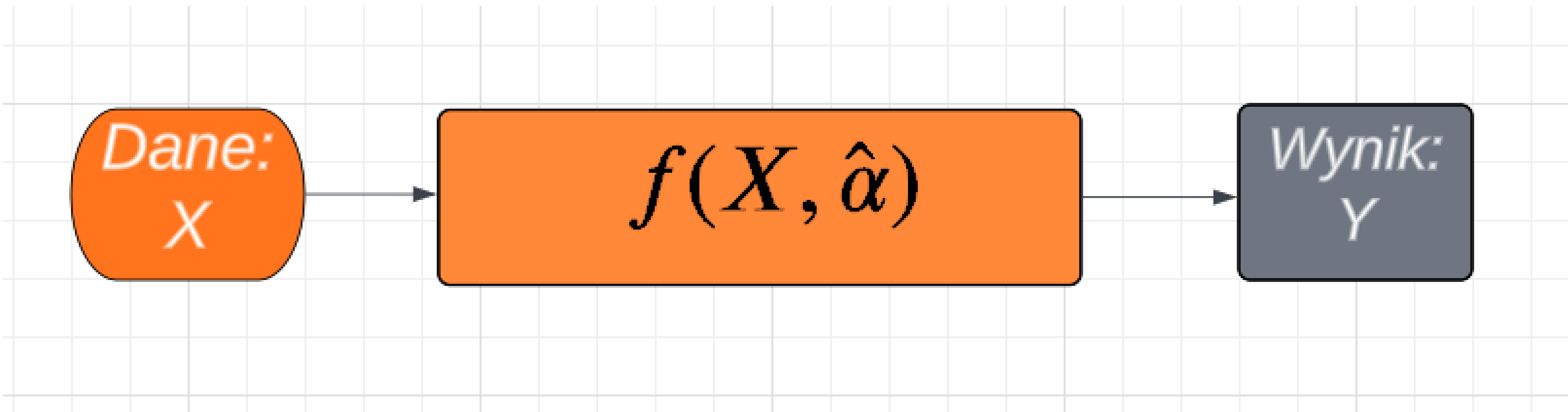
Machine learning in physics

Artur Kalinowski
University of Warsaw

X: n dimensional input data

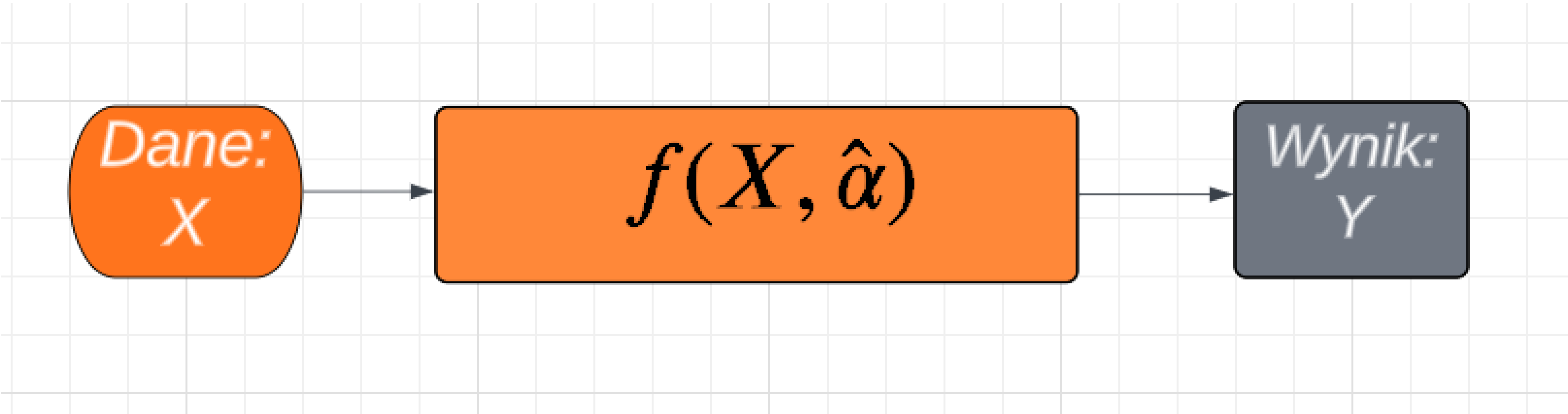
Y: k dimensional output data



Function fitting

X: n dimensional input data

Y: k dimensional output data



Function fitting:

- **n~2, k~1**
- **basic function given *explicite***
- **expansion coefficients *could* be interpretable**

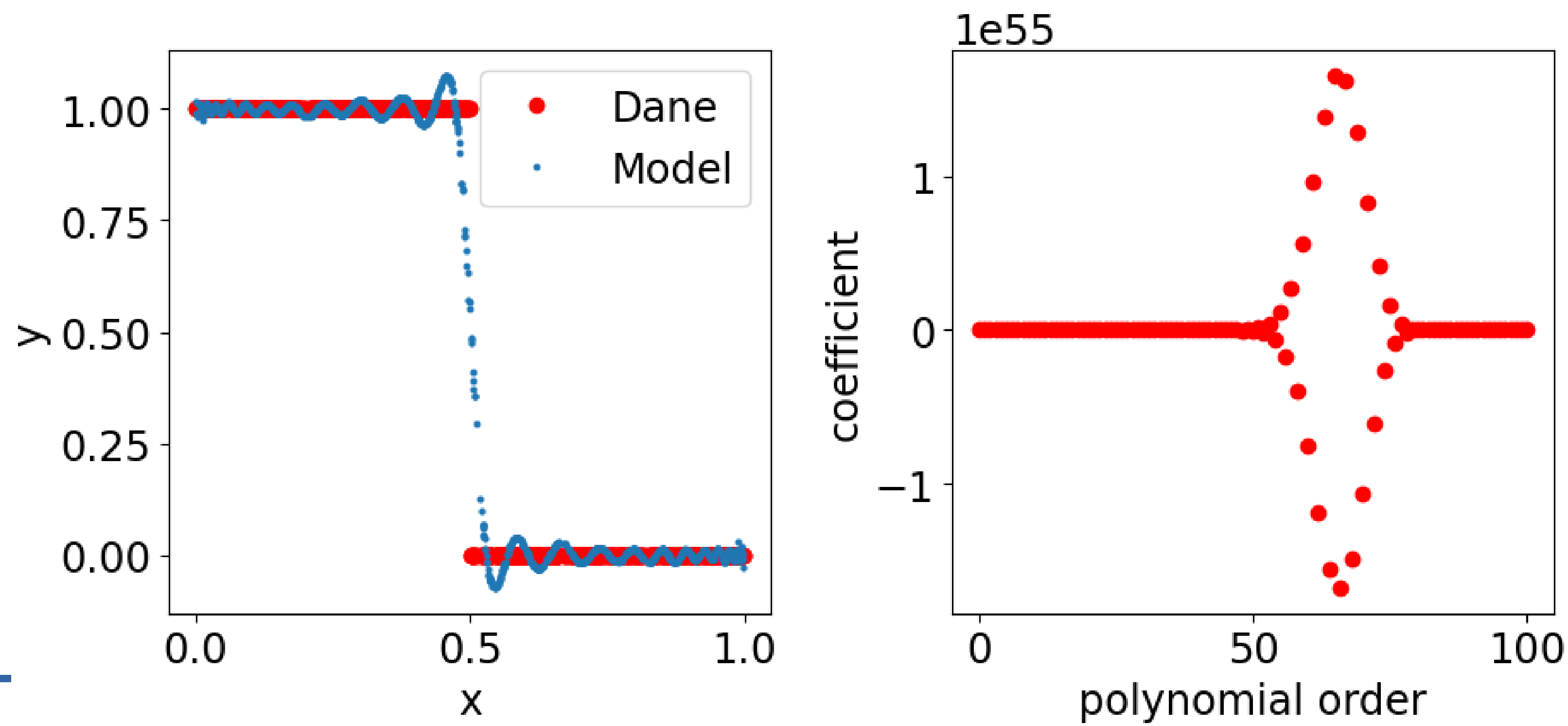
Universal approximator

Taylor theorem (J. Gregory, 1671):

Every, continuous, differentiable, function f(x): R → R can be approximated by a polynomial:

$$f(x,\theta) = \sum_{n=0}^{\infty} \theta_n x^n \simeq \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots + \theta_n x^n$$

Coefficients θ_i are derivatives of f(x).
In the case of unknown function ("data") coefficients can be found by a numerical procedure.
Usually…



Universal approximator

Fourier theorem (1807) :

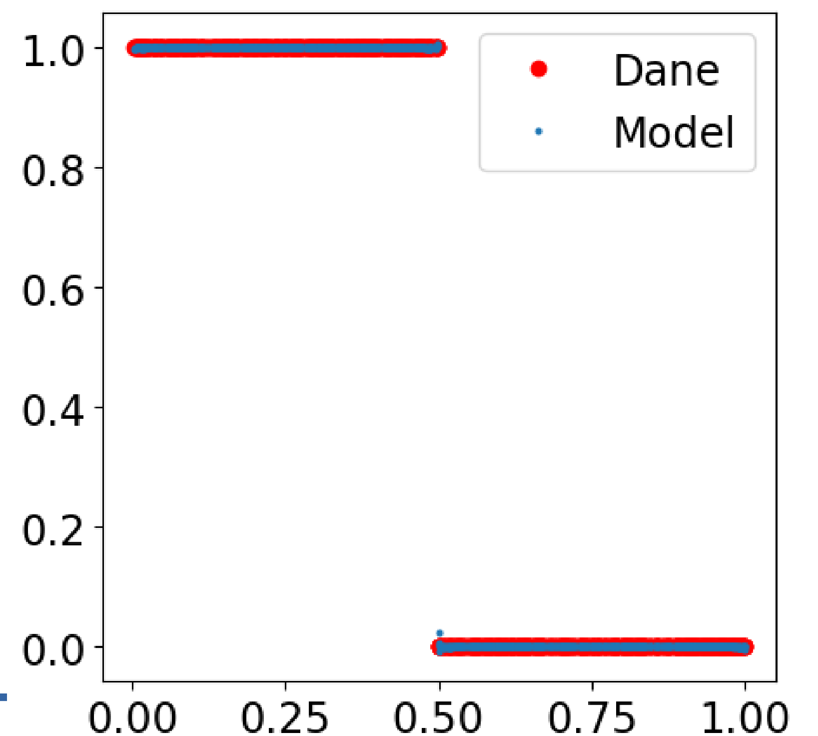
Every continuous, differentiable, and periodic function f(x): R → R can be approximated in a basis of sines i cosines:

$$f(x,\theta) = \frac{\theta_{0,0}}{2} + \sum_{n=1}^{\infty} \theta_{0,n}\cos(n\omega x) + \theta_{1,n}\sin(n\omega x),\ \omega = \frac{2\pi}{T}$$

Coefficients θ_i can be found analytically.
In the case of unknown function ("data") coefficients can be found by a numerical procedure.

```
nMax = 25000
omega = 2*np.pi/1.0

Y_model = np.full_like(X,0.5)
for n in range(1,nMax+1, 2):
    Y_model += 2.0/np.pi*1/n*np.sin(omega*n*X)
```



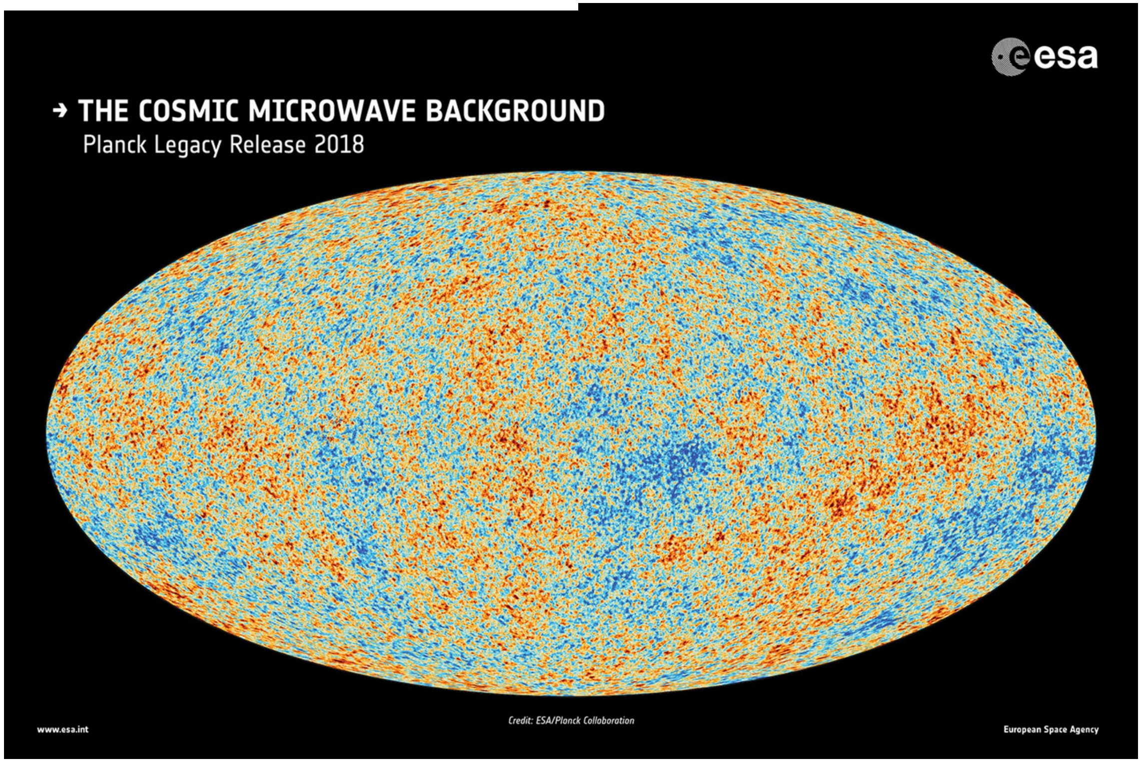
Universal approximator

Spherical harmonics (Laplace, 1782) :

Every, continuous, differentiable, function on sphere f(x): $R^2 \rightarrow R$ can be approximated in a basis of function solving a Laplace'a equation – spherical harmonics:

$$f(\theta, \varphi, w) = \sum_{n=1}^{\infty} w_{l,m} Y_m^l(\theta, \varphi)$$

Coefficients *w* can be found analytically. In the case of unknown function ("data") coefficients can be found by a numerical procedure.



ESA/Planck Collaboration

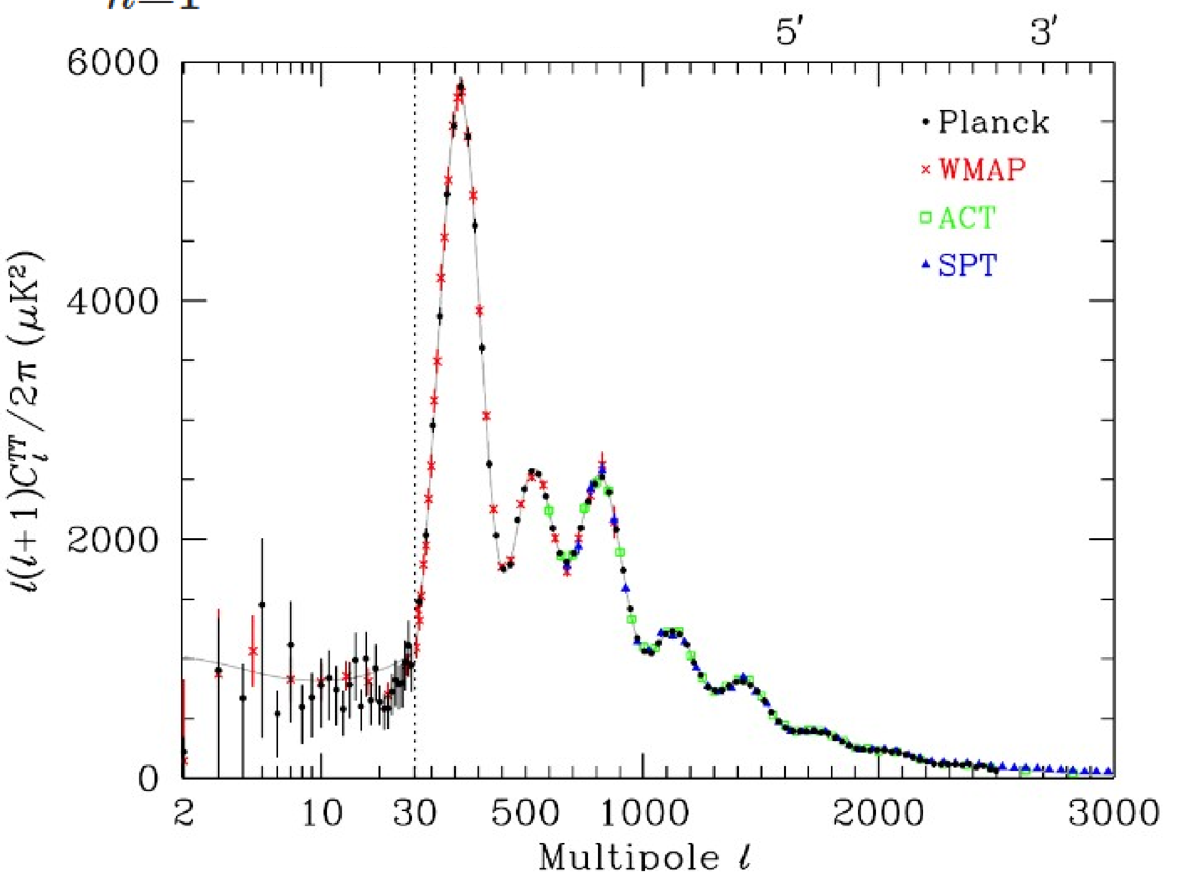
Universal approximator

Spherical harmonics (Laplace, 1782) :

Every, continuous, differentiable function on sphere f(x): $R^2 \rightarrow R$ can be approximated in a basis of function solving a Laplace'a equation – spherical harmonics:

$$f(\theta, \varphi, w) = \sum_{n=1}^{\infty} w_{l,m} Y_m^l(\theta, \varphi)$$

Coefficients w can be found analytically. In the case of unknown function ("data") coefficients can be found by a numerical procedure.

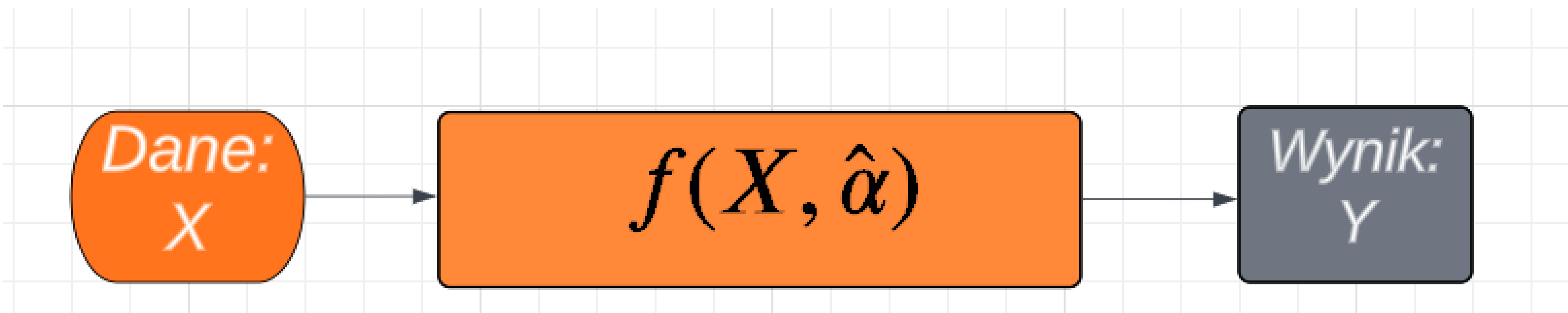


R.L. Workman et al. (Particle Data Group), Prog. Theor. Exp. Phys. 2022, 083C01 (2022) and 2023 update

Machine learning

X: n dimensional input space

Y: k dimensional output space



- **n~10^l, k~10^m, l,m~6**
- **basis functions defined *implicite* - through data flow architecture**
- **expansion coefficients are uninterpretable**

Universal approximator

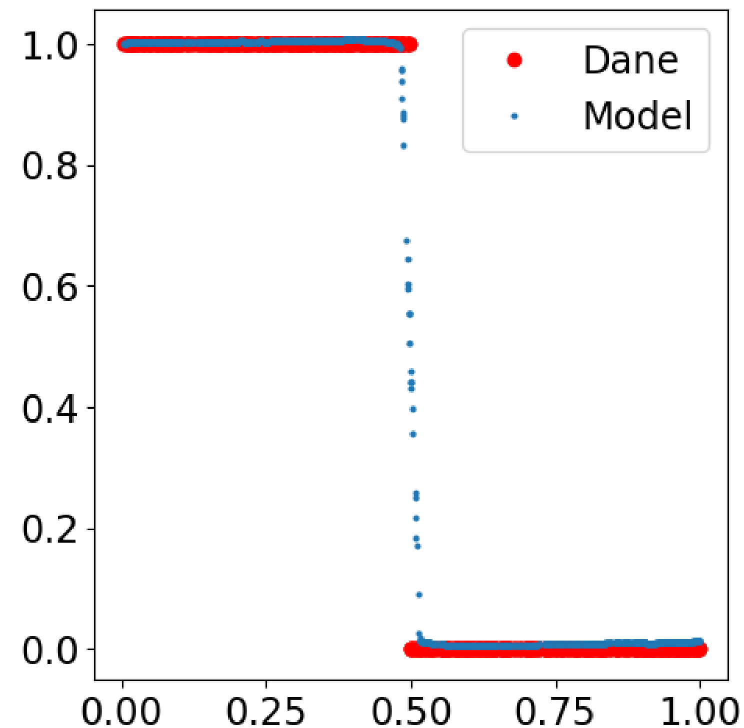
A sigmoid function: any non polynomial function fulfilling conditions:

$$A(\theta, x) = A(\sum_{i=1}^{n} \theta_i x_i + b) \qquad \lim_{x_i \to -\infty} A(x) \to 0 \quad \lim_{x_i \to +\infty} A(x) \to 1$$

Universal approximator theorem (Cybenko, 1989):

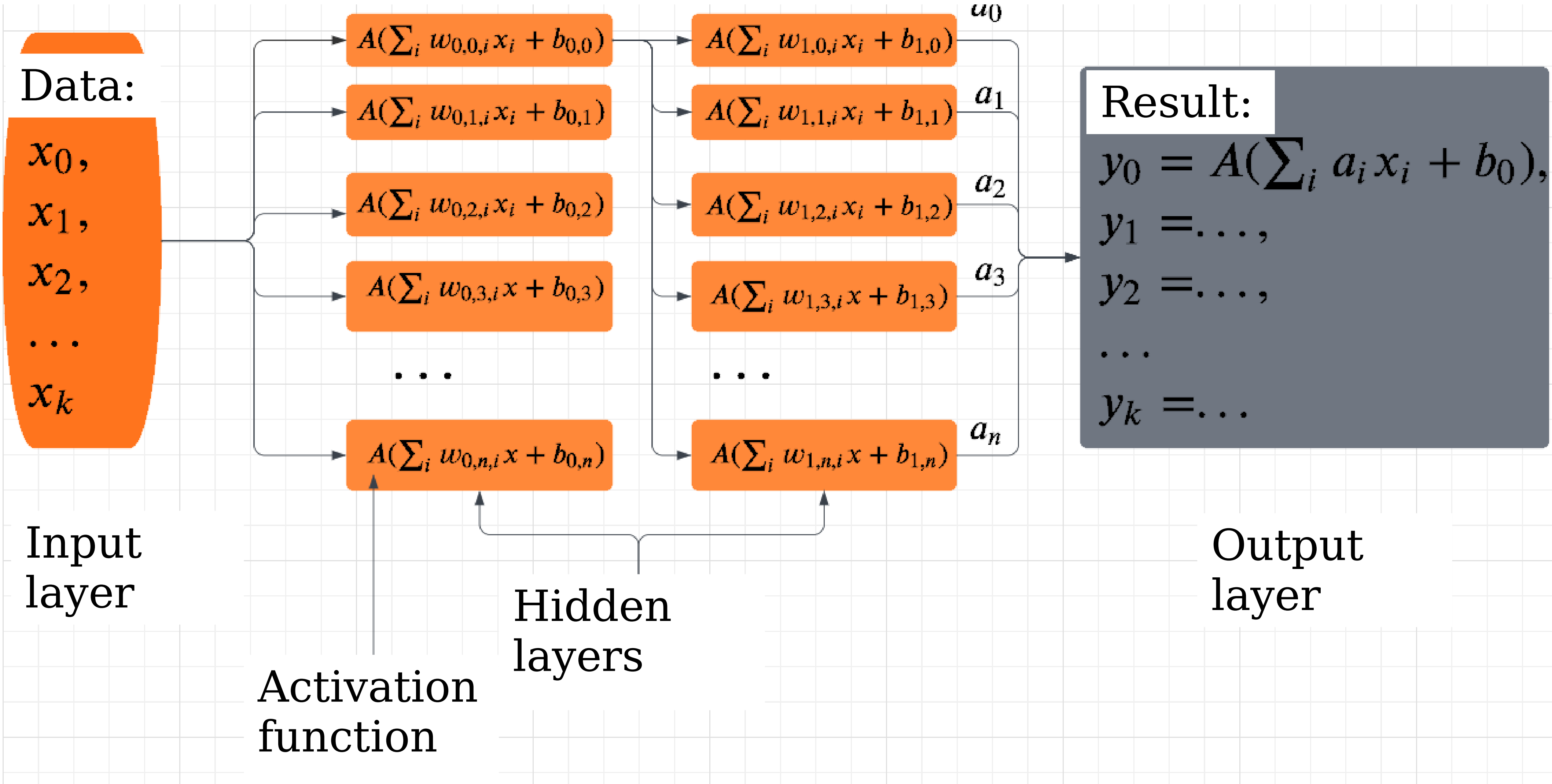
Every continuous function f(x) $R^n \to R$ can be approximated in basis of sigmoidal functions:

$$f(x, \theta) \simeq \sum_n w_n A(\theta_n, x)$$



Coefficients θ_i, w_i do not have in general an analytic form, but can be found using a numerical procedure

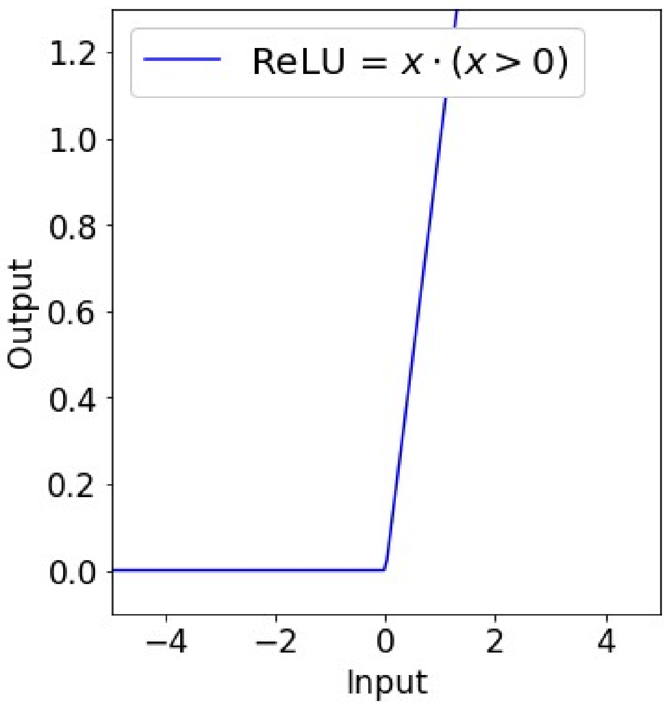
Architecture: fully connected

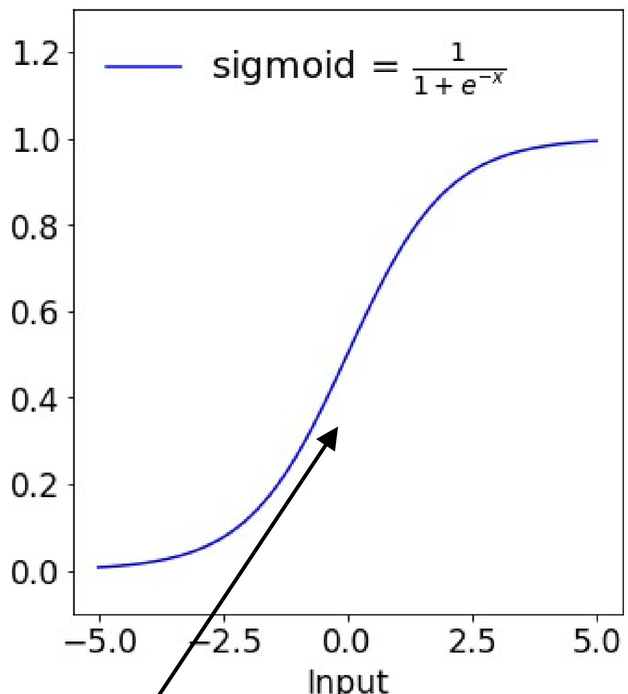


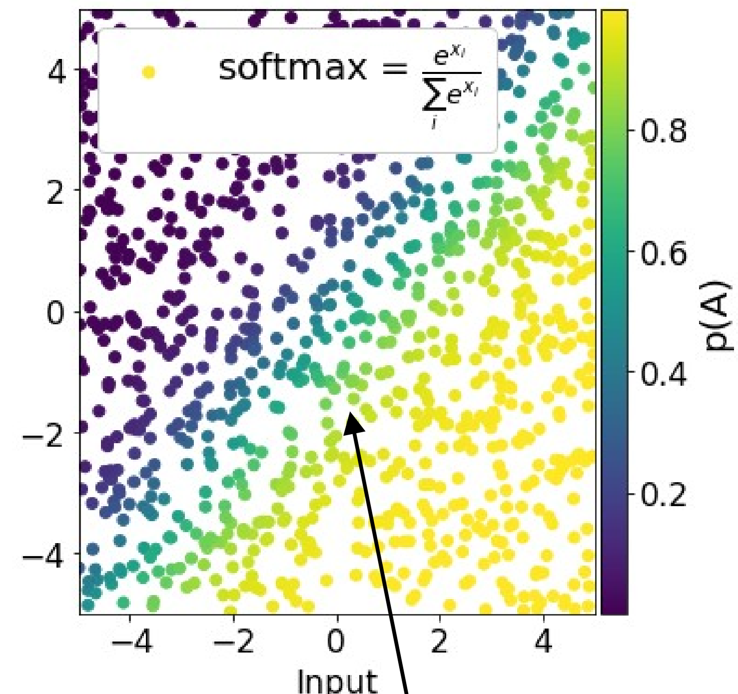
Activation function

standard for hidden layers

standard for output layers (when probability is the target)







two classes probability

many classes probability

Example: materials physics

The task: superconductivity compound classification:
Is superconductive? **YES/NO**

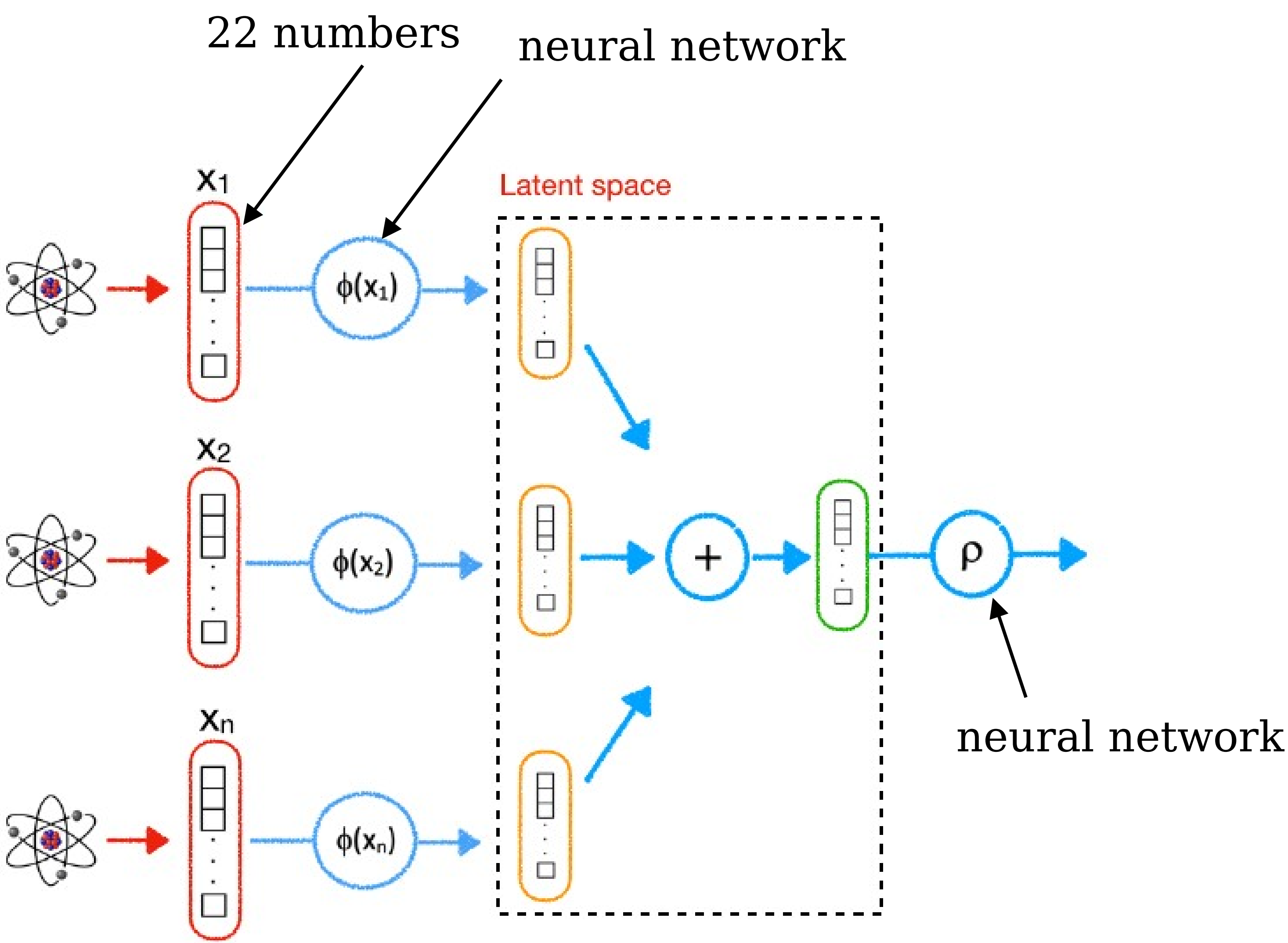
Input data:
- set of compound elements,
- 22 features per element + stoichiometric coefficients

Data sets:
Training and validation:
- 16 395 superconductive compounds,
- 50 000 normal compounds

Test:
- 207 compounds including 39 superconductors

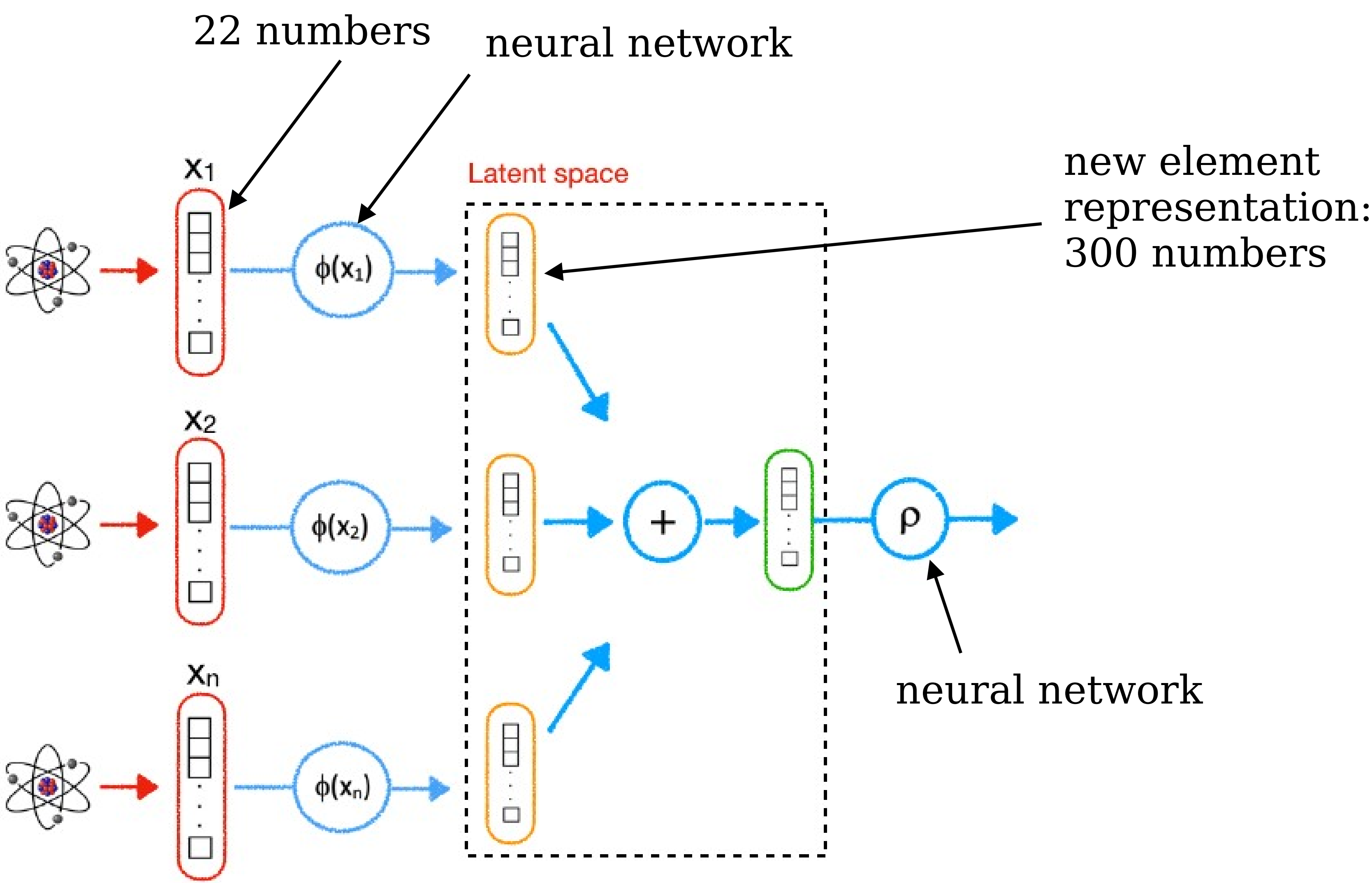
Pereti, C., Bernot, K., Guizouarn, T. et al. From individual elements to macroscopic materials: in search of new superconductors via machine learning. npj Comput Mater 9, 71 (2023).
https://doi.org/10.1038/s41524-023-01023-6

Example: materials physics



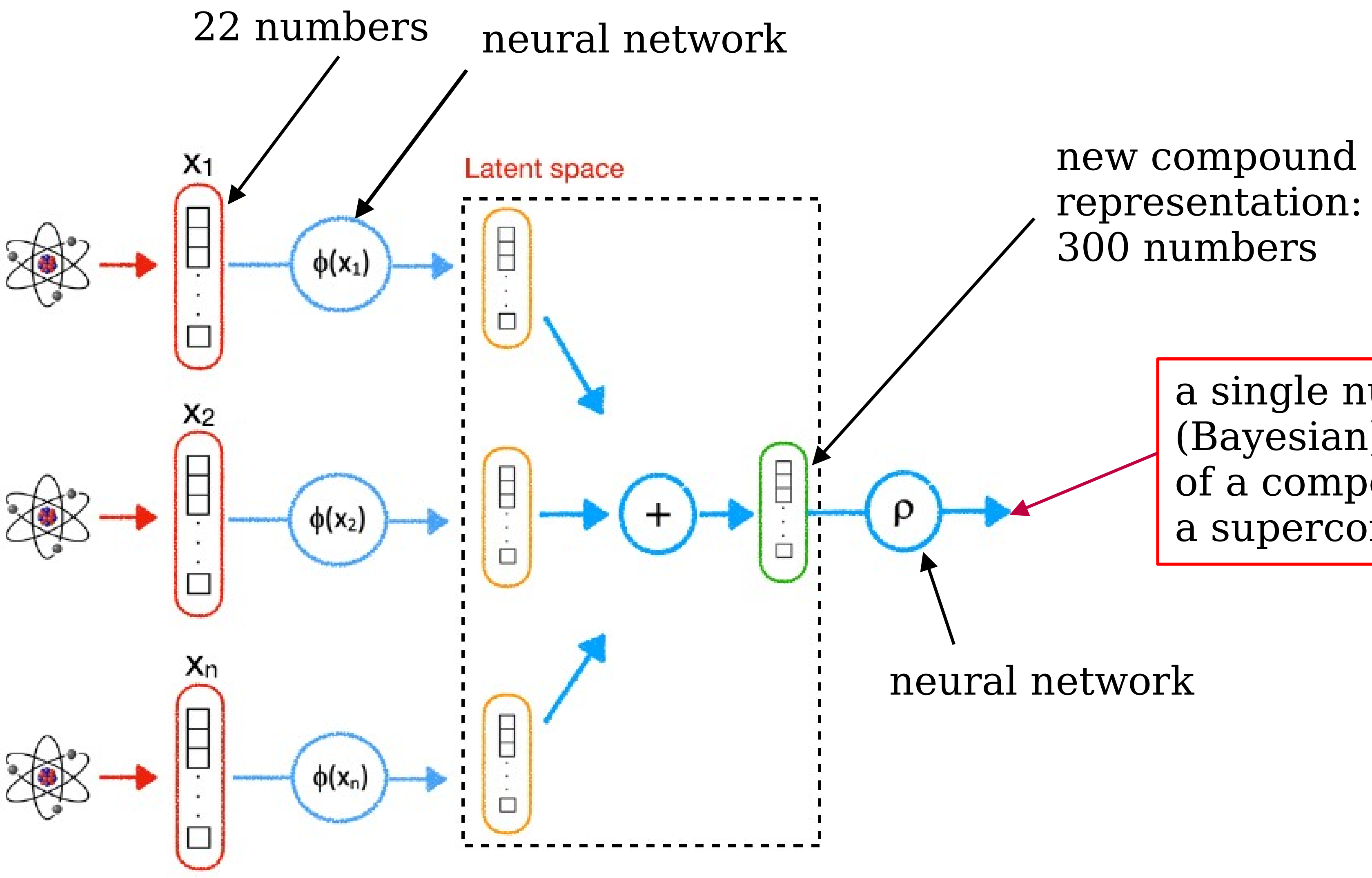
Pereti, C., Bernot, K., Guizouarn, T. et al. From individual elements to macroscopic materials: in search of new superconductors via machine learning. npj Comput Mater 9, 71 (2023). https://doi.org/10.1038/s41524-023-01023-6

Example: materials physics



Pereti, C., Bernot, K., Guizouarn, T. et al. From individual elements to macroscopic materials: in search of new superconductors via machine learning. npj Comput Mater 9, 71 (2023). https://doi.org/10.1038/s41524-023-01023-6

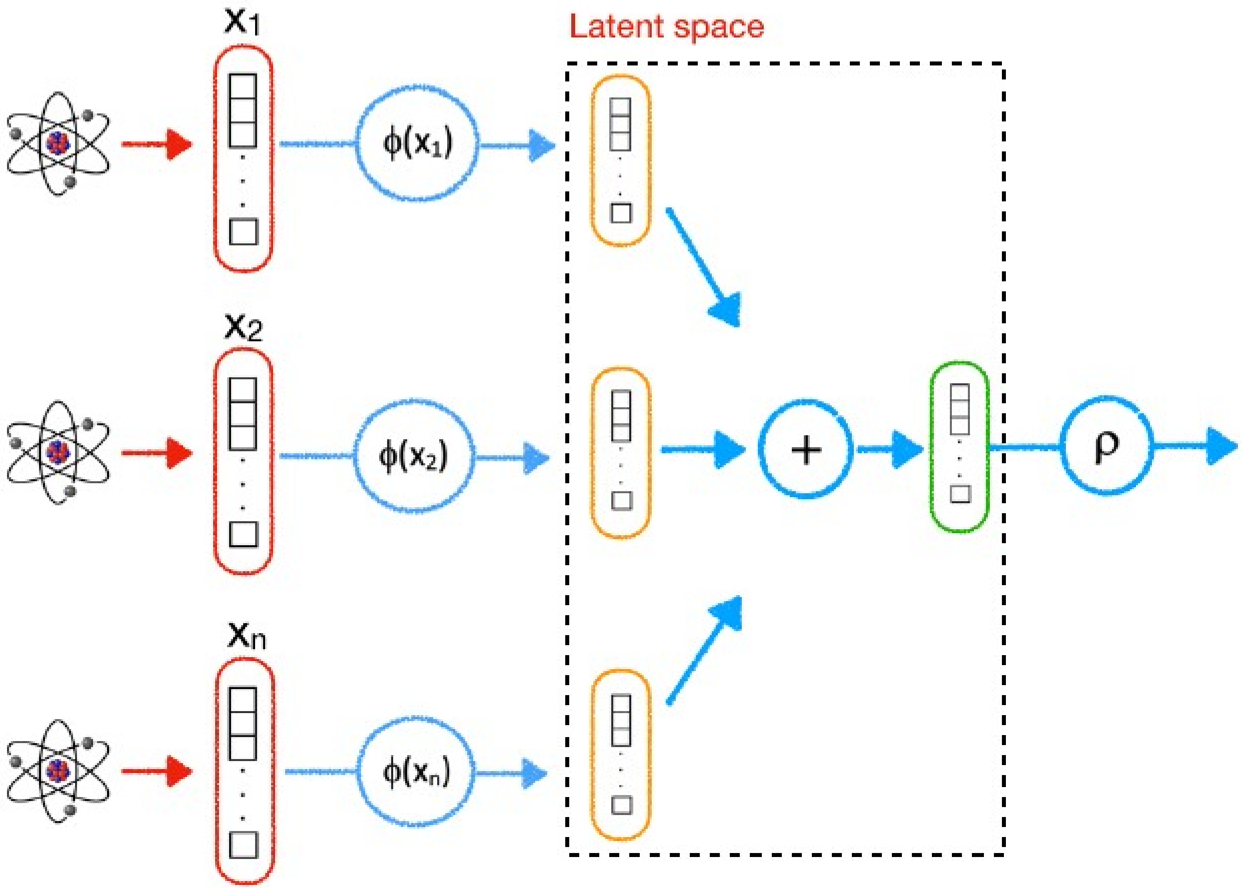
Example: materials physics



a single number:
(Bayesian) probability
of a compound being
a superconductor

Pereti, C., Bernot, K., Guizouarn, T. et al. From individual elements to macroscopic materials: in search of new superconductors via machine learning. npj Comput Mater 9, 71 (2023).
https://doi.org/10.1038/s41524-023-01023-6

Example: materials physics

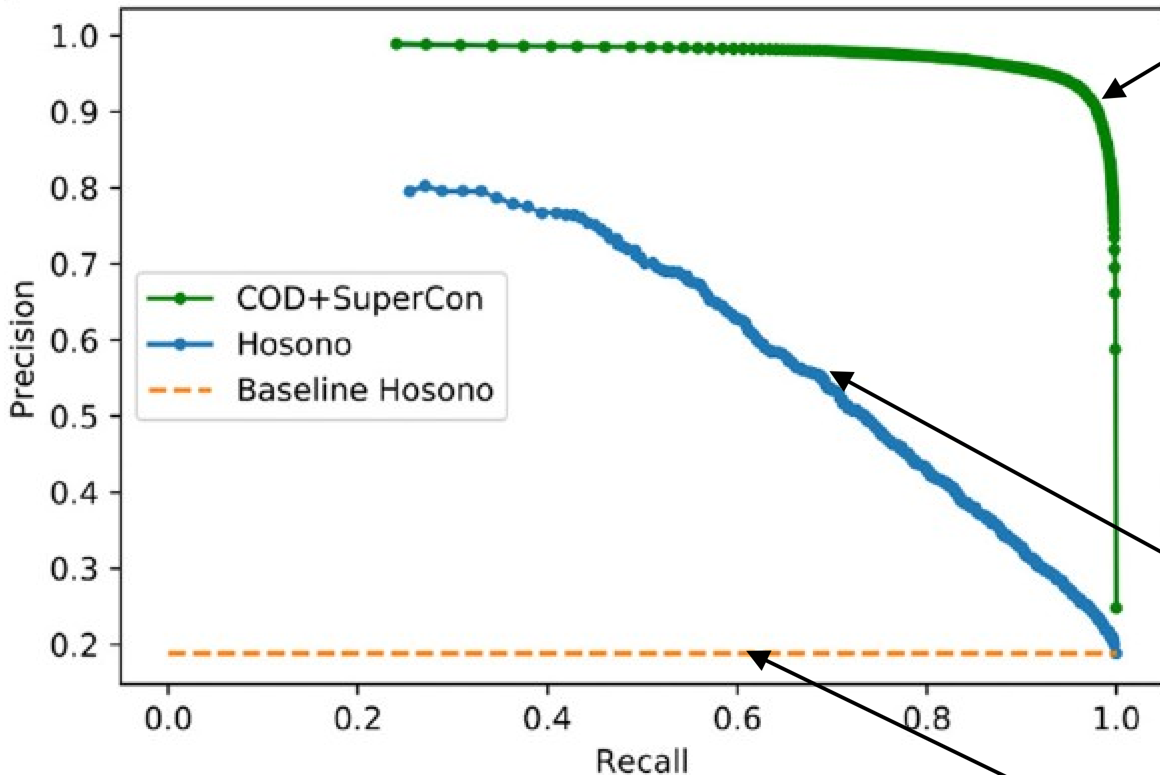


Training:

60 sessions with random split into training and validation datasets in 80:20 proportions

Pereti, C., Bernot, K., Guizouarn, T. et al. From individual elements to macroscopic materials: in search of new superconductors via machine learning. npj Comput Mater 9, 71 (2023). https://doi.org/10.1038/s41524-023-01023-6

Example: materials physics



SuperCon – superconductors
COD – normal compounds

Compounds from this dataset were used during the training

independent set, not used during the training

the Hosono set contains 20% of superconductors

decision – **average** from 60 models responses

Pereti, C., Bernot, K., Guizouarn, T. et al. From individual elements to macroscopic materials: in search of new superconductors via machine learning. npj Comput Mater 9, 71 (2023). https://doi.org/10.1038/s41524-023-01023-6

Example: materials physics

decision – **average** from 60 models responses

decision – **„voting”** of 60 models

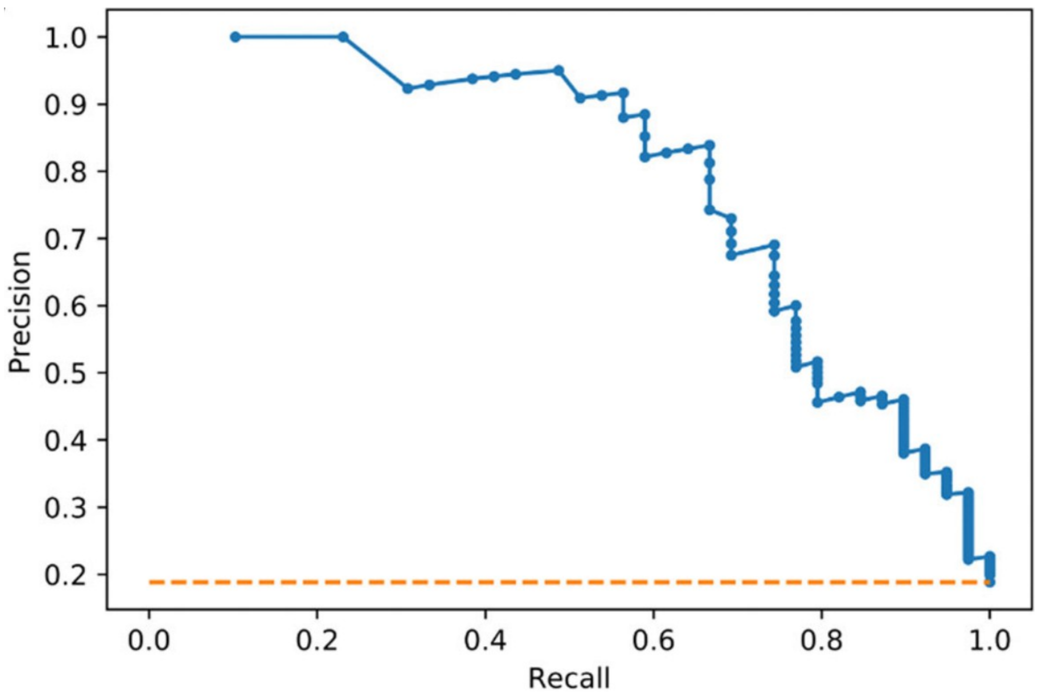
$$\text{avg} = \frac{1}{60}\left[\sum p\right] > 0.5$$

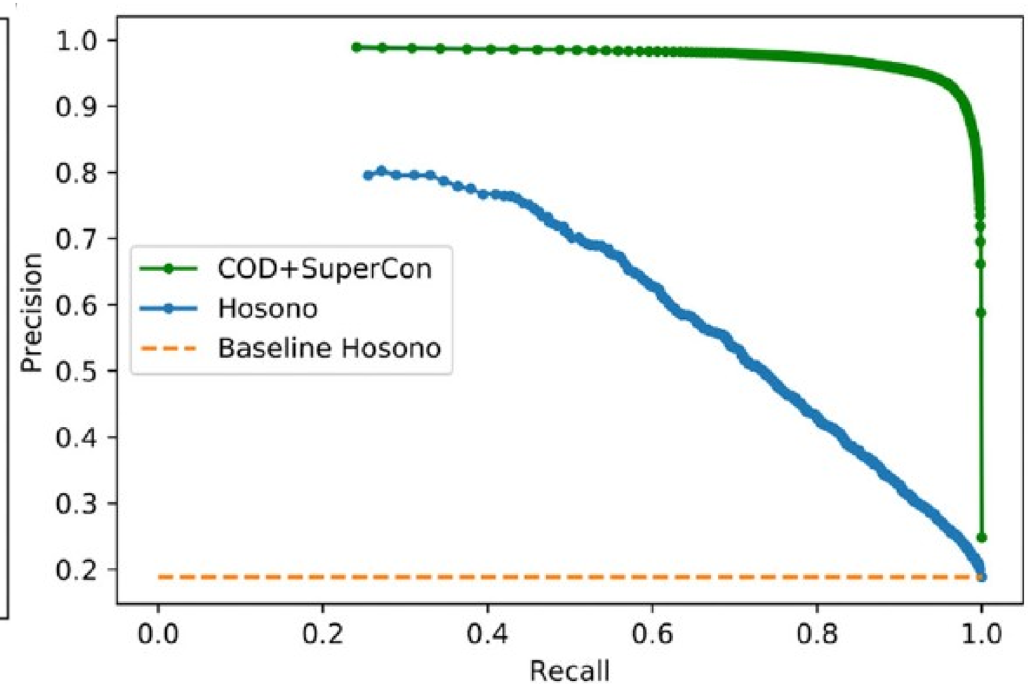
$$\text{vote} = \frac{1}{60}\left[\sum p > 0.5\right] > 0.5$$

$$\text{avg} = \frac{1}{3}(0.1) + \frac{1}{3}(0.51) + \frac{1}{3}(0.51) = \frac{1.12}{3}$$

$$\text{vote} = \frac{1}{3}(0.1 > 0.5) + \frac{1}{3}(0.51 > 0.5) + \frac{1}{3}(0.51 > 0.5) = \frac{2}{3}$$

Example: materials physics

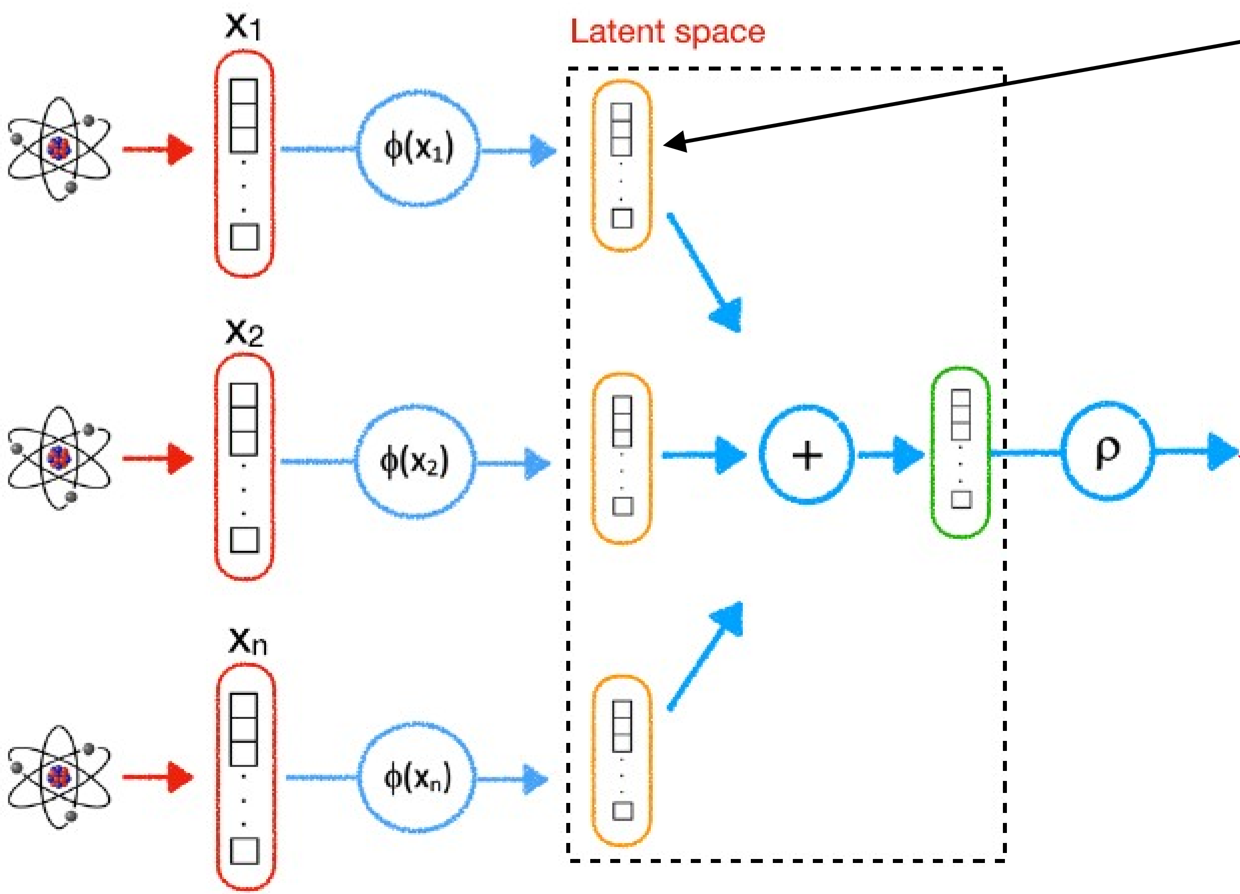




decision – **„voting”** of 60 models

decision – **average** from 60 models responses

Example: materials physics



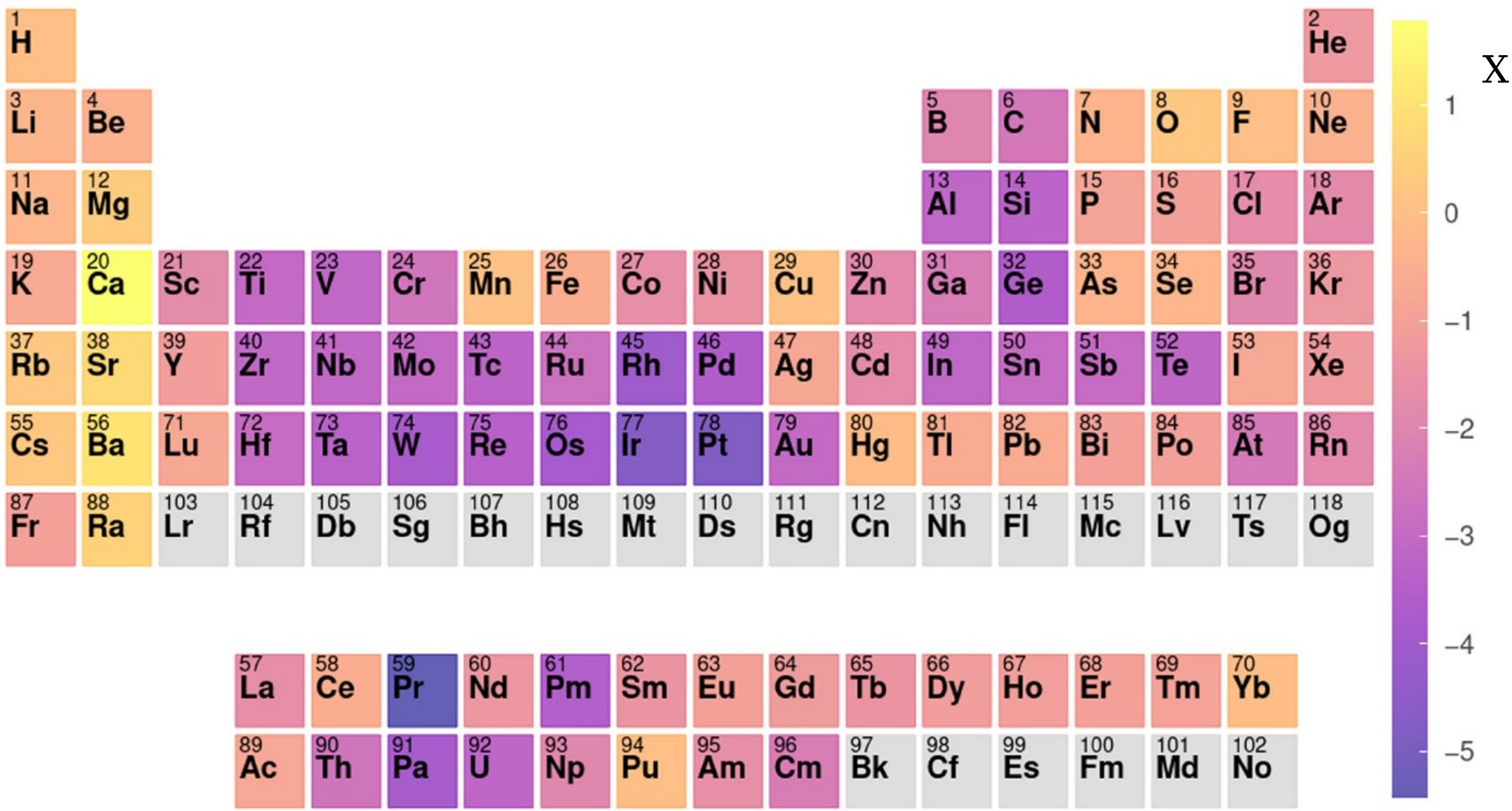
Single dimensional element representation X → some interpretation **possible**

critical superconductor temperature

Pereti, C., Bernot, K., Guizouarn, T. et al. From individual elements to macroscopic materials: in search of new superconductors via machine learning. npj Comput Mater 9, 71 (2023). https://doi.org/10.1038/s41524-023-01023-6

Example: materials physics

Contribution of an element on critical temperature value:
large X → increased T_c



Pereti, C., Bernot, K., Guizouarn, T. et al. From individual elements to macroscopic materials: in search of new superconductors via machine learning. npj Comput Mater 9, 71 (2023).
https://doi.org/10.1038/s41524-023-01023-6

Encoding/representation

Task: seek a function: **E(x): $\mathbf{R^n \rightarrow R^m}$, $\mathbf{m \neq n}$**, such that keeps maximum amount of information from the original input data

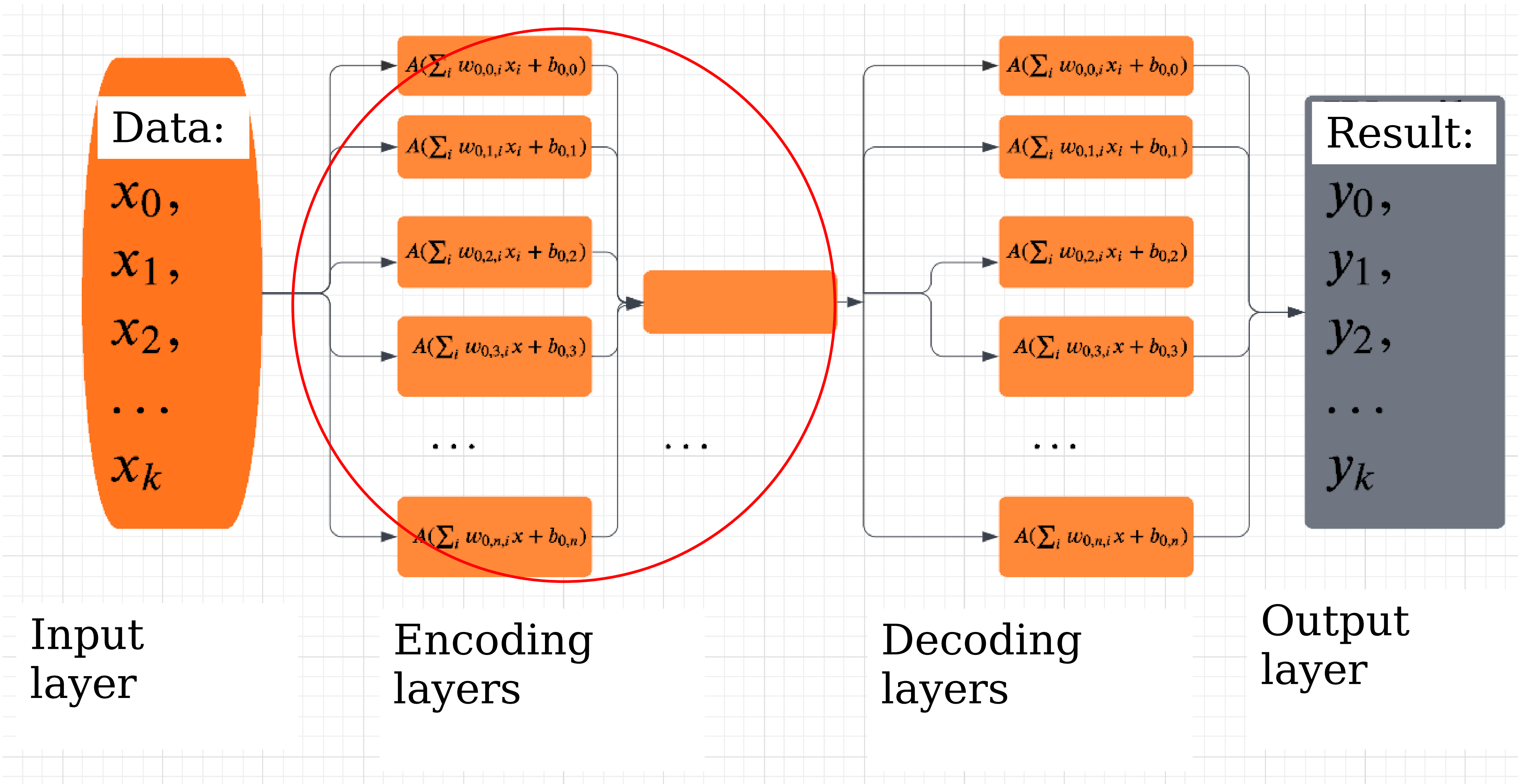
Solution: creation of two functions: E(x): $x \rightarrow z$, $D(z) = E^{-1}(z)$: $z \rightarrow x$

$$x = D(E(x))$$

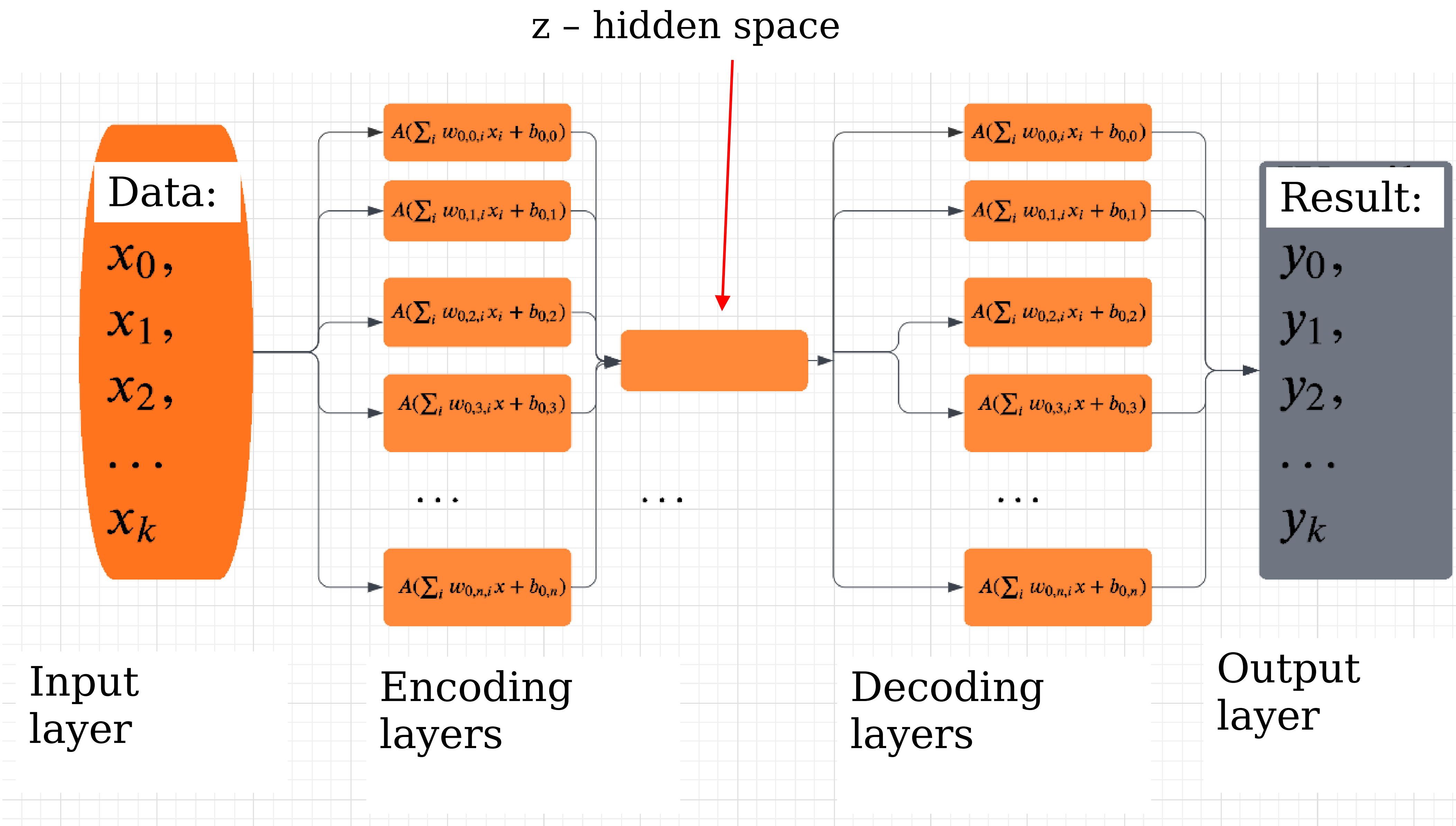
E(x) and D(z) parametrization: given *implicite* by the neural network connections

Architecture: autoencoder

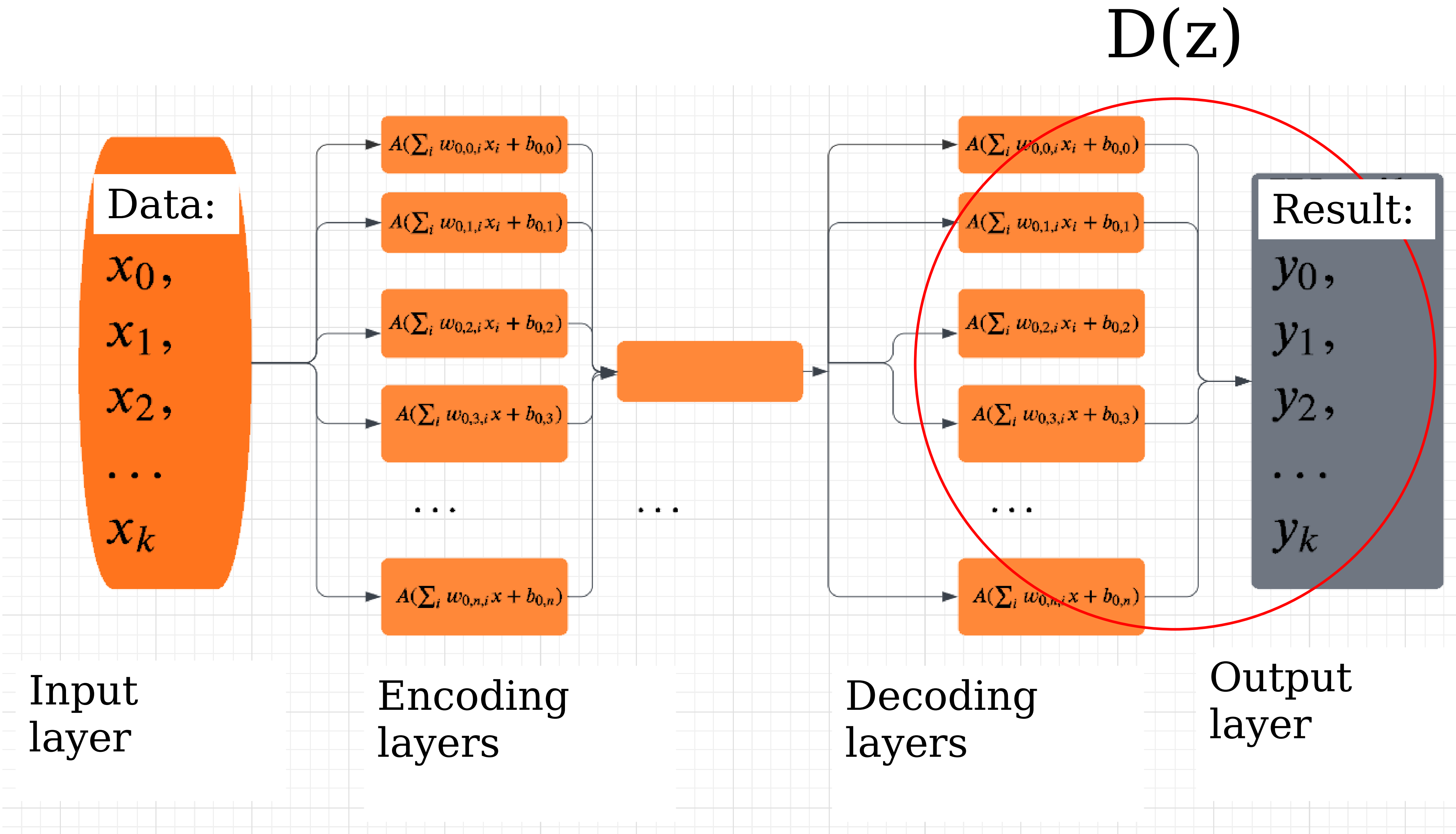
E(x)



Architecture: „autoencoder”



Architektura: „autoencoder”



Example: astronomy

The task:

identification of active galaxy nuclei (AGN) with high redshift z~3

Input data:

- light spectra in form of intensity in 914 wave length bins

Datasets:

Training and validation:

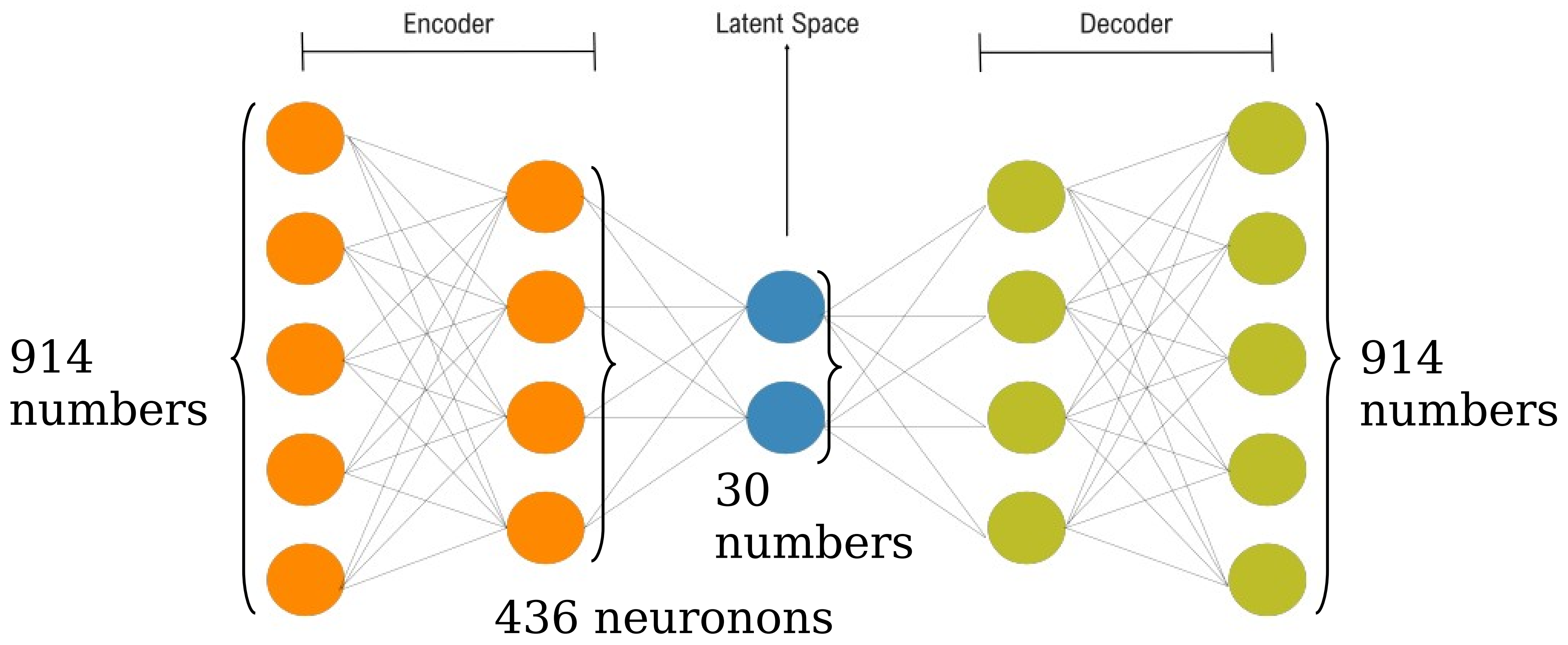
- 2458 spectra: 23% AGN, 22% stars,
  55% normal galaxies with (High-z) or low (Low-z) z
- Division: 4:1

Untagged set:

- 716 objects with z~3 (Photo-z)

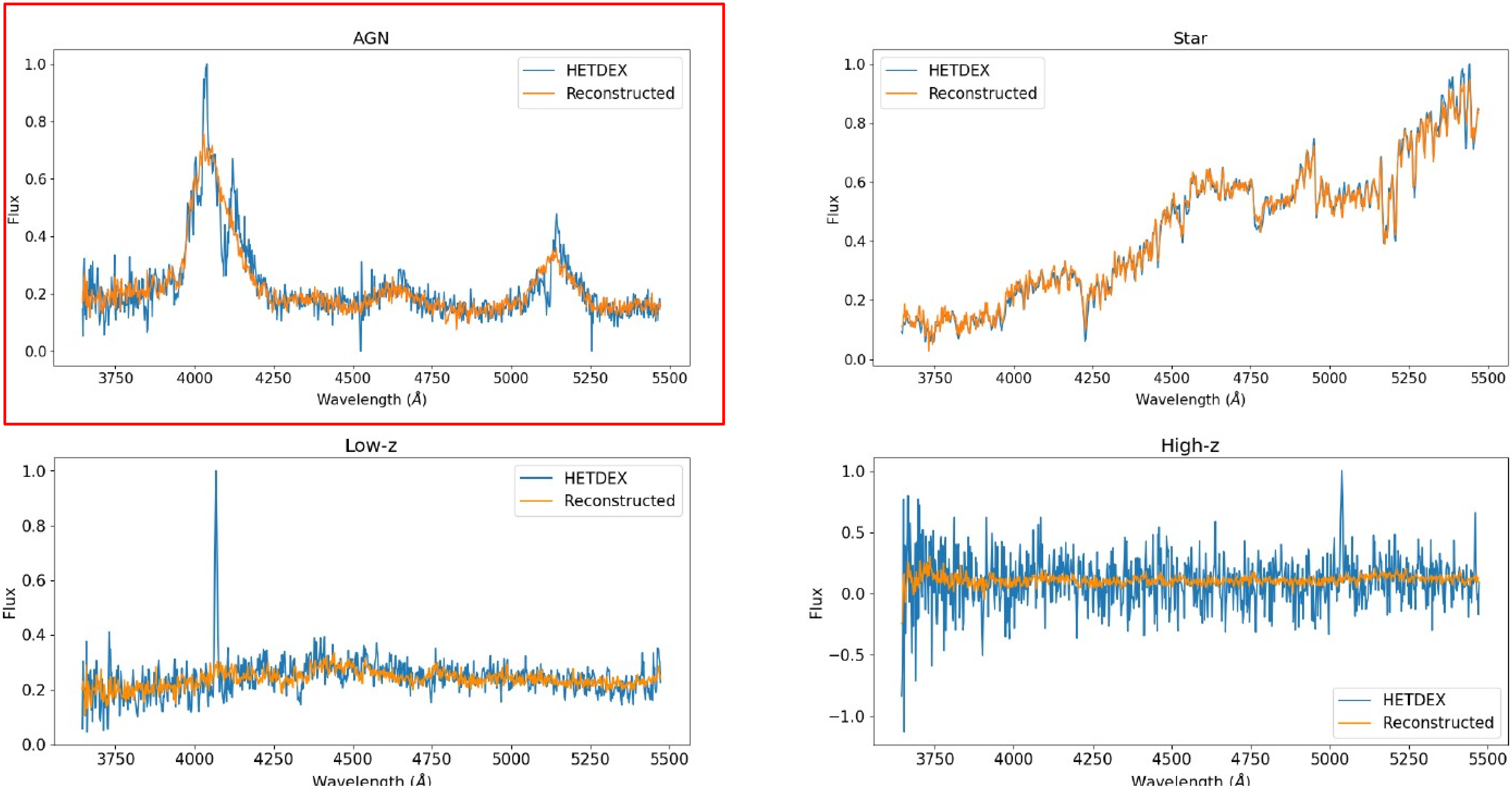
Identifying Active Galactic Nuclei at z~3 from the HETDEX Survey Using Machine Learning
arXiv:2302.11092 [astro-ph.GA]

Example: astronomy

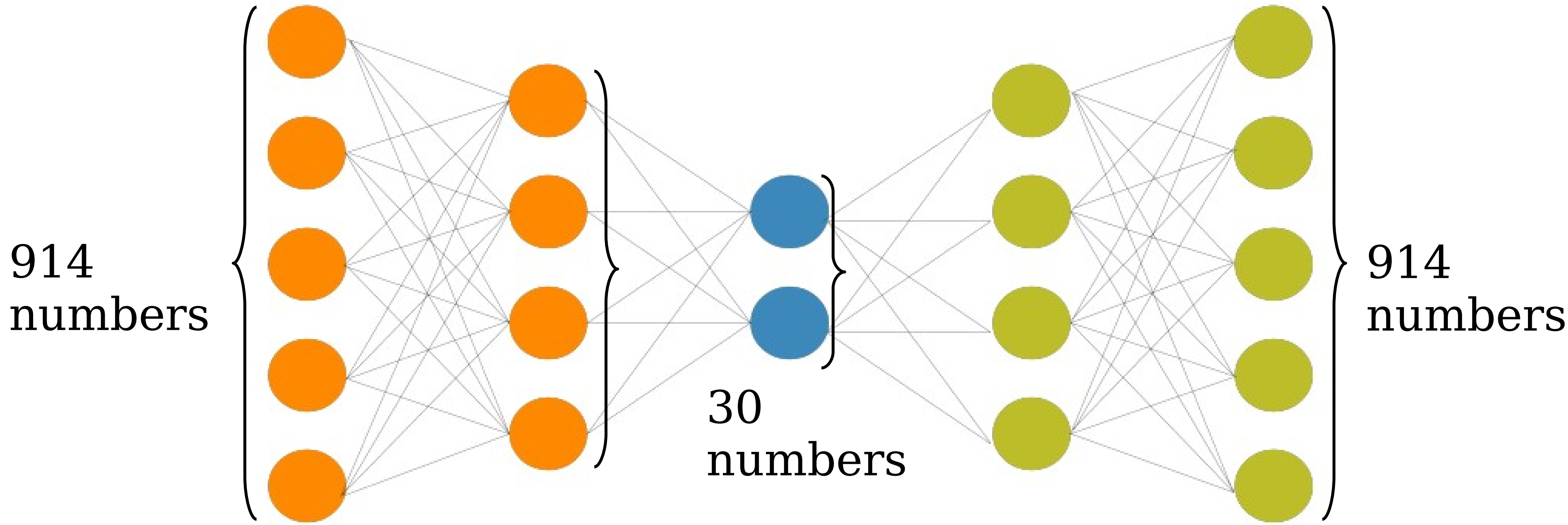


Identifying Active Galactic Nuclei at z~3 from the HETDEX Survey Using Machine Learning
arXiv:2302.11092 [astro-ph.GA]

Example: astronomy



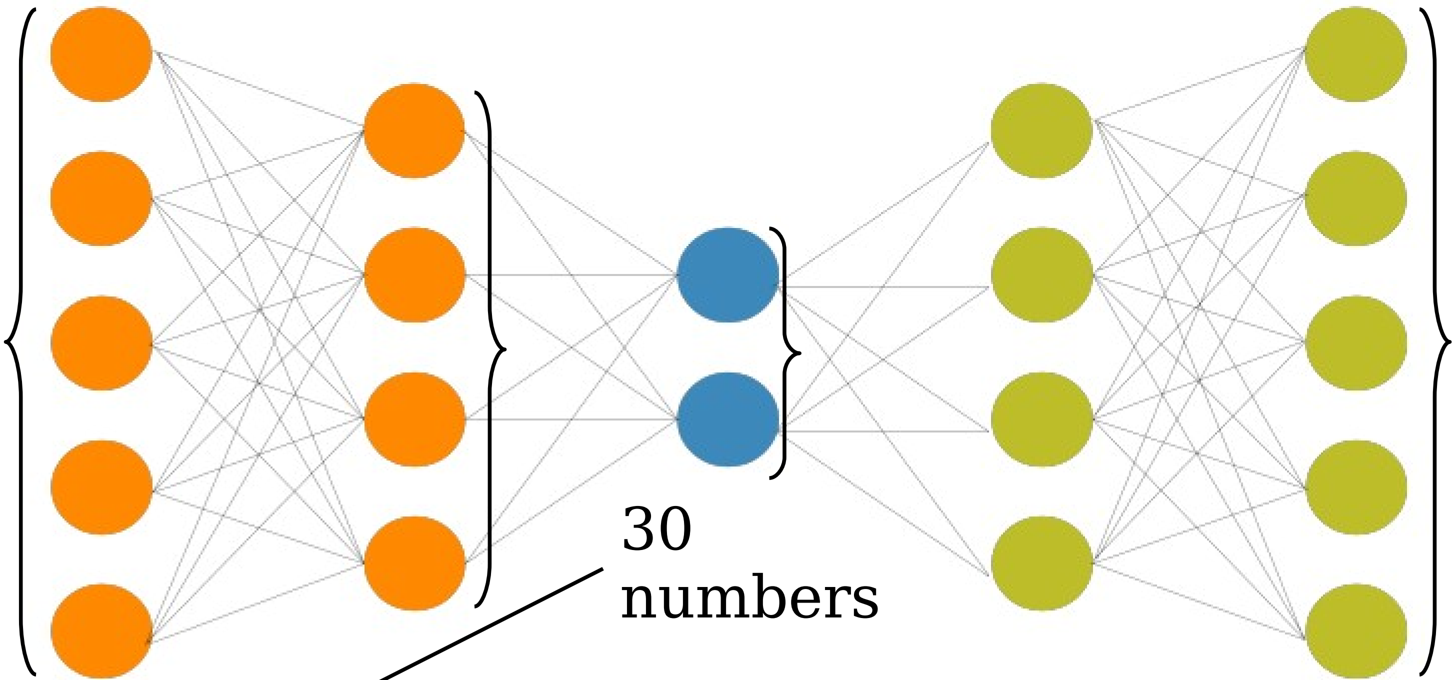
Identifying Active Galactic Nuclei at z~3 from the HETDEX Survey Using Machine Learning
arXiv:2302.11092 [astro-ph.GA]



Identifying Active Galactic Nuclei at z~3 from the HETDEX Survey Using Machine Learning
arXiv:2302.11092 [astro-ph.GA]

Example: astronomy

914
numbers



914
numbers

30
numbers

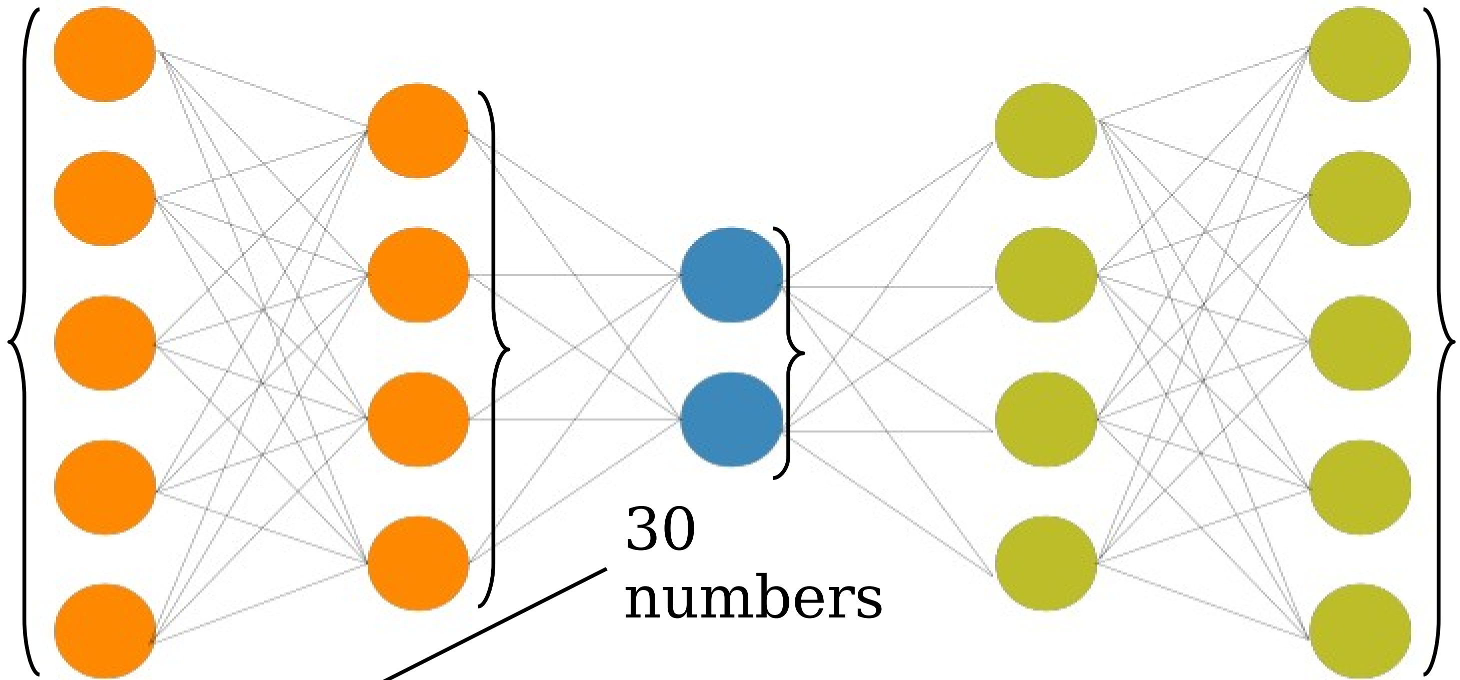
tSNE

2
numbers

Identifying Active Galactic Nuclei at z~3 from the HETDEX Survey Using Machine Learning
arXiv:2302.11092 [astro-ph.GA]

Example: astronomy

914 numbers



914 numbers

30 numbers

tSNE

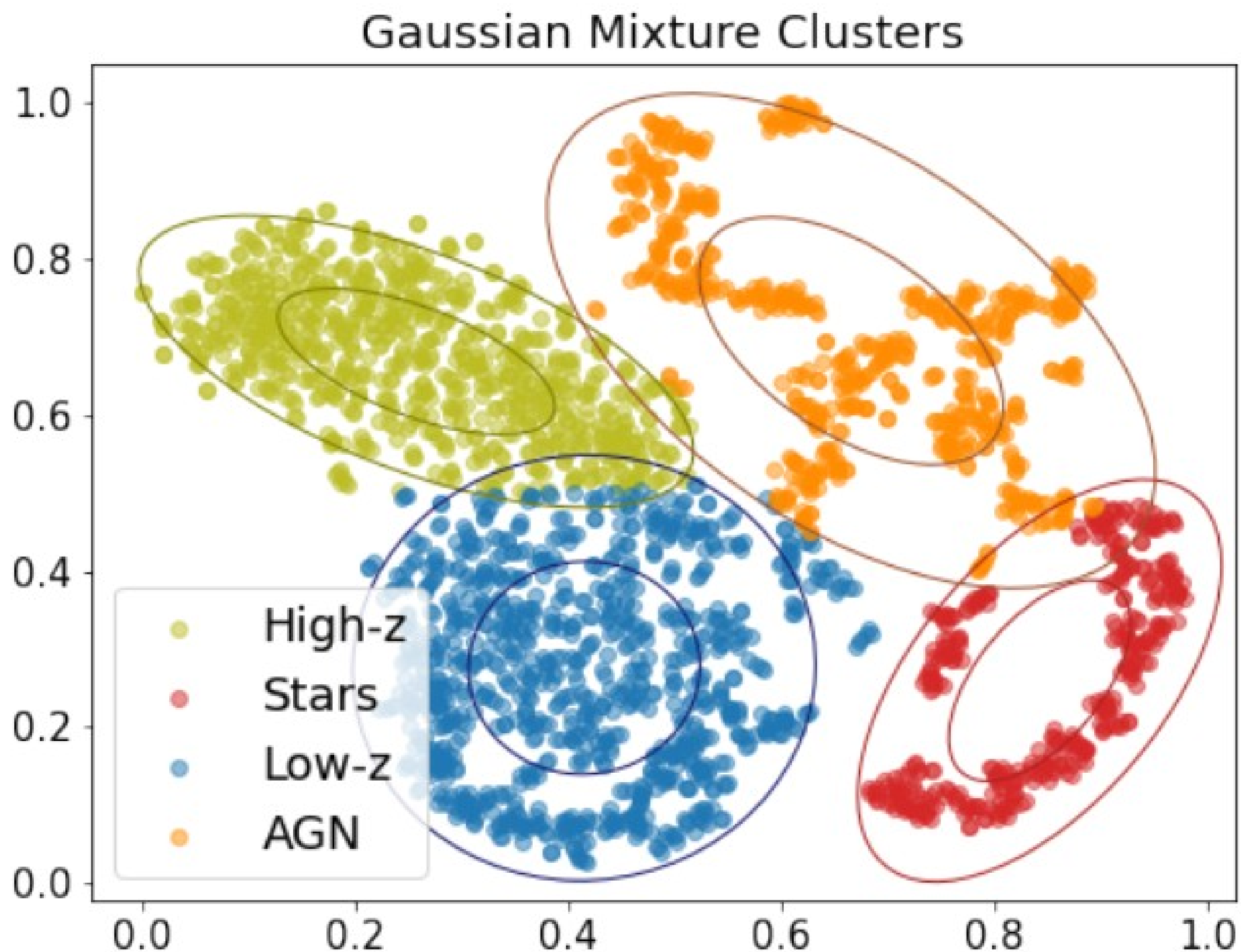
2 numbers

Gaussian mixture – one Gaussian for each category

$$p(x_1, x_2) = \sum_{i=0}^{3} \alpha_i \mathrm{Gauss}(\vec{\mu}_i, \mathrm{Cov}_i)$$

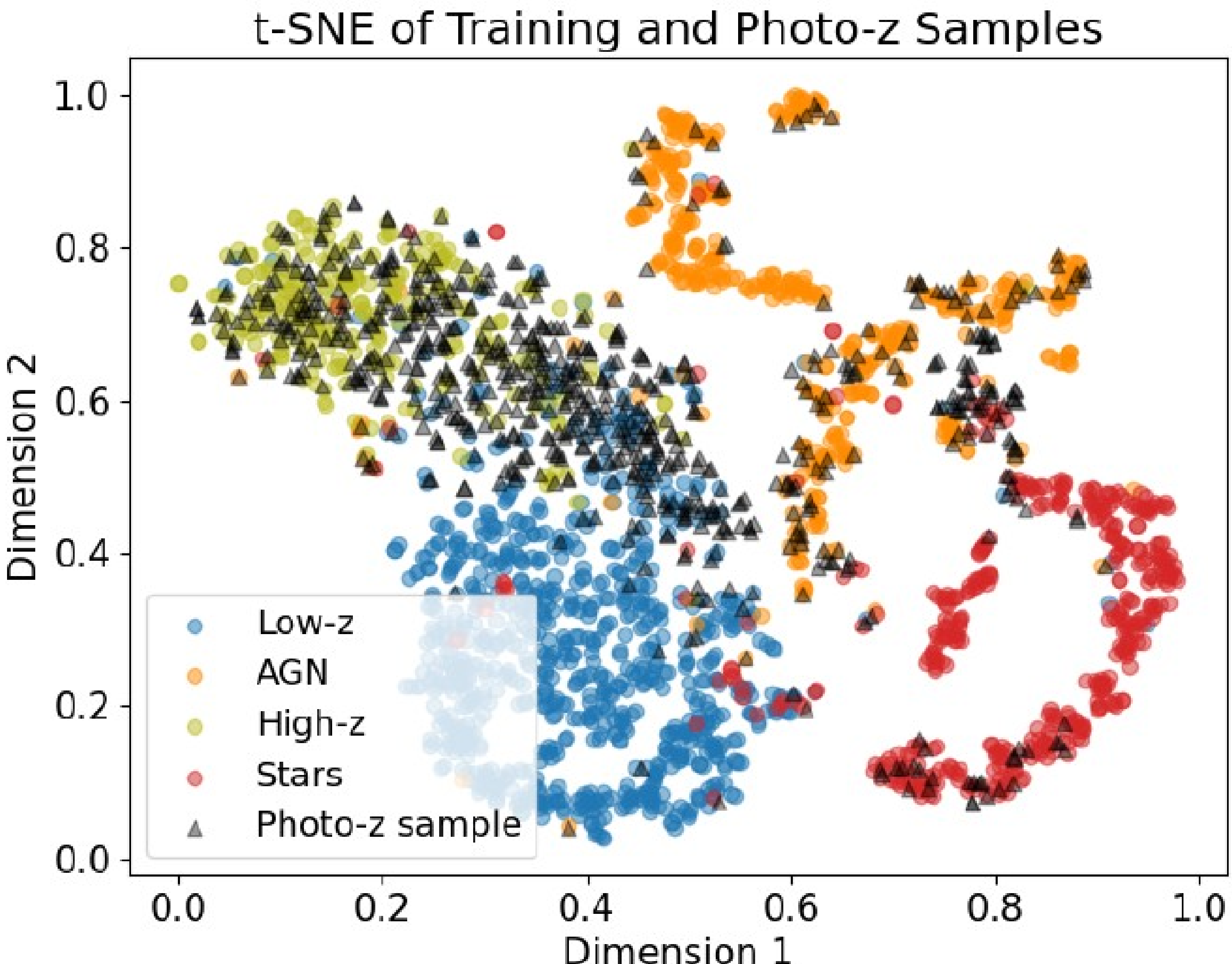
Identifying Active Galactic Nuclei at z~3 from the HETDEX Survey Using Machine Learning
arXiv:2302.11092 [astro-ph.GA]

Example: astronomy



Identifying Active Galactic Nuclei at z~3 from the HETDEX Survey Using Machine Learning
arXiv:2302.11092 [astro-ph.GA]

Example: astronomy



Identifying Active Galactic Nuclei at z~3 from the HETDEX Survey Using Machine Learning
arXiv:2302.11092 [astro-ph.GA]

Example: HEP

Warsaw CMS group work

The task:

muon transverse momentum calculation (p_T)

Input data:

• "hit pattern" - muon position at four points, momentum direction in two points

Datasets:

Training:

- 5.5 M muons with various p_T

Testing:

- 0.5M muons with various p_T

Naive Bayes:

- calculate hit configuration likelihood assuming hit positions are independent between layers:

$$L(\text{hit configuration}|p_T) = \prod_{\text{layers}} p(\text{hit position in layer i}|p_T)$$

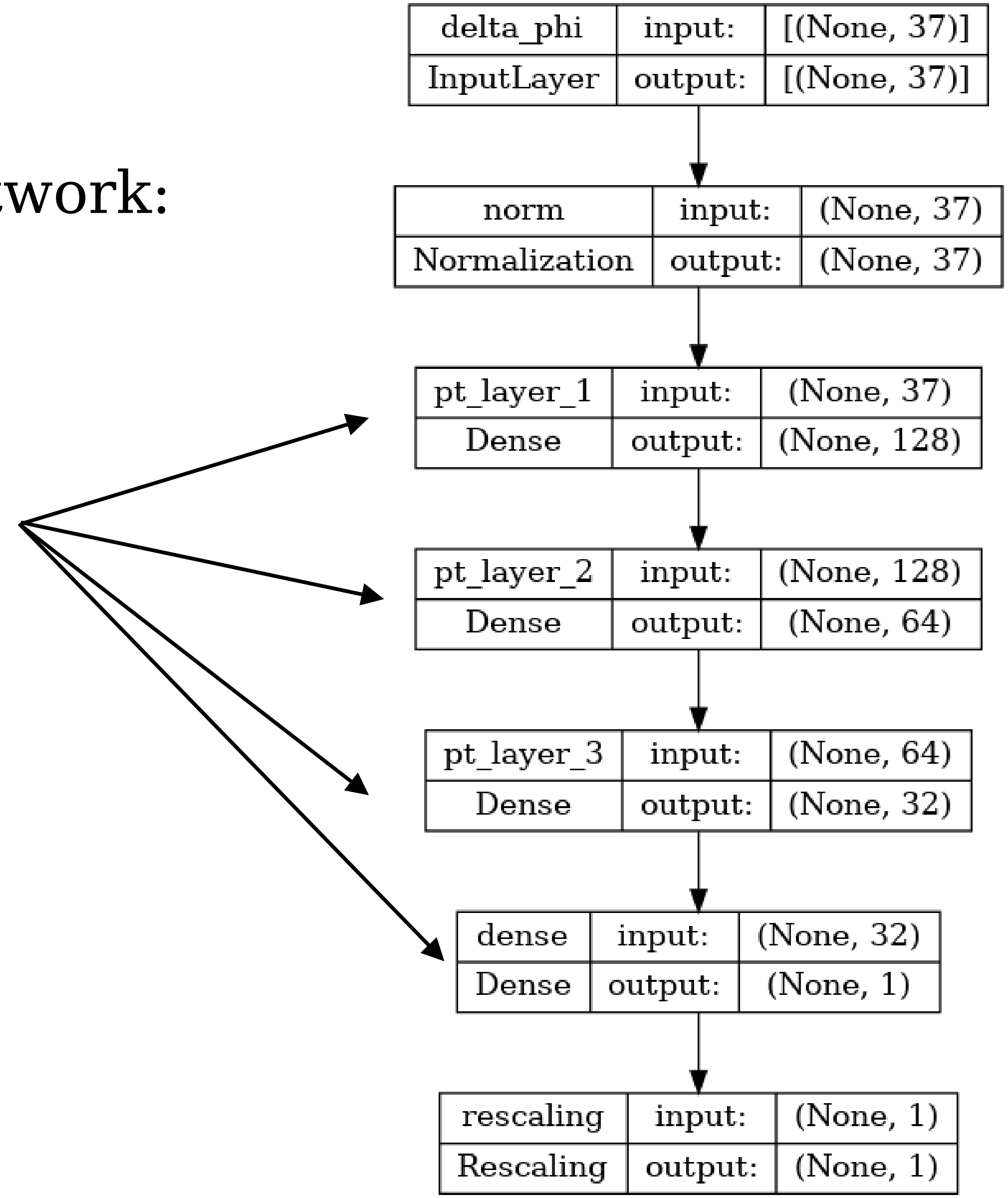
- select p_T giving the largest likelihood value

Example: HEP

Machine Learning:

- very standard fully connected network:

4 layers of neurons



Example: HEP

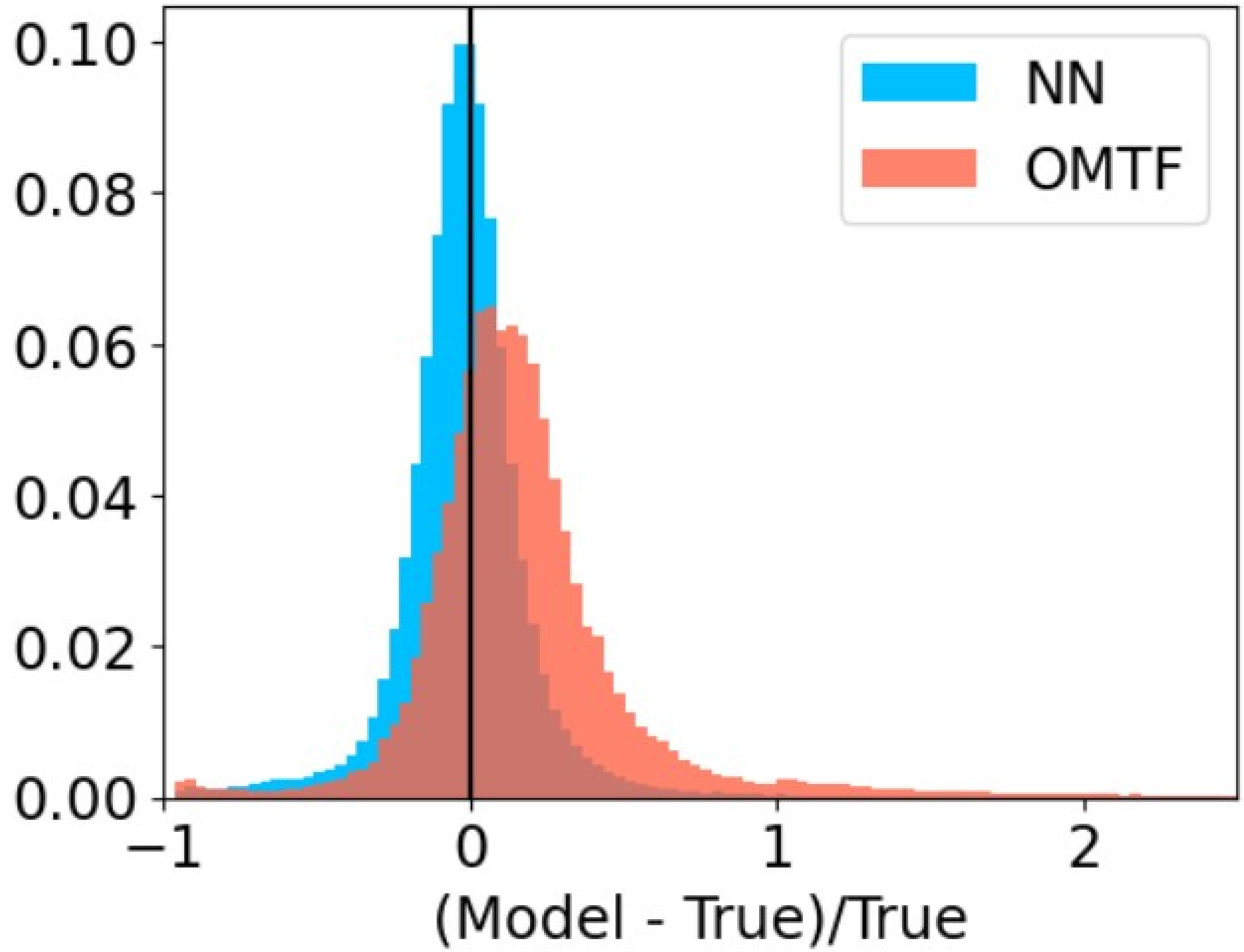
NN value centered at true value:

- good for measurement

$$< p_T^{measured} >= p_T^{true}$$

- bad for event selection

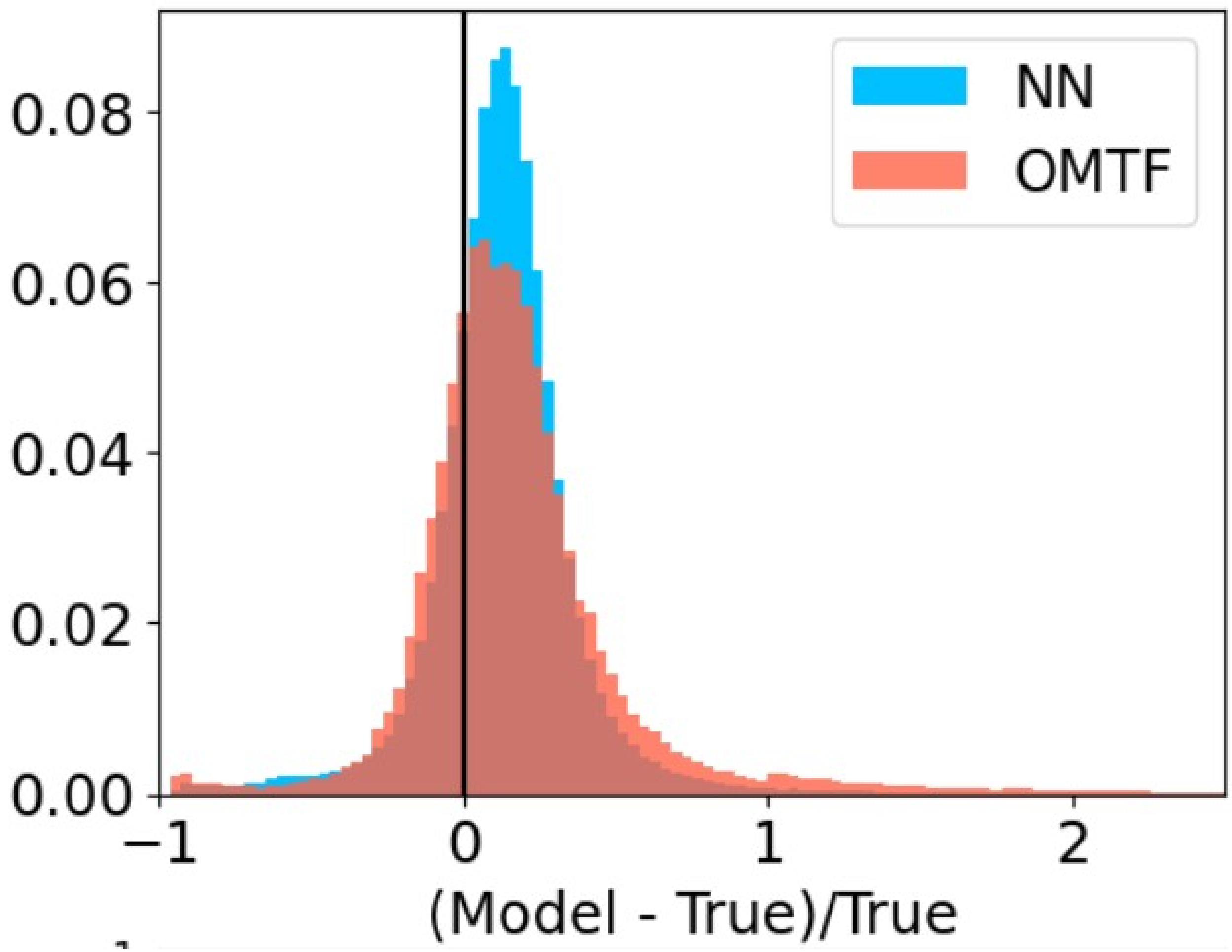
$$P(p_T^{measured} \geq p_T^{true}) = 0.5$$



Example: HEP

NN value centered at true value → “bug/feature” cab ne easily fixed with manual scaling:

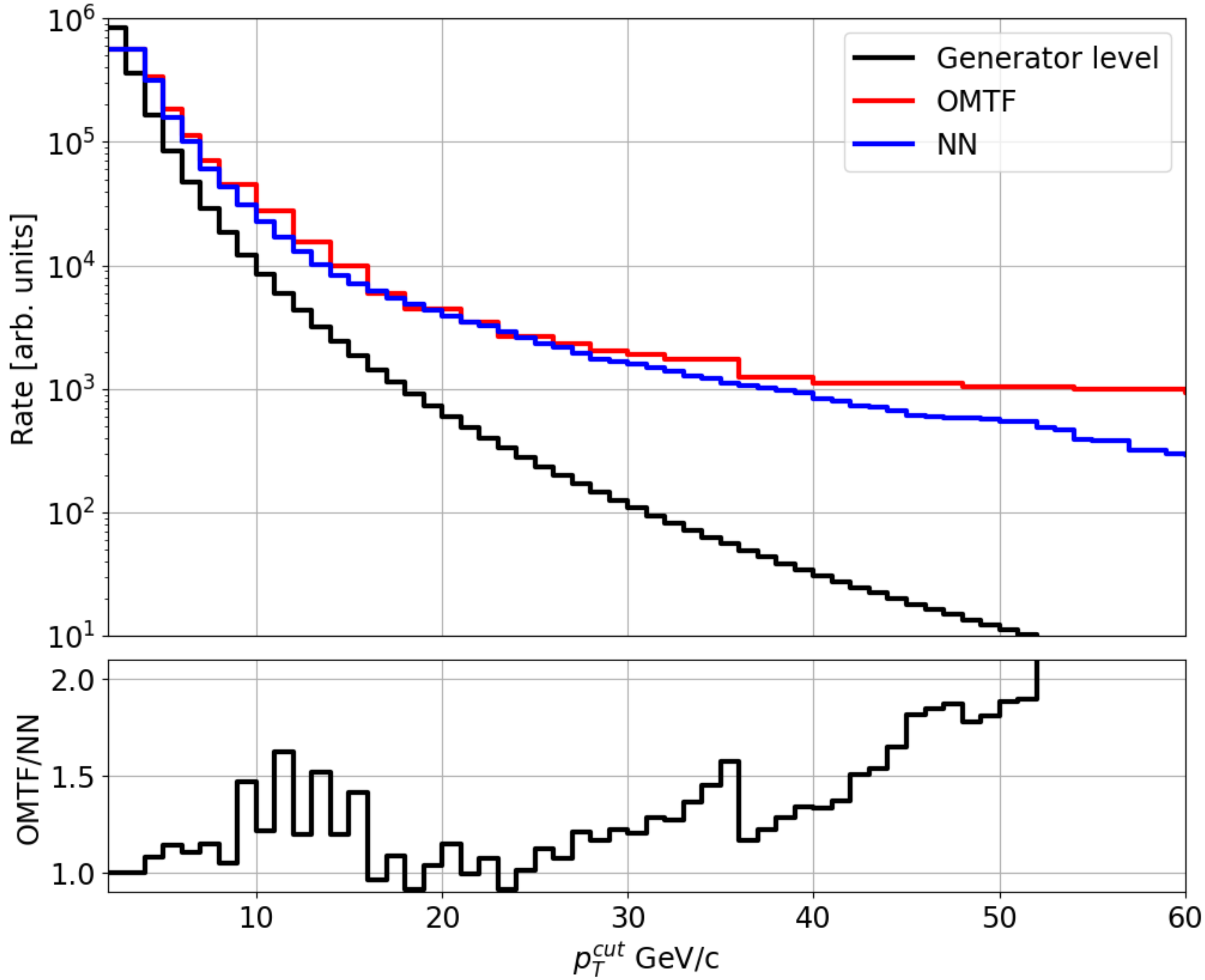
$$p_T = 1.15 \cdot p_T^{NN}$$



Example: HEP

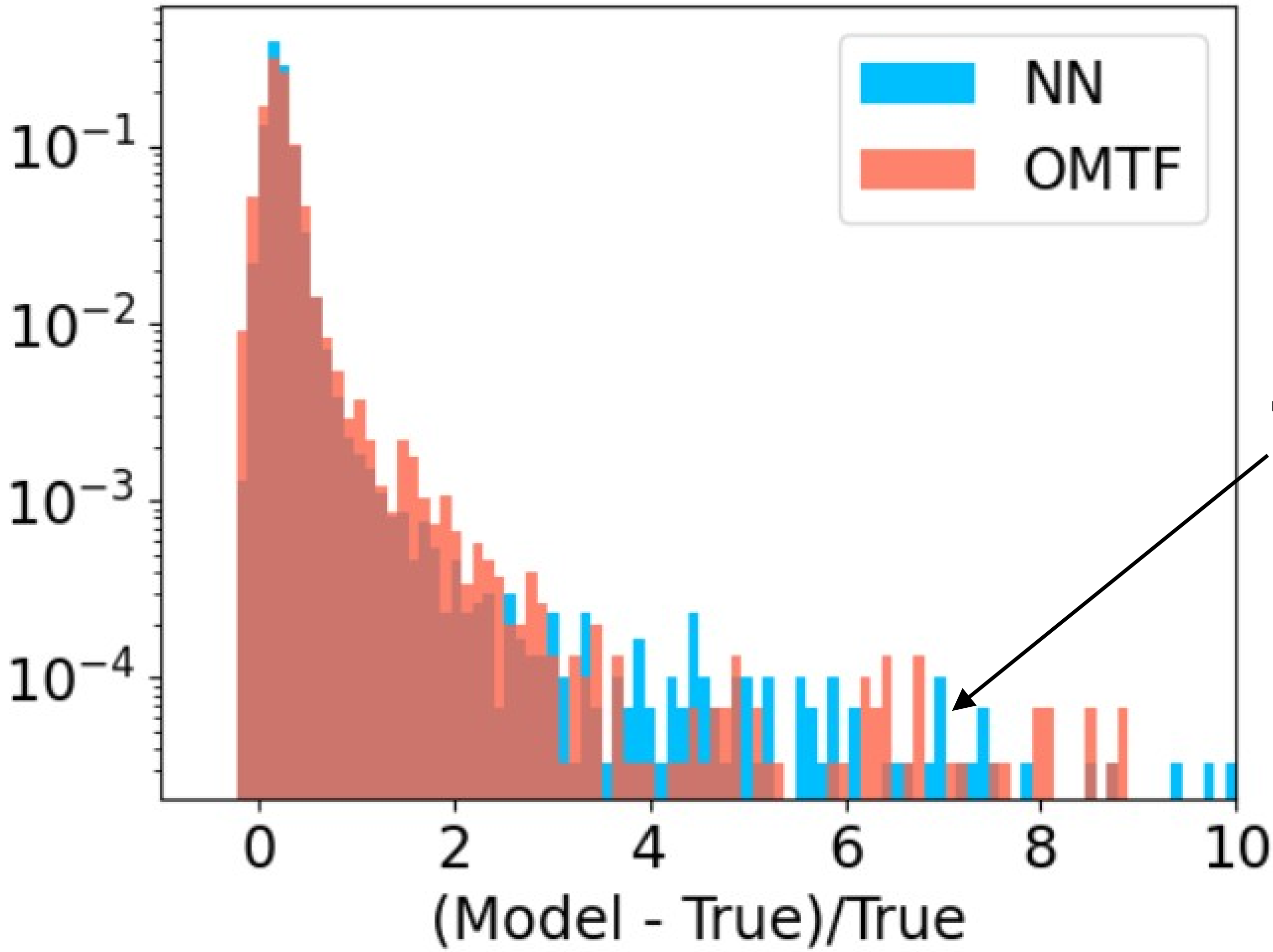
Figure of merit:
accepted event rate

$$\mathrm{R[ev/s] = N[ev/s]} : p_T^{measured} > p_T^{cut}$$



Example: HEP

$$p_T^{generated} < 5\ \mathrm{GeV/c}$$

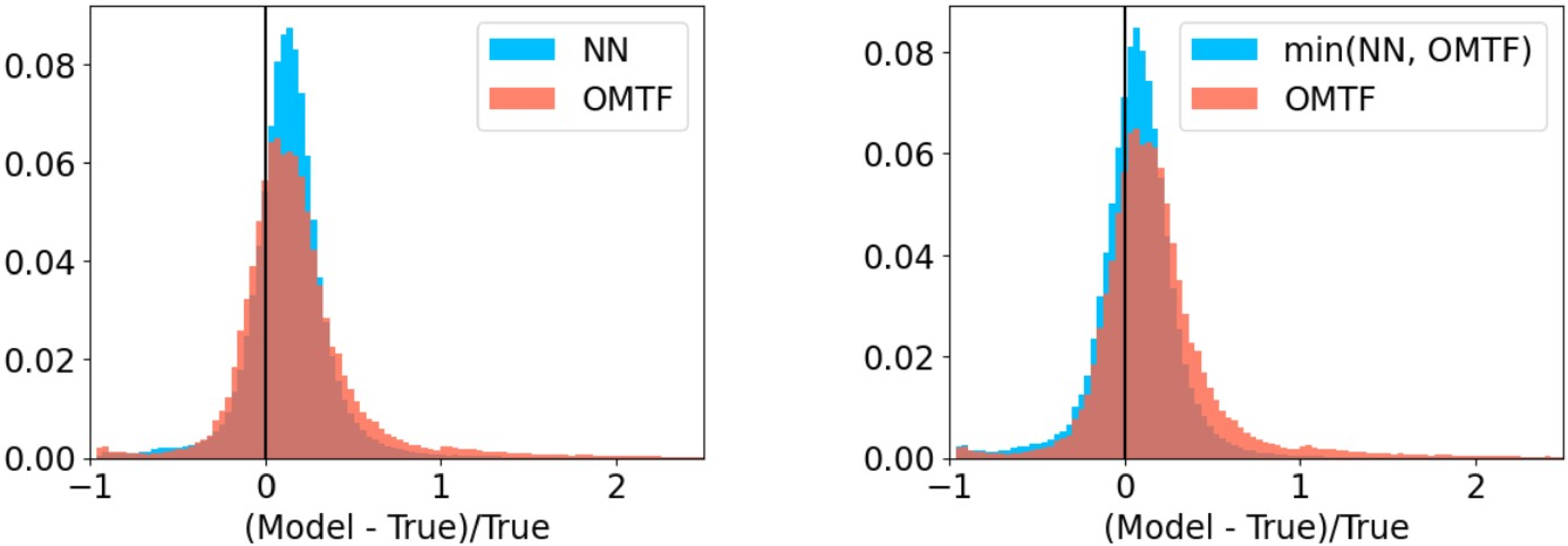


The rate is dominated by highly overestimated low p_T muons

Example: HEP

Solution:

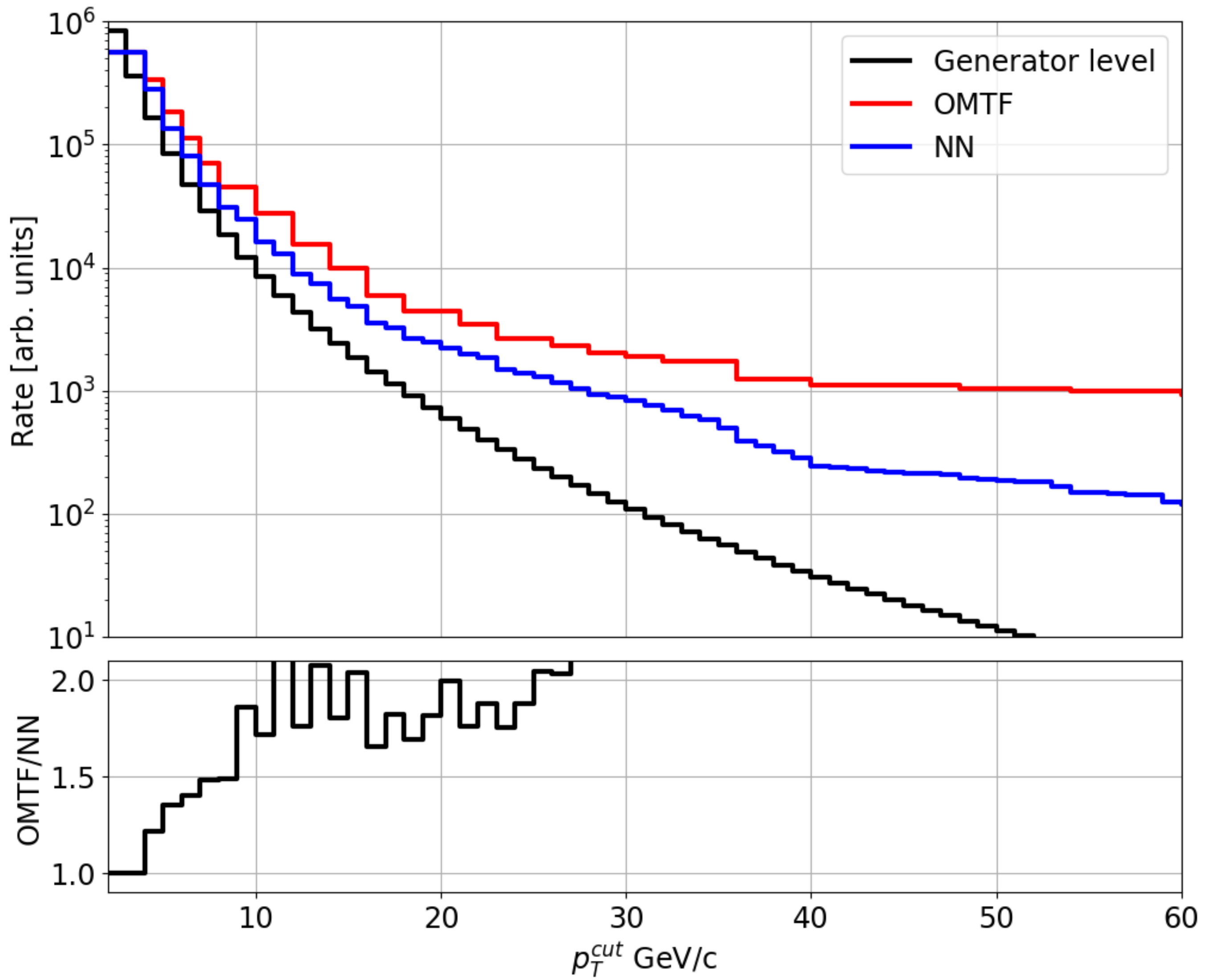
$$p_T^{measured} = min(\text{Naive Bayes},\ \text{NN})$$



Example: HEP

Figure of merit:
accepted event rate

$$\mathrm{R[ev/s]} = \mathrm{N[ev/s]} : p_T^{measured} > p_T^{cut}$$



Summary

- Machine learning is not a magic ward – this is yet another technology

but keep in mind what Arthur C. Clare said: “Any sufficiently advanced technology is indistinguishable from magic.”

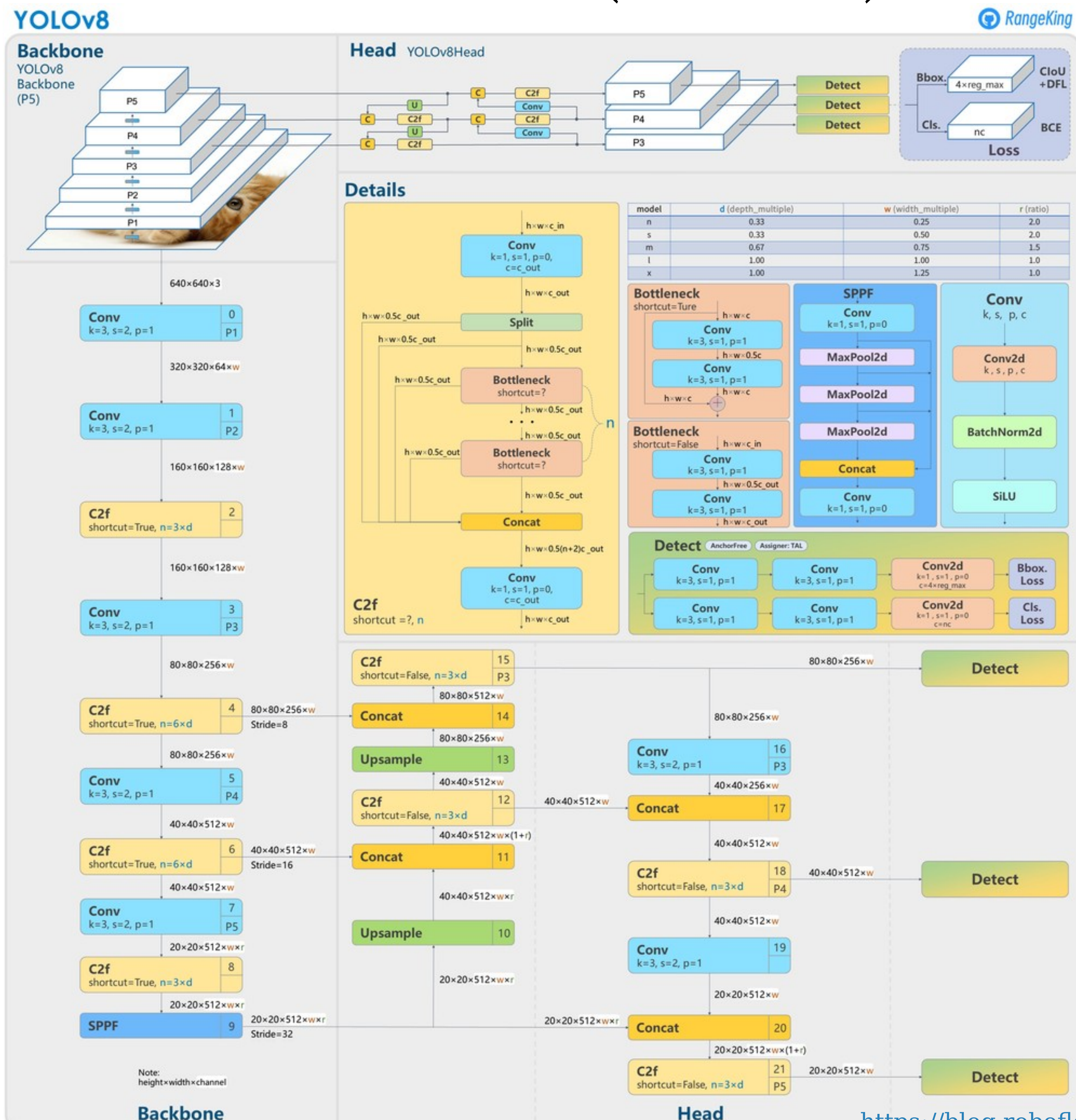
- „ordinary” ML users should concentrate on creative problem formulation instead of attempting to invent a new, complicated, architecture

- for some time ML algorithms will require a human assist in result post processing

but I do not think this will be longer than LHC Run 5 time scale

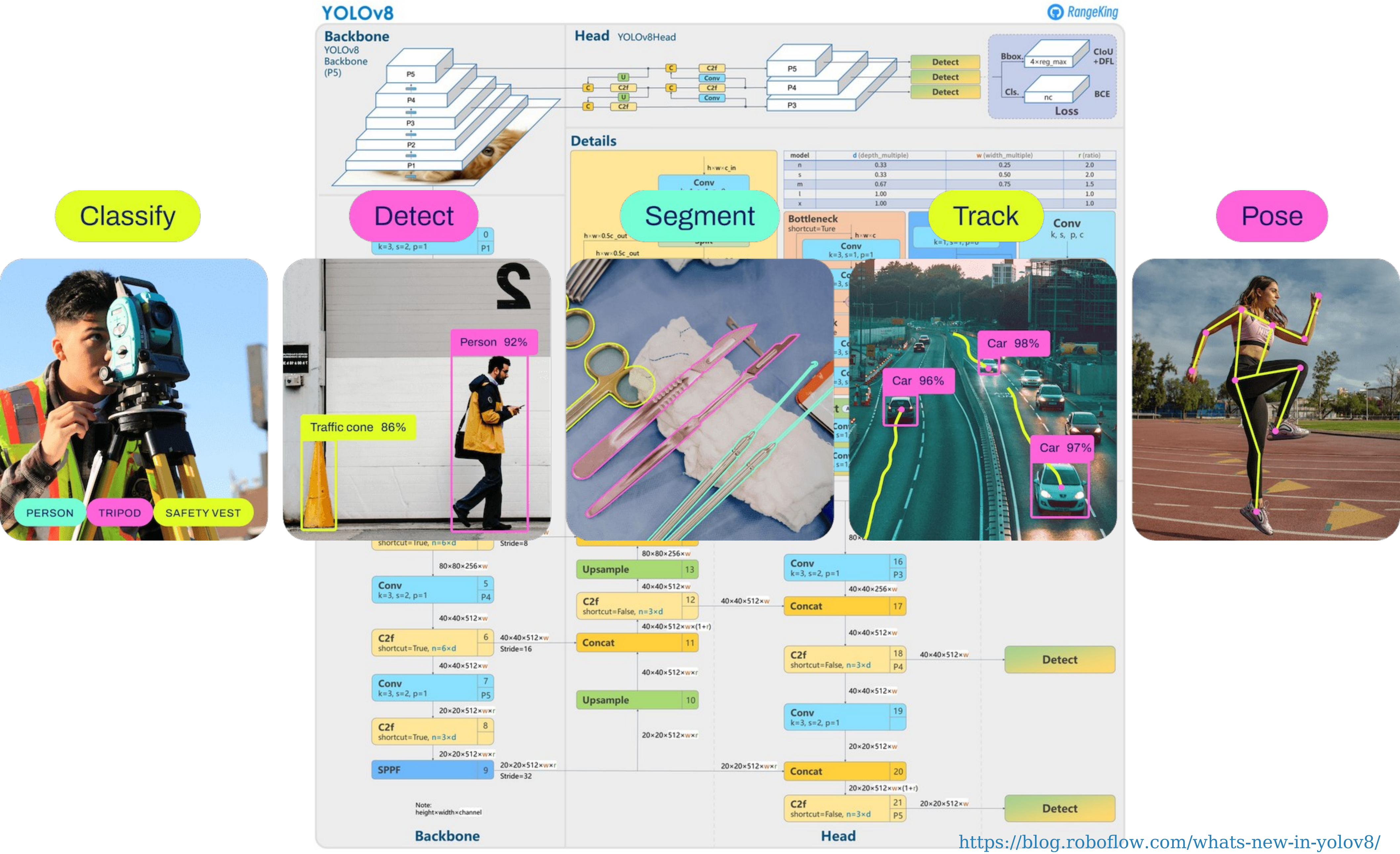
Backup slides

Architecture: You Only Look Once (YOLO)



https://blog.roboflow.com/whats-new-in-yolov8/

You Only Look Once (YOLO)



https://blog.roboflow.com/whats-new-in-yolov8/

You Only Look Once (YOLO)

Classification

Model	size (pixels)	acc top1	acc top5	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B) at 640
YOLOv8n-cls	224	66.6	87.0	12.9	0.31	2.7	4.3
YOLOv8s-cls	224	72.3	91.1	23.4	0.35	6.4	13.5
YOLOv8m-cls	224	76.4	93.2	85.4	0.62	17.0	42.7
YOLOv8l-cls	224	78.0	94.1	163.0	0.87	37.5	99.7
YOLOv8x-cls	224	78.4	94.3	232.0	1.01	57.4	154.8

Analysis of 1M events takes 64 CPU h

https://blog.roboflow.com/whats-new-in-yolov8/

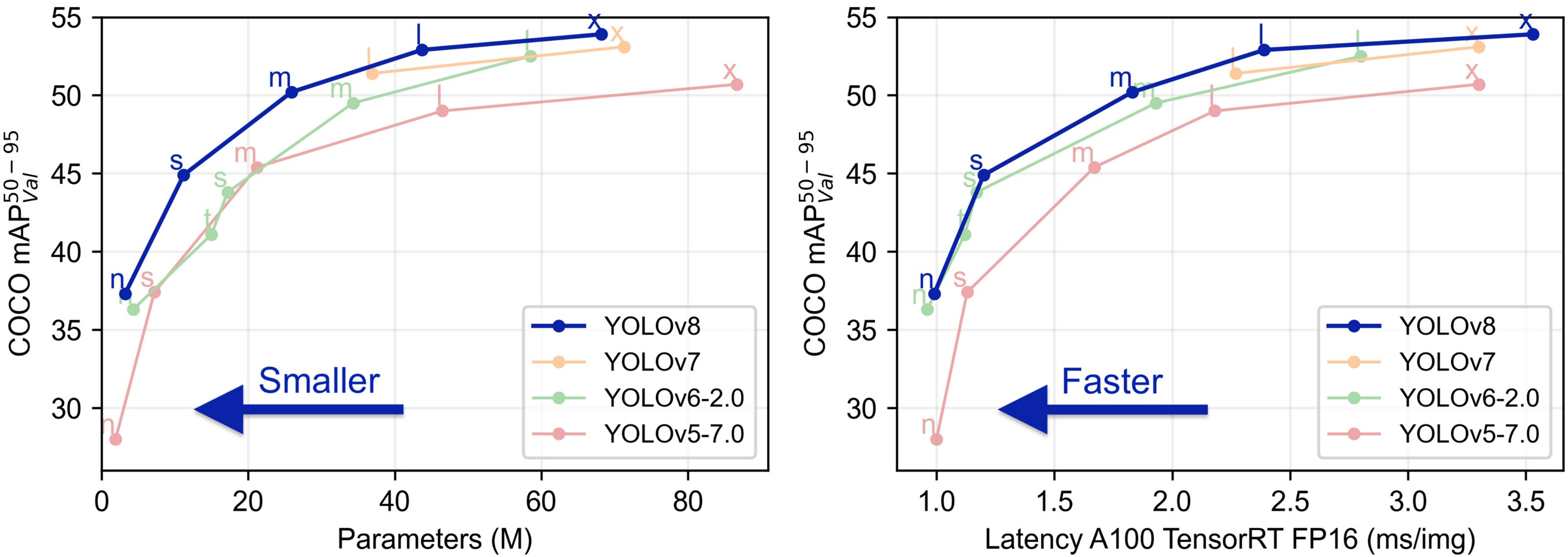
You Only Look Once (YOLO)

Detection

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

https://blog.roboflow.com/whats-new-in-yolov8/

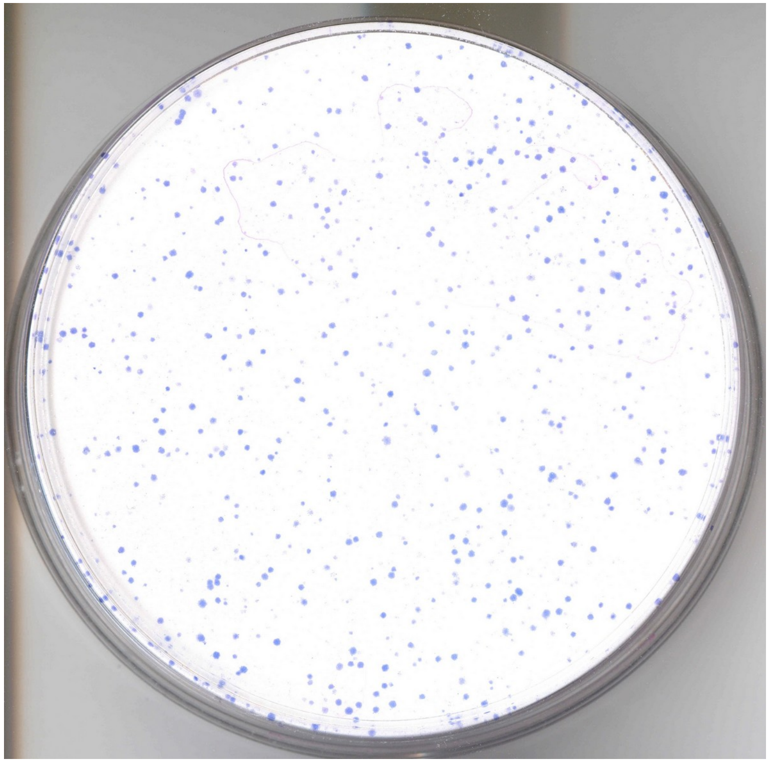
You Only Look Once (YOLO)



https://blog.roboflow.com/whats-new-in-yolov8/

Example: medical physics

Task: cell colony counting



Input data:

- 57 Petri dish high resolution (4390x5059) photos

Datasets:

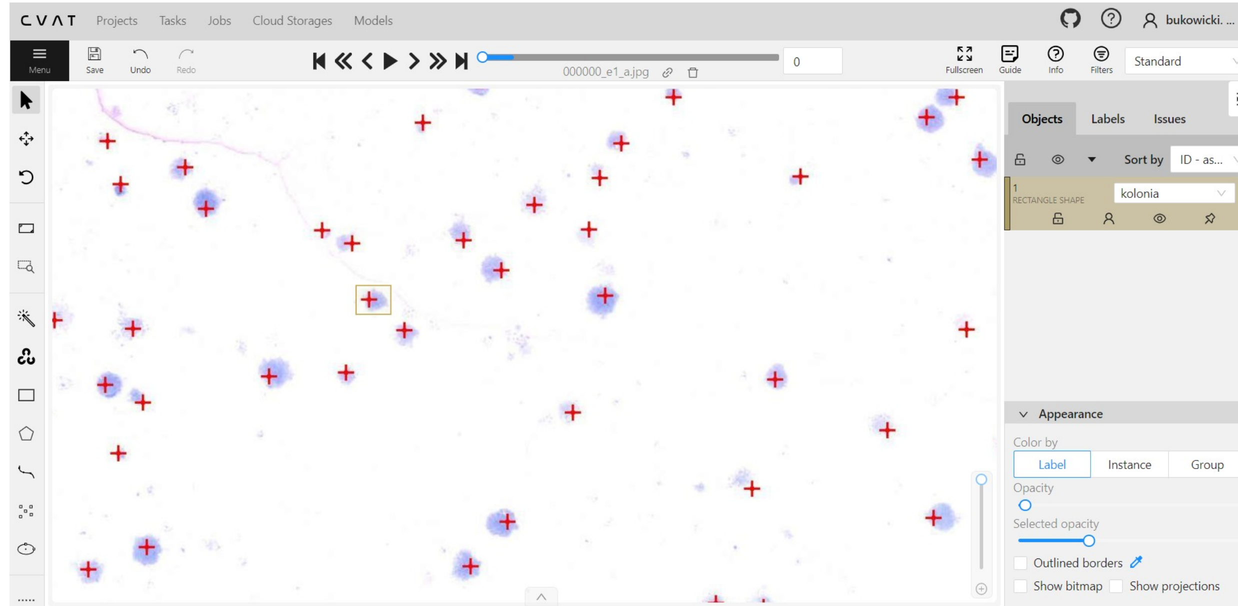
Training and validation: 243 + 73 slices of 640x640

Testing: 32 slices

Zastosowanie uczenia maszynowego do automatycznej analizy testu klonogennego przeprowadzonego na komórkach ssaków

YOLO : procedure

- labeling



- training (single GPU, Google Colaboratory)

```
# Build a new model from YAML, transfer pretrained weights to it and start training
yolo detect train data=coco128.yaml model=yolov8n.yaml pretrained=yolov8n.pt epochs=100 imgsz=640
```

YOLO : procedure

Center4ML

- prediction with a 640x640 window

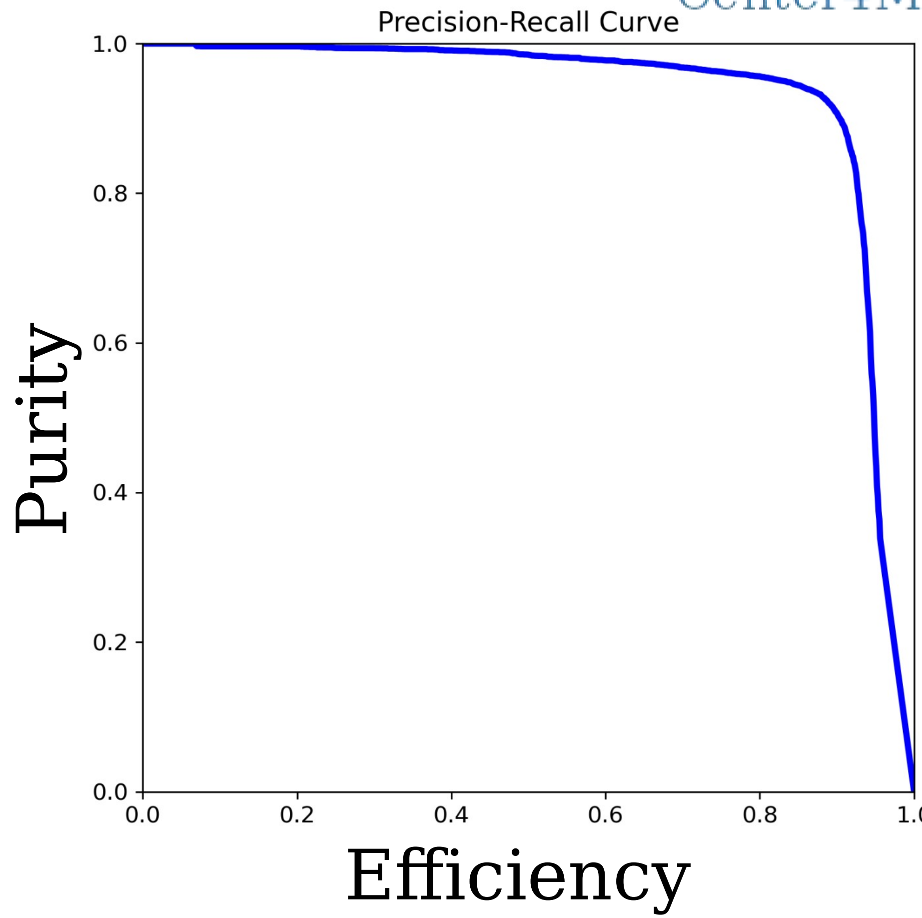
Zastosowanie uczenia maszynowego do automatycznej analizy testu klonogennego przeprowadzonego na komórkach ssaków

YOLO : procedure

- merged image

YOLO : procedure

- the ROC is close to ideal



YOLO provides a convenient user interface easing the training and prediction steps

Center4ML is a IDUB funded service that provide a ready to use ML models for any UW based group.

The service is free of charge.

Adv. break: Center4ML

Center4ML is a IDUB funded service that provides a ready to use ML models for any UW based group.

The service is **free of charge.**