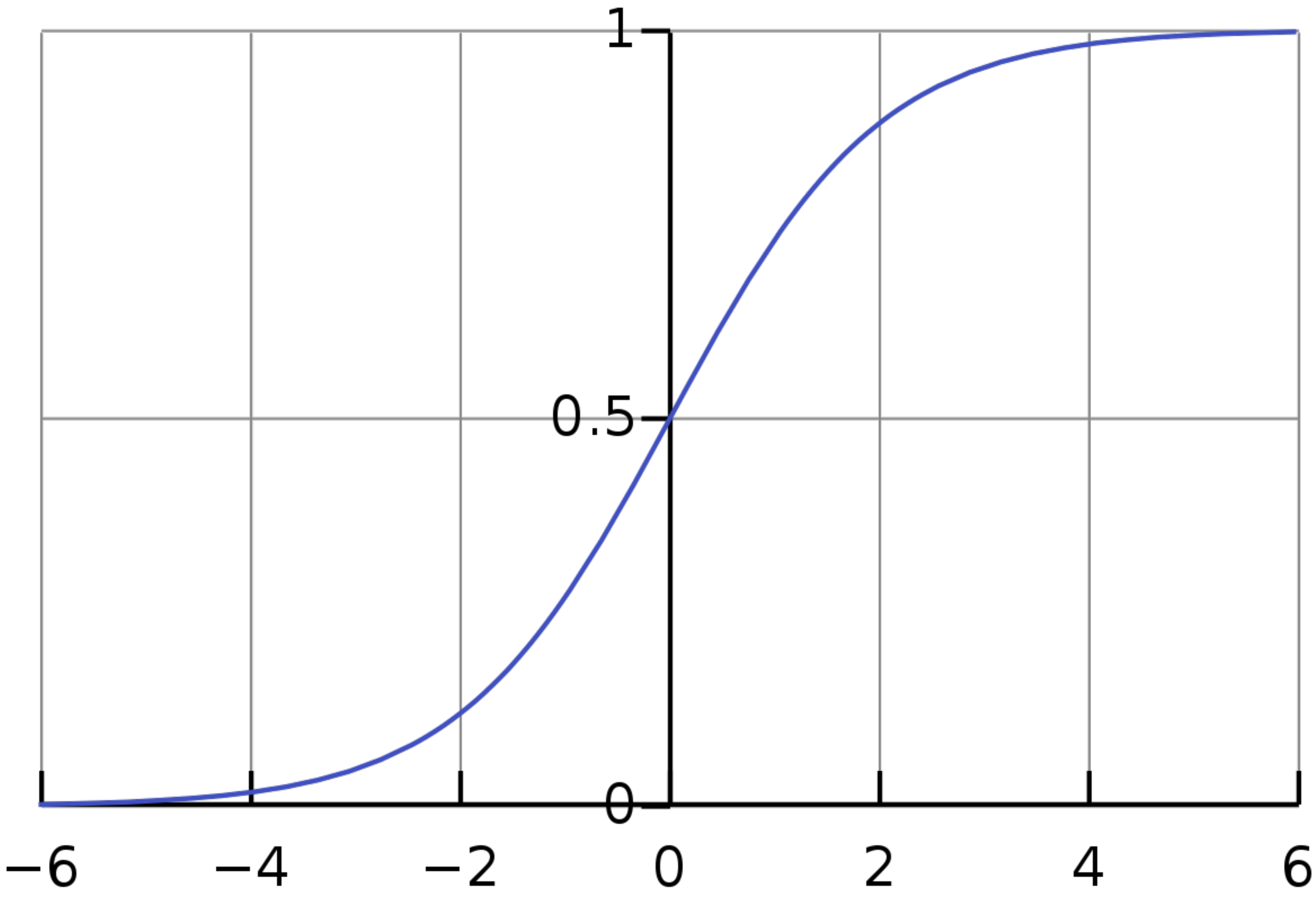


$$f = \frac{1}{1 + e^{-(x_1 w_1 + x_2 w_2)}}$$

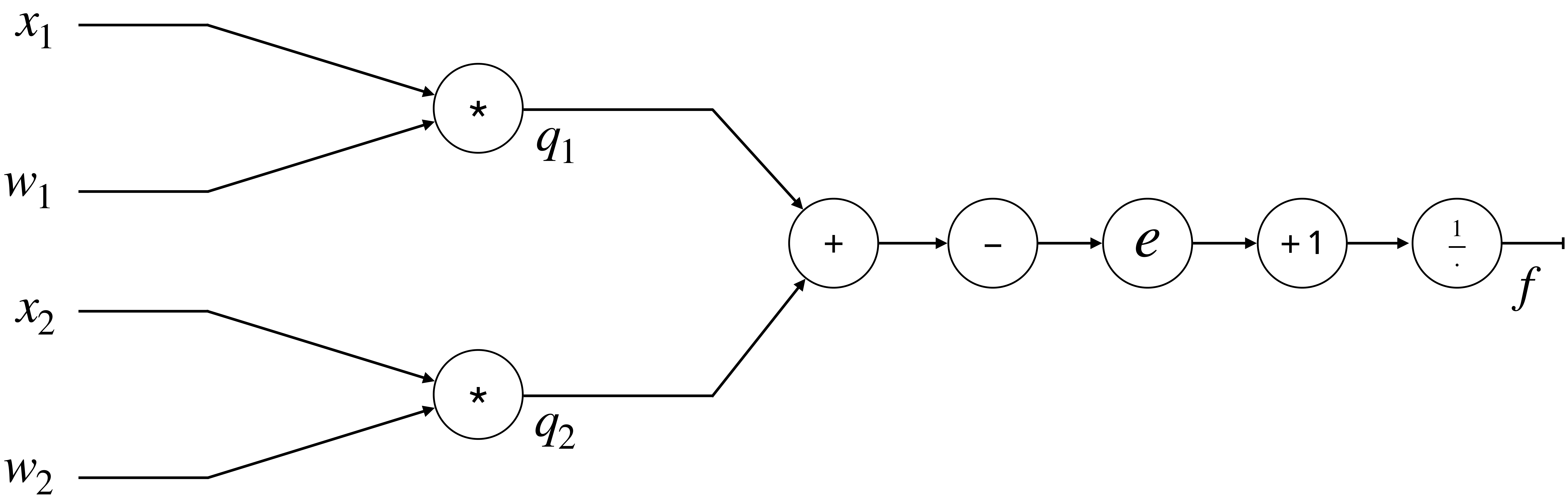
LOGISTIC REGRESSION



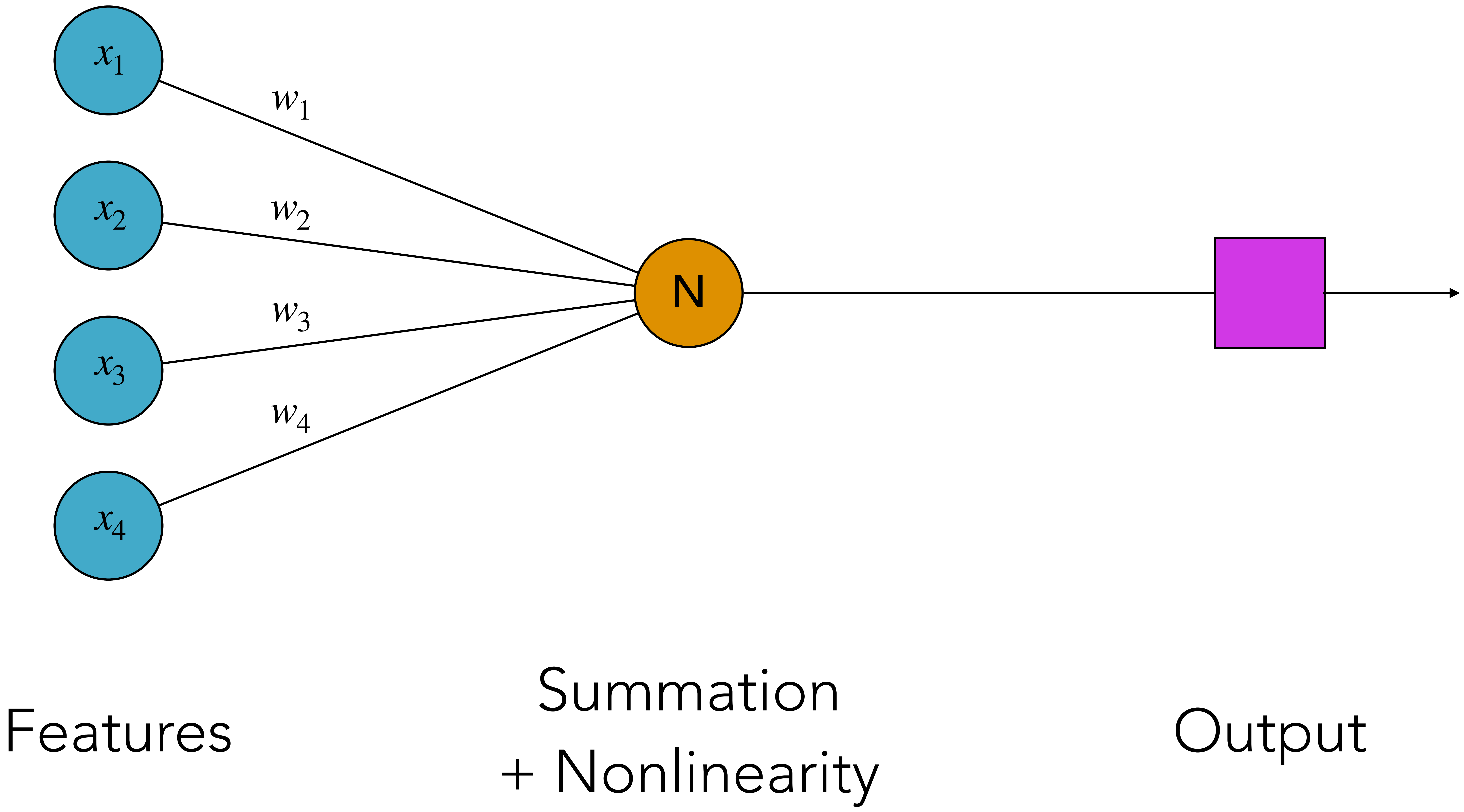
CLASSIFICATION



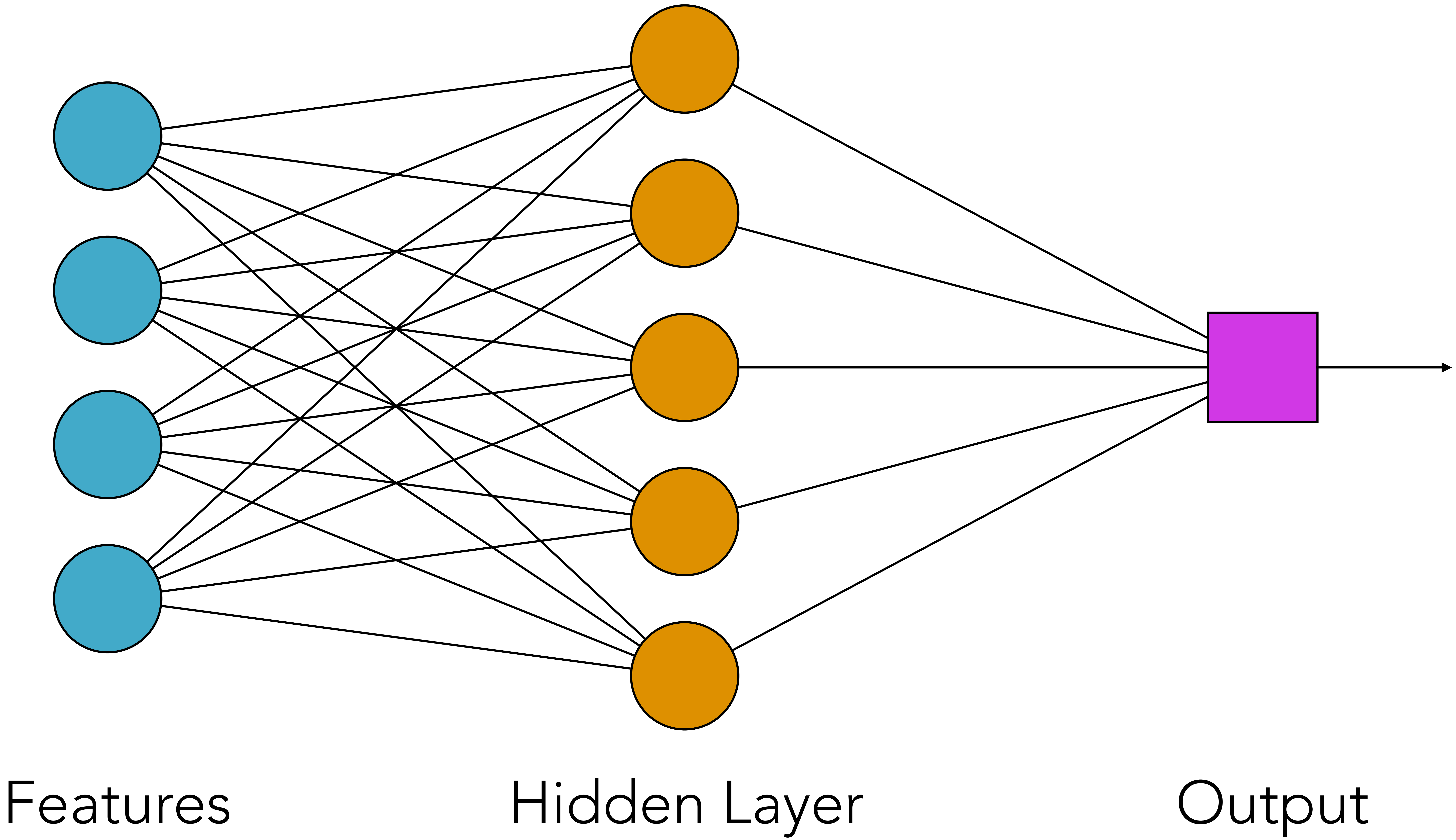
LOGISTIC REGRESSION



$$f = \frac{1}{1 + e^{-(x_1 w_1 + x_2 w_2)}}$$

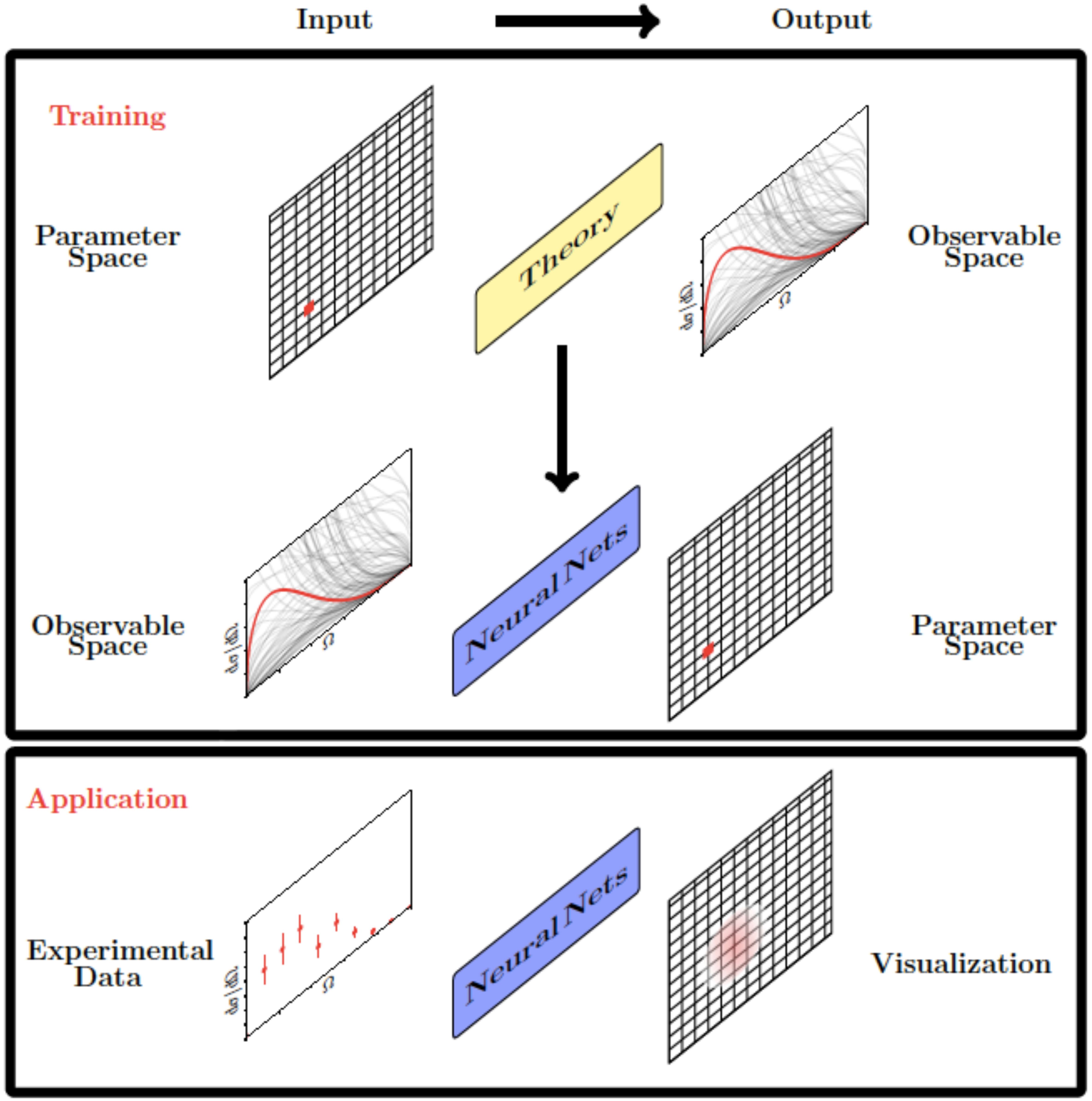


x_1
x_2
x_3
x_4
w_1
w_2
w_3
w_4
N
Features
Summation
+ Nonlinearity
Output


Features
Hidden Layer
Output

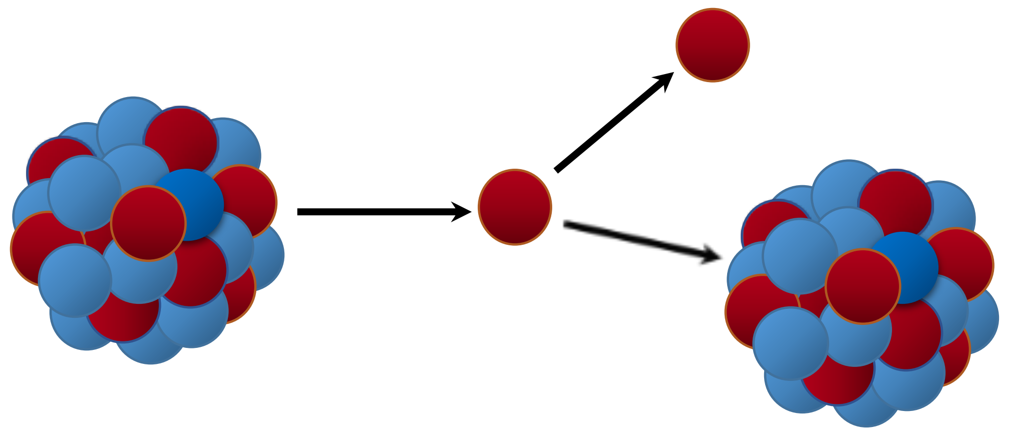
Application 1: How can experimental observables constrain theoretical models?

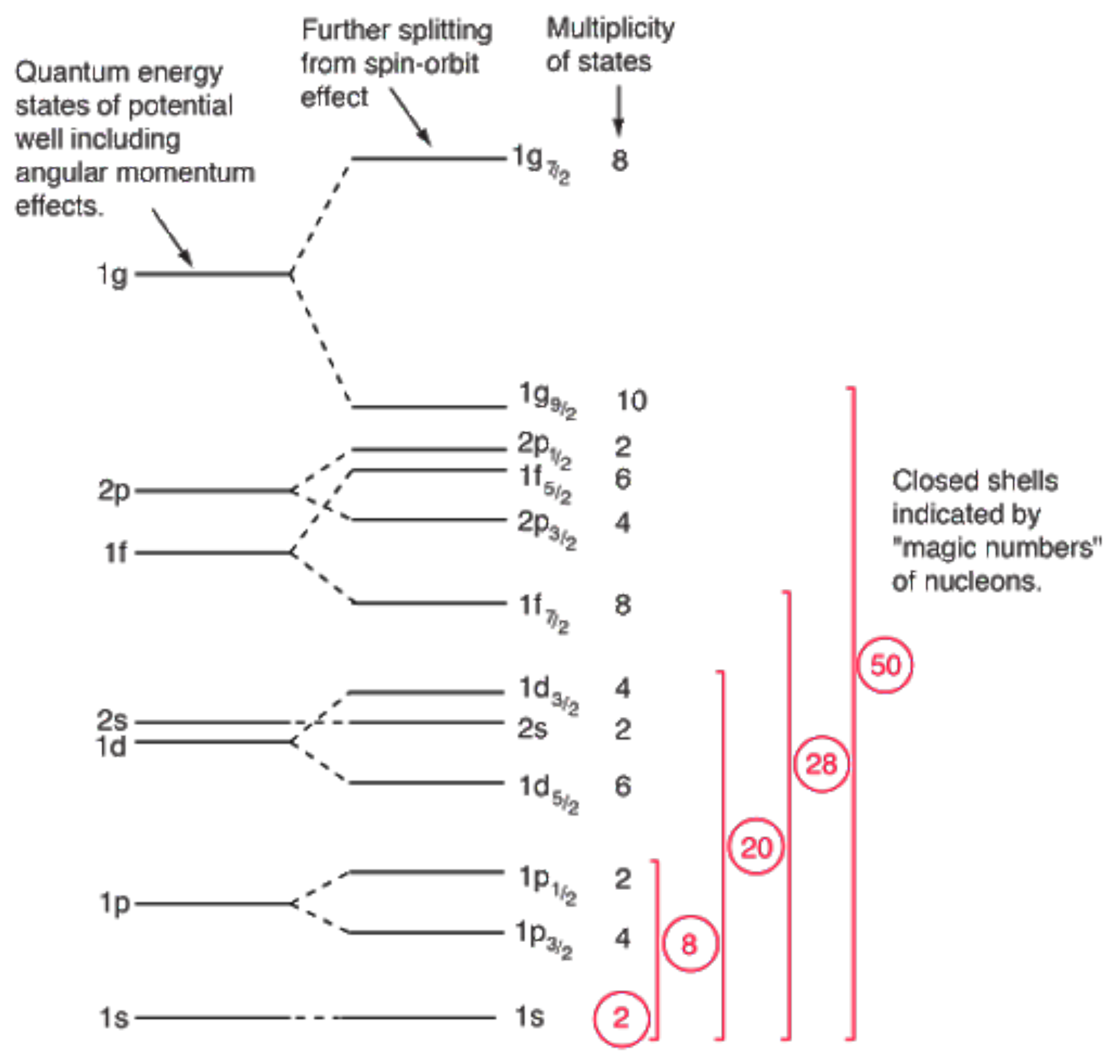
THEORY ⇆ EXPERIMENT

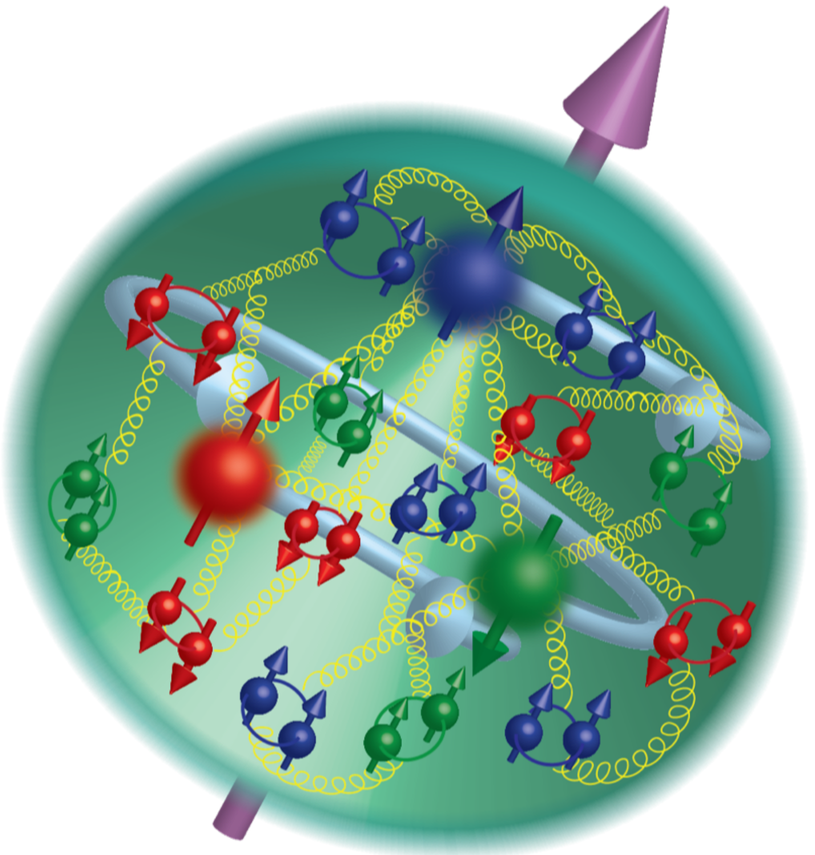


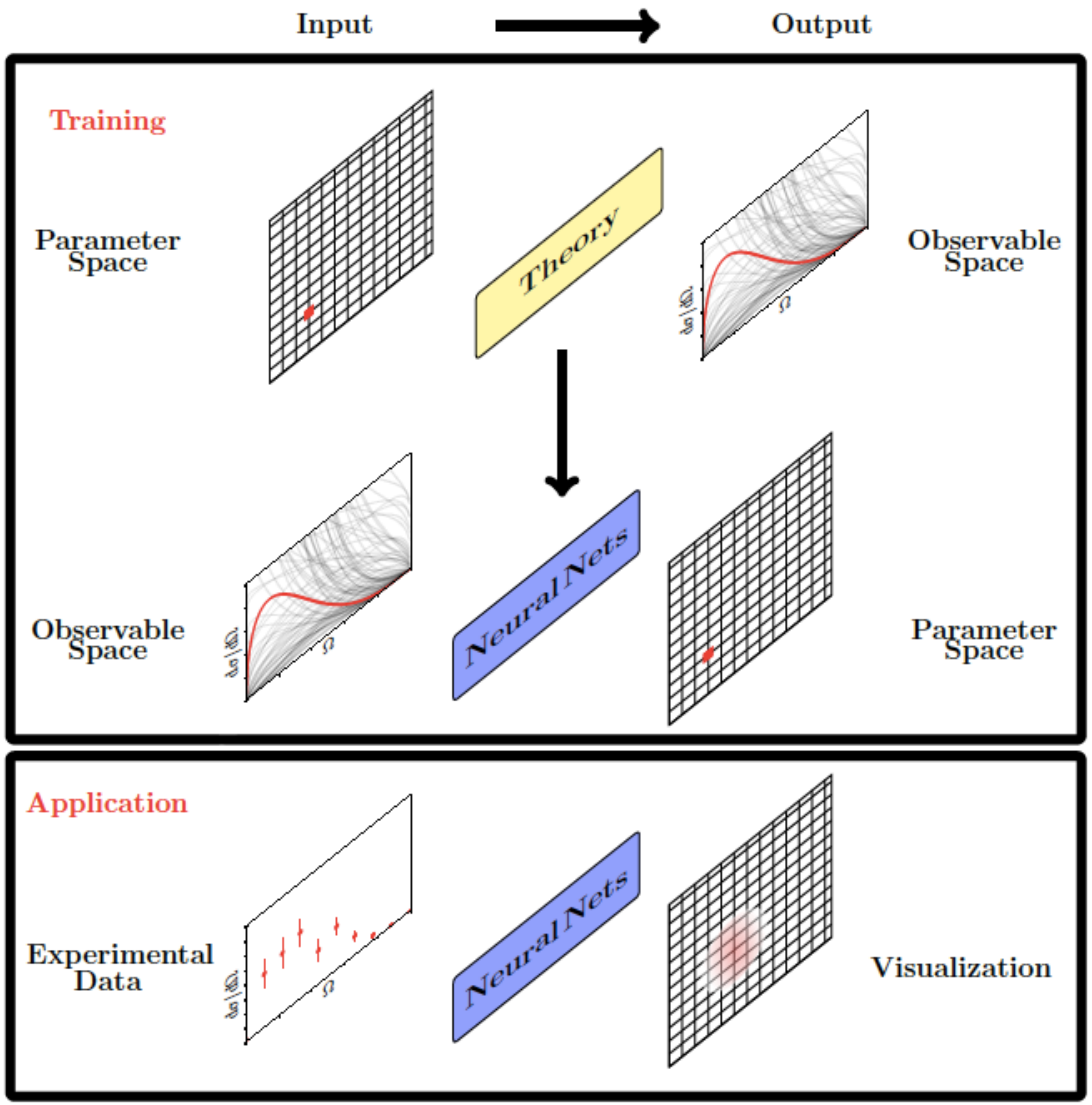
N. SATO, JEFFERSON LAB

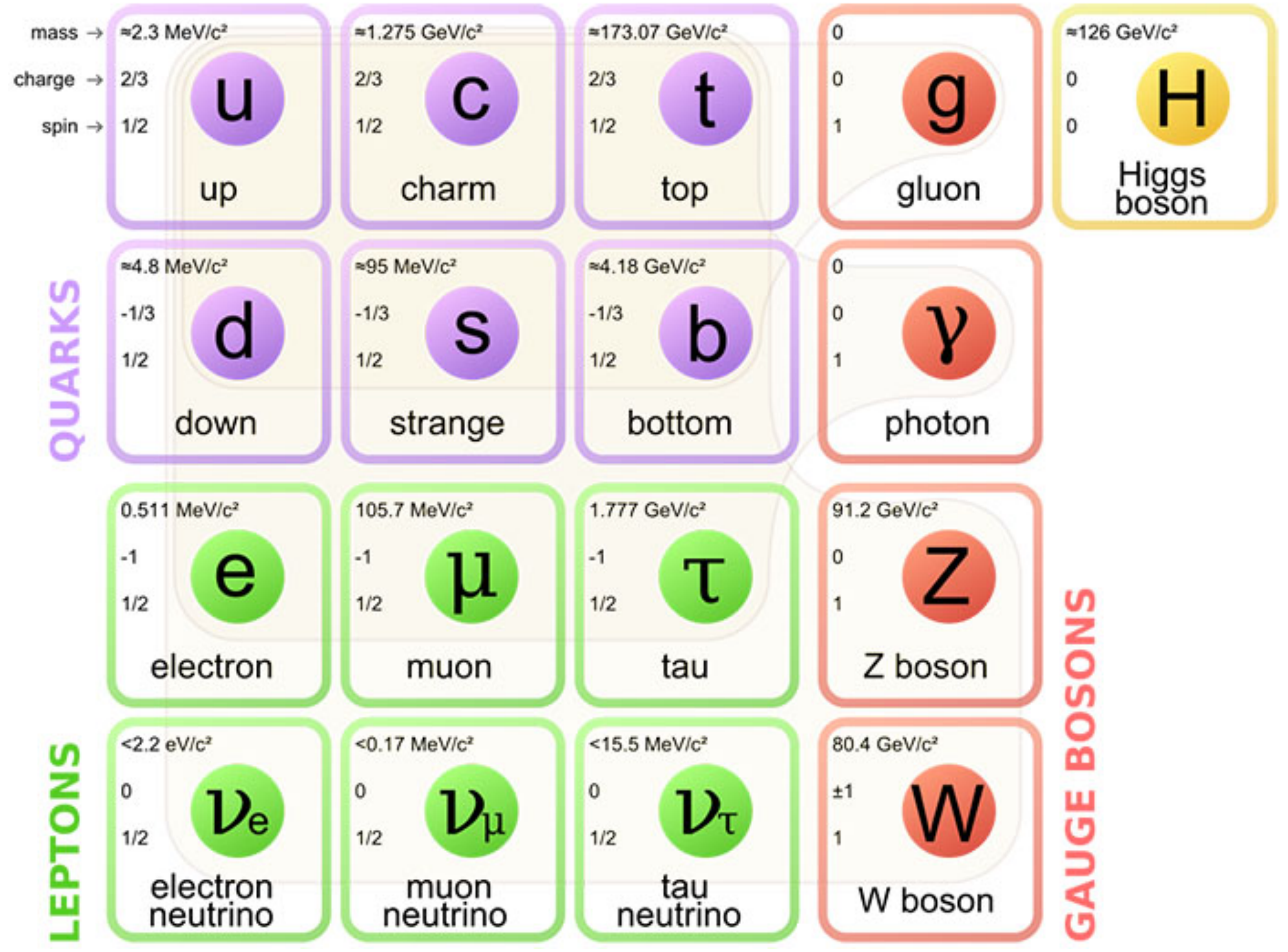
THEORY ⇆ EXPERIMENT

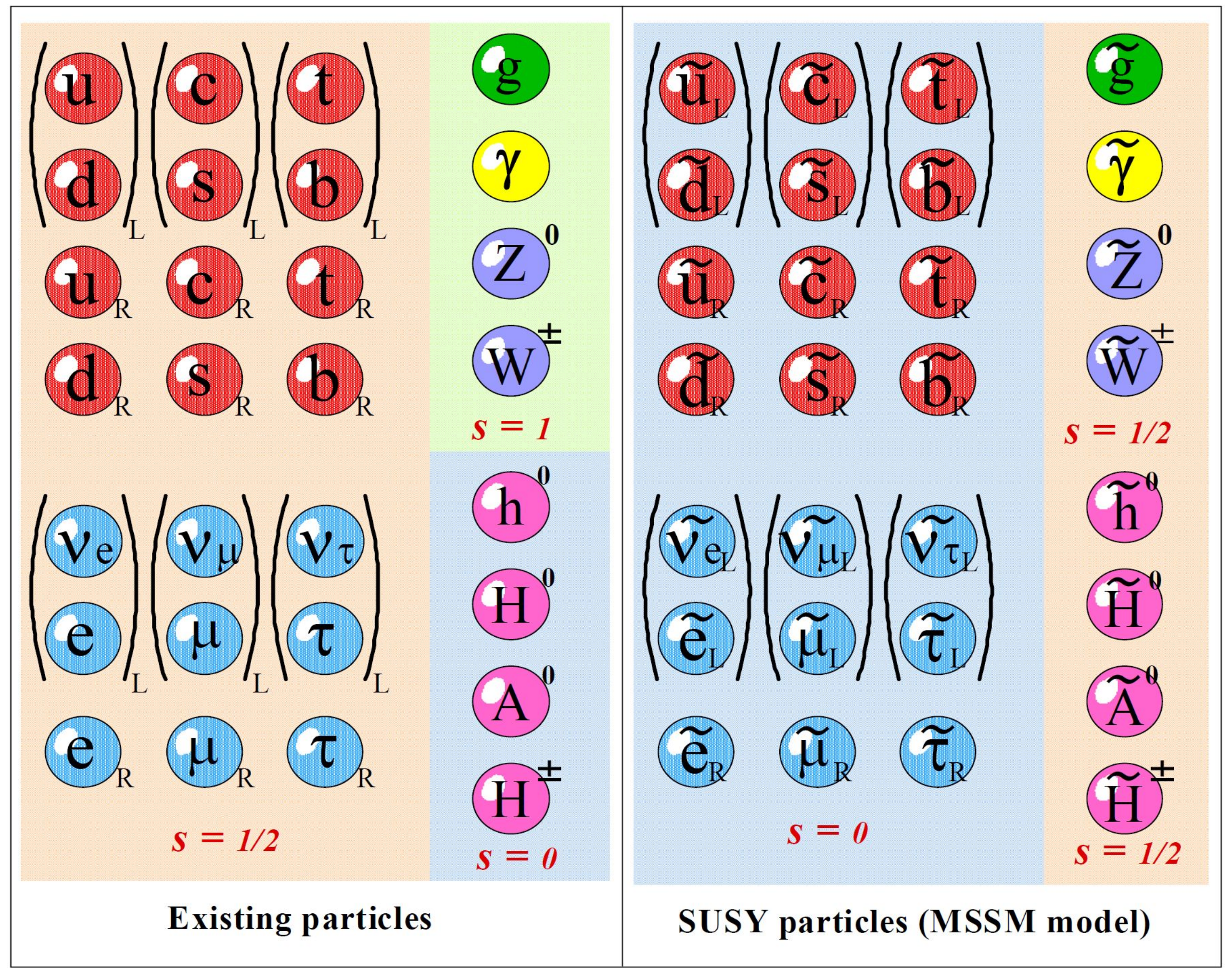




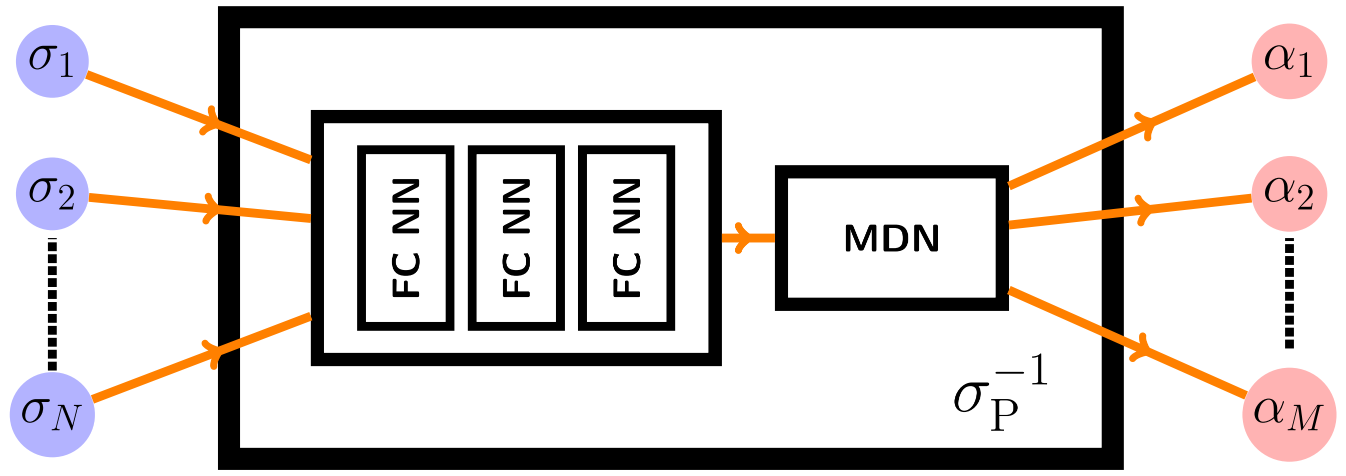








MIXTURE DENSITY NETWORK



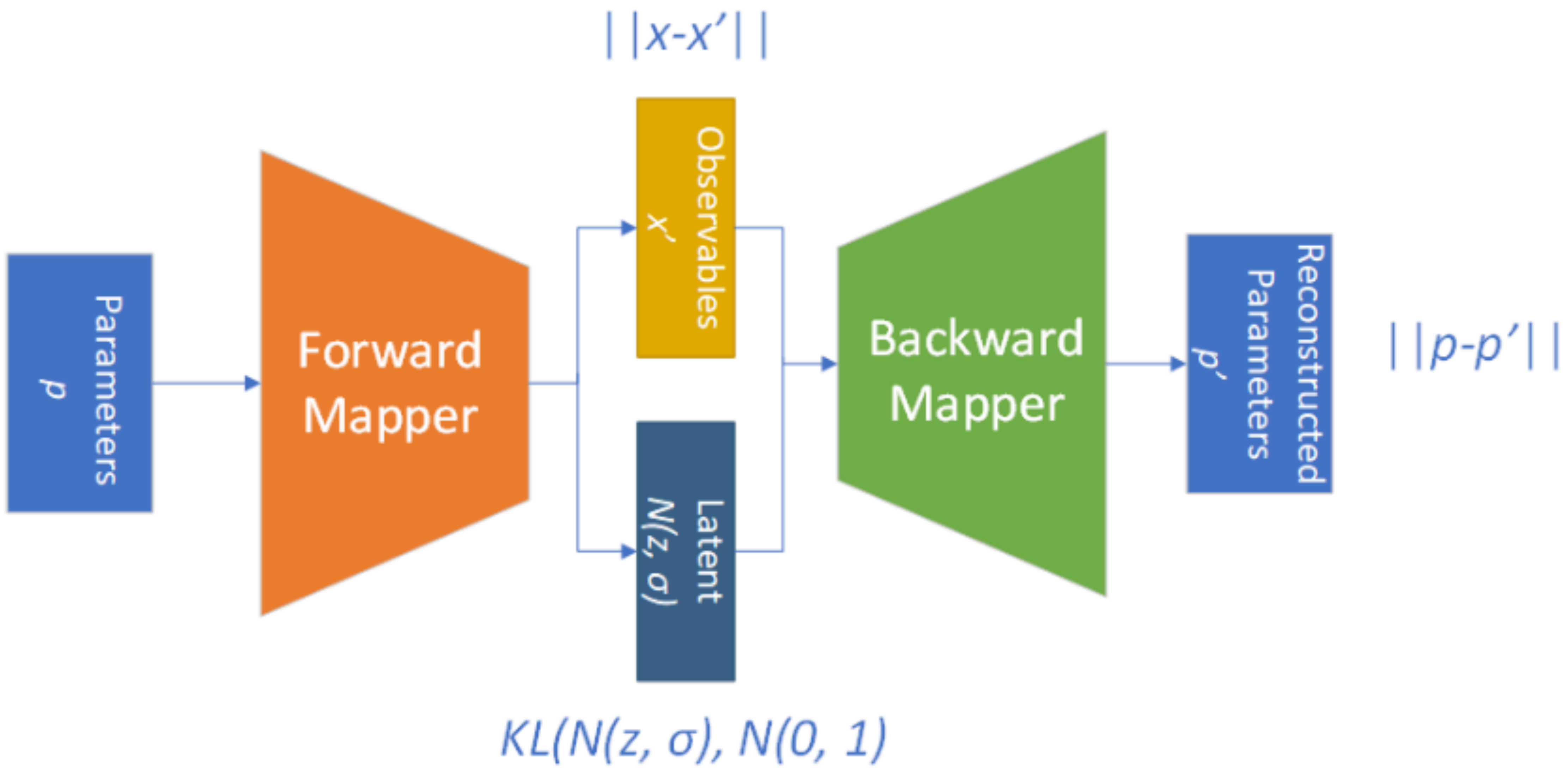
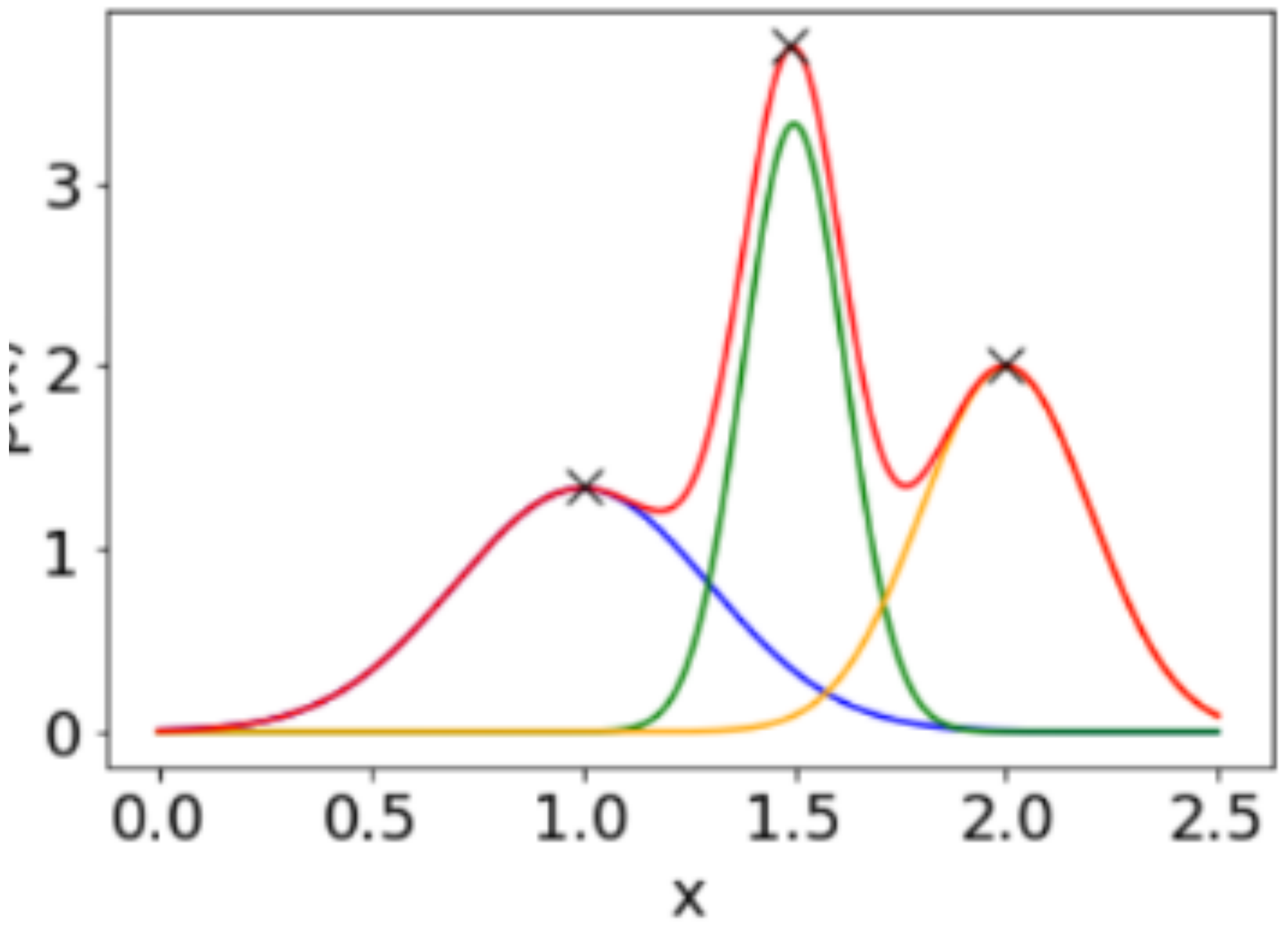
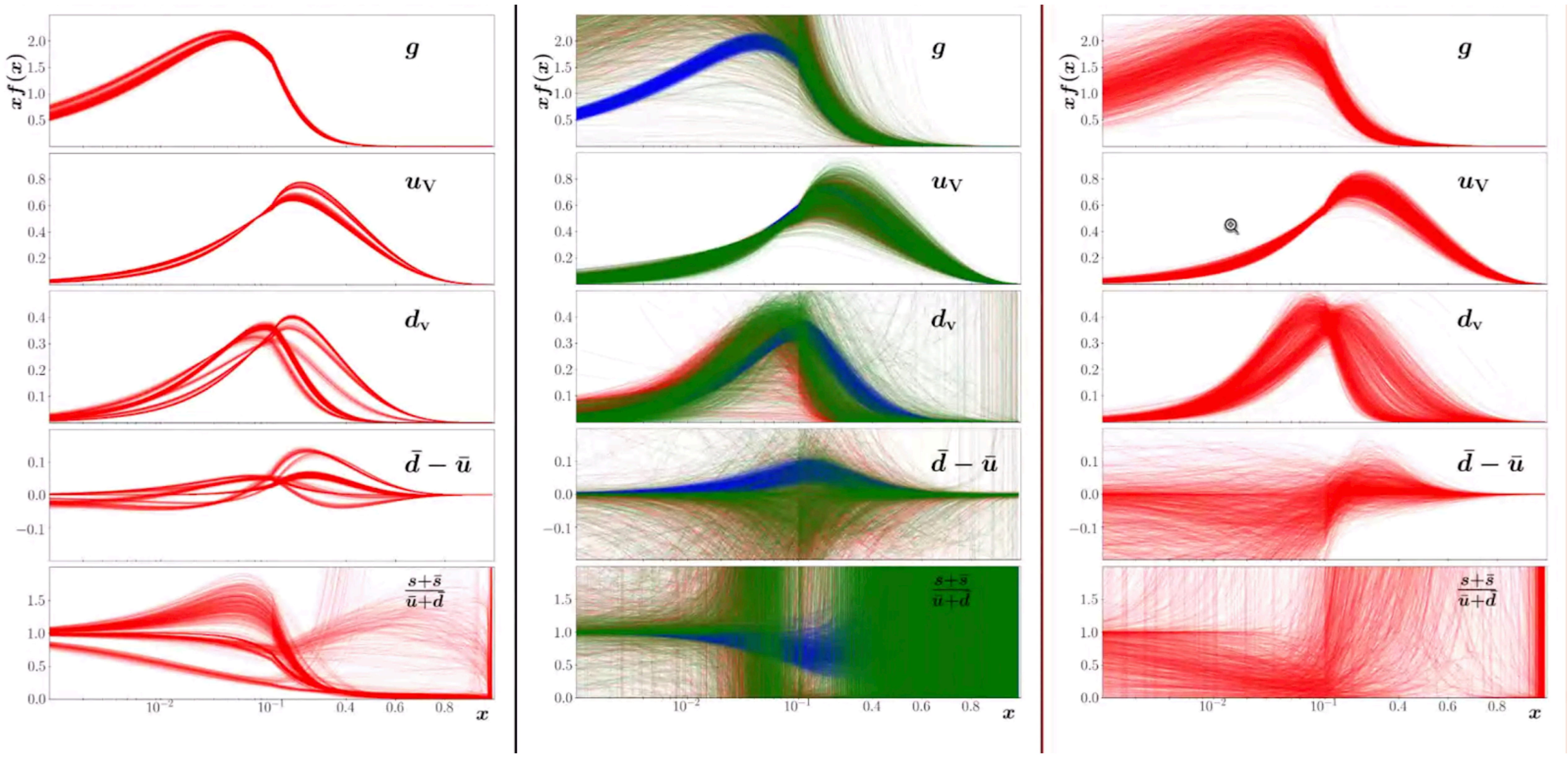


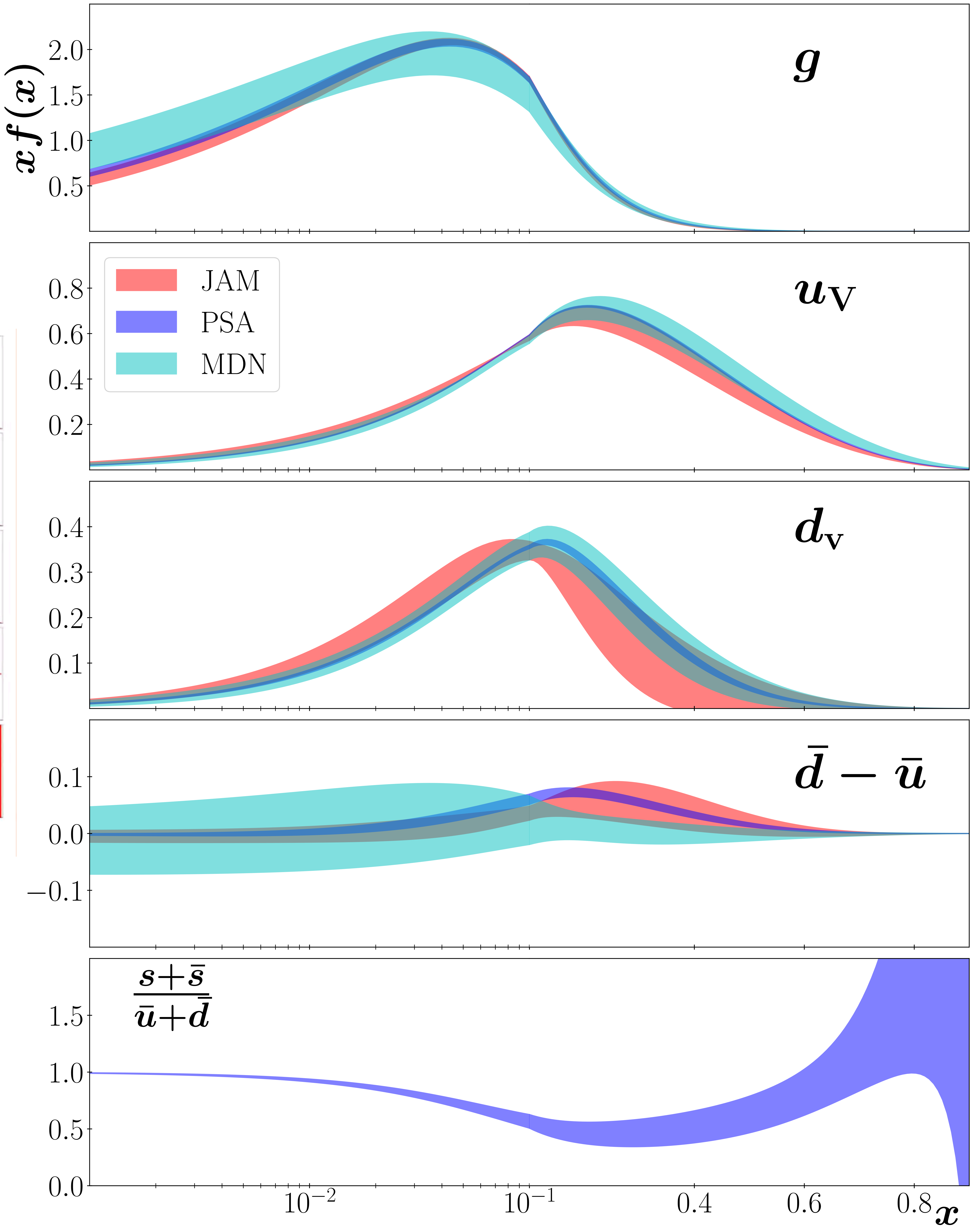
Figure 2: Architecture of the kinematics-independent inverse mapper.



Output Layer Interpretation:

$$p(\mathbf{t}|\mathbf{x}) = \sum_{k=1}^{K} \pi_k(\mathbf{x}) \mathcal{N}\left(\mathbf{t}|\boldsymbol{\mu}_k(\mathbf{x}), \sigma_k^2(\mathbf{x})\right)$$





$xf(x)$
g
u_V
d_v
$\bar{d}-\bar{u}$
$\frac{s+\bar{s}}{\bar{u}+\bar{d}}$
JAM
PSA
MDN
x
10^{-2}
10^{-1}
0.4
0.6
0.8

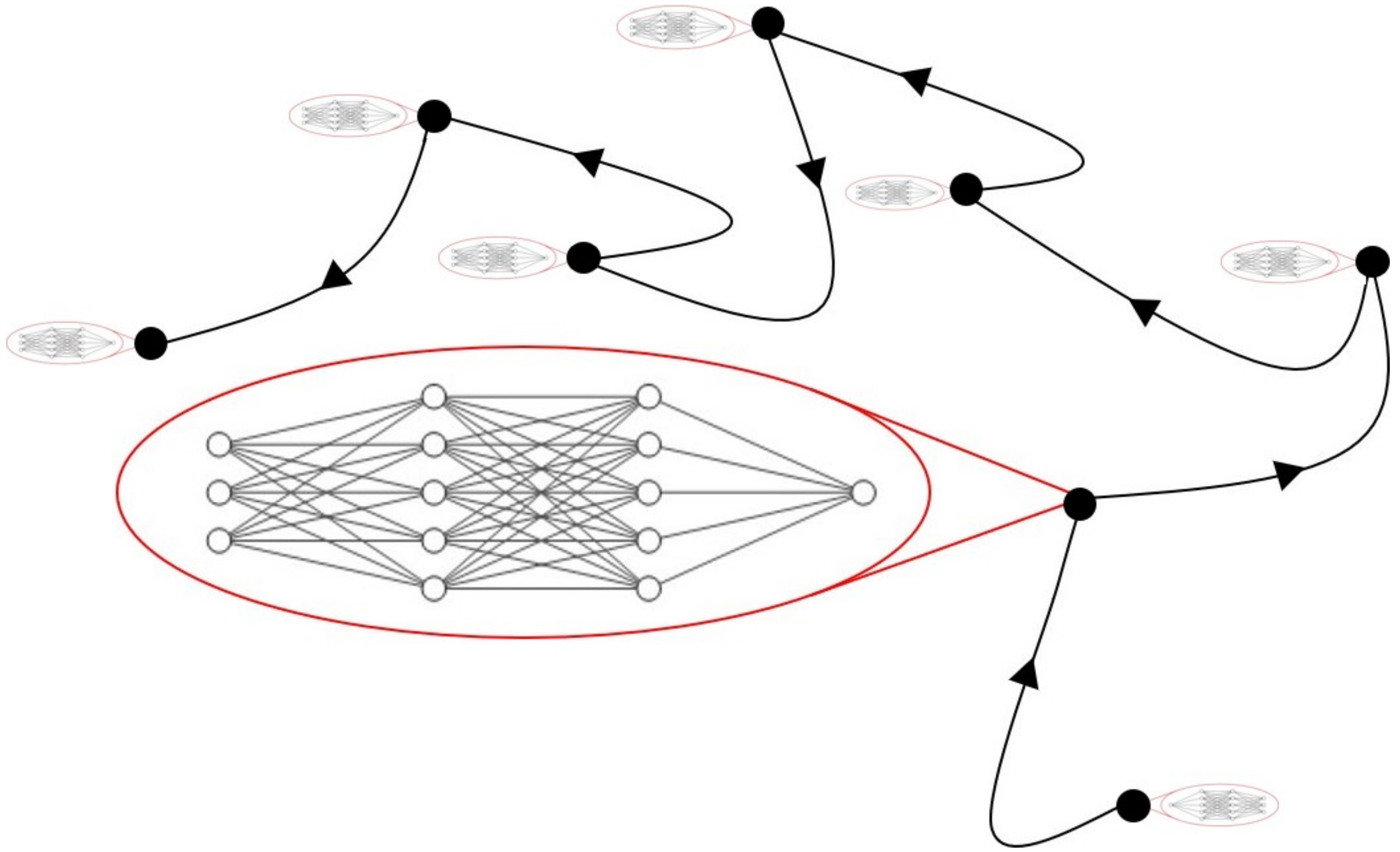
FAST MAPPING TO THEORETICAL PARAMETERS

Bayesian Neural Networks

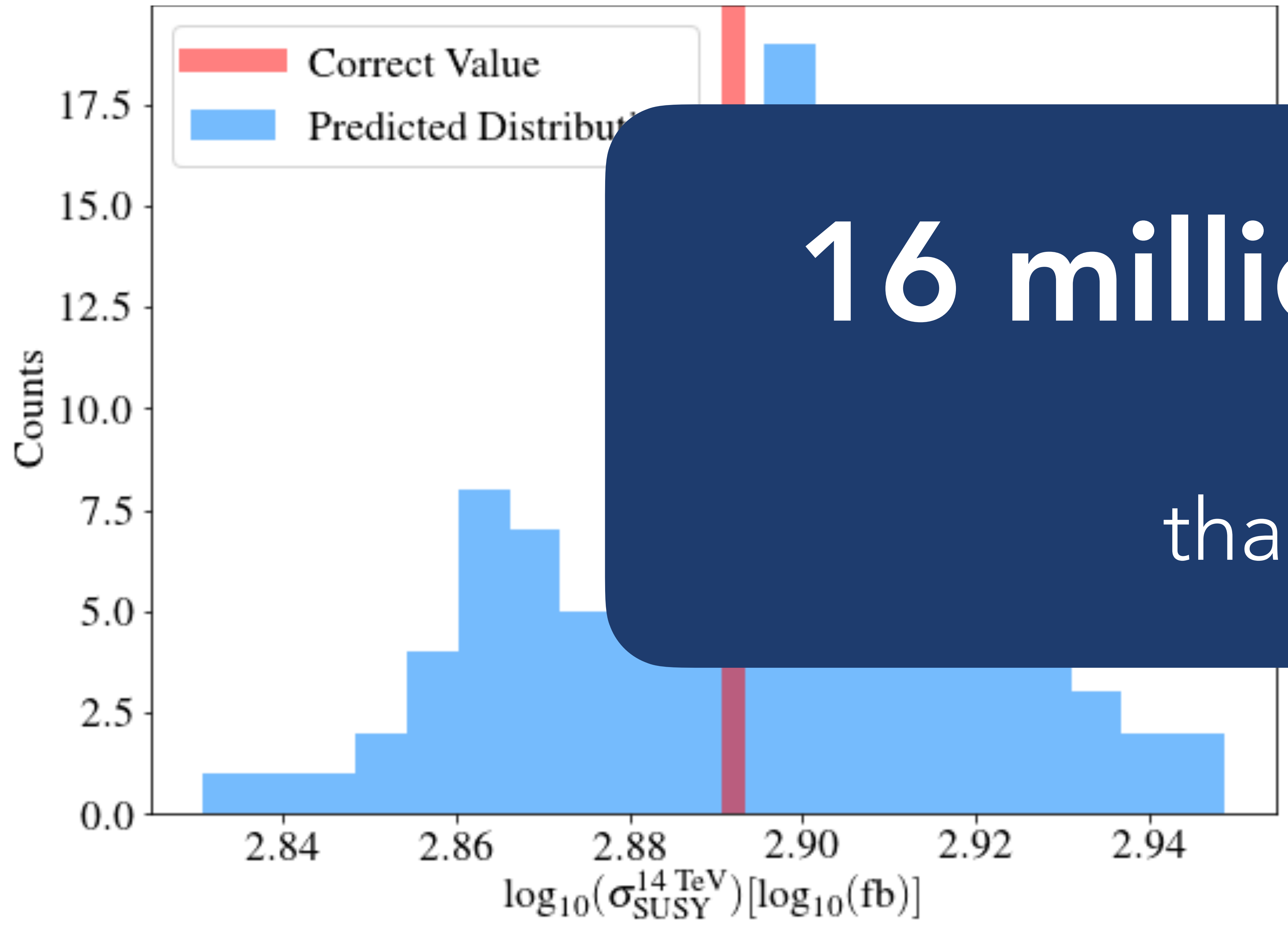
Training — Bayesian inference

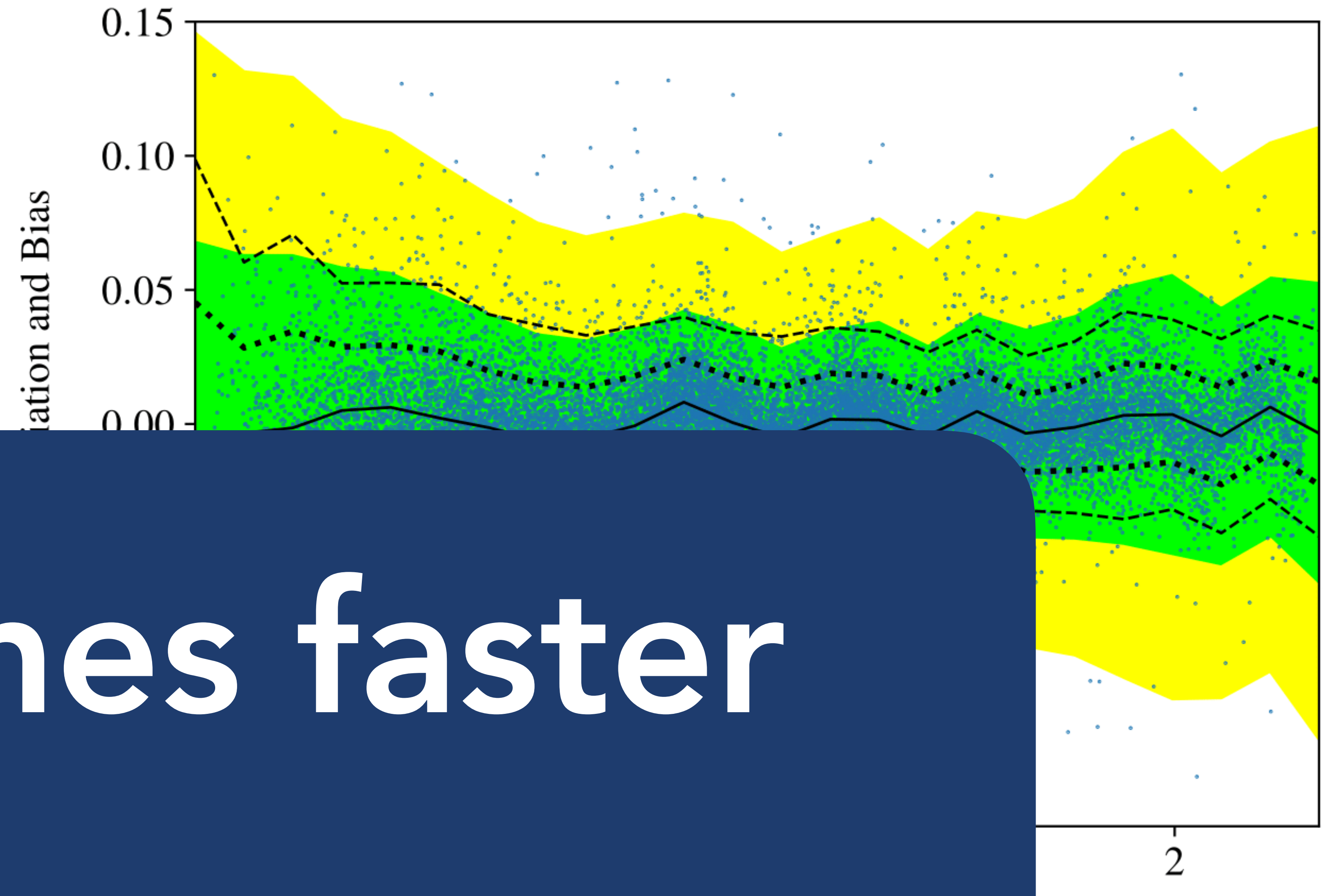
Can we make predictions with accurate error estimates?

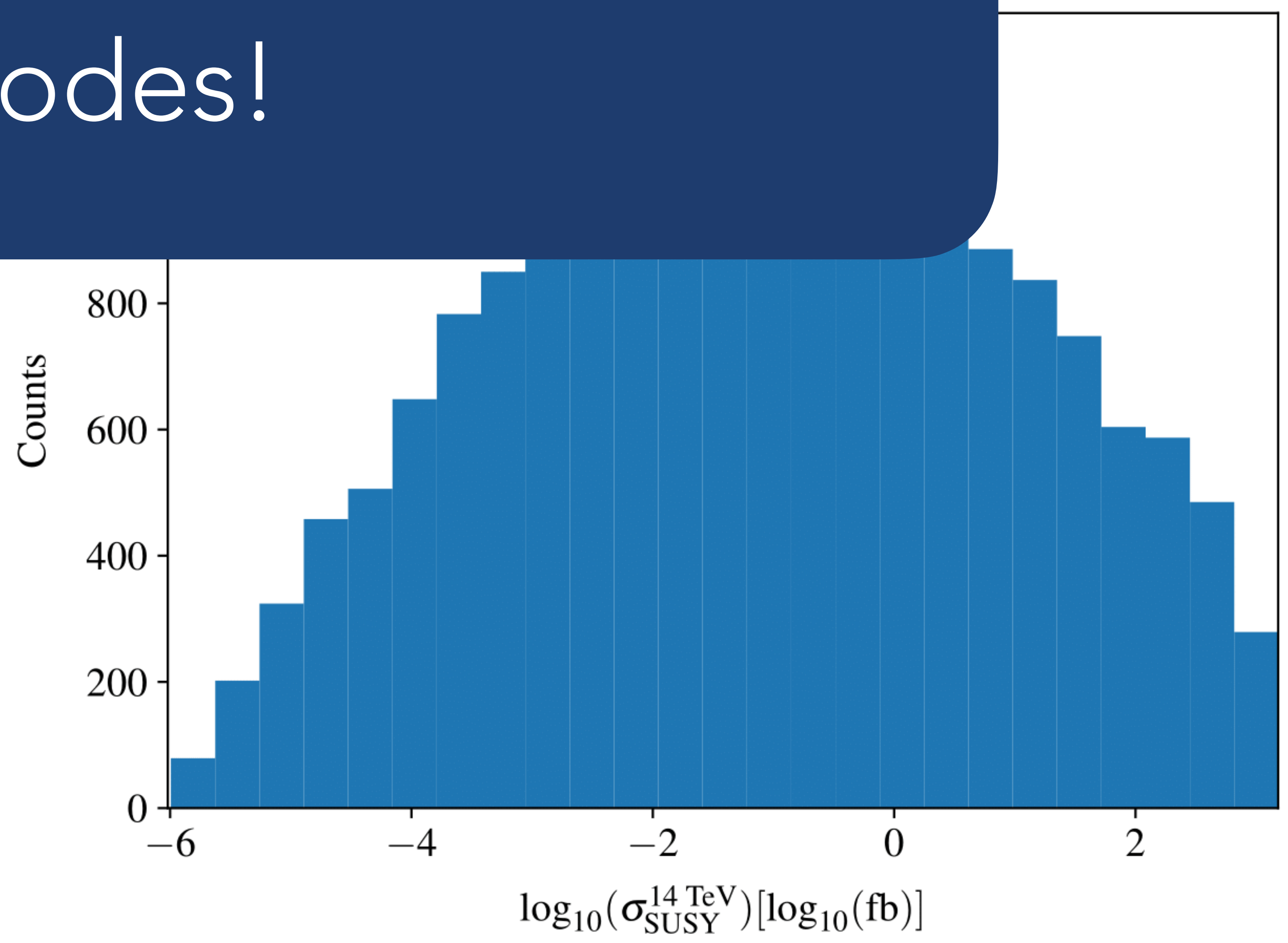
pMSSM parameters → total SUSY cross section



FAST MAPPING TO THEORETICAL PARAMETERS







16 million times faster

than theory codes!

B.S. Kronheim, M.P. Kuchera, H.B. Prosper, A. Karbo, Bayesian neural networks for fast SUSY predictions, Physics Letters B, Volume 813, 2021, 136041, ISSN 0370-2693, https://doi.org/10.1016/j.physletb.2020.136041.

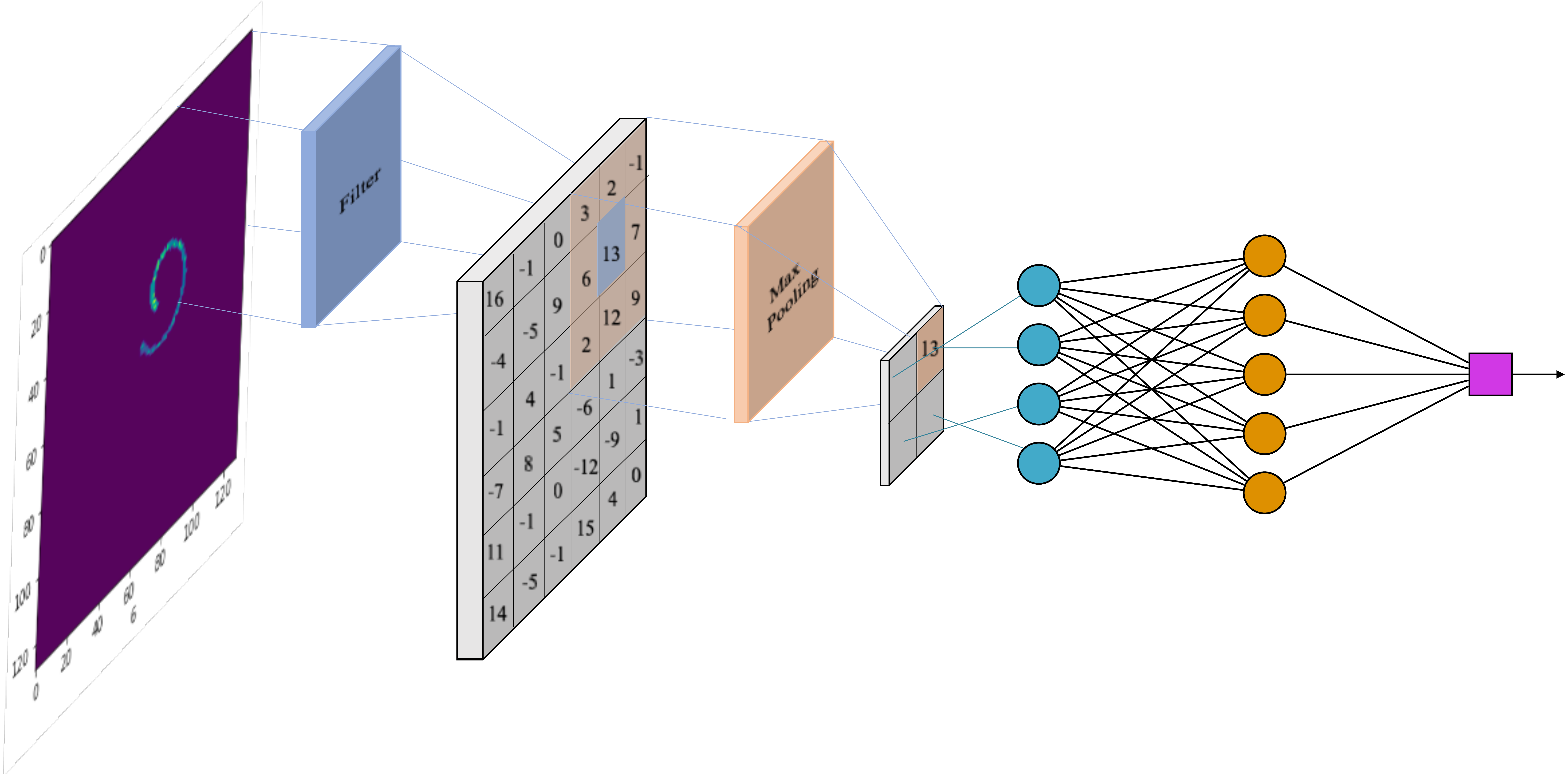
https://arxiv.org/abs/2009.14393

https://alpha-davidson.github.io/TensorBNN

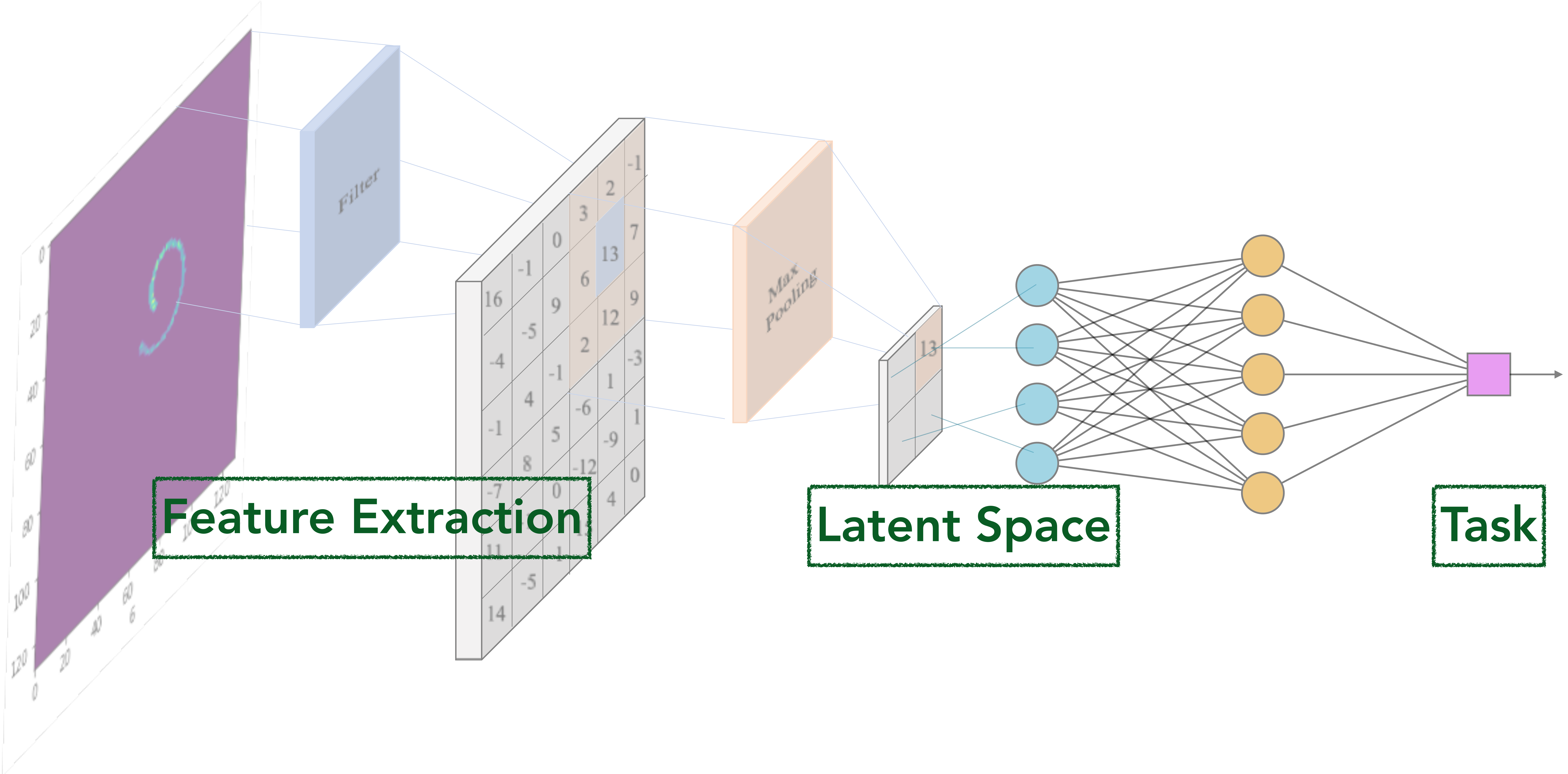
CONVOLUTIONAL NEURAL NETWORKS

CLASSIFICATION

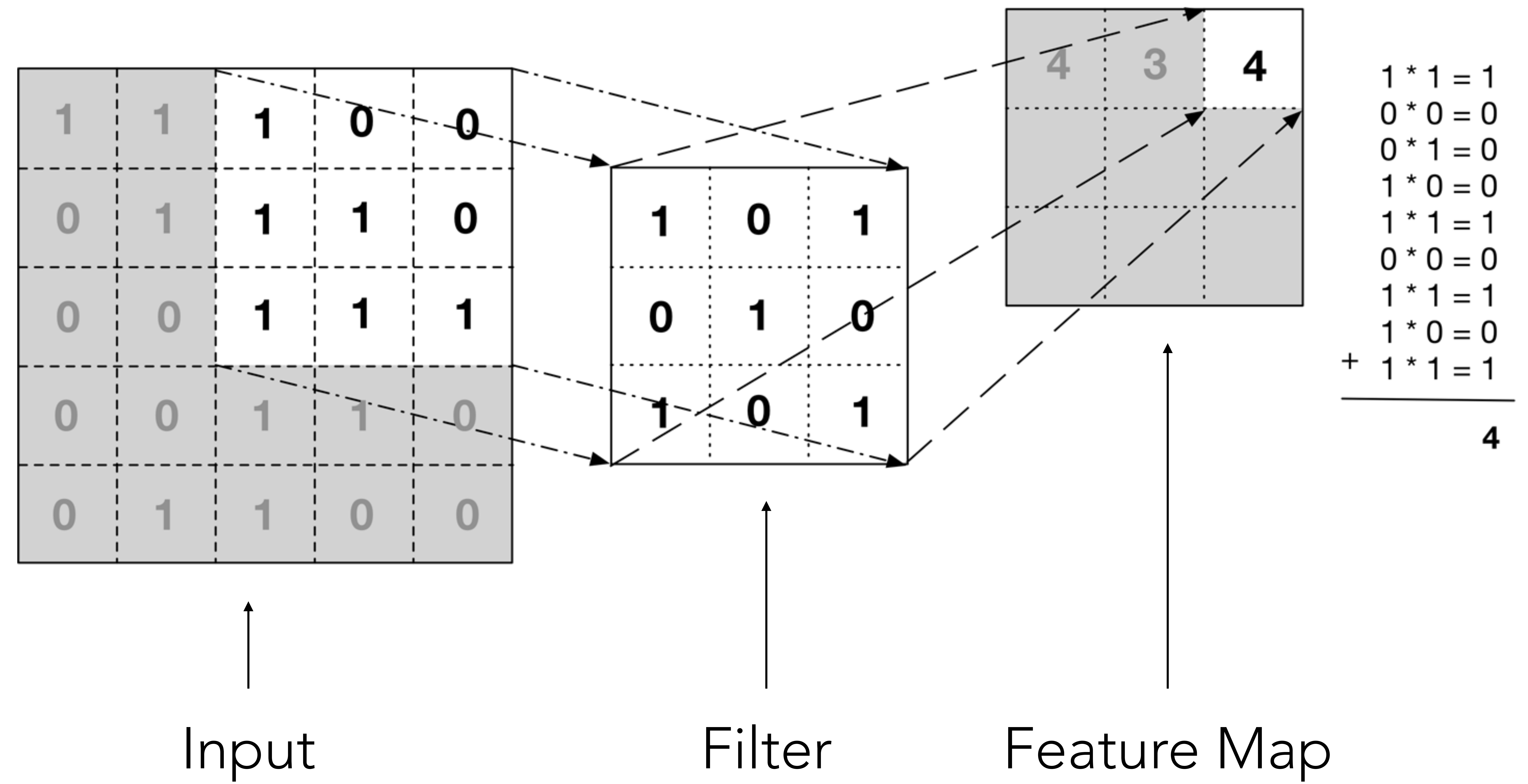
CONVOLUTIONAL NEURAL NETWORKS



CONVOLUTIONAL NEURAL NETWORKS

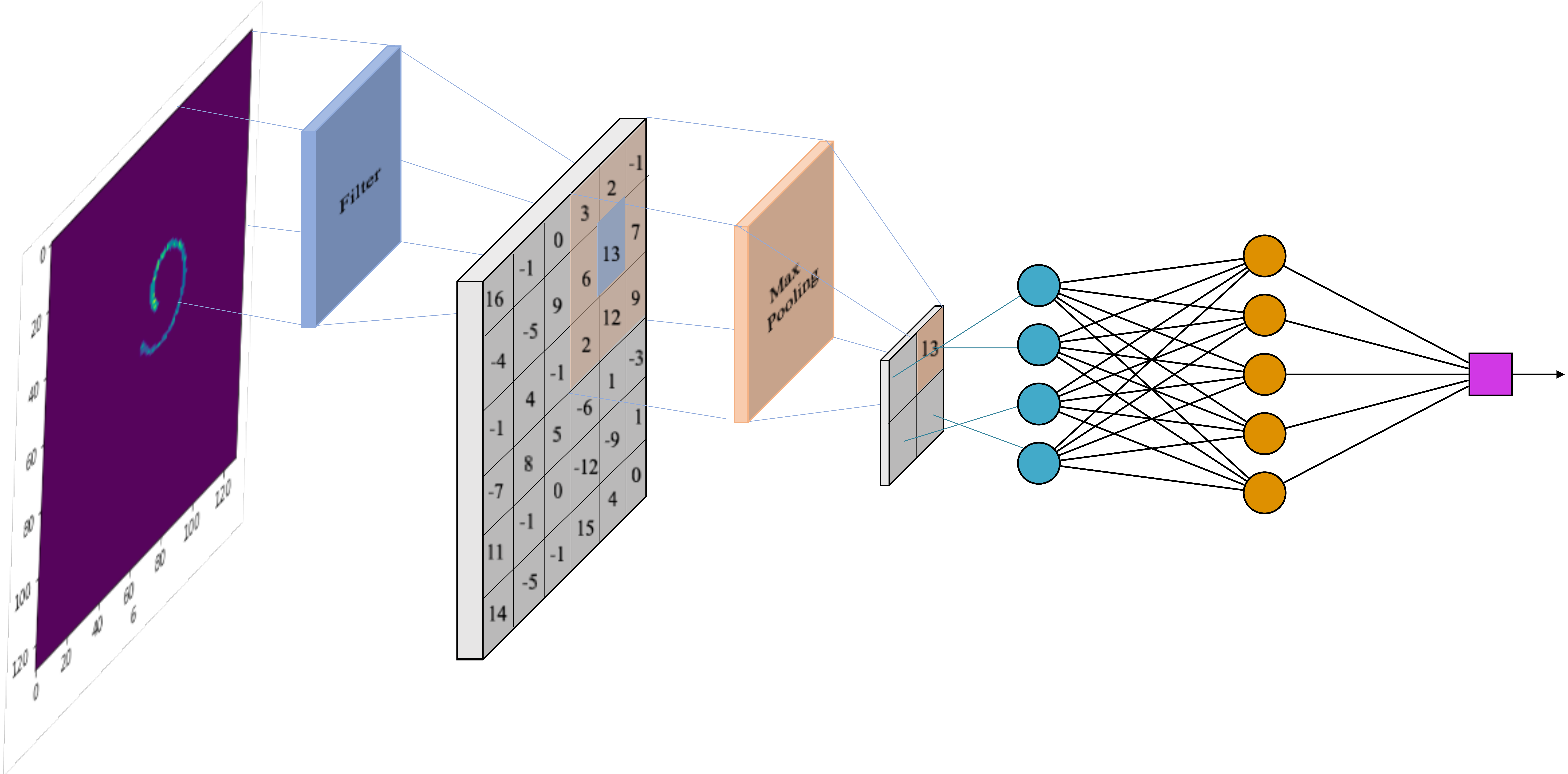


DISCRETE CONVOLUTION



ADAPTED FROM *DEEP LEARNING*, ADAM GIBSON & JOSH PATTERSON

CONVOLUTIONAL NEURAL NETWORKS

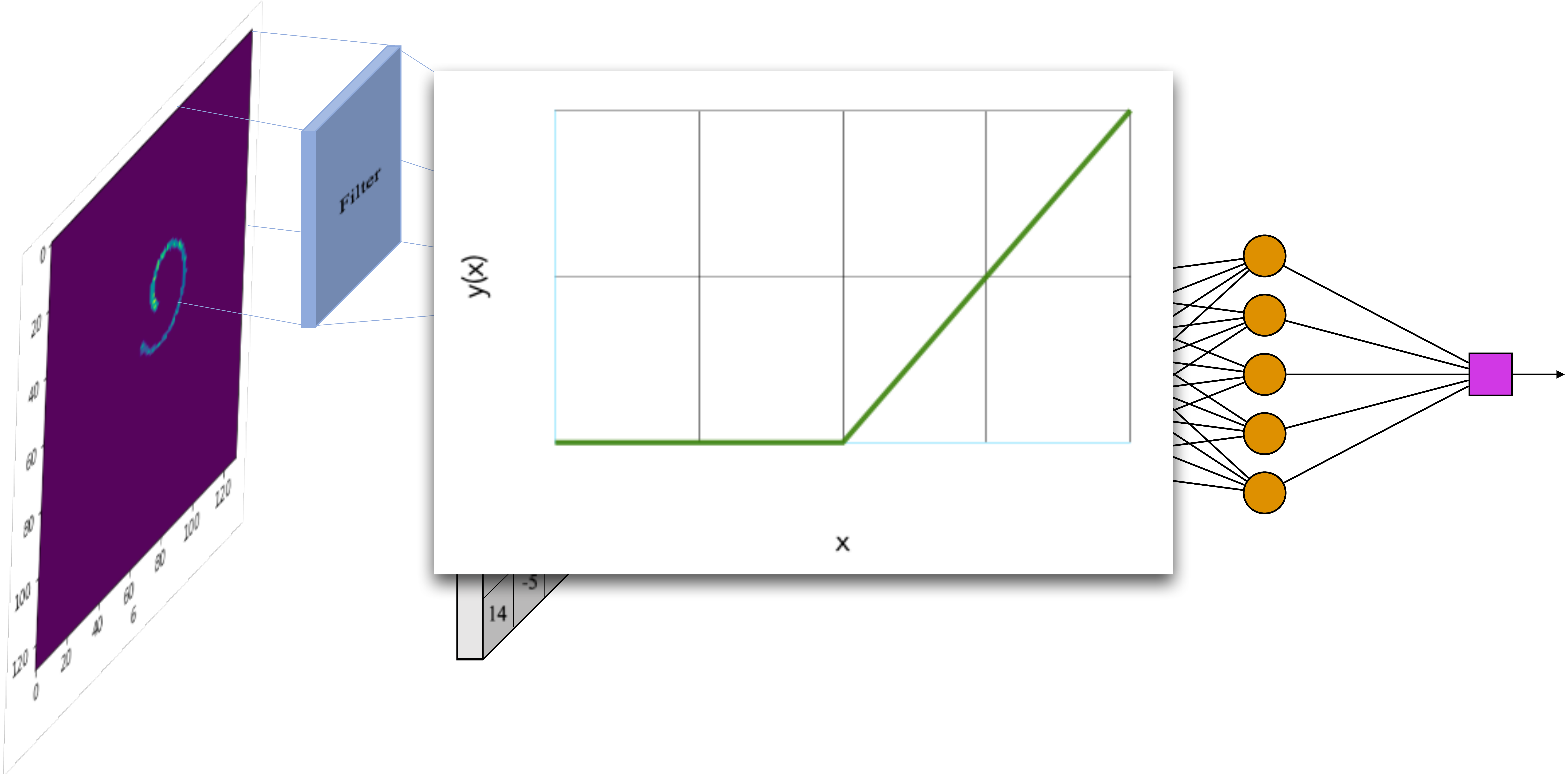



1156

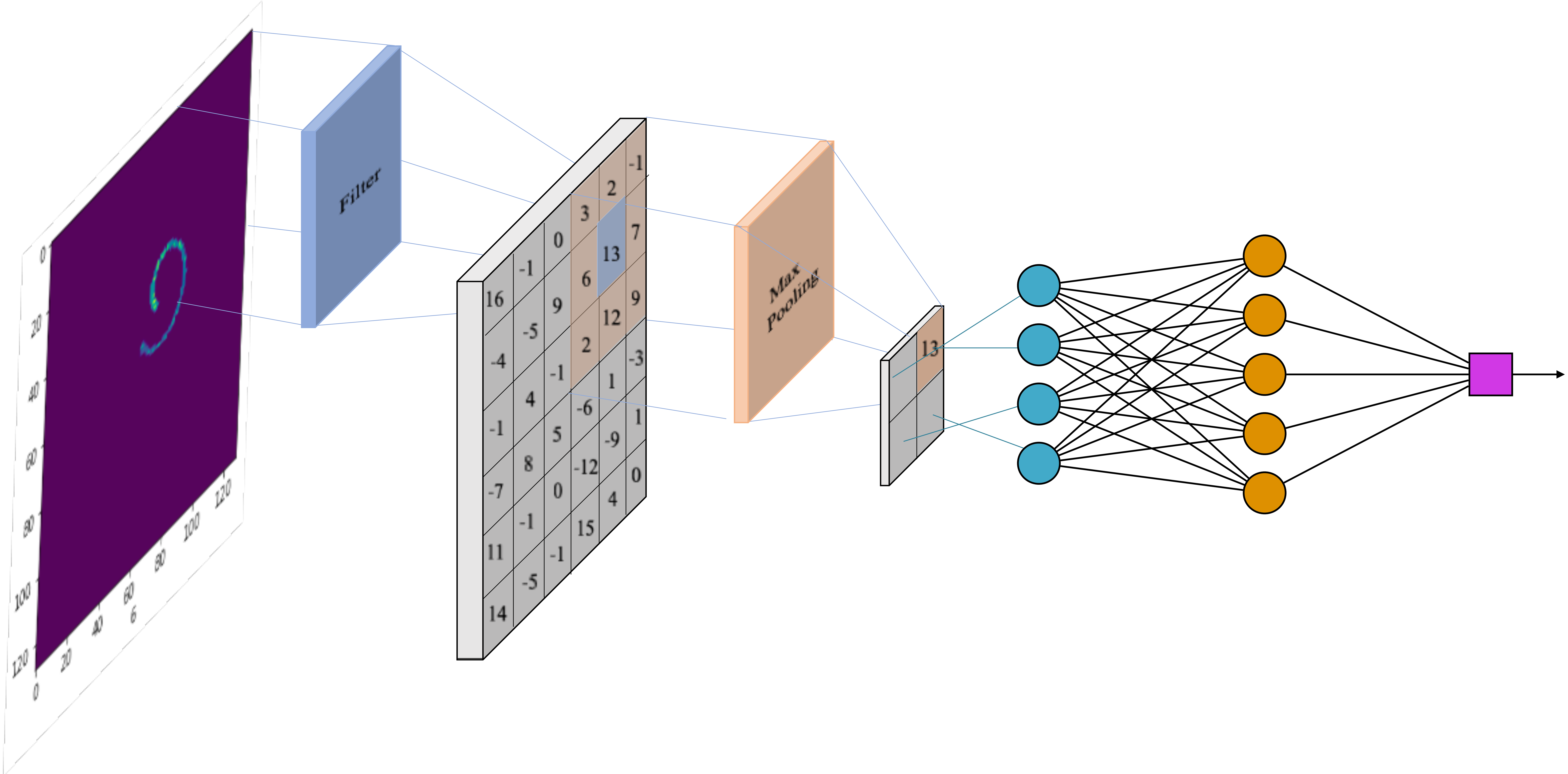
-1	-1	-1	-1	-1
-1	-1	-1	-1	-1
5	5	5	5	5
-1	-1	-1	-1	-1
-1	-1	-1	-1	-1

-1	-1	5	-1	-1
-1	-1	5	-1	-1
-1	-1	5	-1	-1
-1	-1	5	-1	-1
-1	-1	5	-1	-1

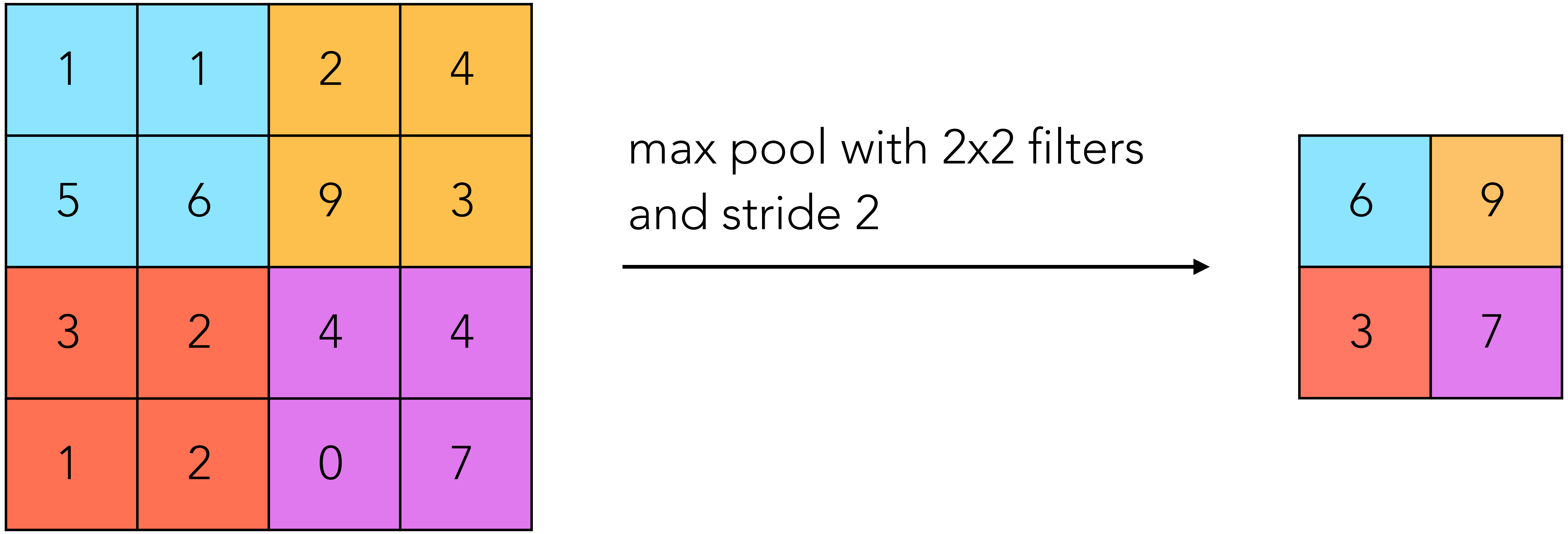
CONVOLUTIONAL NEURAL NETWORKS



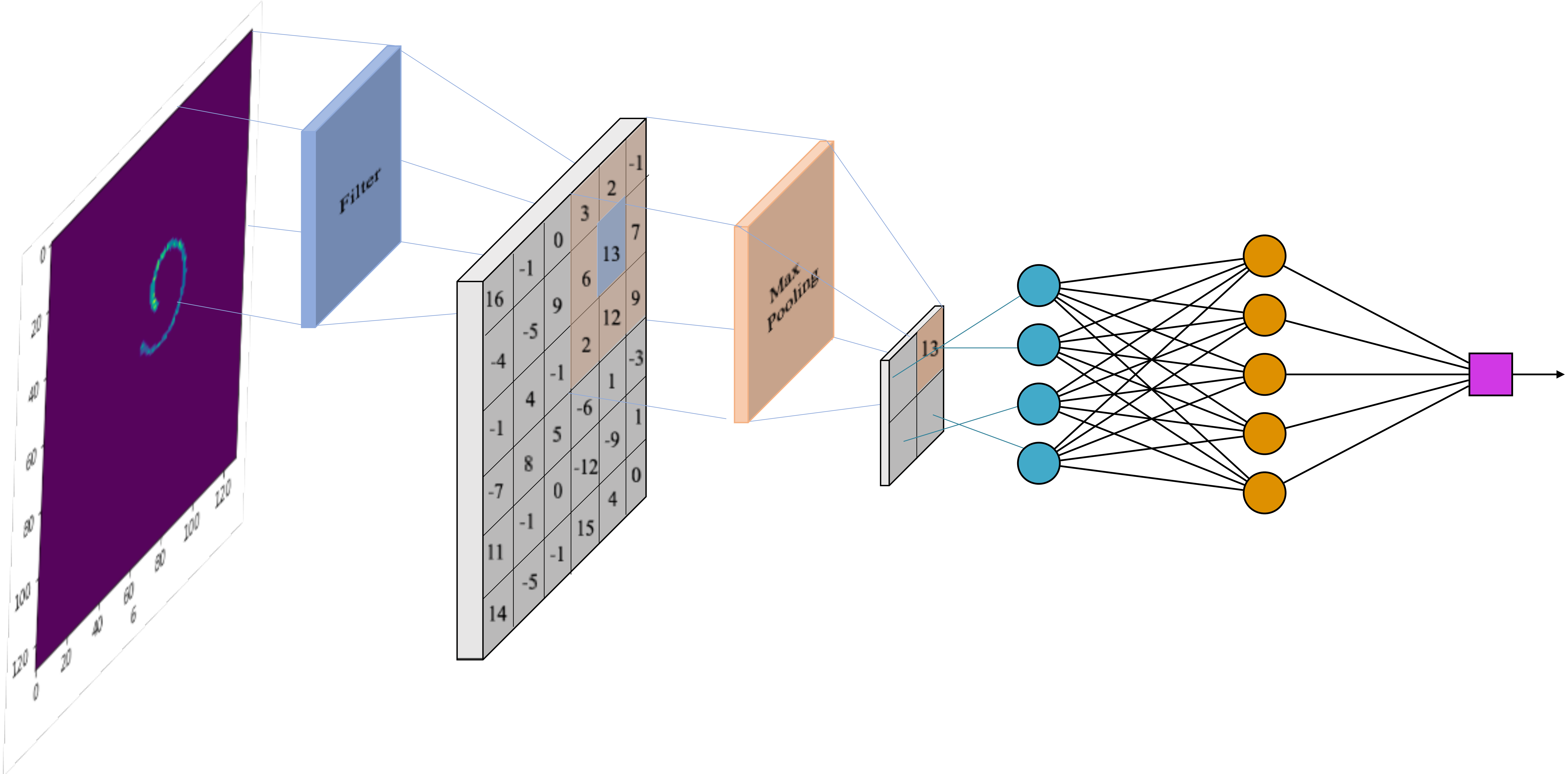
CONVOLUTIONAL NEURAL NETWORKS



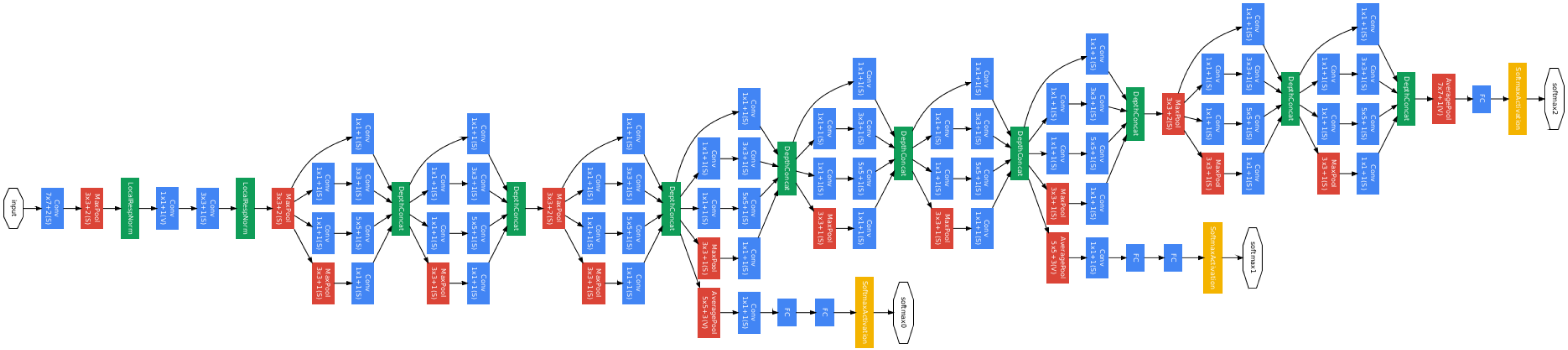
MAX POOLING


1 1 2 4
5 6 9 3
3 2 4 4
1 2 0 7
max pool with 2x2 filters
and stride 2
6 9
3 7

CONVOLUTIONAL NEURAL NETWORKS



"GoogLeNet network with all the bells and whistles"



CHRISTIAN SZEGEDY ET. AL. GOING DEEPER WITH CONVOLUTIONS.

CHOOSING AN ARCHITECTURE

HOW MANY LAYERS?

HOW MANY NODES PER LAYER?

LEARNING RATE

DROPOUT?

WHAT ACTIVATION FUNCTION(S)?

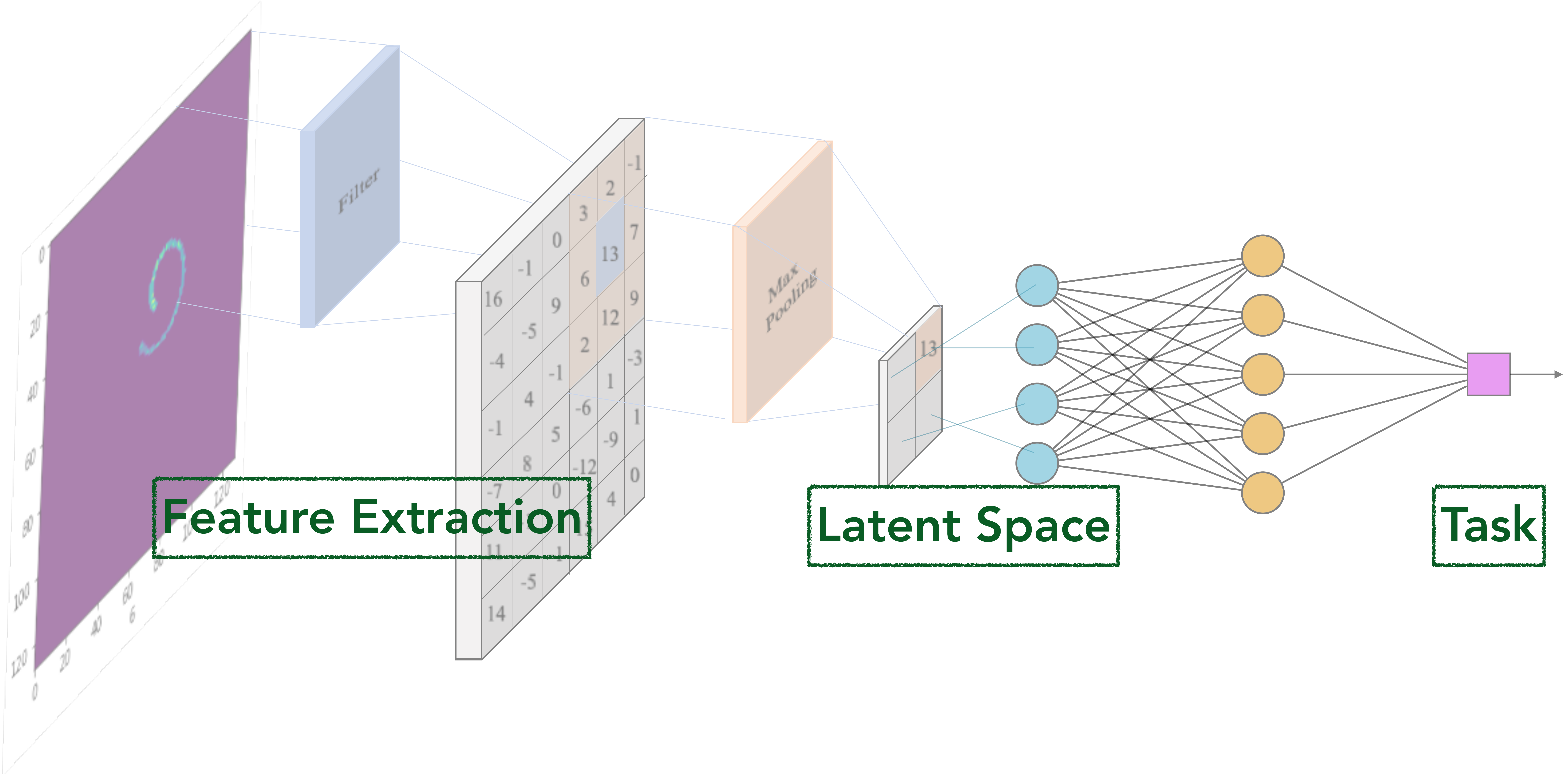
HOW MANY CONVOLUTION LAYERS?

FILTER SIZE?

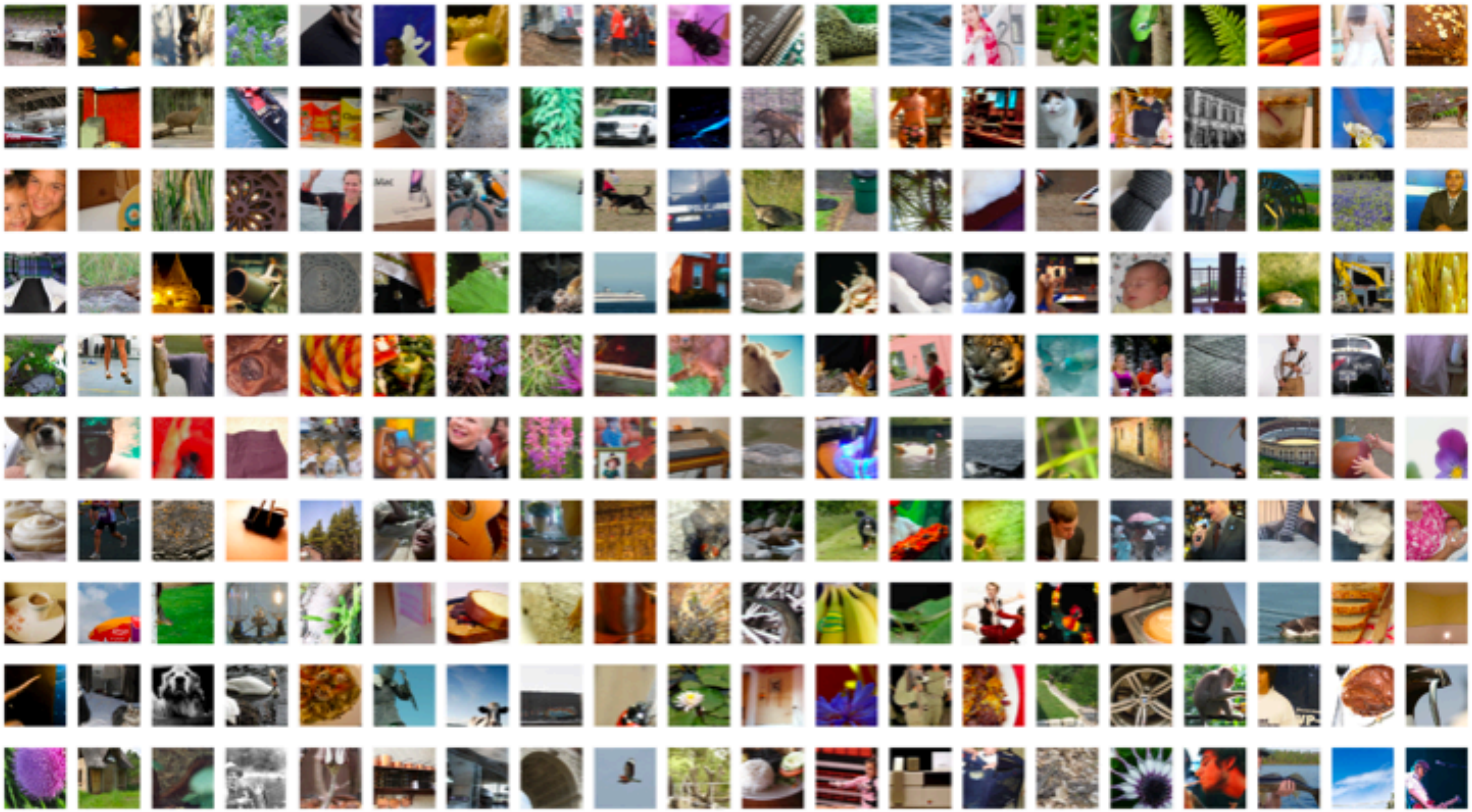
STRIDE?

POOLING?

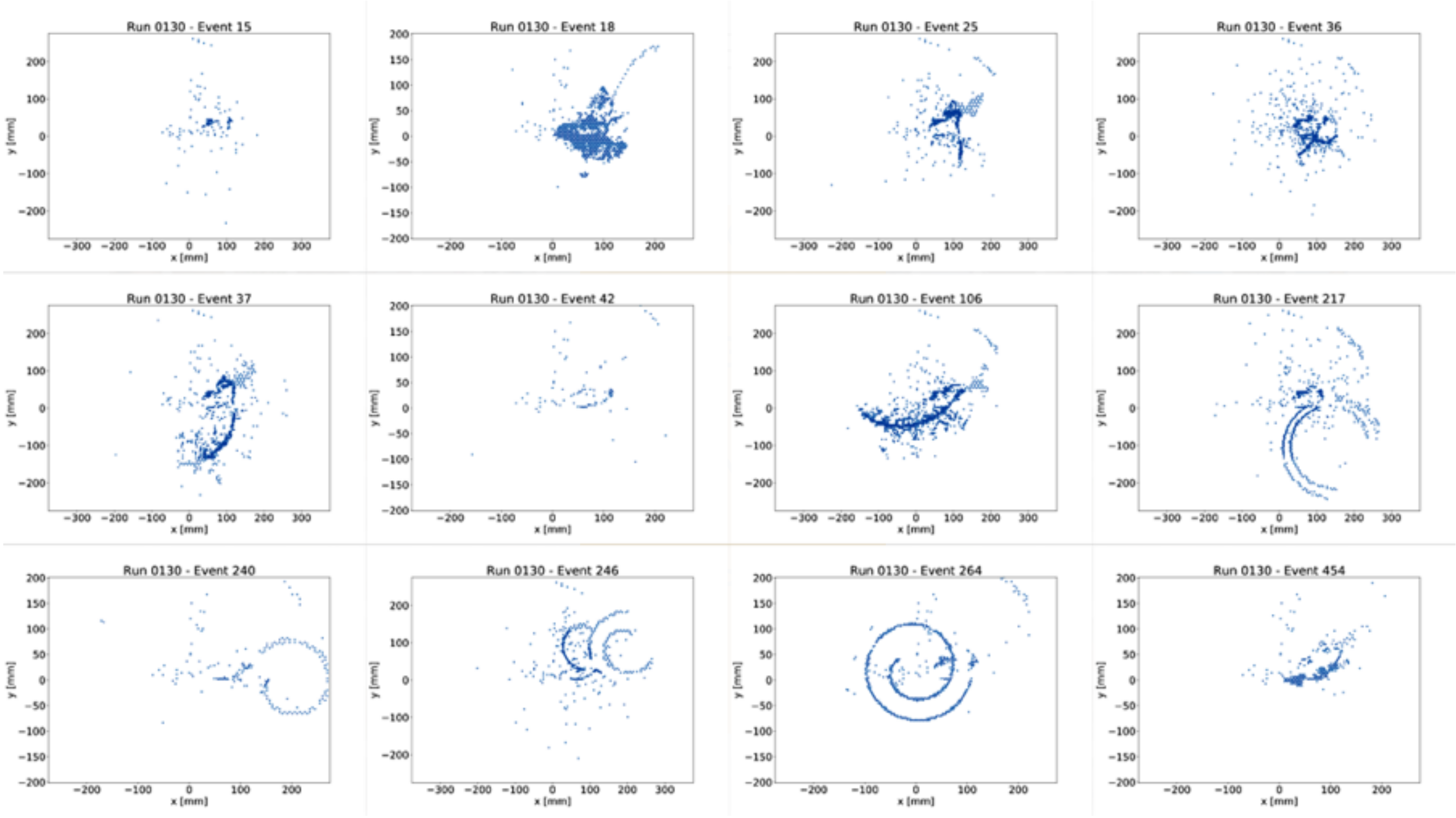
PRE-TRAINED MODELS

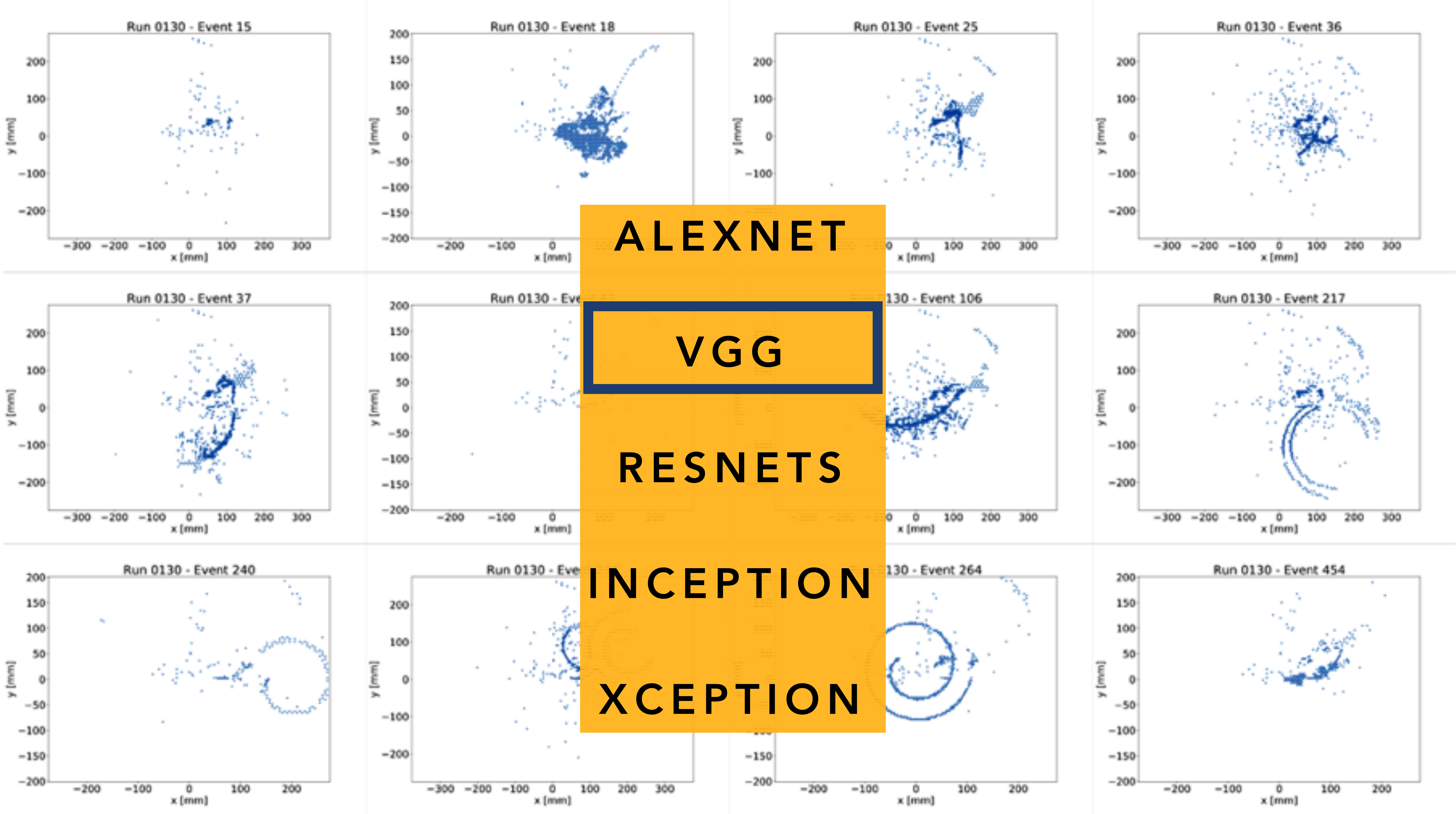


PRE-TRAINED MODELS



PRETRAINED MODELS


Run 0130 - Event 15
Run 0130 - Event 18
Run 0130 - Event 25
Run 0130 - Event 36
Run 0130 - Event 37
Run 0130 - Event 42
Run 0130 - Event 106
Run 0130 - Event 217
Run 0130 - Event 240
Run 0130 - Event 246
Run 0130 - Event 264
Run 0130 - Event 454
x [mm]
y [mm]



ALEXNET

VGG

RESNETS

INCEPTION

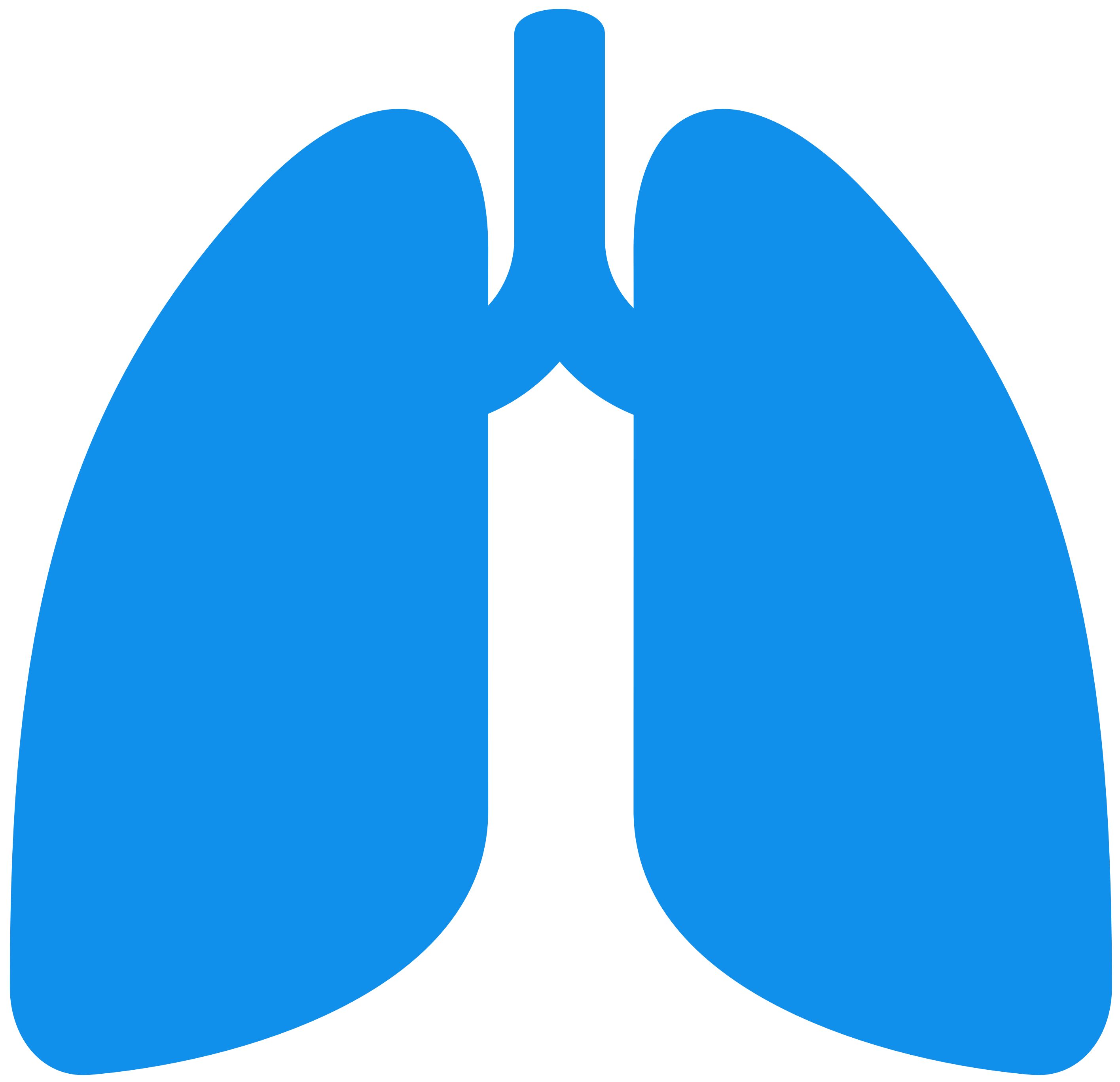
XCEPTION

J. Z. TAYLOR, HONOR'S THESIS, DAVIDSON COLLEGE

M.P. KUCHERA, ET. AL. NUCLEAR INSTRUMENTATION AND METHODS A (2019)

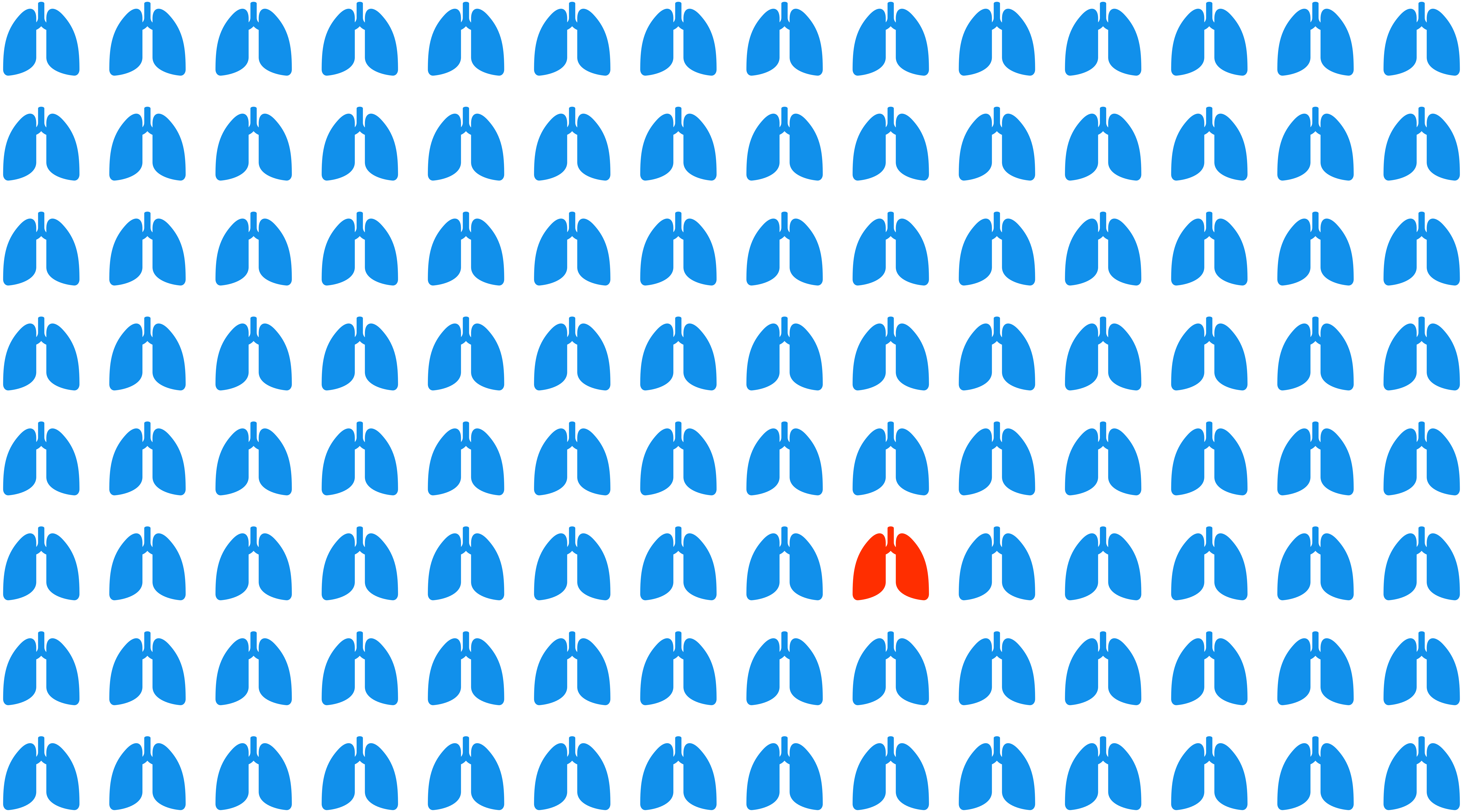
Application 2: Can we use machine learning to **accurately** classify events in detectors?

Metrics



Detect Lung Cancer

99% Accuracy



		PREDICTED	
		Proton	Not Proton
TRUE	Proton	TRUE POSITIVE (TP)	FALSE NEGATIVE (FN)
	Not Proton	FALSE POSITIVE (FP)	TRUE NEGATIVE (TN)

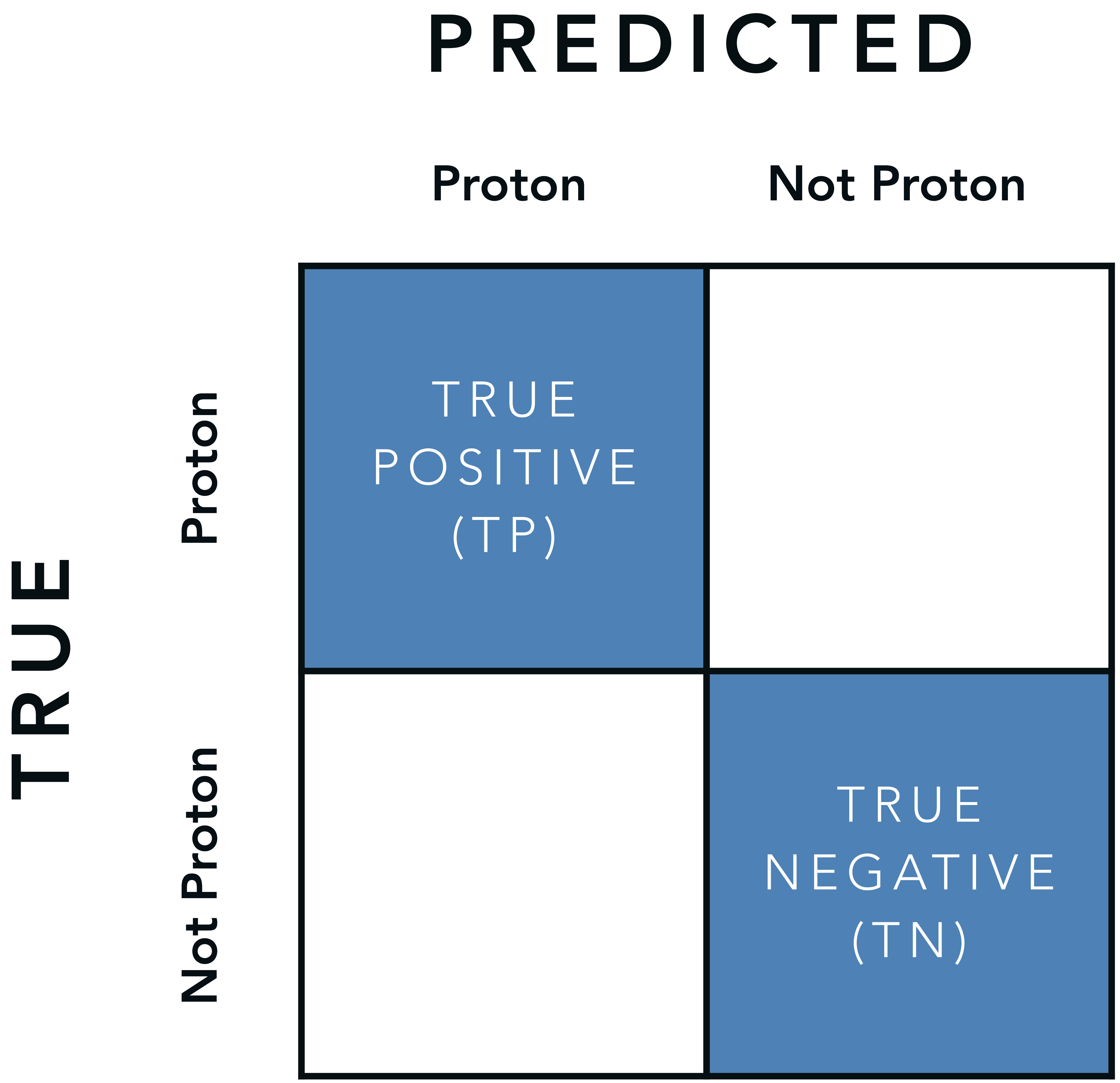
TRUE \ PREDICTED	Proton	Not Proton
Proton	TRUE POSITIVE (TP)	FALSE NEGATIVE (FN)
Not Proton	FALSE POSITIVE (FP)	TRUE NEGATIVE (TN)

$$\text{accuracy} = \frac{TP + TN}{TP + FN + FP + TN}$$

$$\text{precision} = \frac{TP}{TP + FP}$$

$$\text{recall} = \frac{TP}{TP + FN}$$

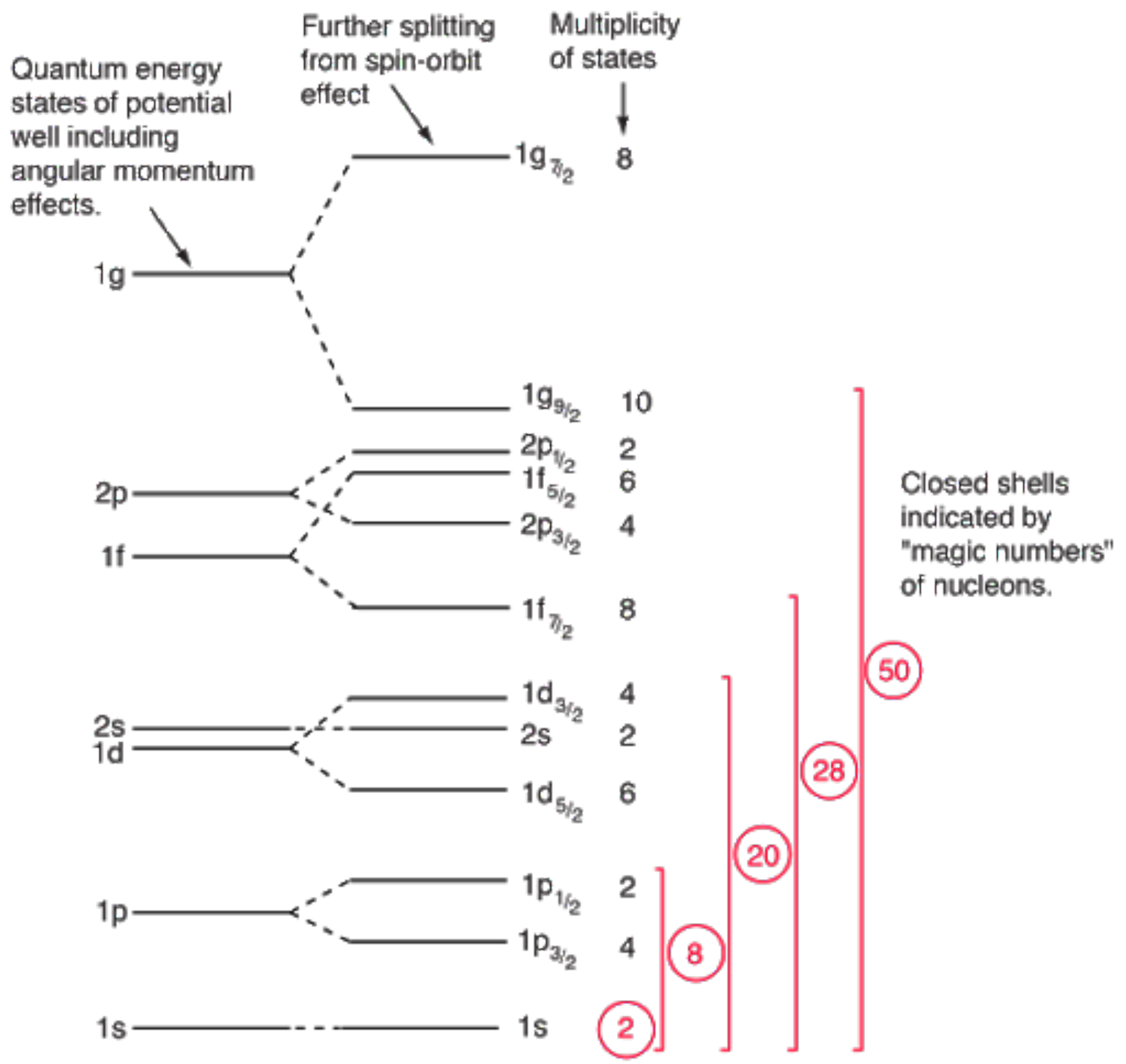
$$\text{F1} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

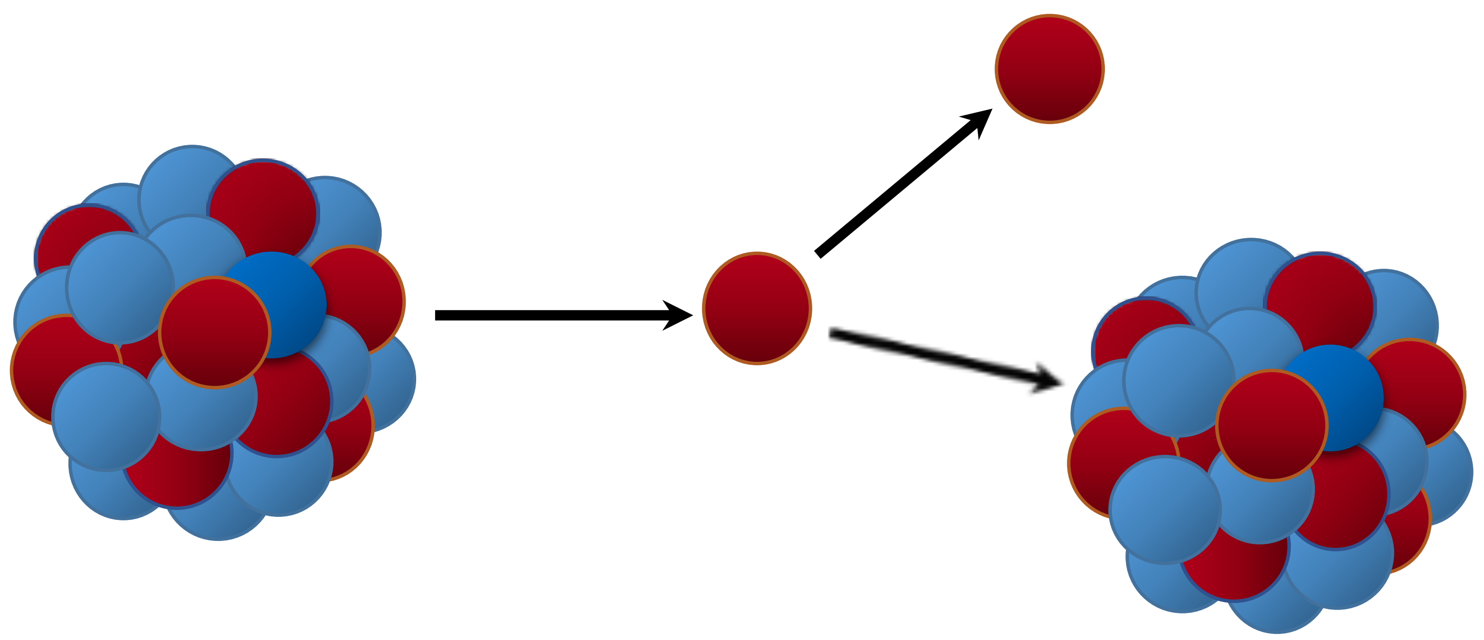


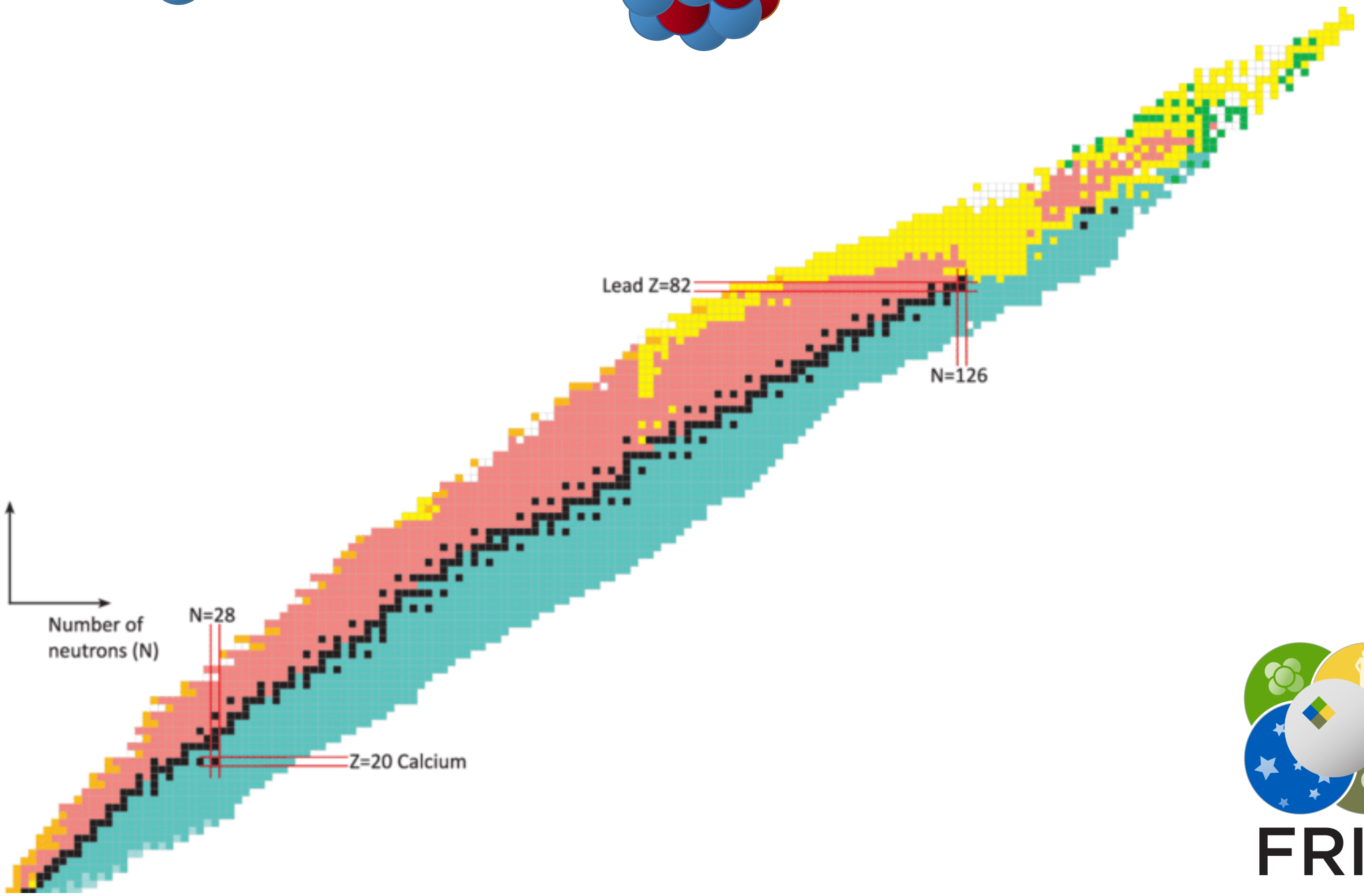
PERFECT MODEL

Application 2: Can we use machine learning to **accurately** classify events in detectors?

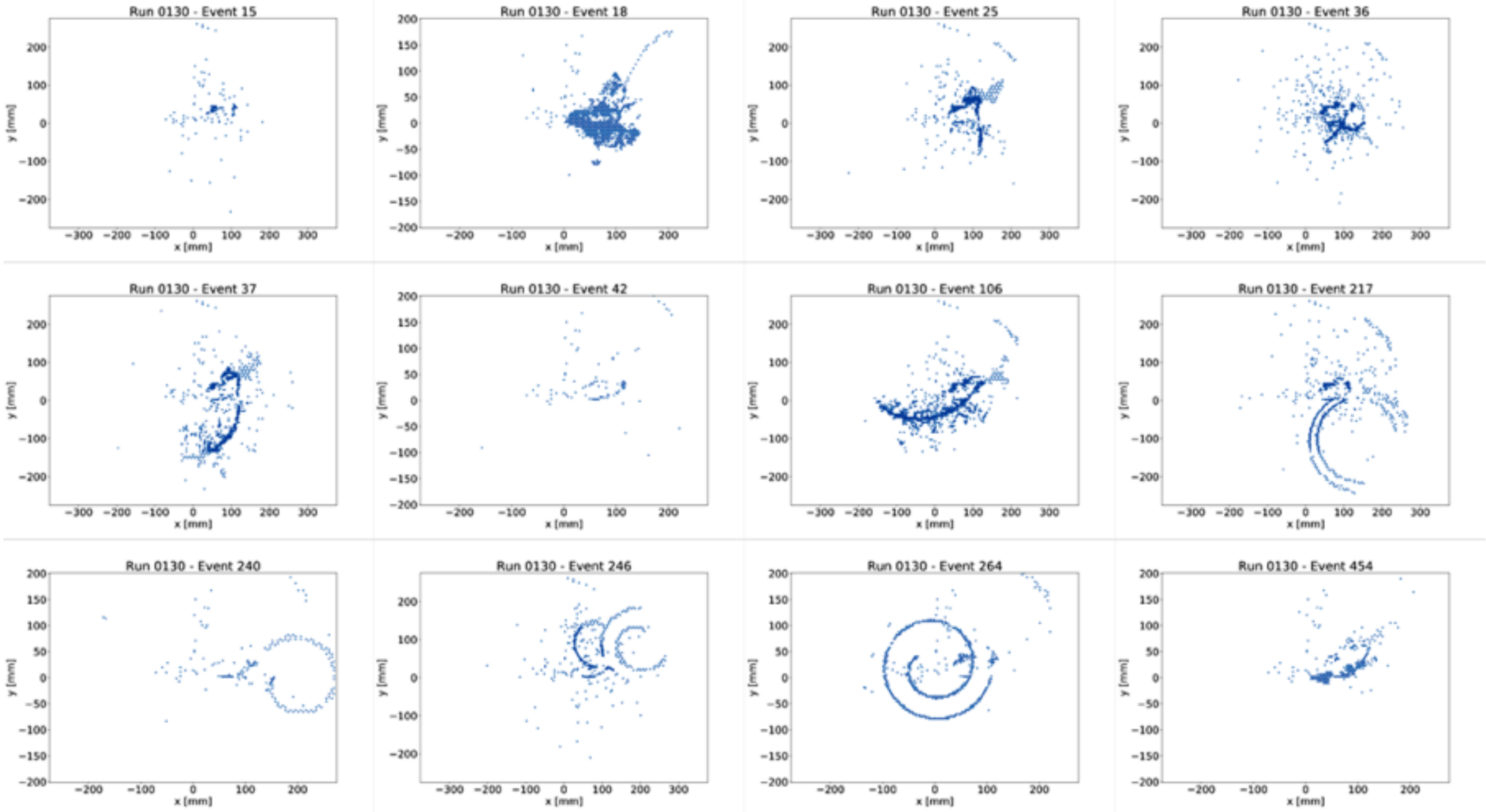
ACTIVE-TARGET TIME PROJECTION CHAMBER (AT-TPC)

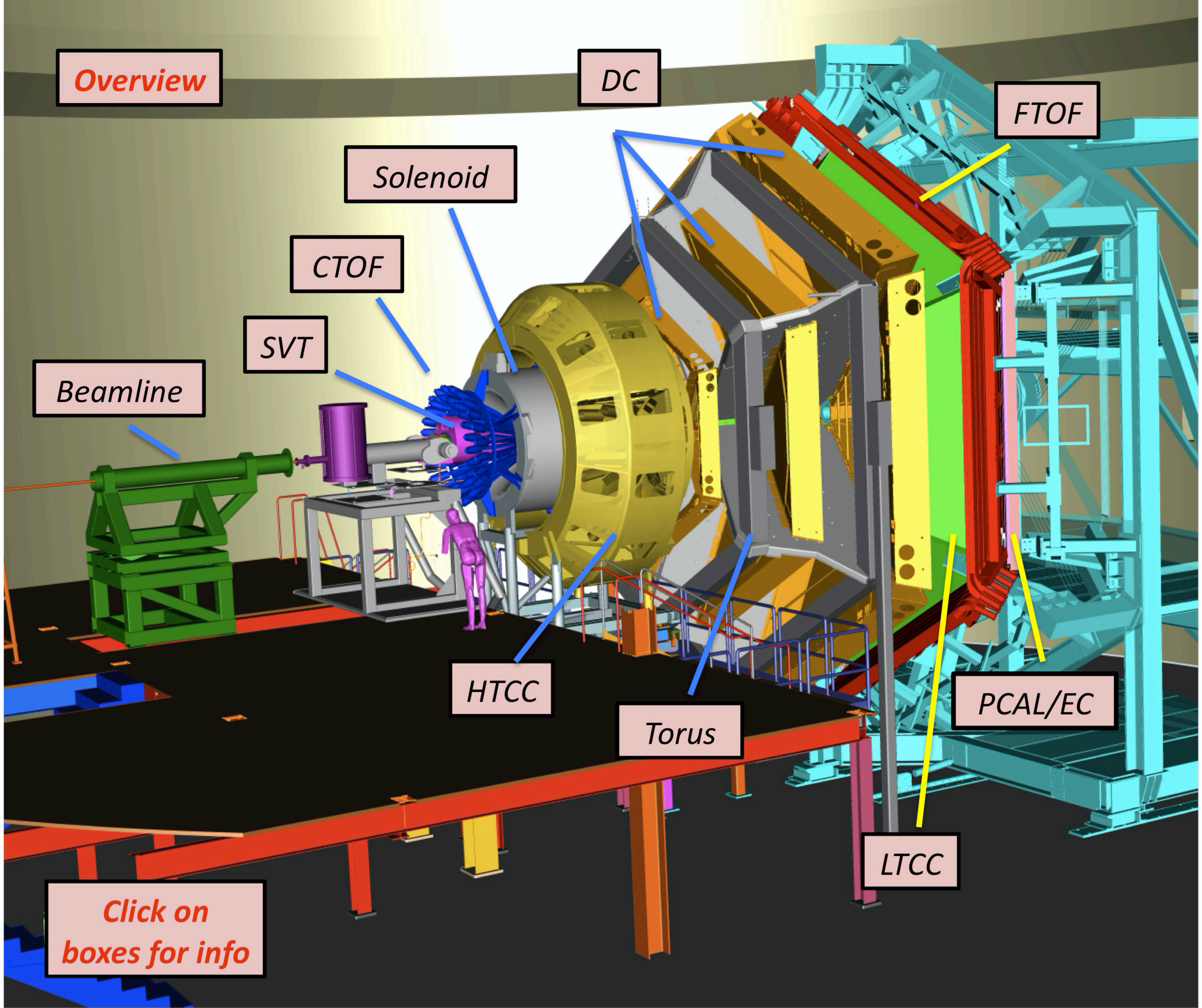






FRIB


Run 0130 - Event 15
Run 0130 - Event 18
Run 0130 - Event 25
Run 0130 - Event 36
Run 0130 - Event 37
Run 0130 - Event 42
Run 0130 - Event 106
Run 0130 - Event 217
Run 0130 - Event 240
Run 0130 - Event 246
Run 0130 - Event 264
Run 0130 - Event 454
x [mm]
y [mm]

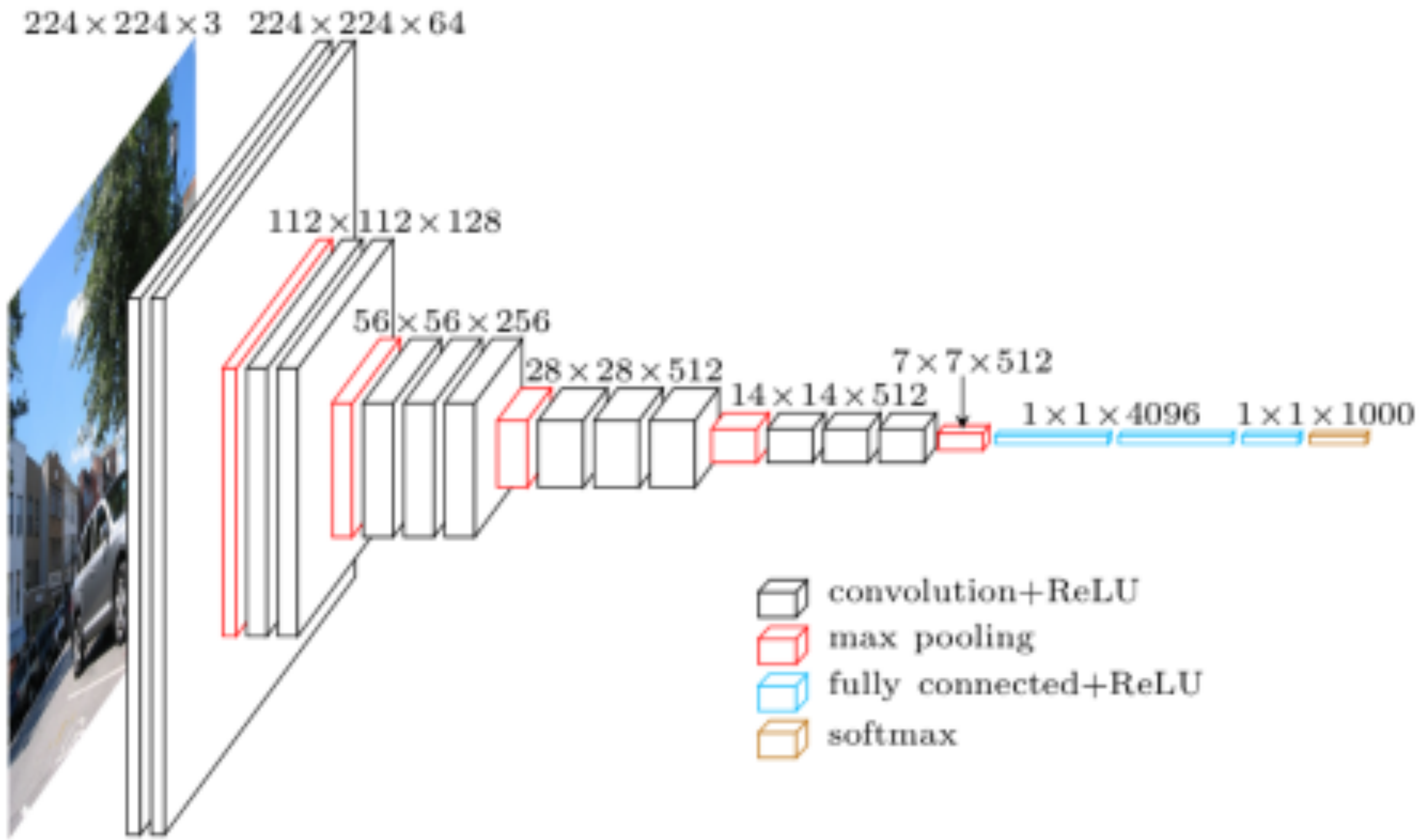

Overview
DC
FTOF
Solenoid
CTOF
SVT
Beamline
HTCC
Torus
PCAL/EC
LTCC
Click on boxes for info

EXPERIMENTAL DATA

AT-TPC

HALL B

VGG16 ARCHITECTURE



PRE-TRAINED ON IMAGENET DATA!

AT-TPC

Experiment	Precision	Recall	F1
Experimental → Experimental	0.96	0.90	0.93
Simulated → Simulated	1.00	1.00	1.00
Simulated → Experimental	0.90	0.60	0.72

HALL B

Precision	Recall	F1
0.97	0.93	0.95

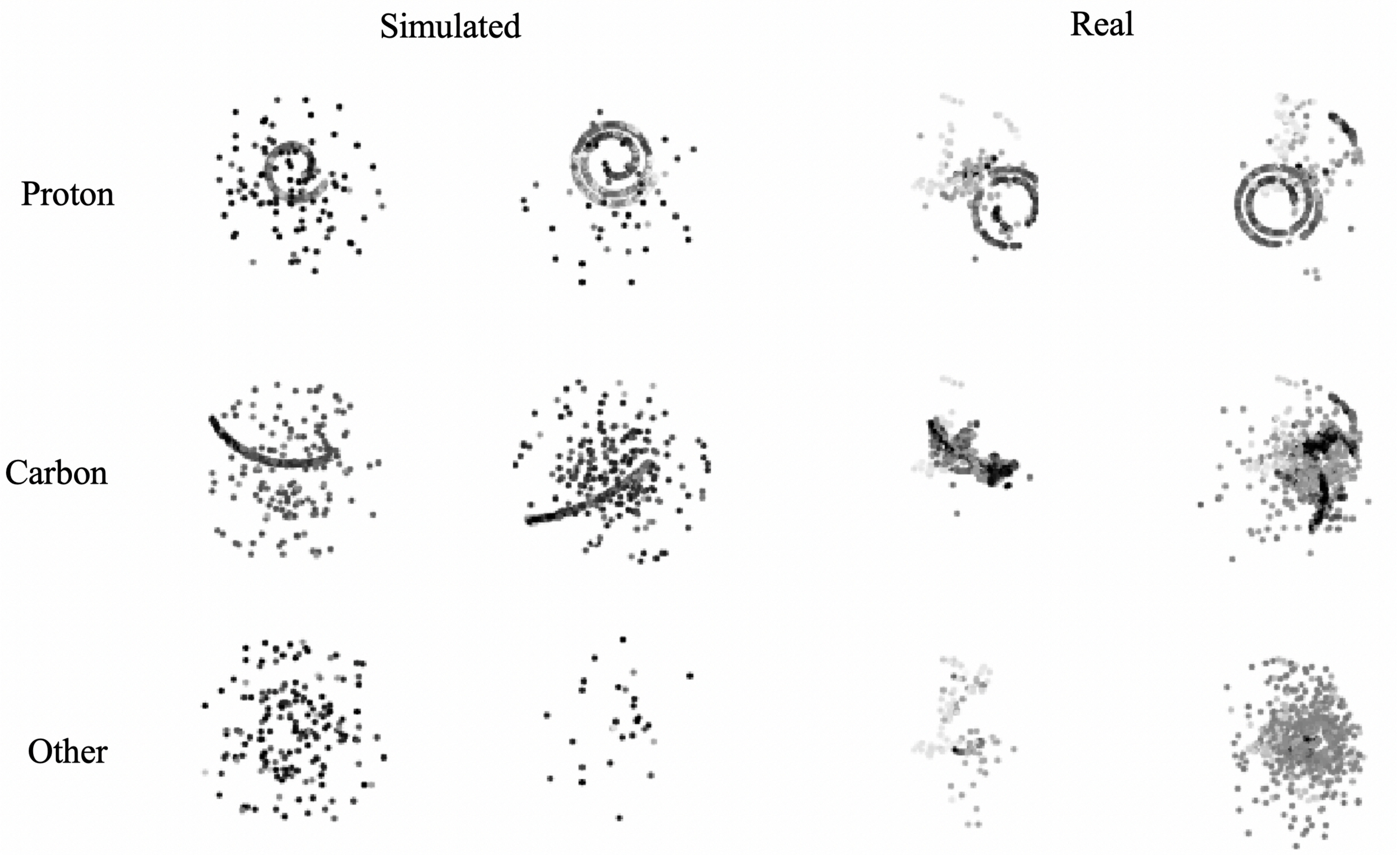
AT-TPC

Experiment	Precision	Recall	F1
Experimental → Experimental	0.96	0.90	0.93
Simulated → Simulated	1.00	1.00	1.00
Simulated → Experimental	0.90	0.60	0.72

HALL B

Precision	Recall	F1
0.97	0.93	0.95

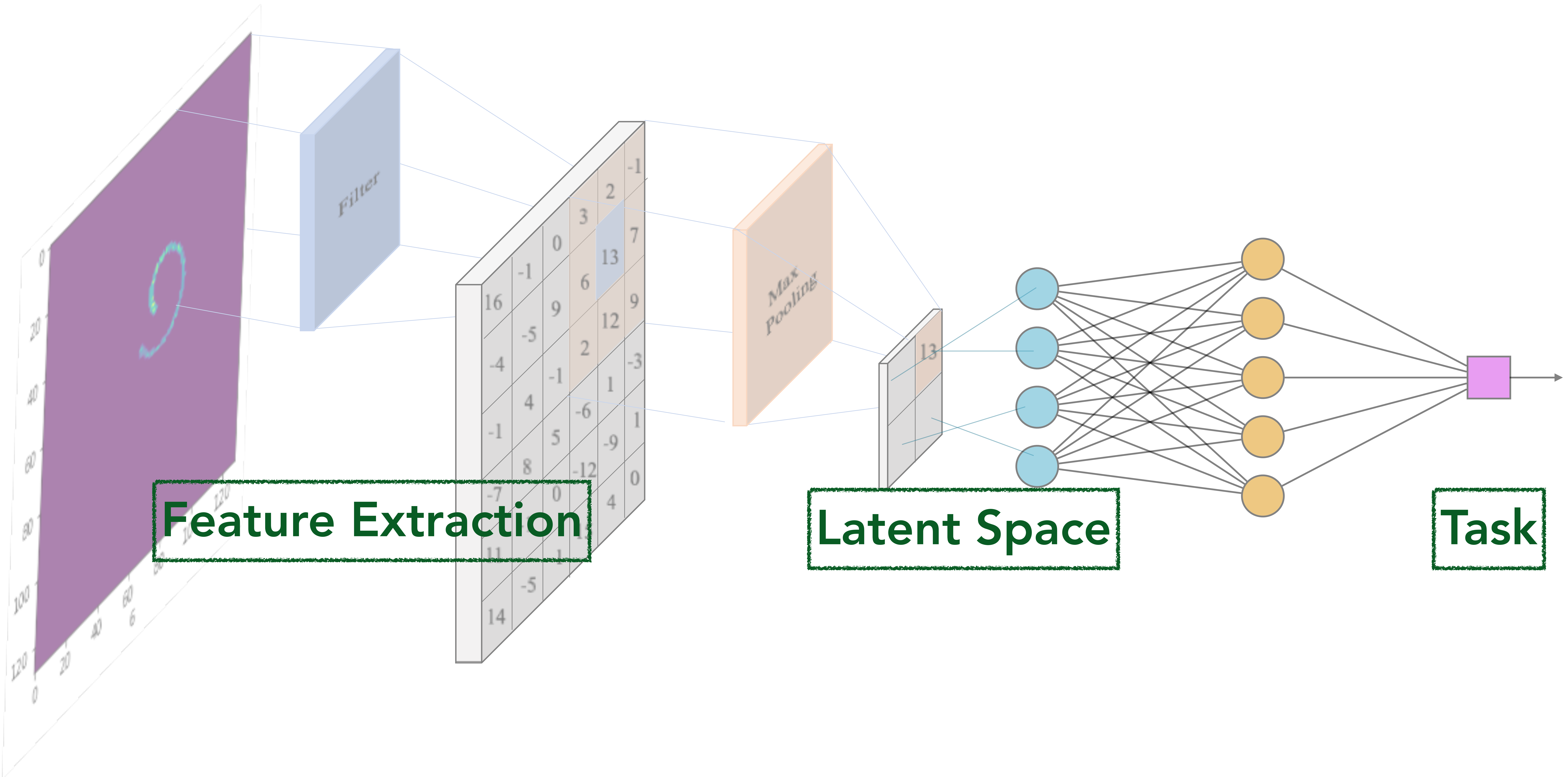
6x faster!


Simulated
Real
Proton
Carbon
Other

MACHINE LEARNING

UNSUPERVISED LEARNING

CONVOLUTIONAL NEURAL NETWORKS



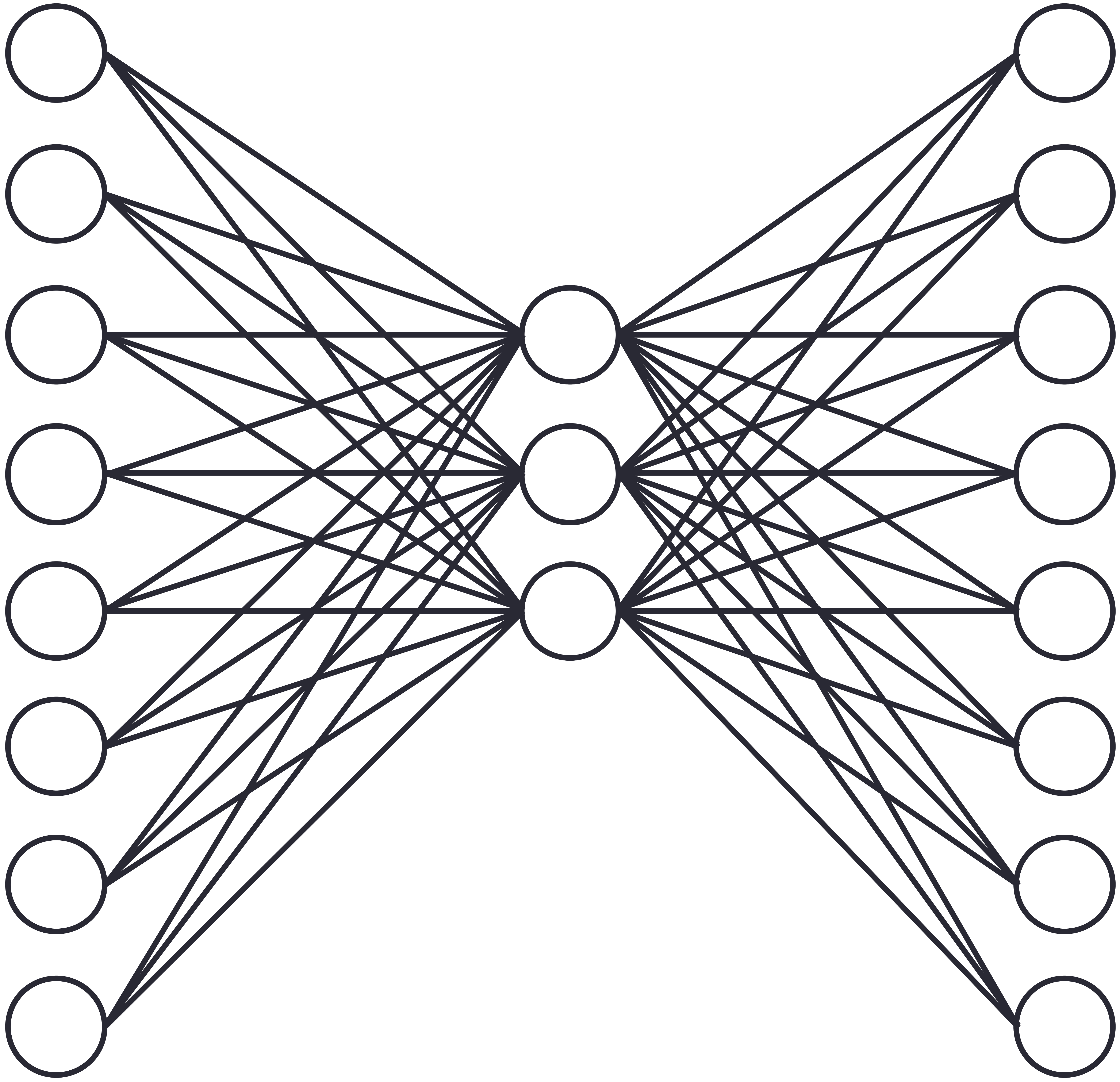
CLUSTERING — KMEANS

Goal: minimize pairwise distances between points in *same* cluster

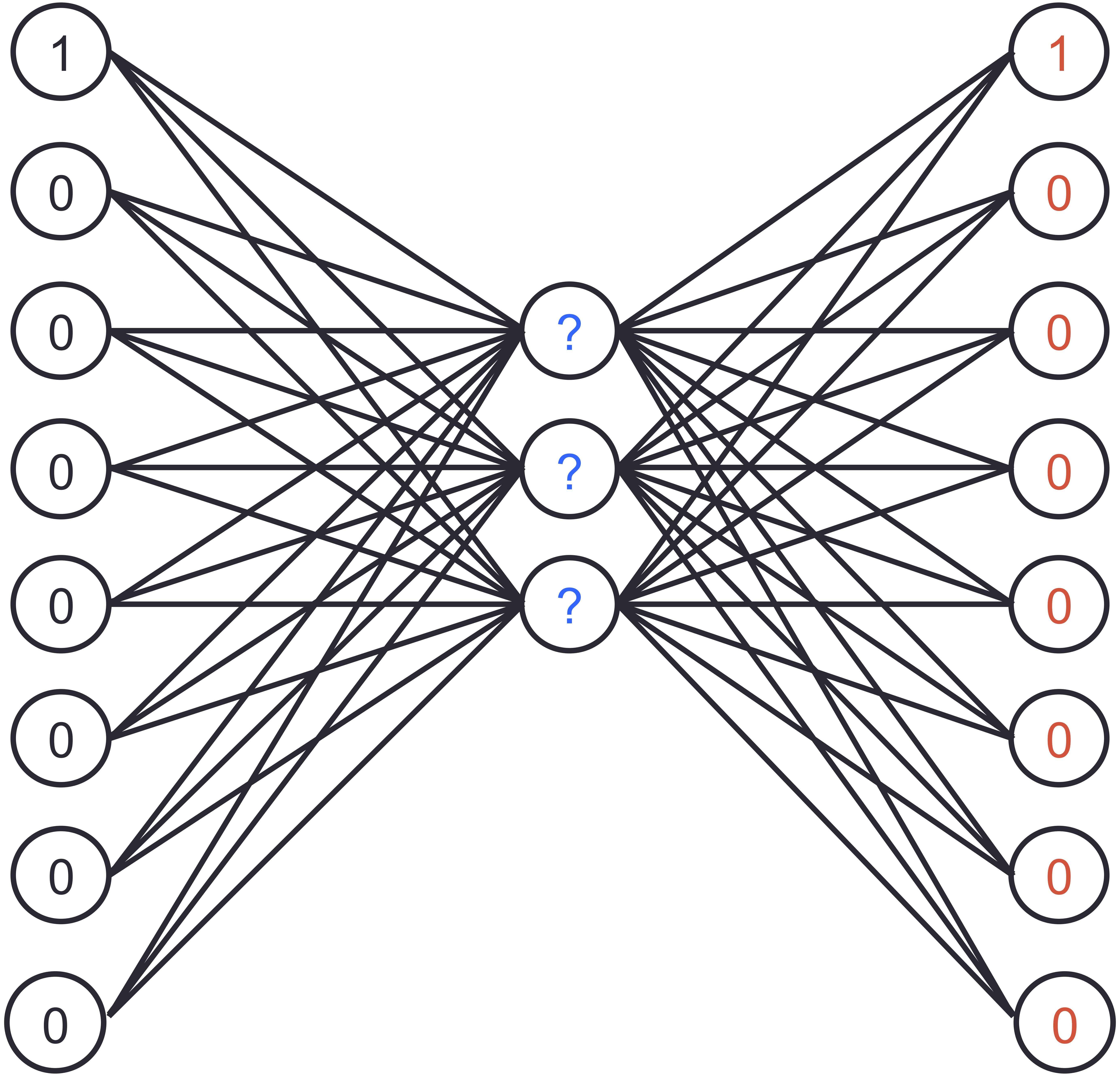
$$\min \sum_{i=1}^{k} \frac{1}{2N} \sum_{x,y,x\neq y}^{N} (\vec{x} - \vec{y})^2$$

Goal: maximize pairwise distances between points in *different* clusters

CLUSTERING — KMEANS

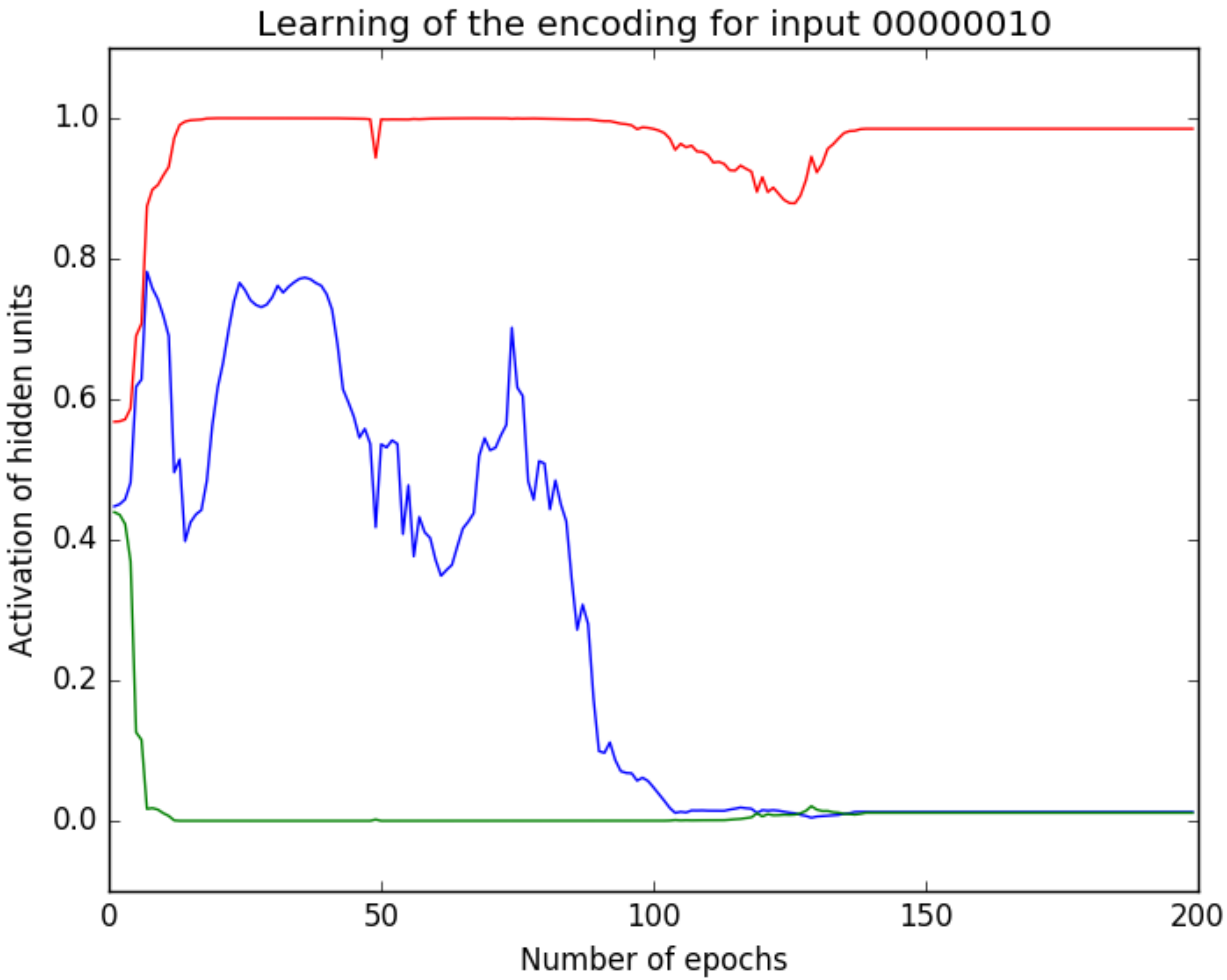


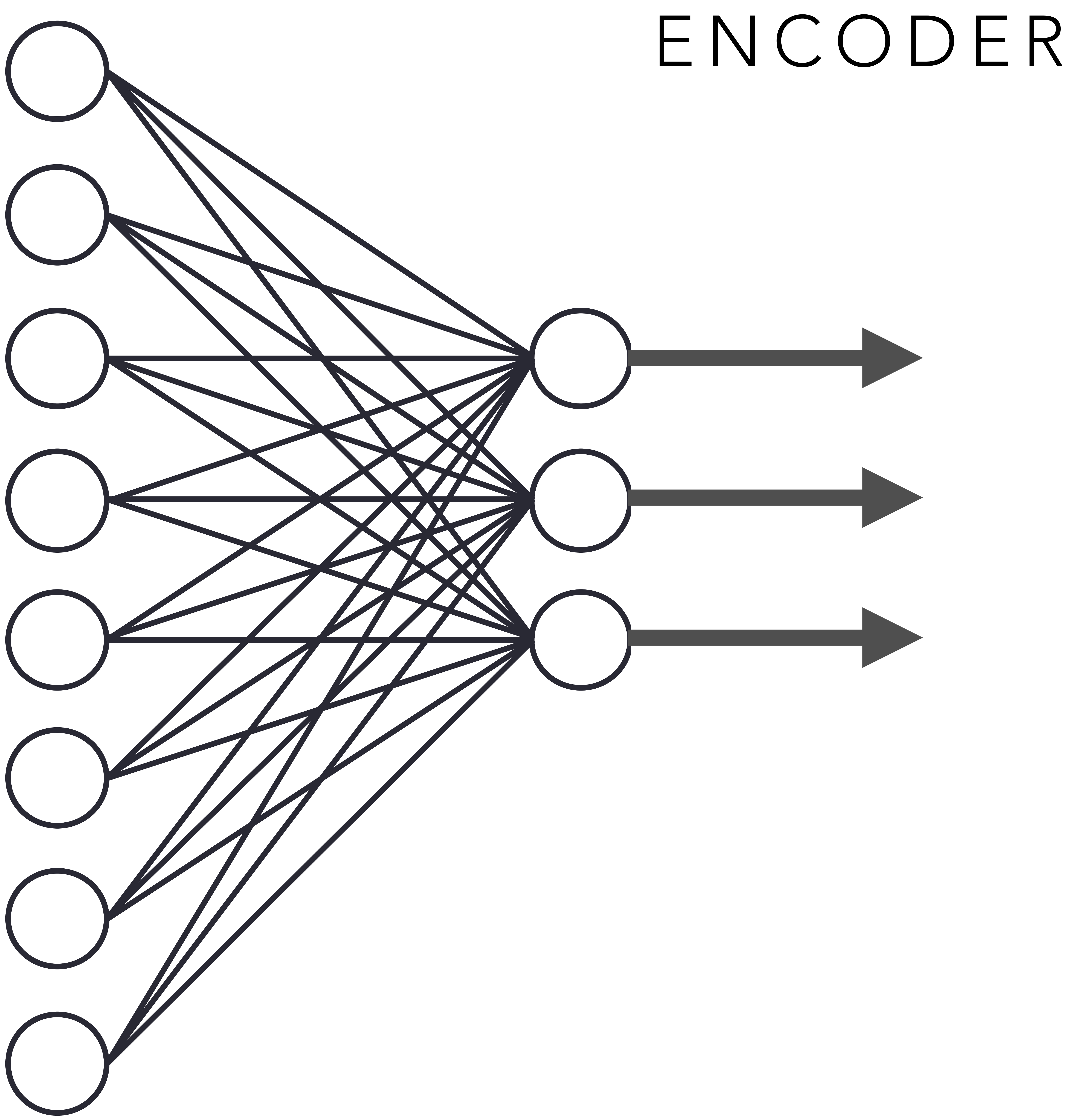
Input	Output
10000000	10000000
01000000	01000000
00100000	00100000
00010000	00010000
00001000	00001000
00000100	00000100


1
0
0
0
0
0
0
0
?
?
?
1
0
0
0
0
0
0
0

Input	Output
10000000	10000000
01000000	01000000
00100000	00100000
00010000	00010000
00001000	00001000
00000100	00000100

Input	A1	A2	A3	Output
10000000	0.9911	0.9869	0.0093	10000000
01000000	0.9892	0.0095	0.0124	01000000
00100000	0.0094	0.0283	0.0122	00100000
00010000	0.9840	0.9836	0.9900	00010000
00001000	0.0139	0.9904	0.0186	00001000
00000100	0.0128	0.9805	0.9868	00000100


Learning of the encoding for input 00000010
Activation of hidden units
Number of epochs
0.0
0.2
0.4
0.6
0.8
1.0
0
50
100
150
200

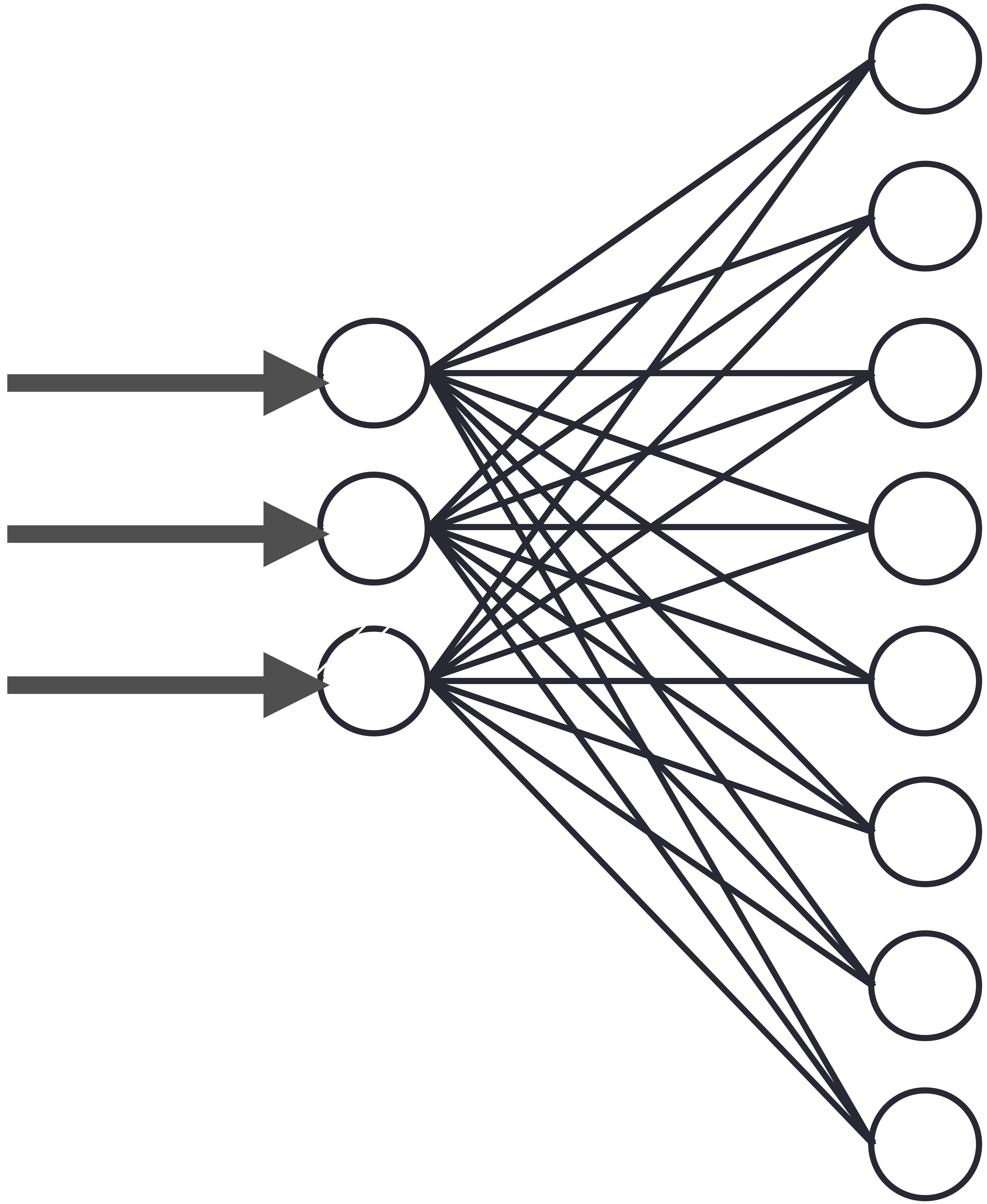

ENCODER

GENERATIVE MODELS

MICHELLE KUCHERA
DAVIDSON COLLEGE

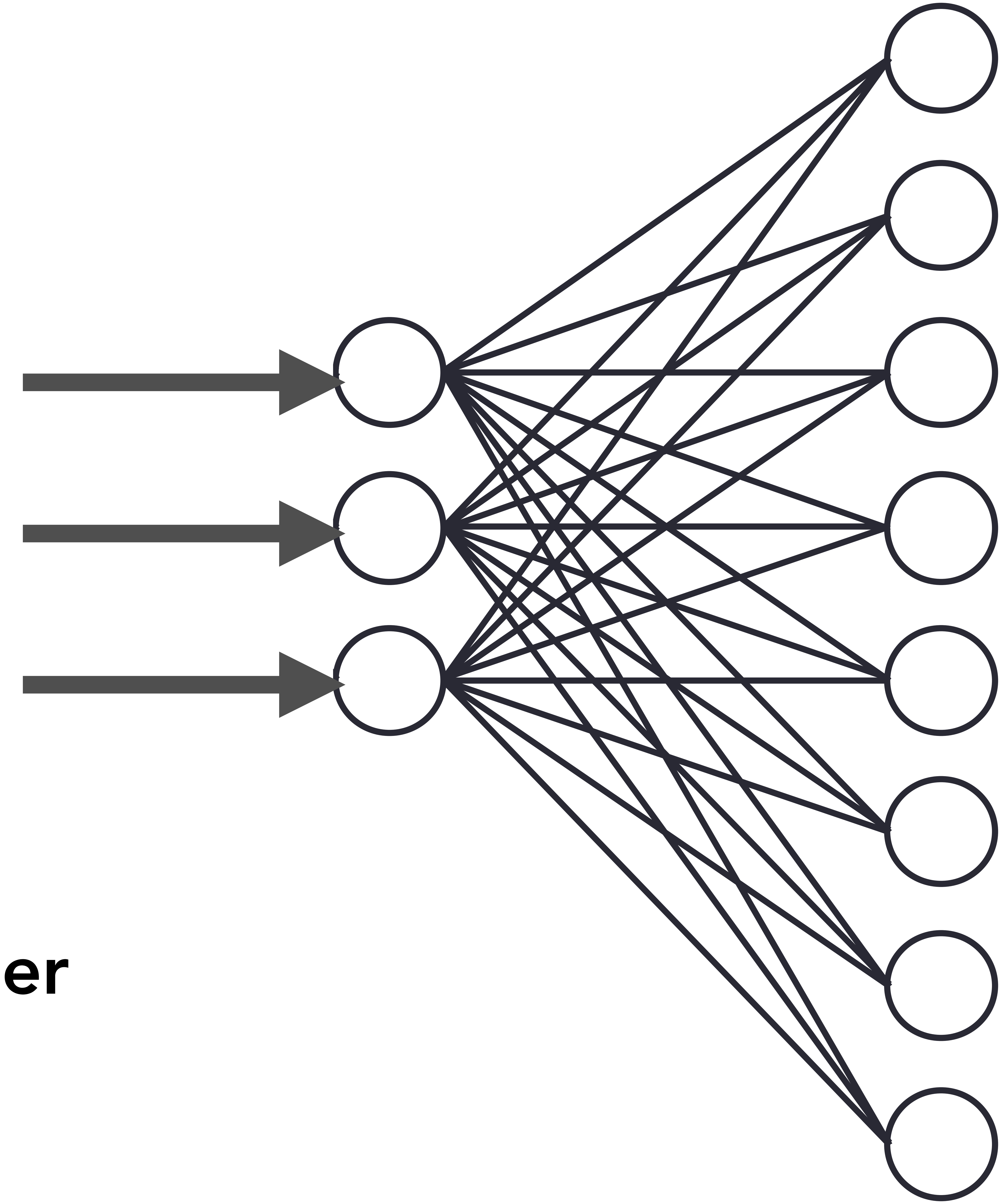
ECT* TALENT SUMMER SCHOOL
02 JULY 2020

DECODER



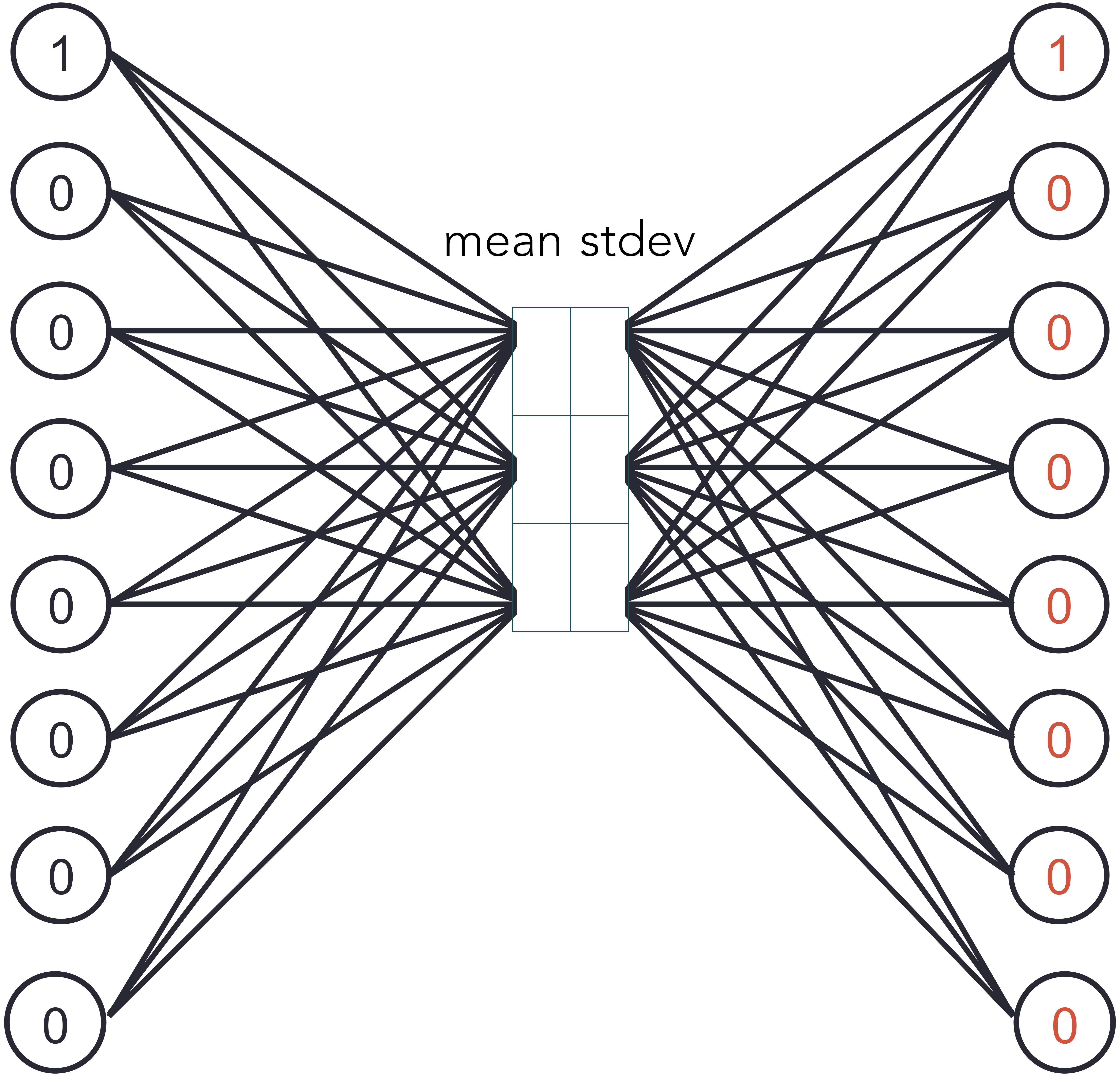
DECODER

How do we know that we are providing a latent vector that represents those seen in training?

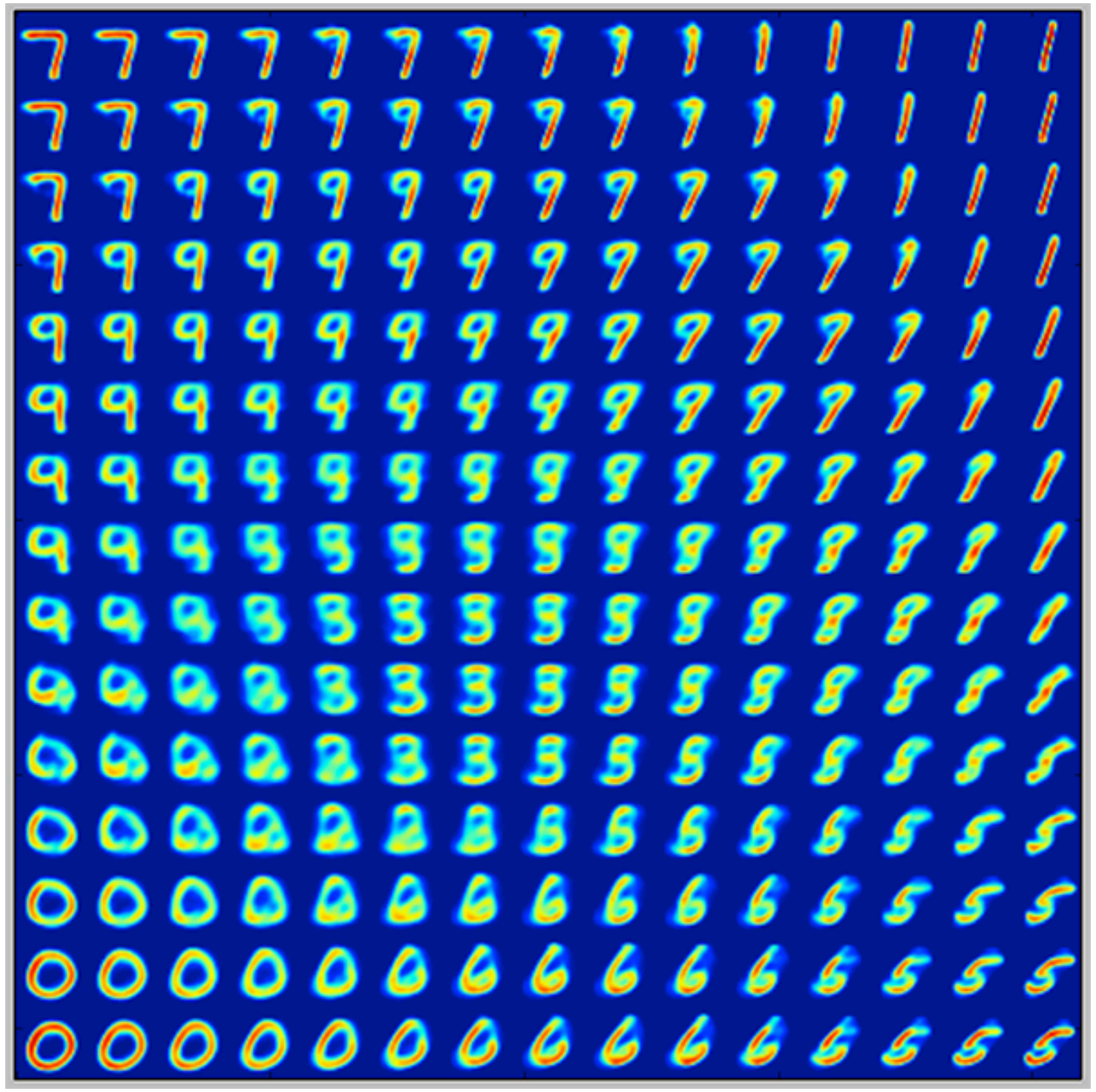


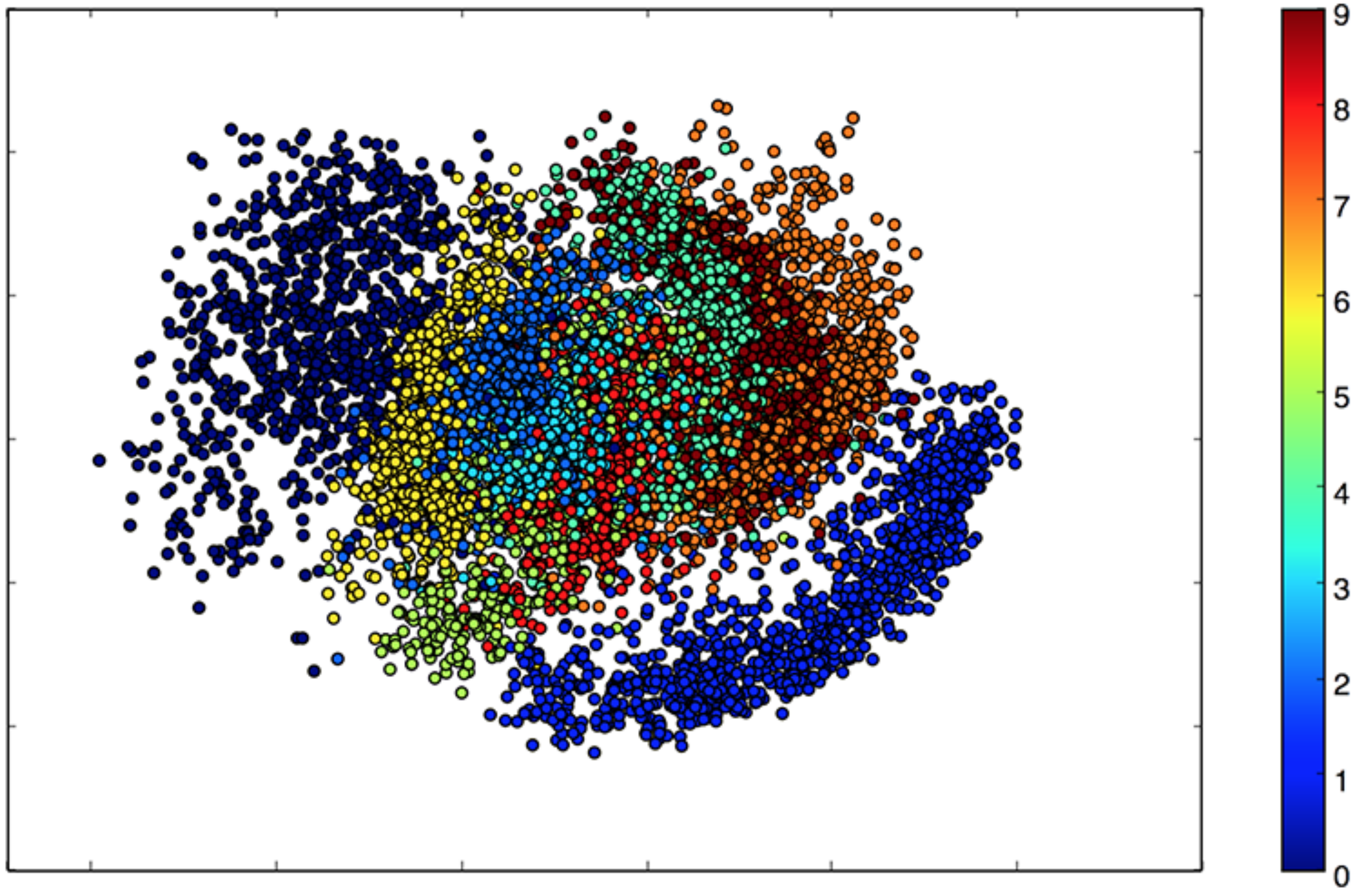
Variational Autoencoder

Encode to two outputs for each latent dimension: mean and stdev



Sample similar points in latent space, decode, and compare with regularization





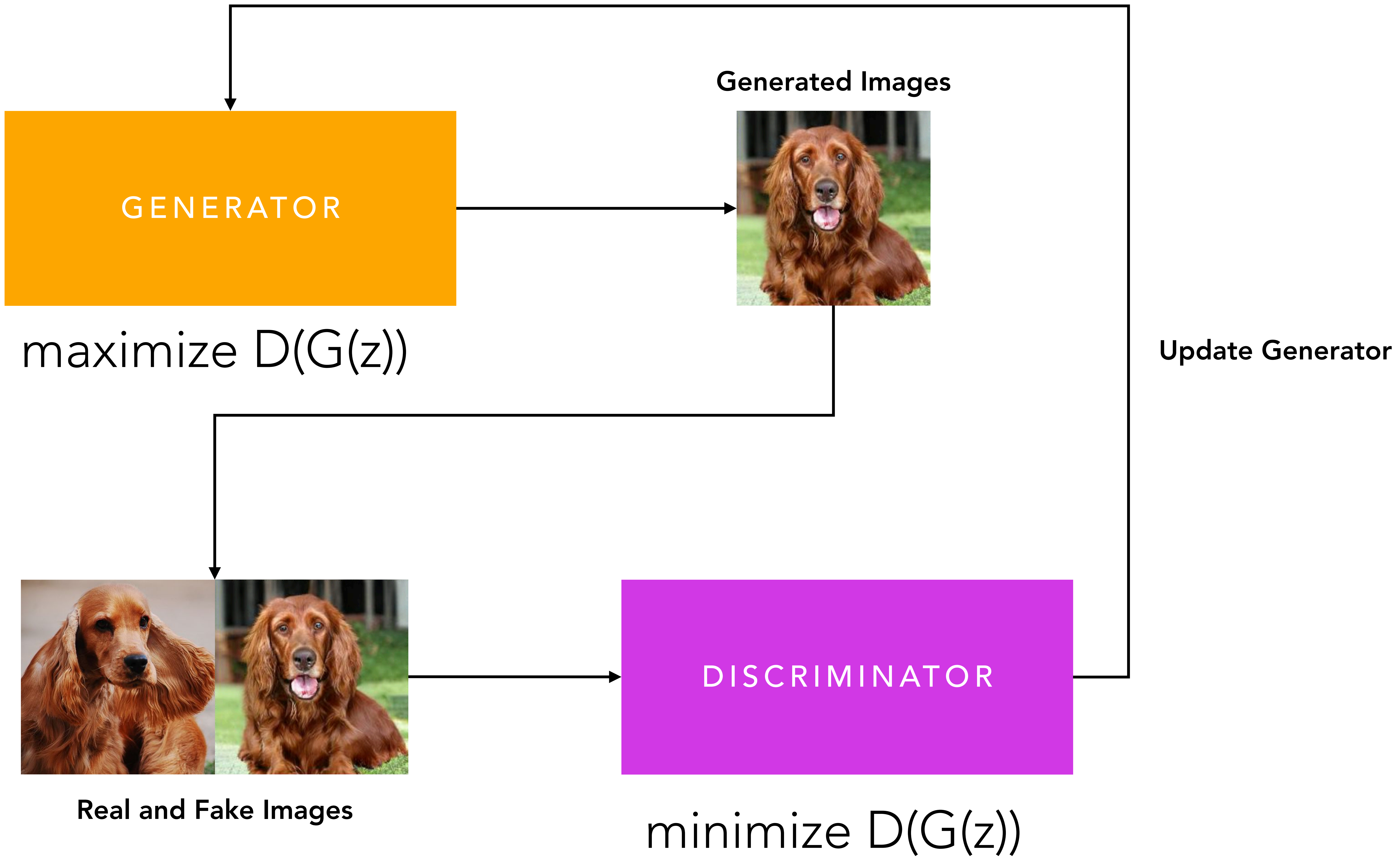
https://blog.keras.io/building-autoencoders-in-keras.html

GENERATIVE MODELS

SIMULATION

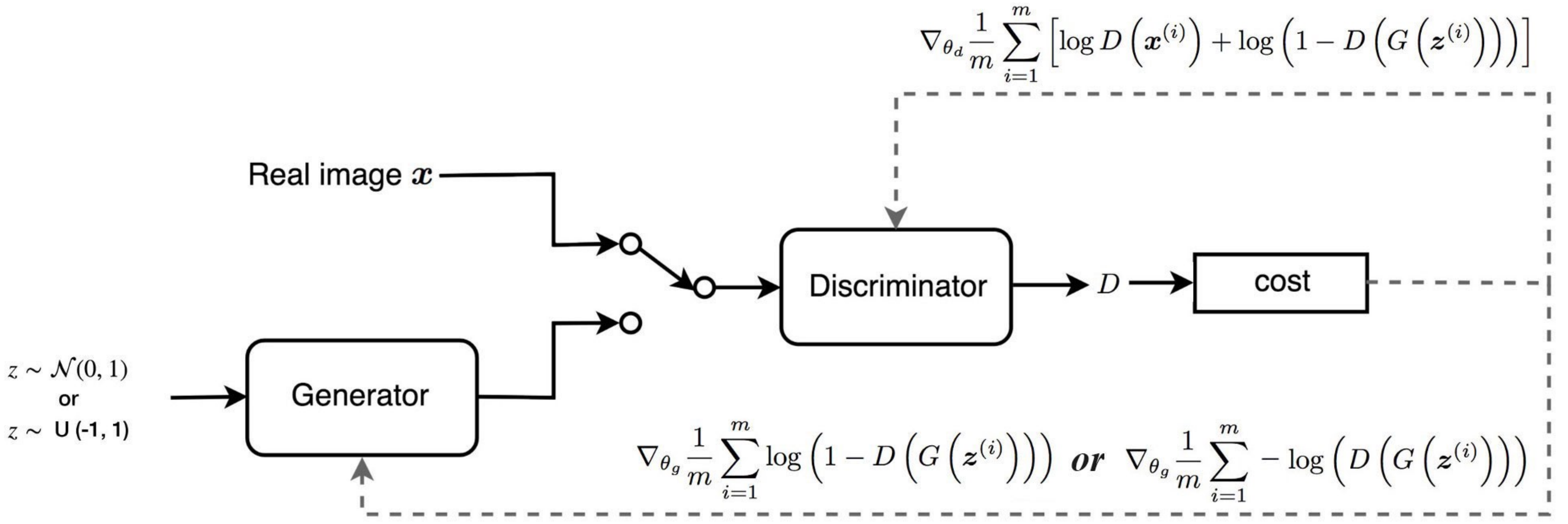
GENERATIVE ADVERSARIAL NETWORKS (GANS)

SIMULATION

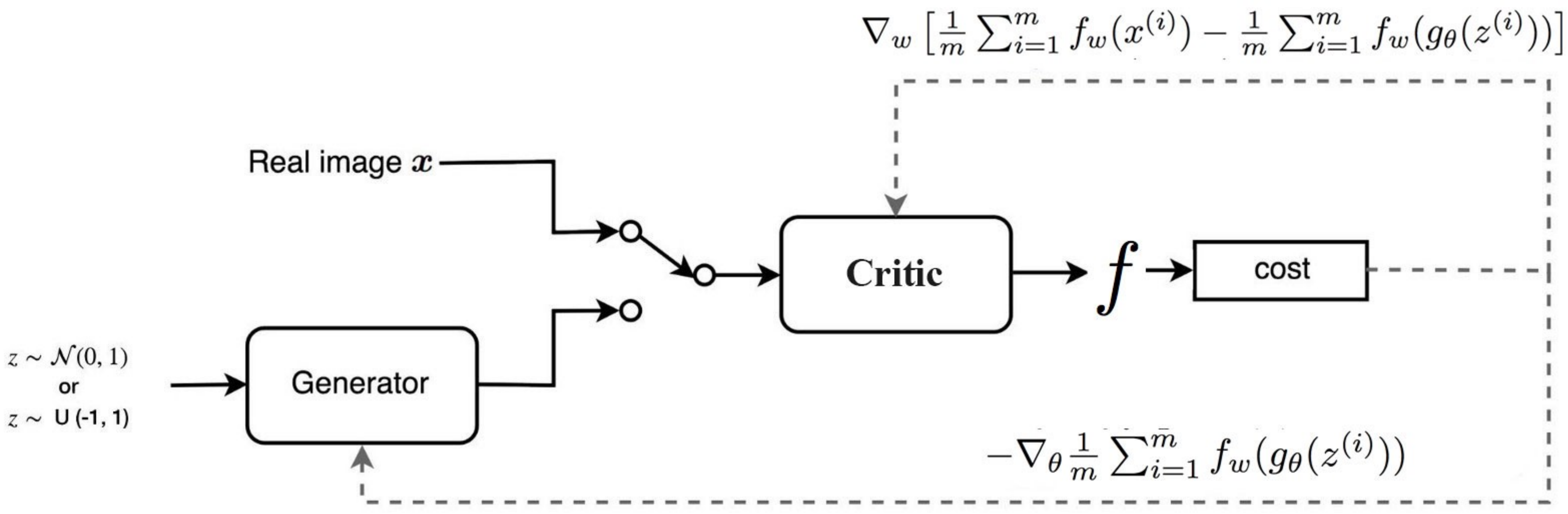


PHOTOS FROM KATRINA S; ANDREW B ET. AL., *BIGGAN*.

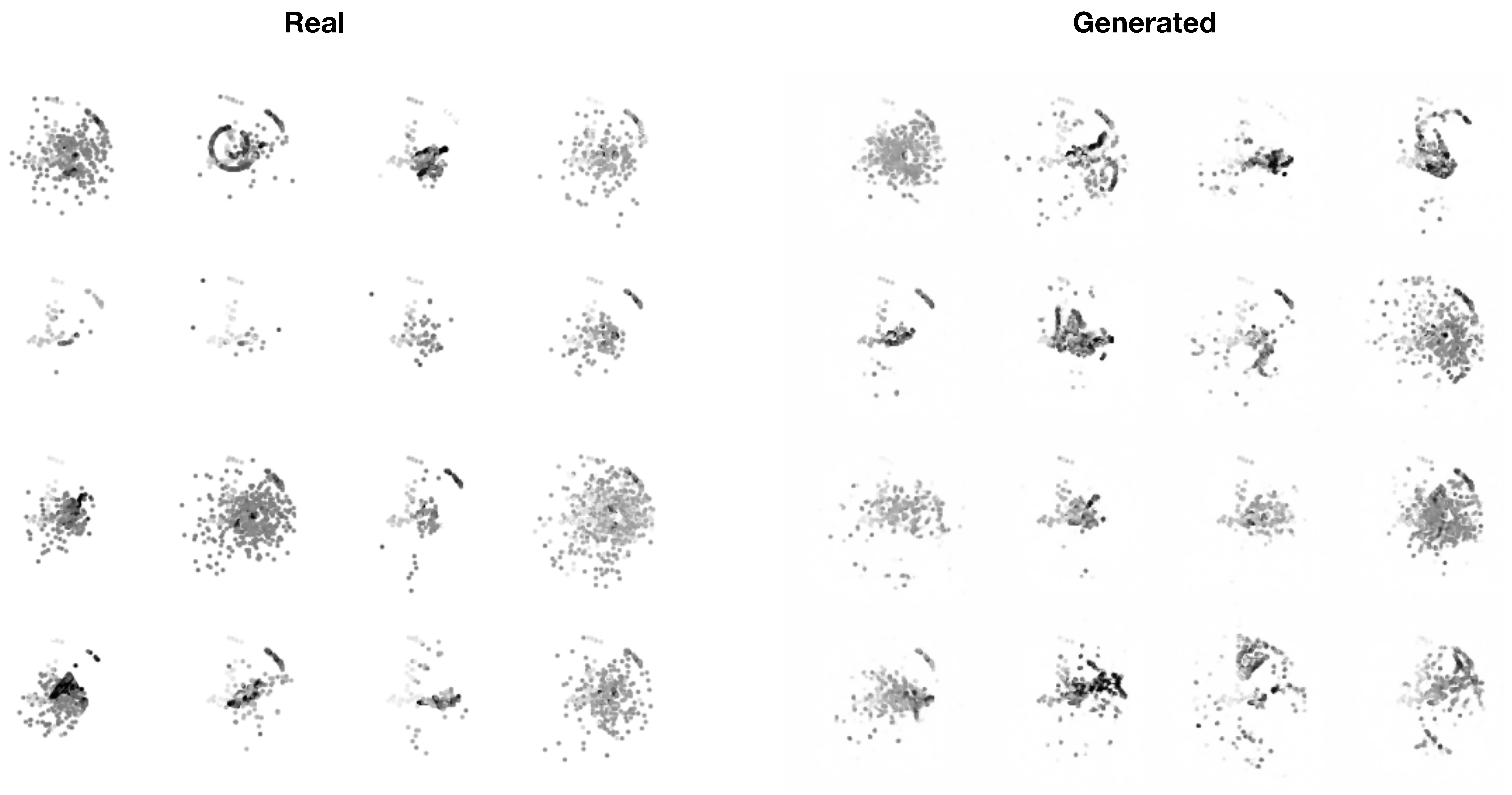
GAN (DCGAN)

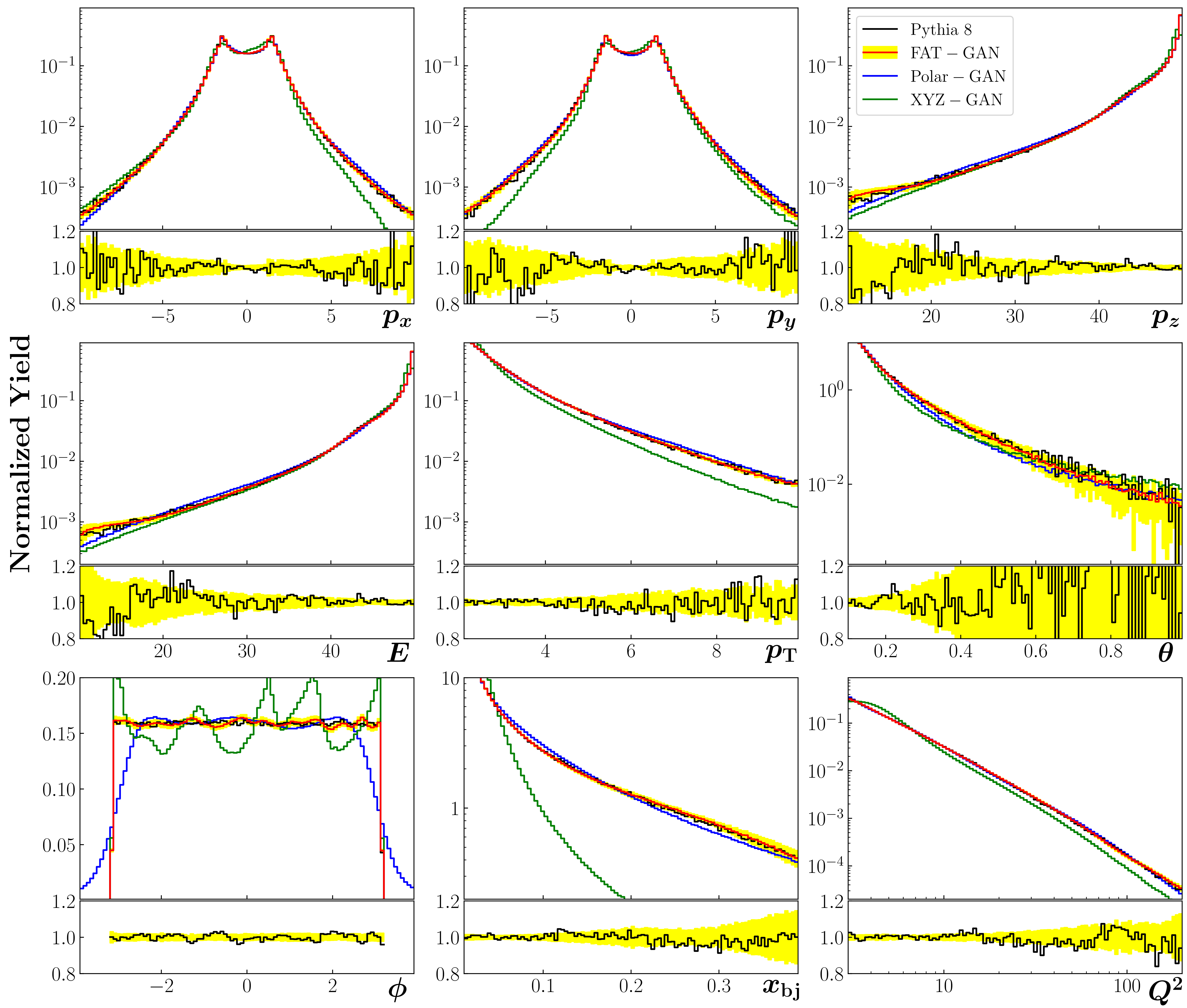


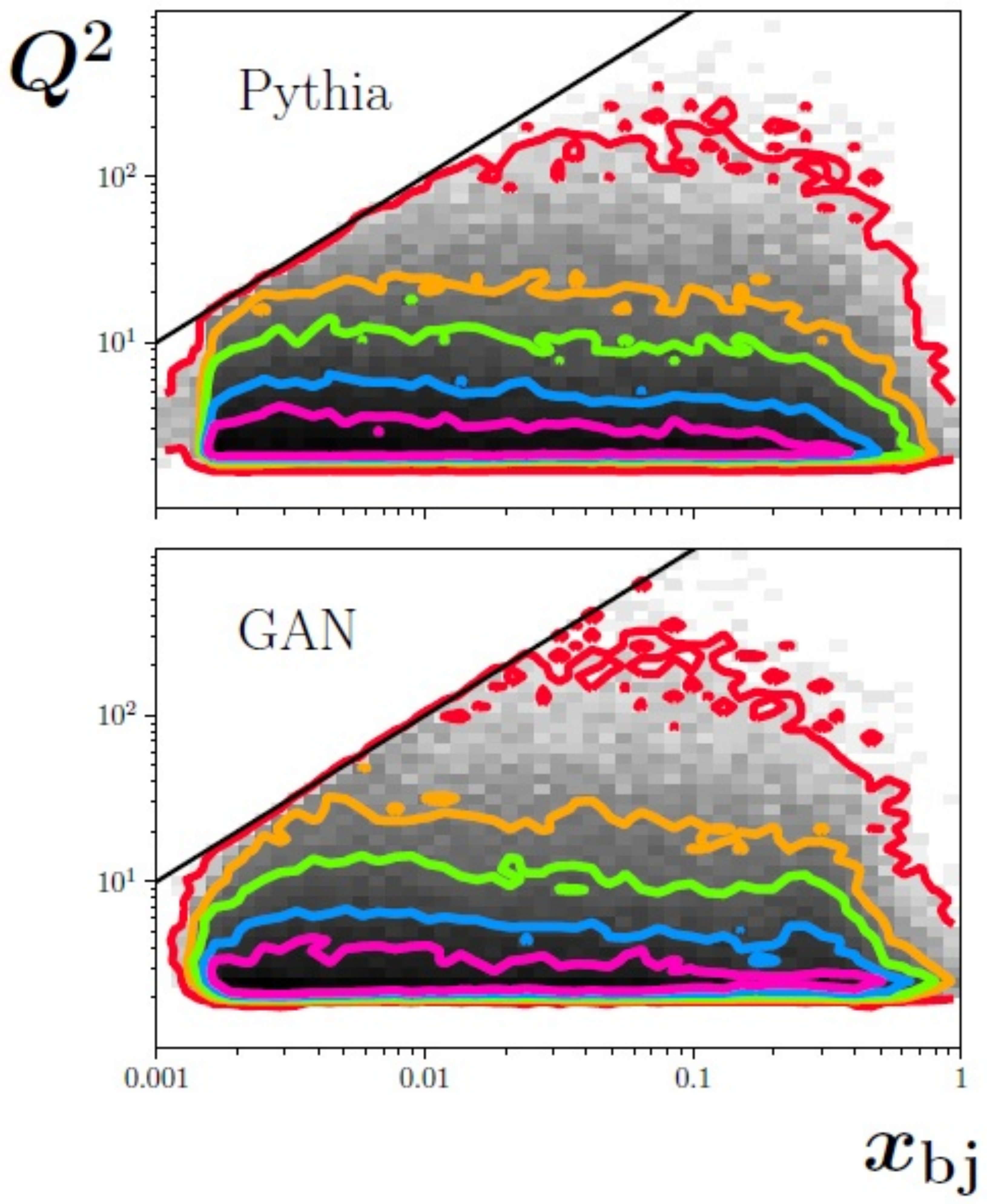
WGAN



Application 3: Can we use machine learning to **simulate data**?

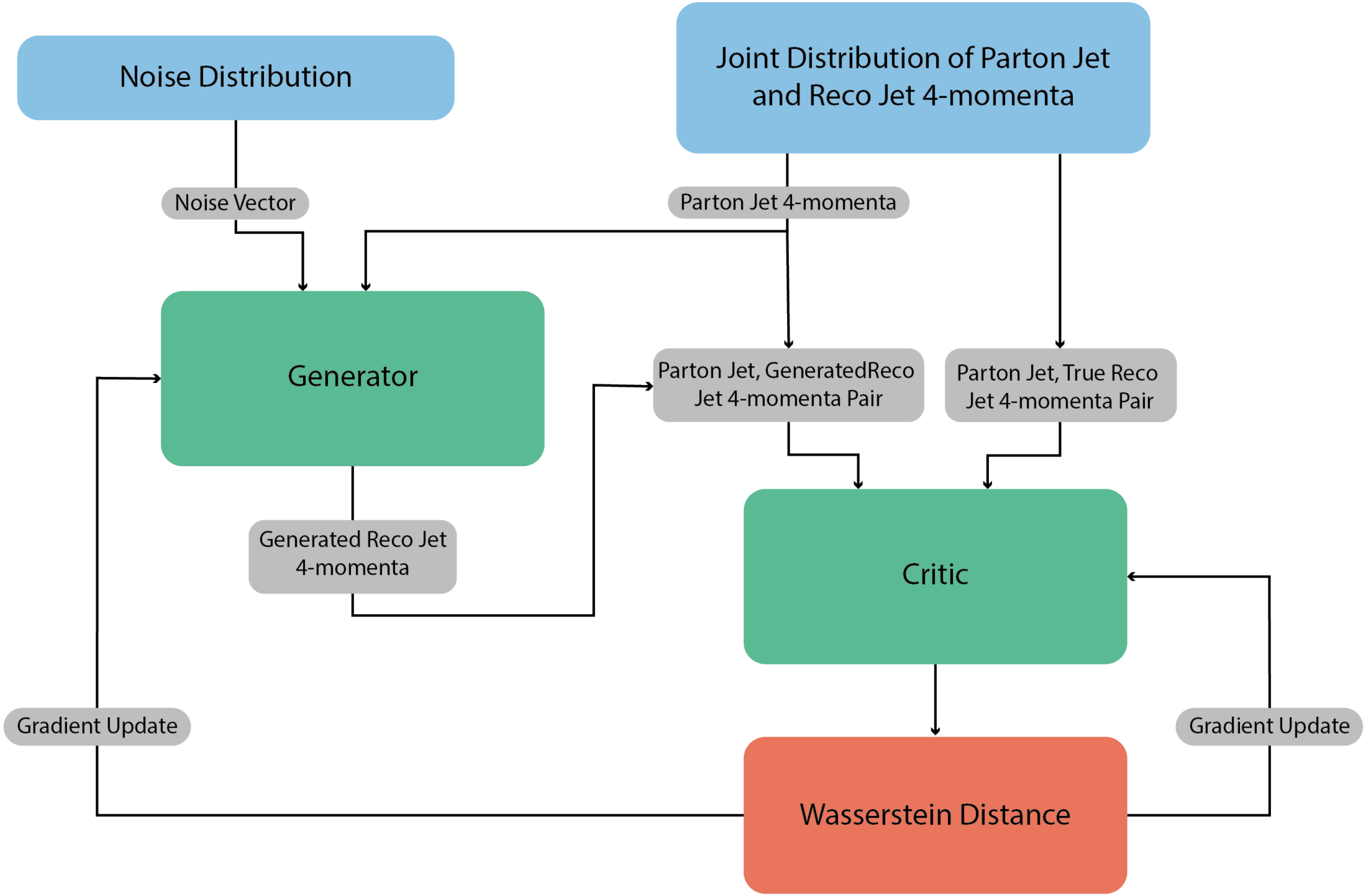

Real
Generated





Simulation of electron-proton scattering events by a Feature-Augmented and Transformed Generative Adversarial Network (FAT-GAN), Y. Alanazi, N. Sato, T. Liu, W. Melnitchouk, M. P. Kuchera, E. Pritchard, M. Robertson, R.R. Strauss, L. Velasco, Y. Li. *accepted*. 30th International Joint Conference on Artificial Intelligence (2021).

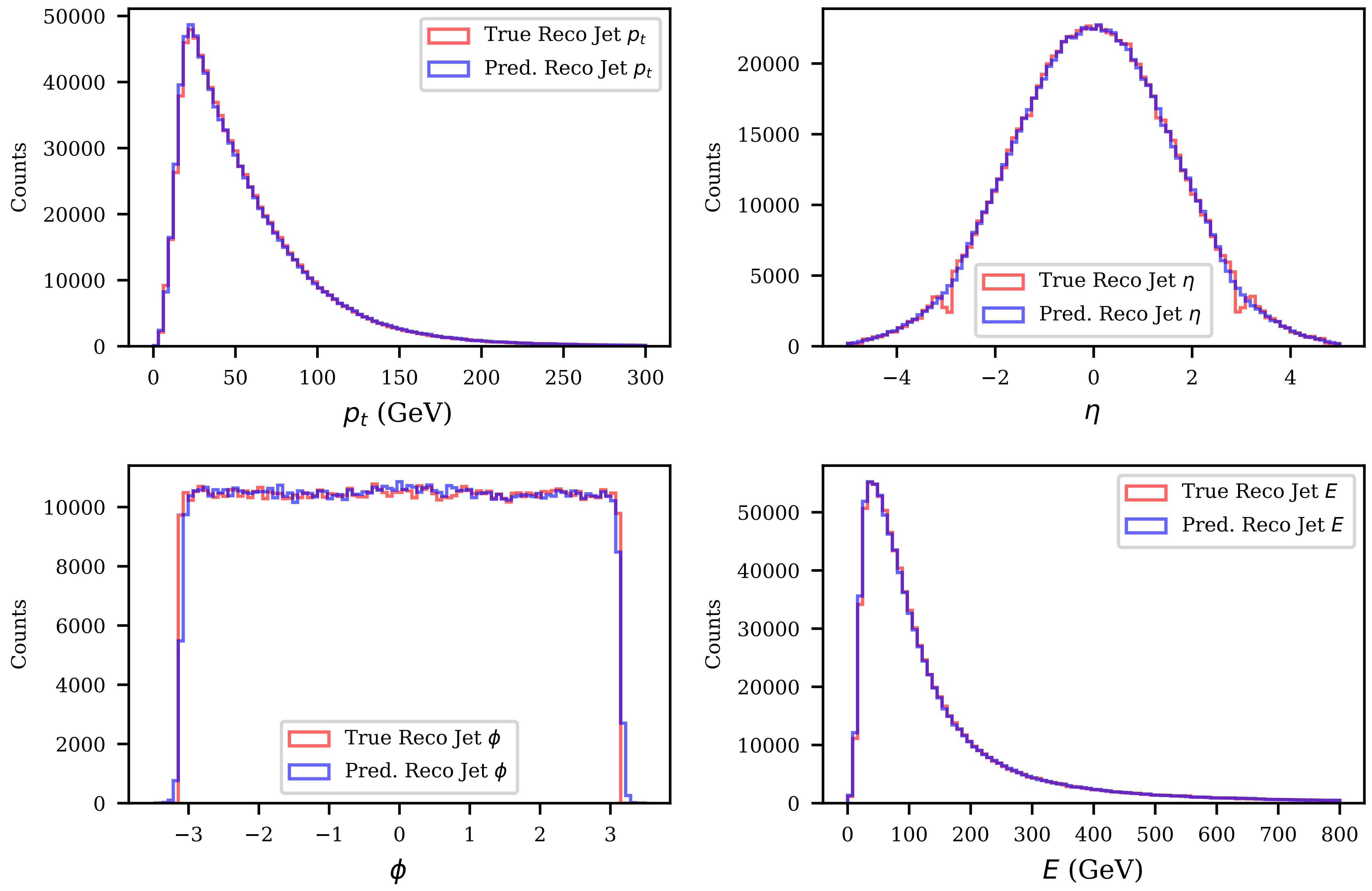
CONDITIONAL GAN



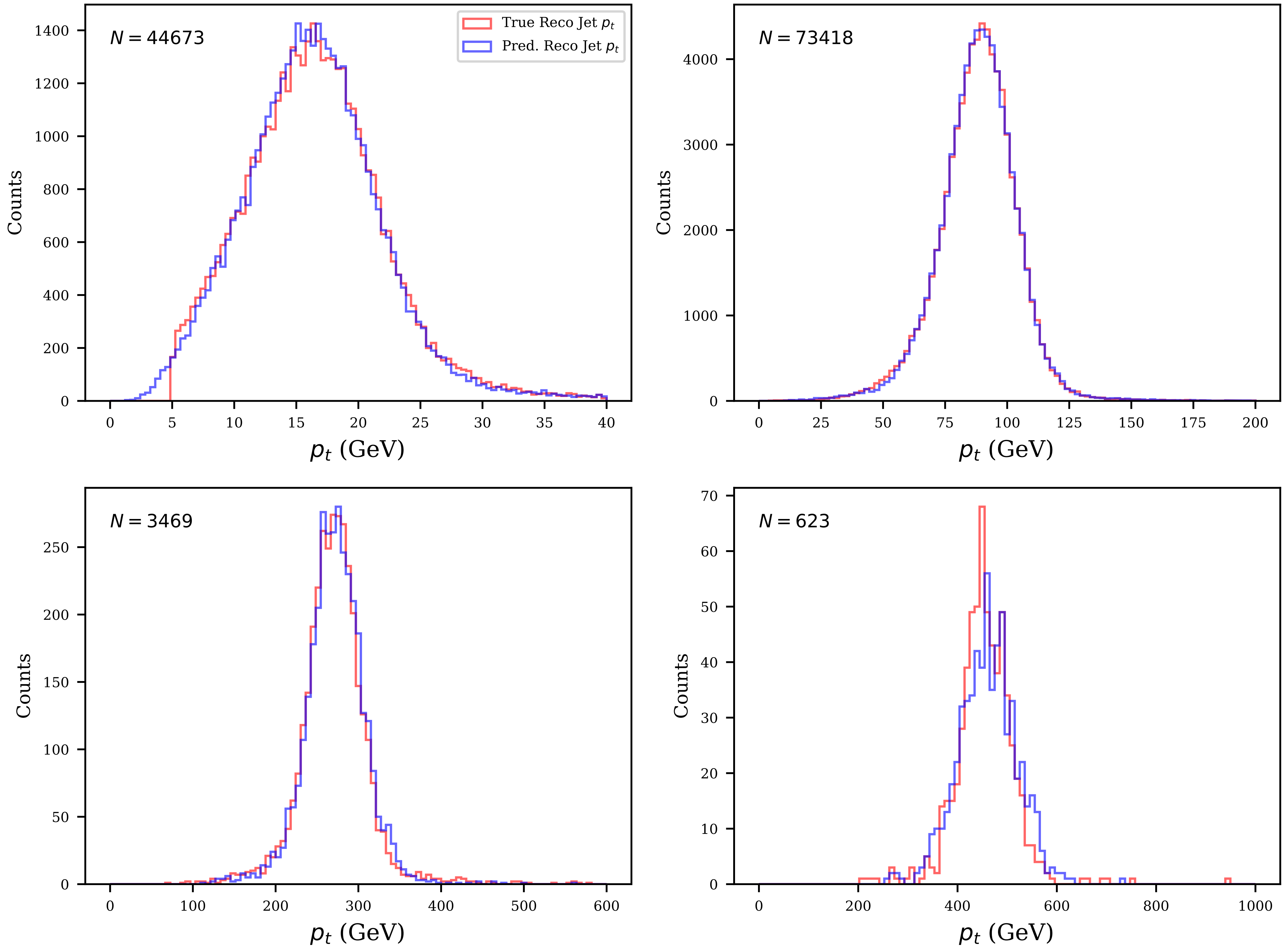
Conditional Wasserstein Generative Adversarial Networks for Fast Detector Simulation. John Blue, Braden Kronheim, Michelle Kuchera, Raghuram Ramanujan
accepted. Computational High Energy and Nuclear Physics (2021)

CONDITIONAL GAN

Total Distributions



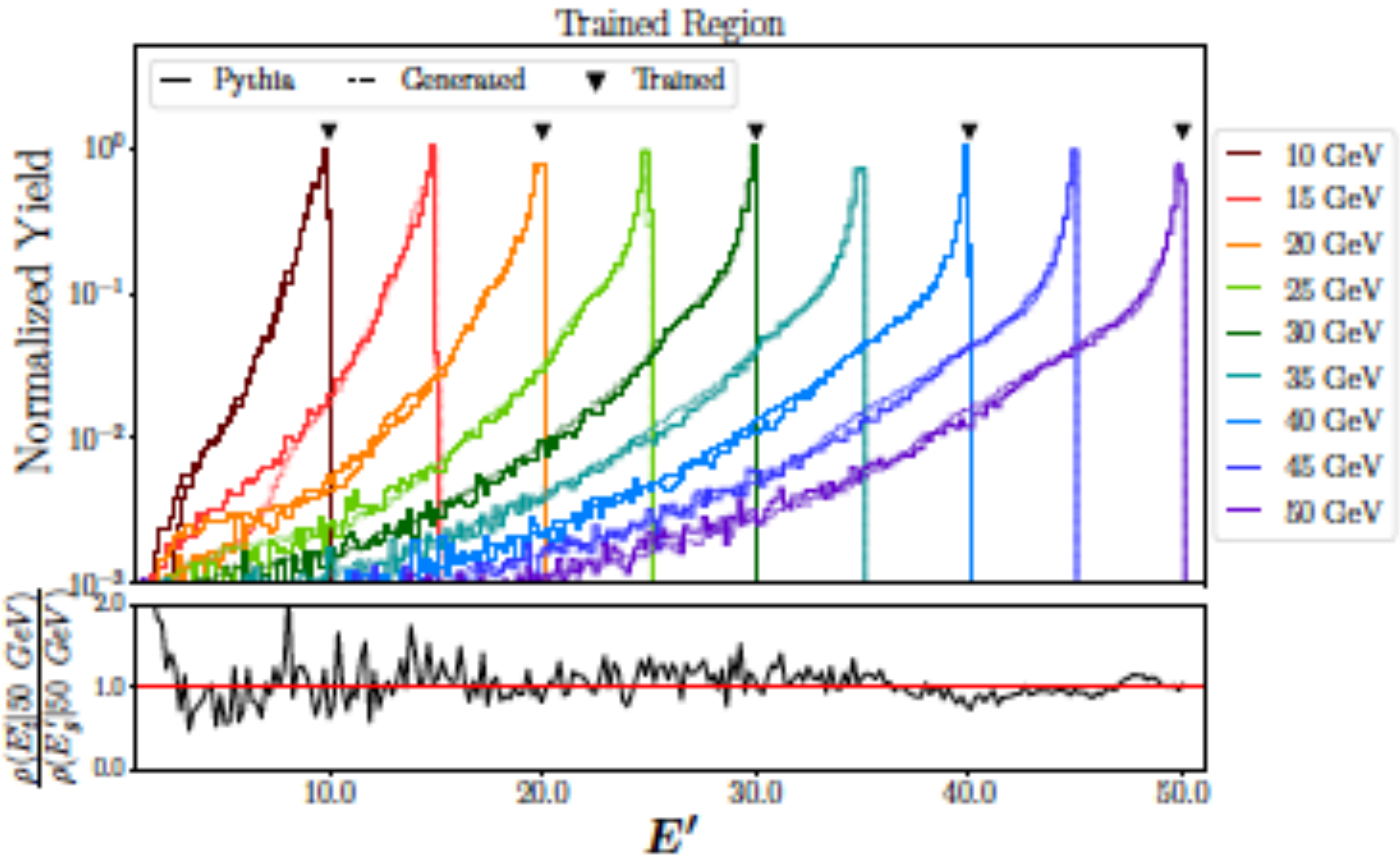
Conditional Jet Distributions



Conditional Wasserstein Generative Adversarial Networks for Fast Detector Simulation. John Blue, Braden Kronheim, Michelle Kuchera, Raghuram Ramanujan
accepted. Computational High Energy and Nuclear Physics (2021)

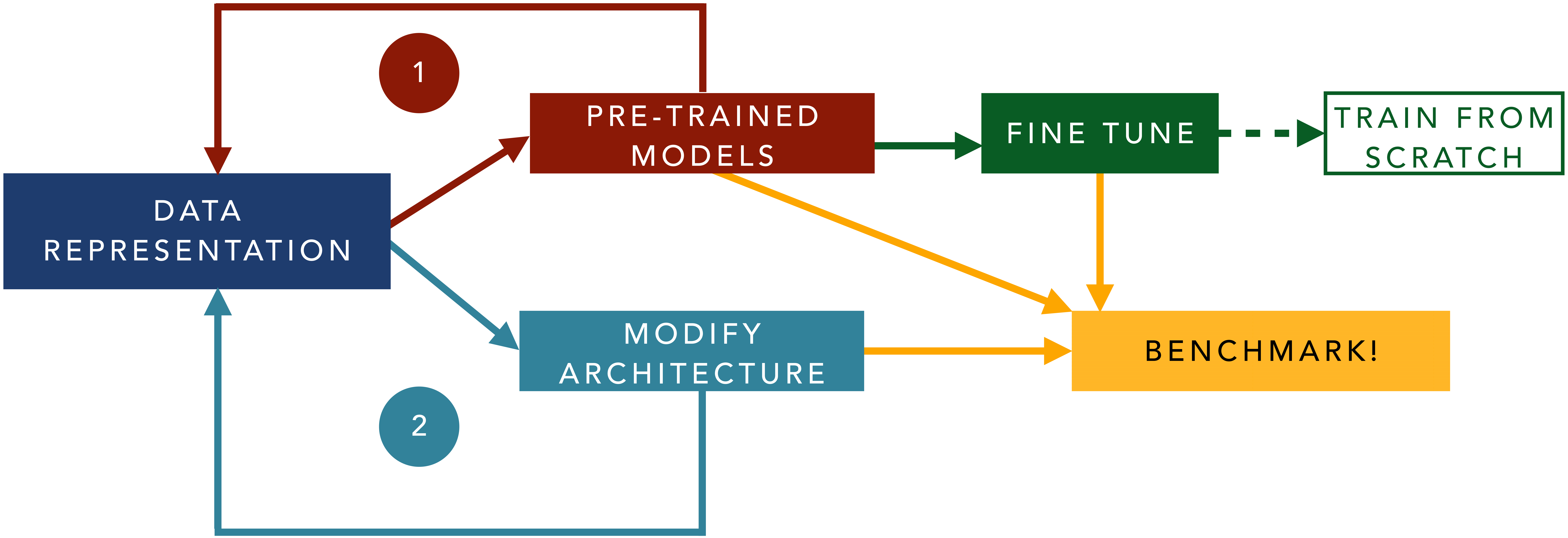
CONDITIONAL GAN

Conditional Distributions

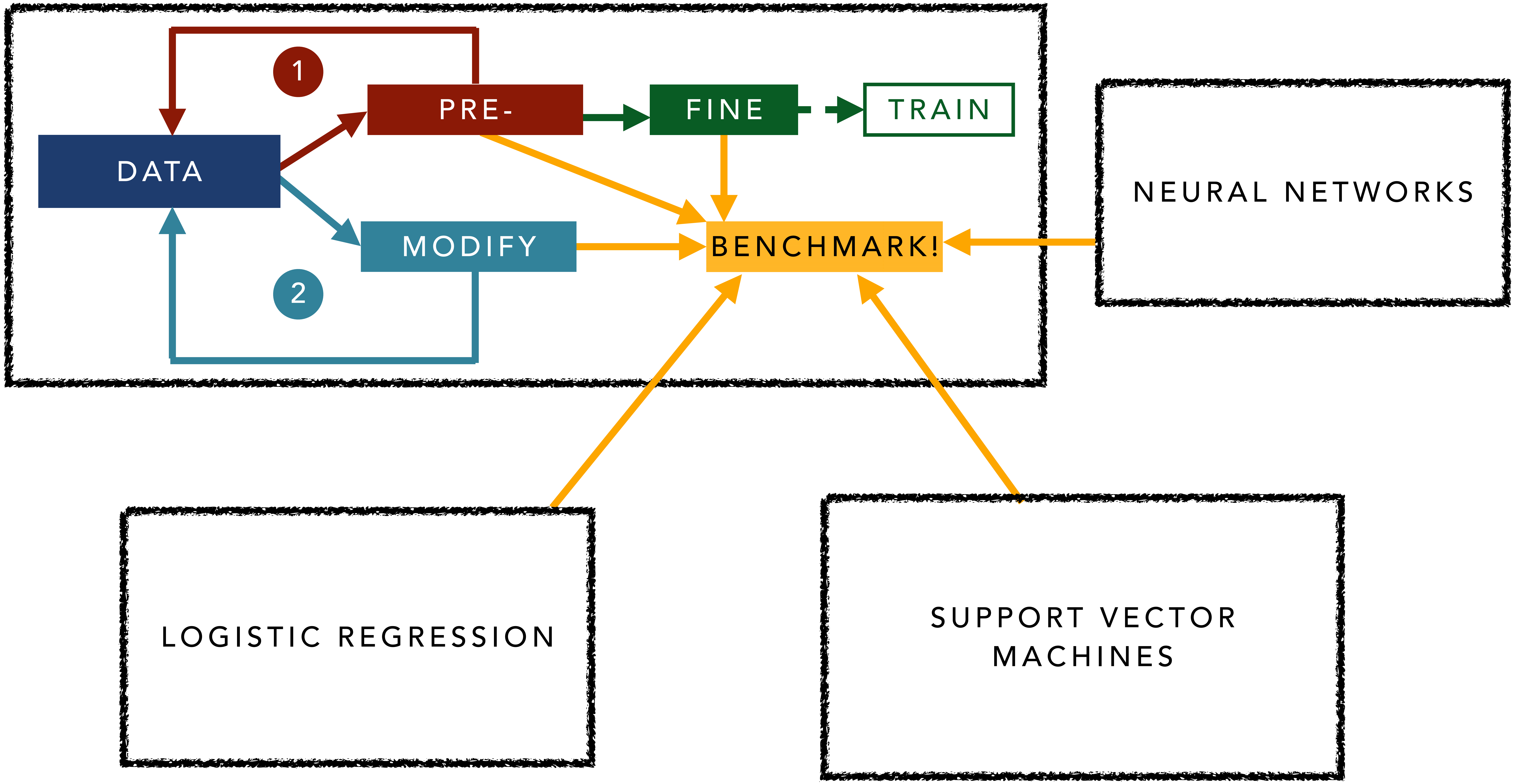


cFAT-GAN: Conditional Simulation of Electron-Proton Scattering Events with Variate Beam Energies by a Feature Augmented and Transformed Generative Adversarial Network L. Velasco, E. McClellan, N. Sato, P. Ambrozewicz, T. Liu, W. Melnitchouk, M.P. Kuchera, Yasir Alanazi, Yaohang Li, 19th IEEE International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 2020, pp. 372-375, doi: 10.1109/ICMLA51294.2020.00066.

EXAMPLE WORKFLOW



EXAMPLE WORKFLOW



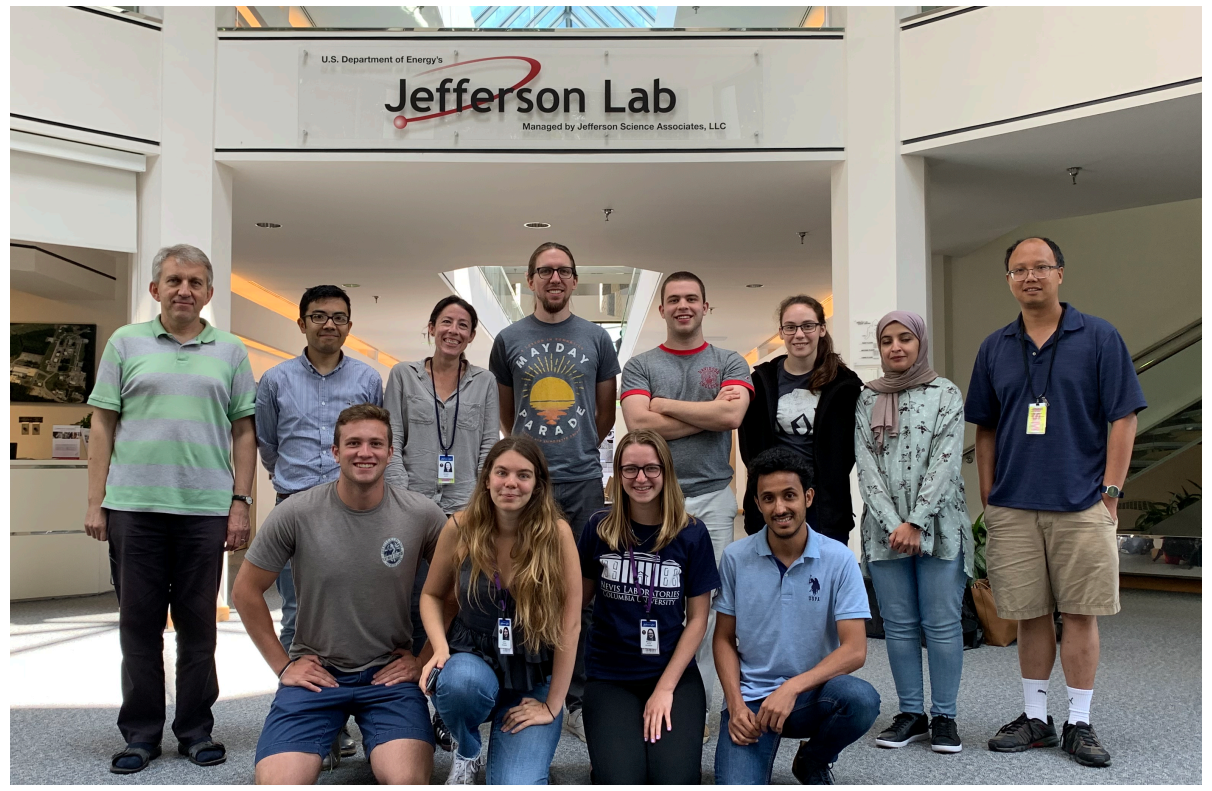
ACKNOWLEDGMENTS

- Raghu Ramanujan, Meg Houck, Eleni Tsitinidi, Jose Cruz, Andrew Hoyle, Michael Robertson, Evan Pritchard, Robert Solli, John Blue, Zach Nussbaum, Ryan Strauss, Jack Taylor

- JLab/ODU - Theory Center, Hall B: Nobuo Sato, Wally Melnitchouk, Yaohang Li, Yasir Alanazi, Manal Almaeen, Gagik Gavalian

- NSCL/FRIB - AT-TPC Group

ALPhA
DAVIDSON COLLEGE



Jefferson Lab

FRIB

DAVIDSON

NSCL