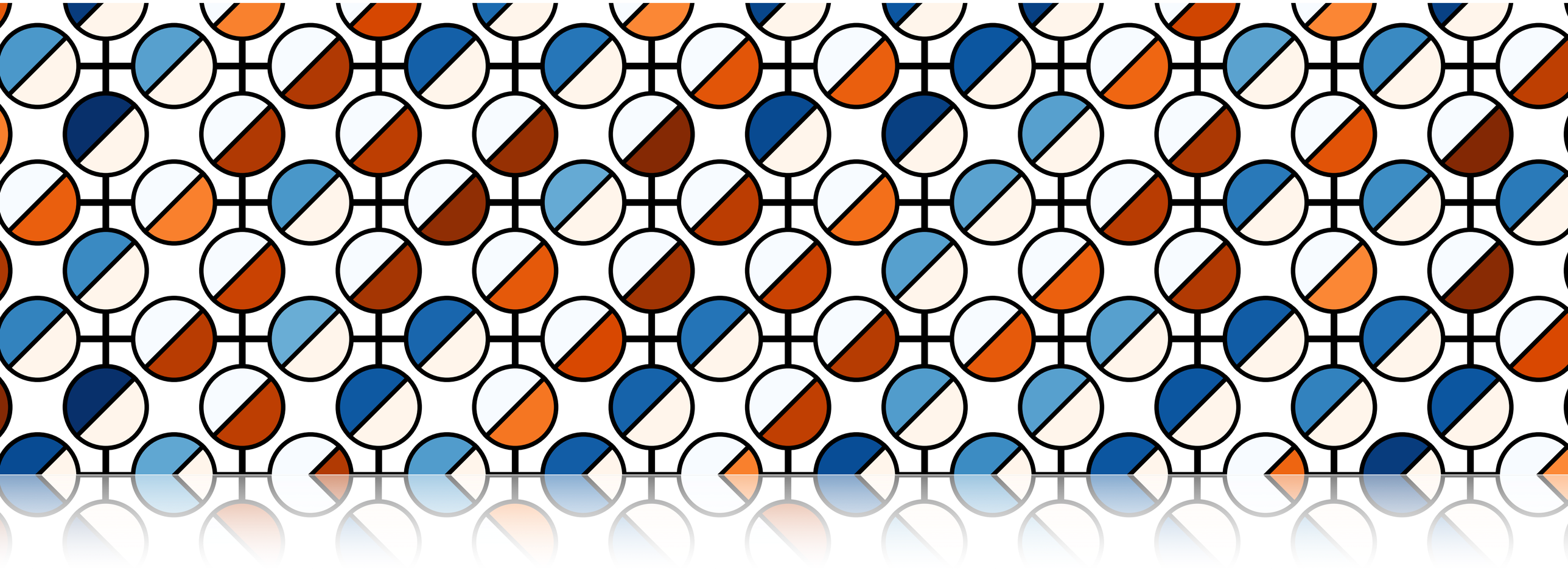


Introduction to machine learning for condensed matter

ML4QT Erlangen

Eliška Greplová, Kavli Institute of Nanoscience Delft

TU Delft Delft University of Technology

QMAI

QuTech

KAVLI INSTITUTE of Nanoscience Delft

Quantum Matter and AI Lab

eliskagreplova.com

What we'll learn

- (classical) ML + quantum: WHY?
- (classical) ML + quantum: HOW?
- Can we machine-learn physics?

Minimal toy models

X

Data-driven learning

Quantum physics: Why is it hard?

2 SPINS = SIZE 4

100 SPINS = SIZE 10^30

ψ

10 SPINS = SIZE 1024

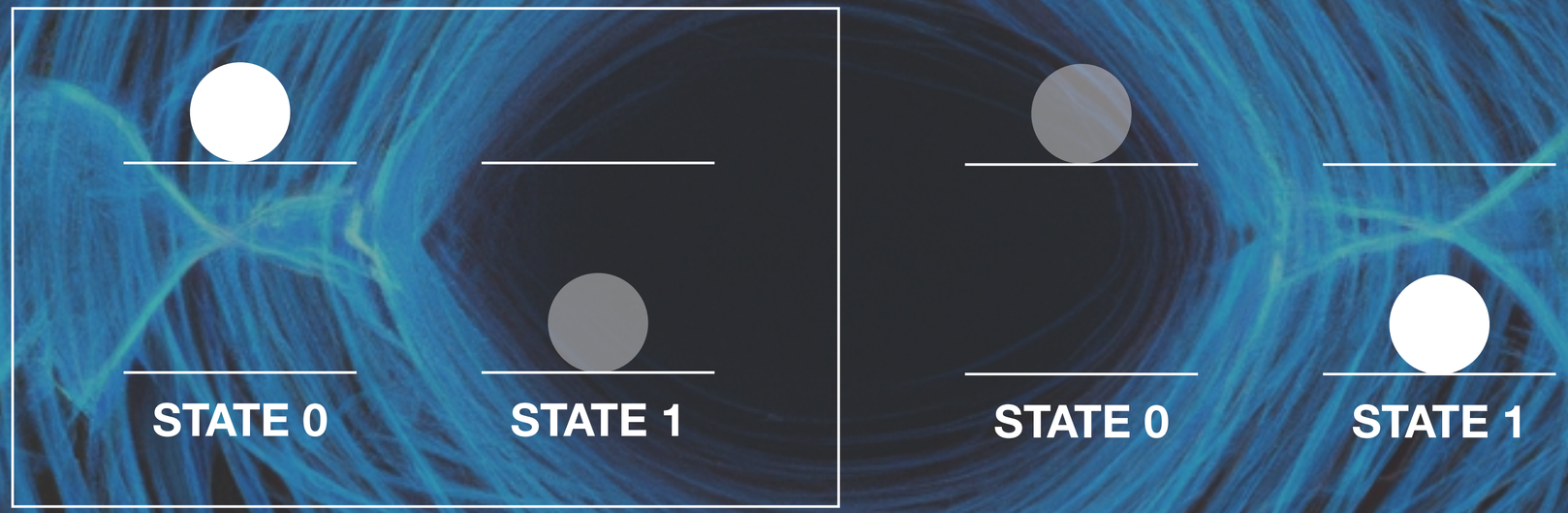
1000 SPINS = SIZE 10^300

50 SPINS = SIZE 10^15

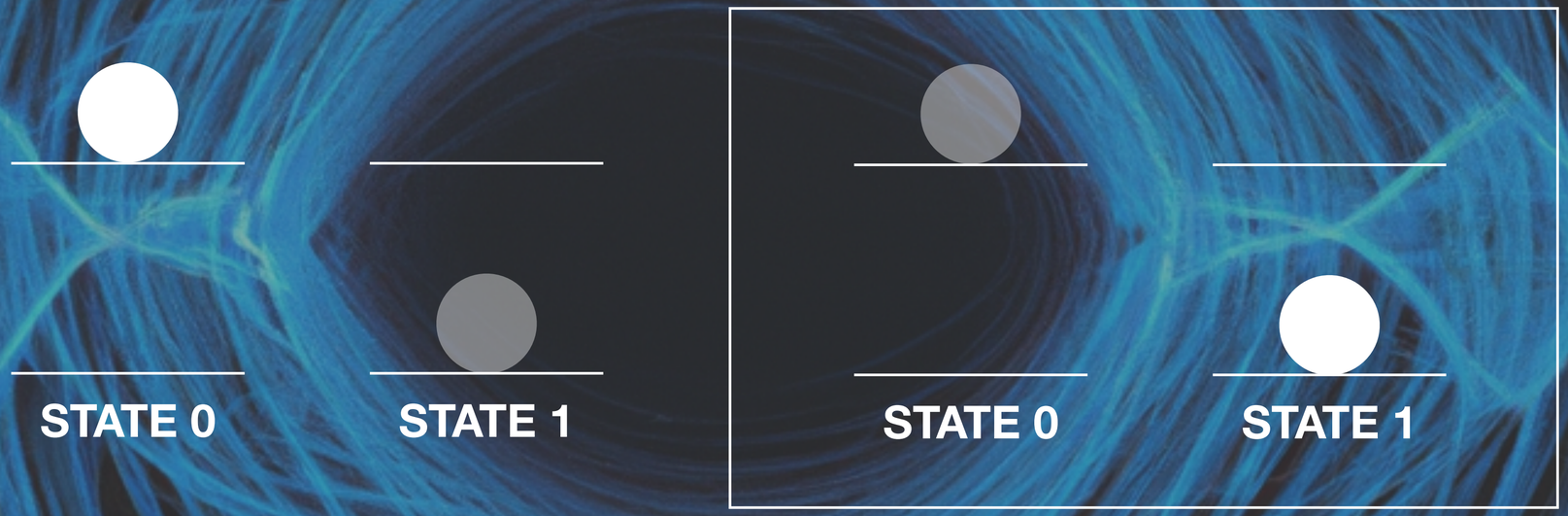
QMAI | TU Delft

Quantum Bits: Qubits

Classical states



Classical states

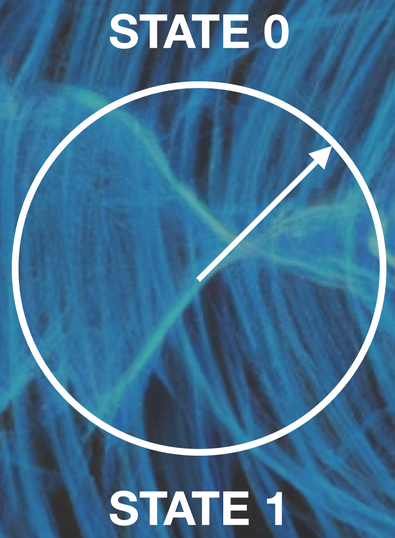


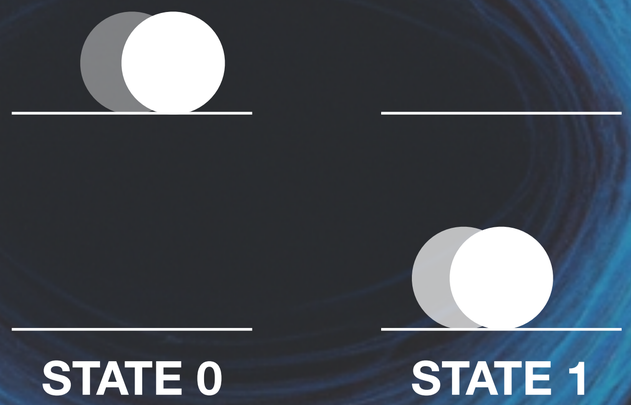
Classical states

2 different state ~ 2 numbers
N different states ~ N numbers

Quantum states

QMAI | TU Delft

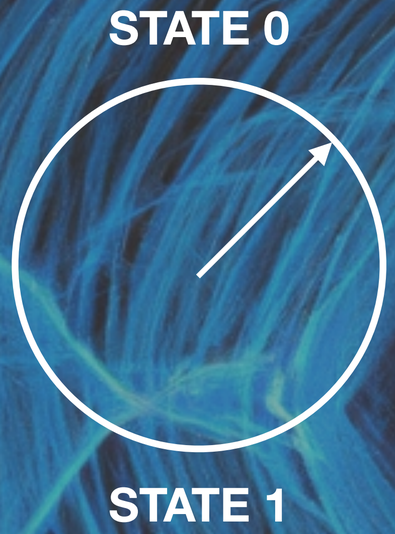




SUPERPOSITION
a (STATE 0) + b (STATE 1)

Quantum states

QMAI | TU Delft

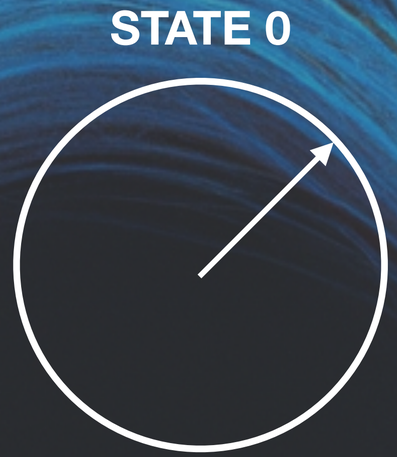


a (STATE 0) + b (STATE 1)

c (STATE 0) + d (STATE 1)

Quantum states

QMAI | TU Delft

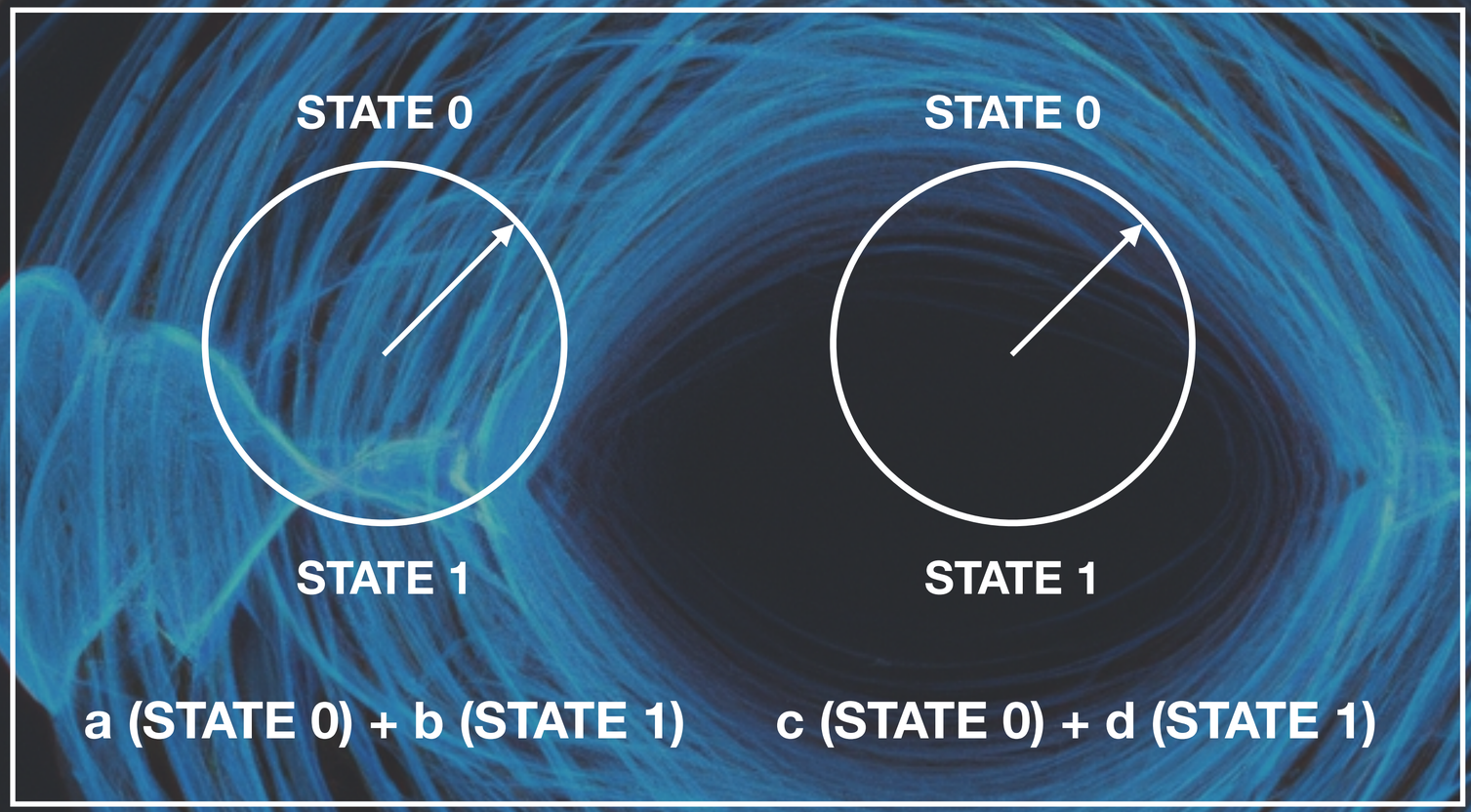


a (STATE 0) + b (STATE 1) c (STATE 0) + d (STATE 1)

COMBINED STATE:
ac (STATE 0, STATE 0) + ad (STATE 0, STATE 1) +
bc (STATE 1, STATE 0) + bd (STATE 1, STATE 1)

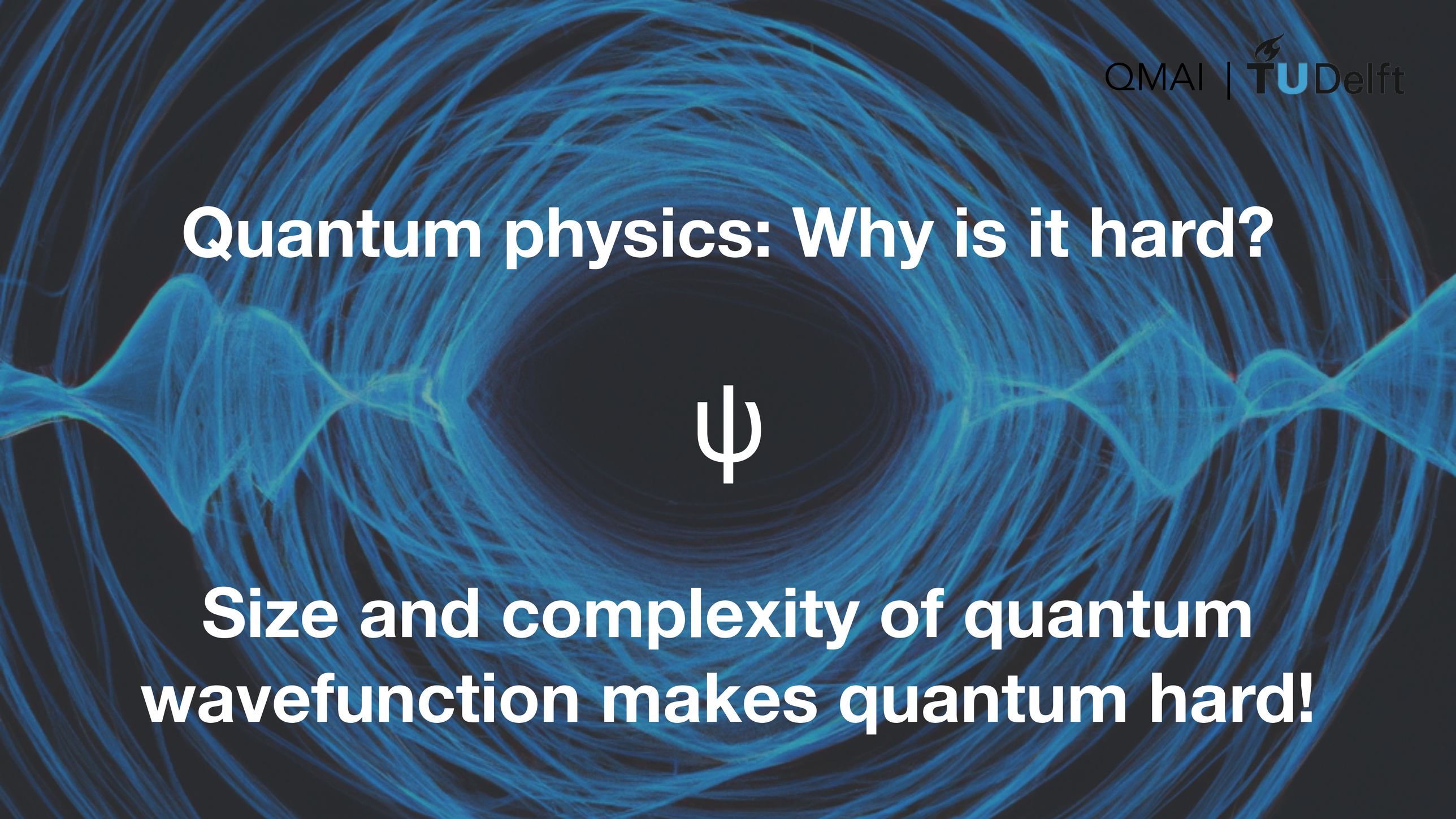
Quantum states

QMAI | TU Delft



COMBINED STATE:
ac (STATE 0, STATE 0) + ad (STATE 0, STATE 1) +
bc (STATE 1, STATE 0) + bd (STATE 1, STATE 1)

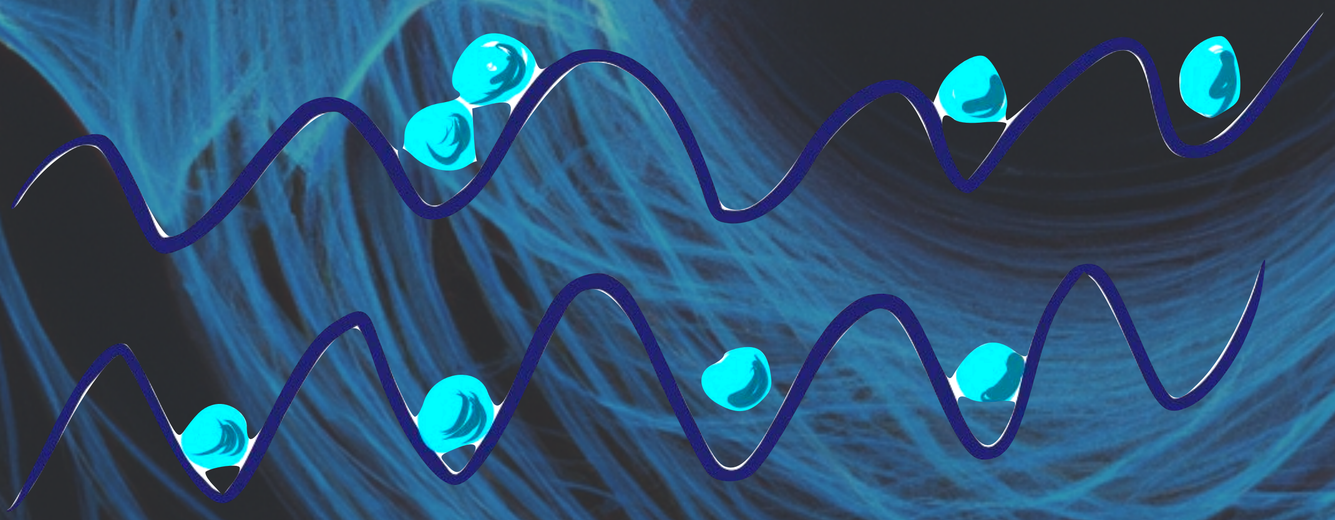
2 SYSTEMS ~ 4 NUMBERS
3 SYSTEMS ~ 8 NUMBERS
N SYSTEMS ~ 2^N NUMBERS


QMAI | TUDelft
Quantum physics: Why is it hard?
ψ
Size and complexity of quantum wavefunction makes quantum hard!

Wavefunction vs. Reality

QMAI | TUDelft

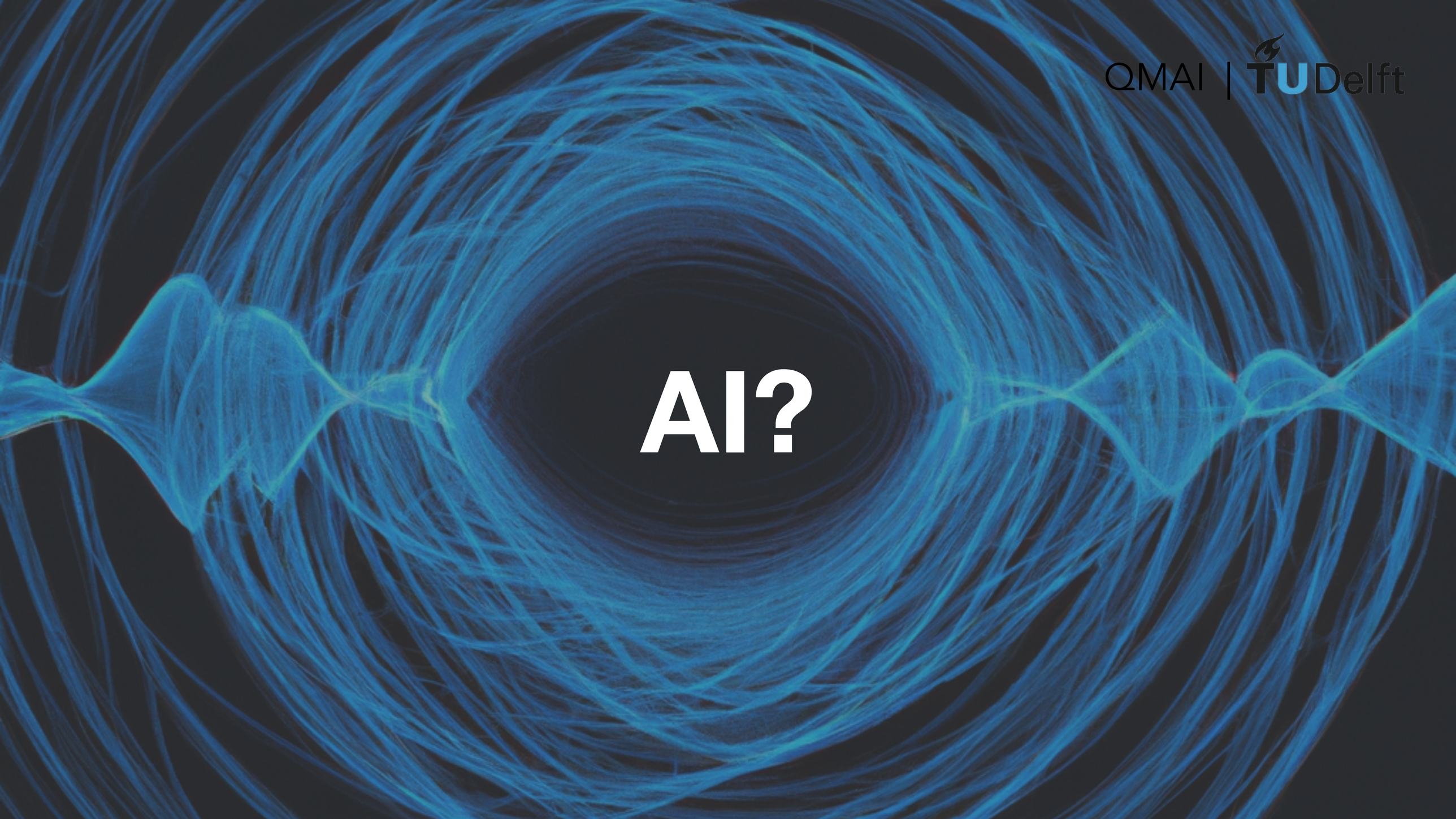
EXPERIMENT



THEORY

ψ

wave function


QMAI | TUDelft
AI?

QMAI | TU Delft

Machine Learning & Quantum Physics

Discover new physics from data

Efficiently Approximate Quantum States

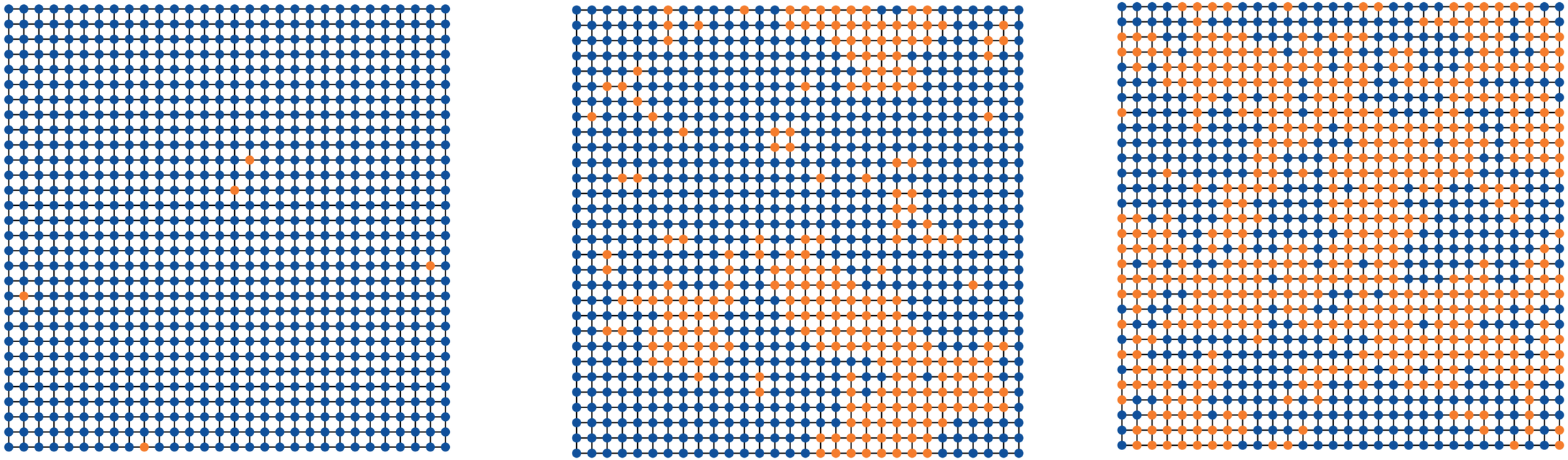
Automated Control of Quantum Devices

Benchmark quantum computers

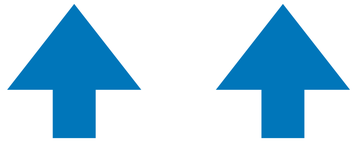
Design quantum experiments and novel materials

Discover and optimise quantum algorithms

Ising model



$$H = -J \sum_{<i,j>} \sigma_i \sigma_j$$

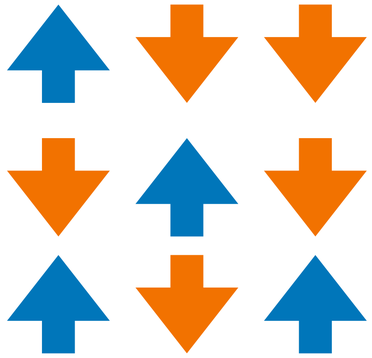


$\sigma_i \sigma_j = 1$ energy = - J

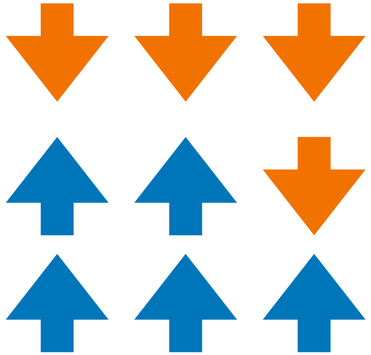
$\sigma_i \sigma_j = -1$ energy = + J

Ising QUIZ

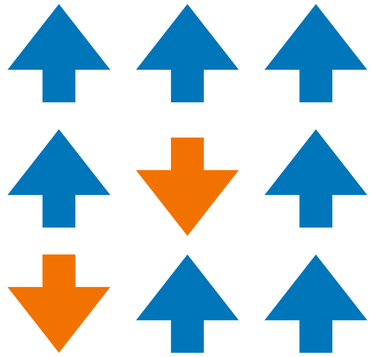
Which of these has the lowest energy?



A



B

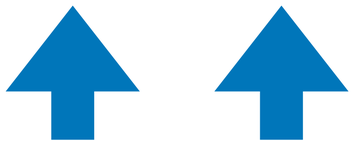


C

$$H = -J \sum_{<i,j>} \sigma_i \sigma_j$$

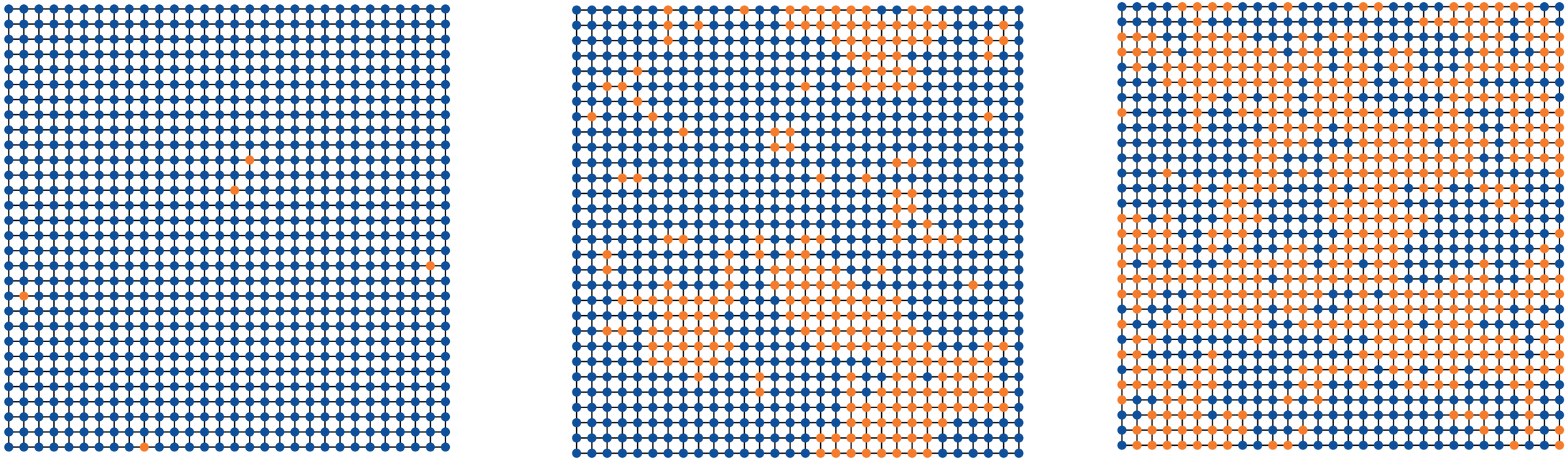
$$p(\sigma) = \frac{e^{-\beta H(\sigma)}}{Z}$$

= probability of configuration σ at the temperature T = 1/β

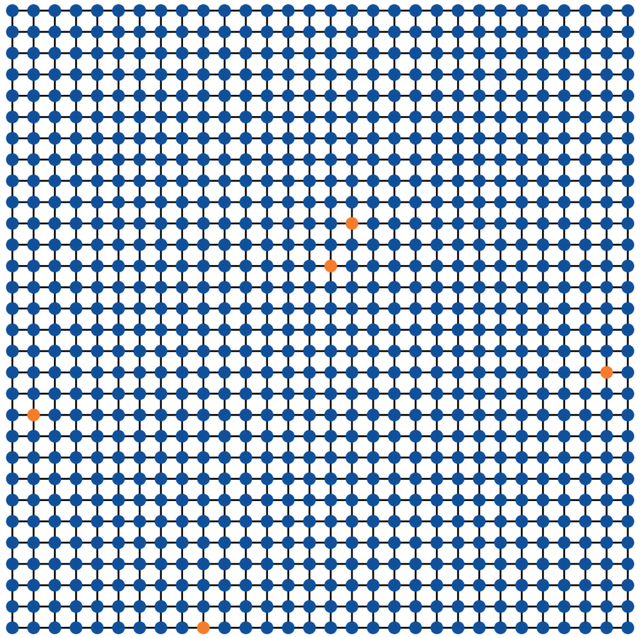


adding temperature makes the difference smaller

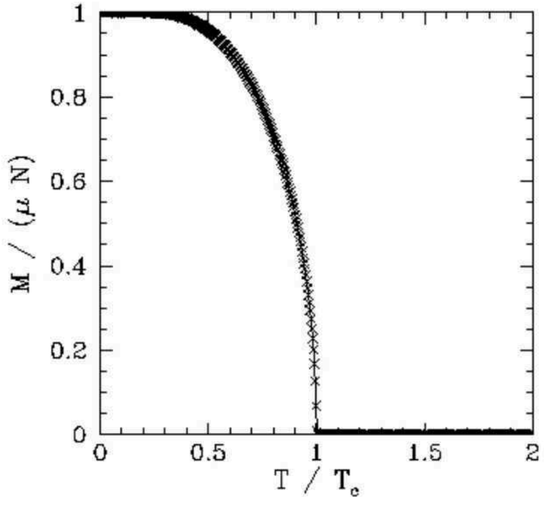
Ising model: Temperature phase transition

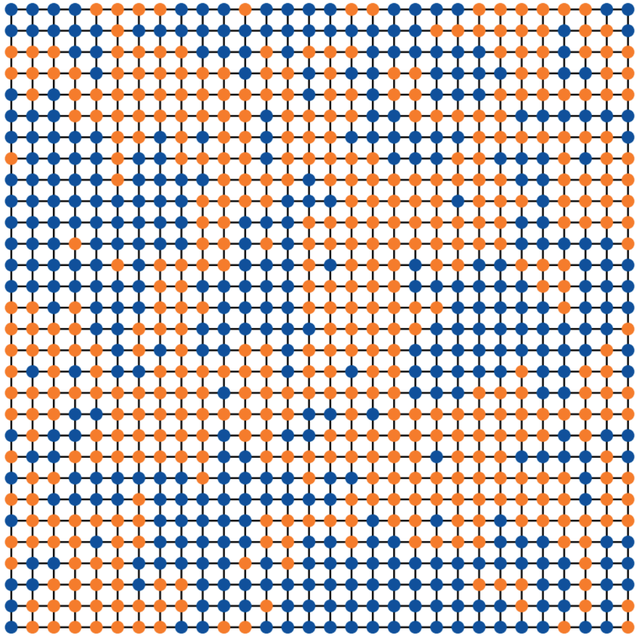


Temperature Phase Transition

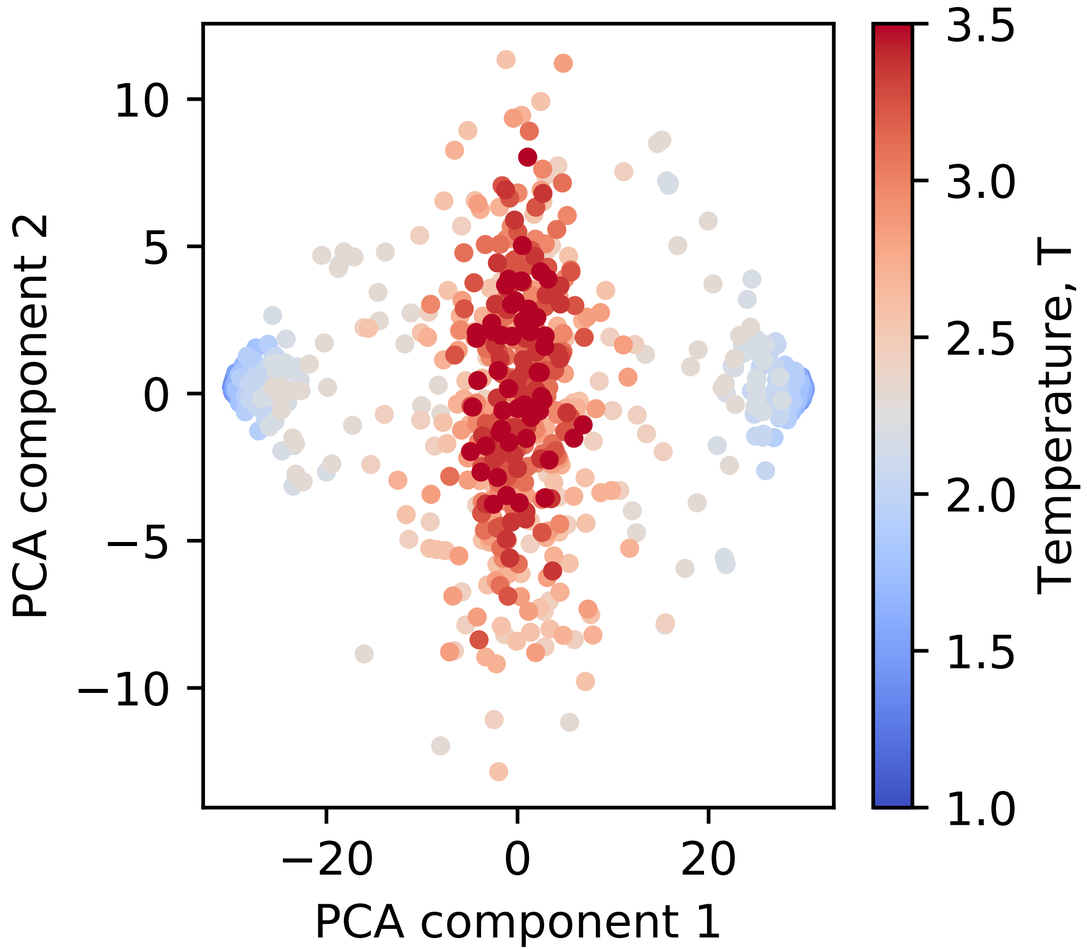


$$T_c = \frac{2J}{k \ln(1+\sqrt{2})}$$





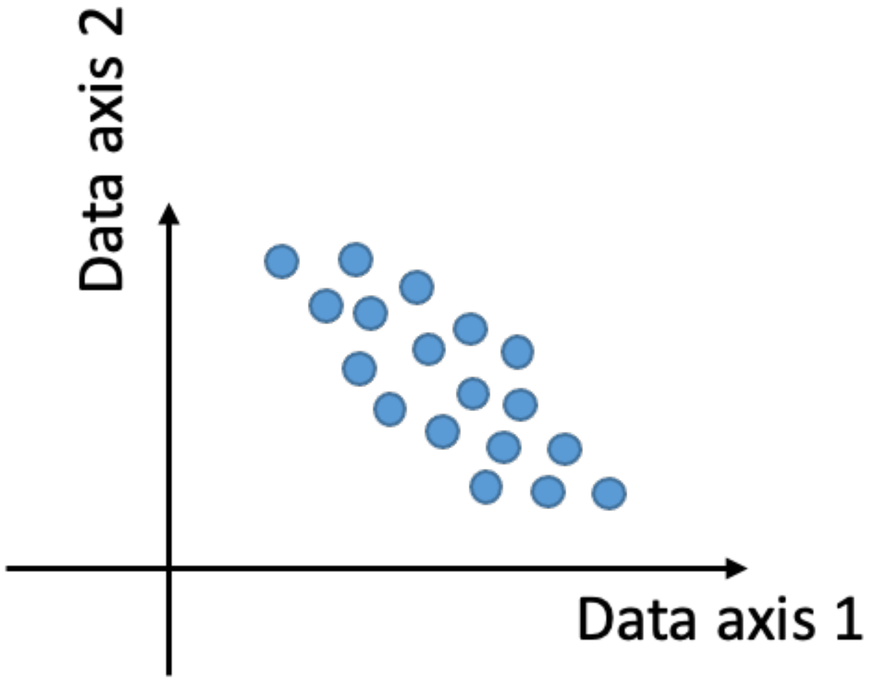
Simple clustering algorithm



Lei Wang, PHYSICAL REVIEW B 94, 195105 (2016)

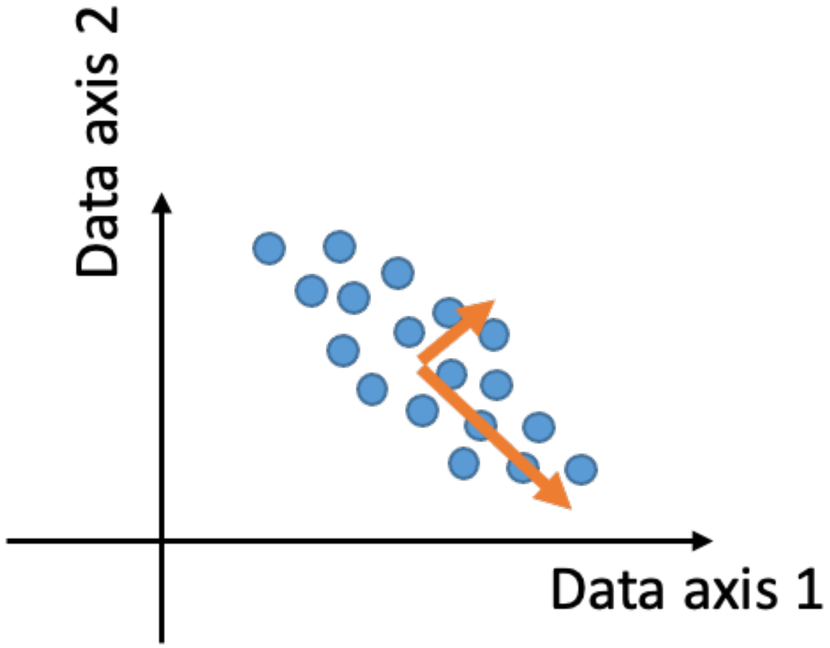
Principal component analysis

Data



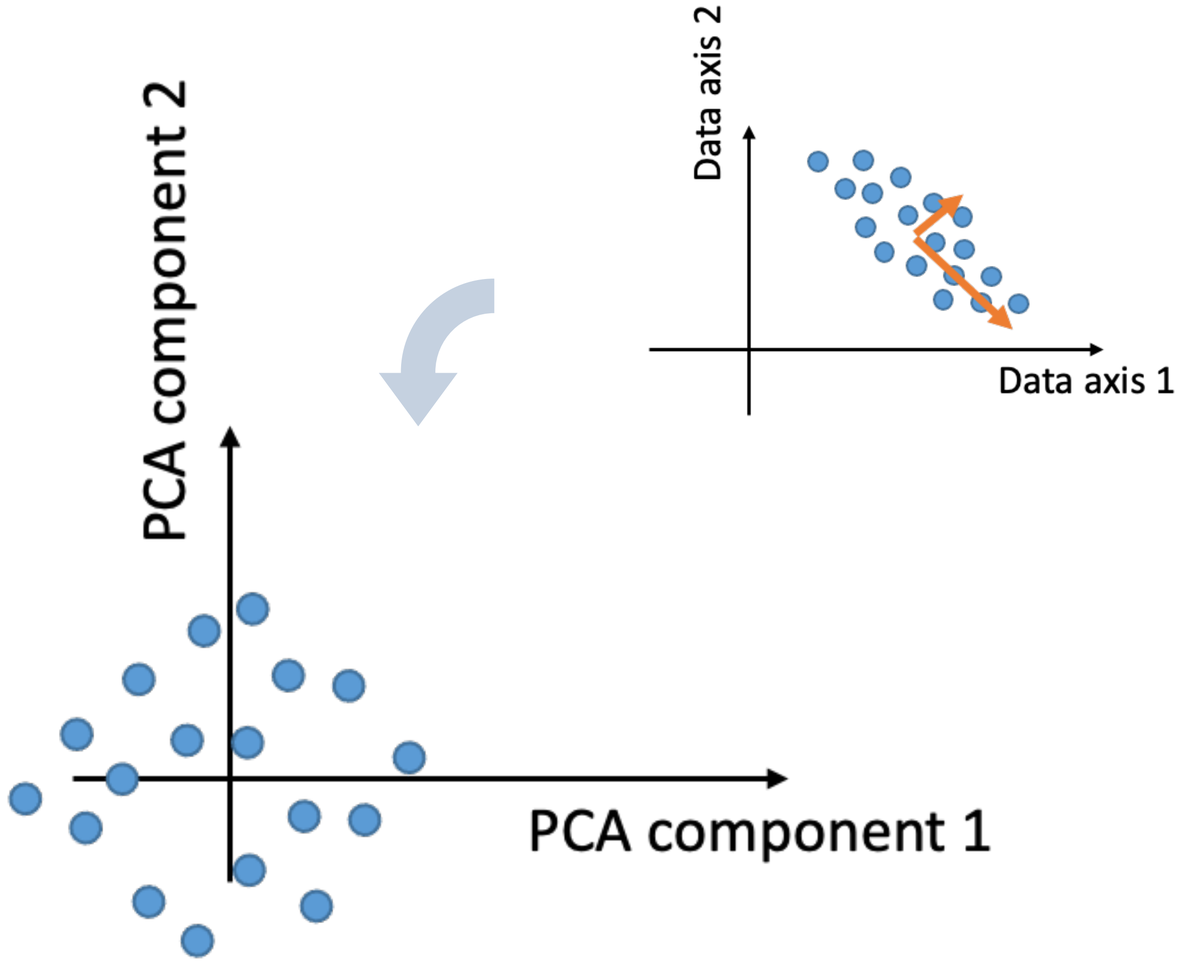
Basic Idea of PCA

Determine directions of maximal variance

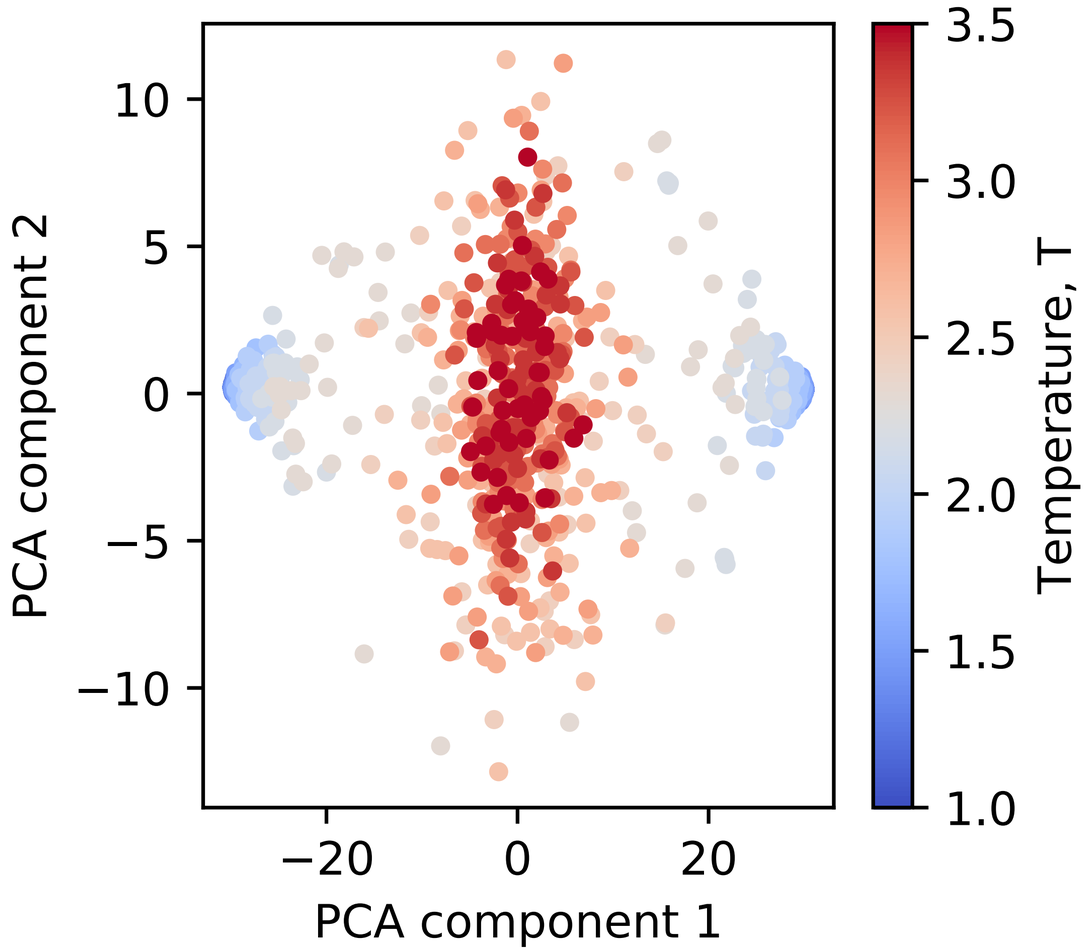


Basic Idea of PCA

Project data into the PCA axes

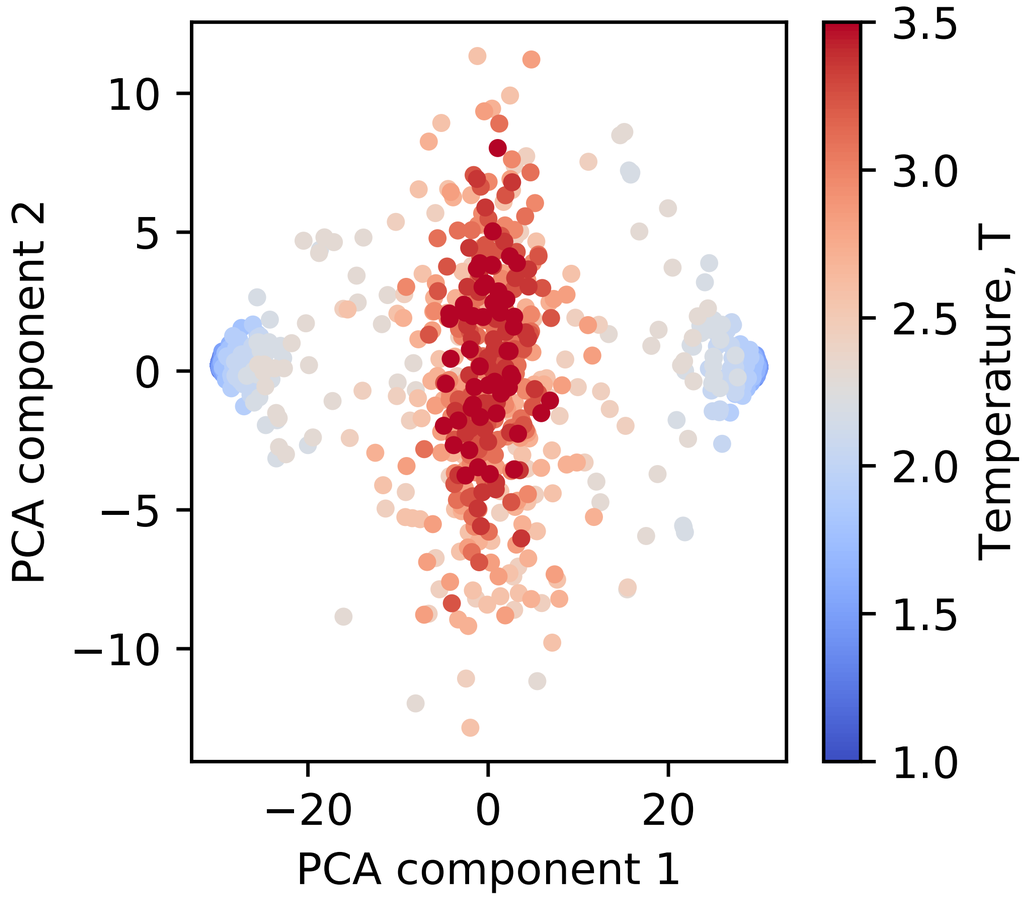


Ising model: PCA



Lei Wang, PHYSICAL REVIEW B 94, 195105 (2016)

Data driven vs. toy model driven



$$H = -J \sum_{<i,j>} \sigma_i \sigma_j$$

$$p(\sigma) = \frac{e^{-\beta H(\sigma)}}{Z}$$

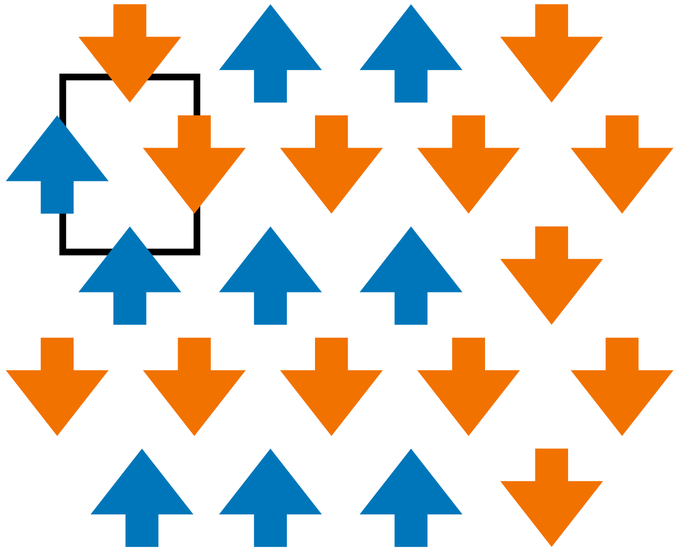
$$T_c = \frac{2J}{k \ln(1+\sqrt{2})} \qquad \sim 2.27$$

Lei Wang, PHYSICAL REVIEW B 94, 195105 (2016)


QMAI | TUDelft
Let’s just discover all
the physics from data!

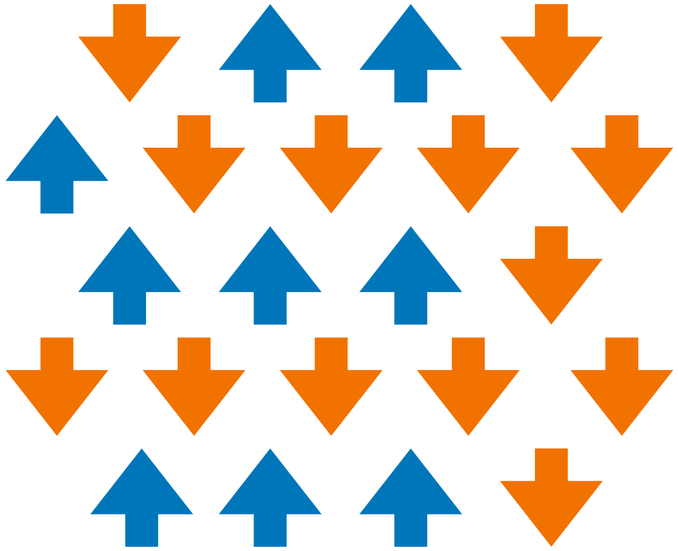
Let's try a new Hamiltonian: Ising Gauge Theory

$$H_{IGT} = -J \sum_{p} \prod_{i \in p} \sigma_i^z$$

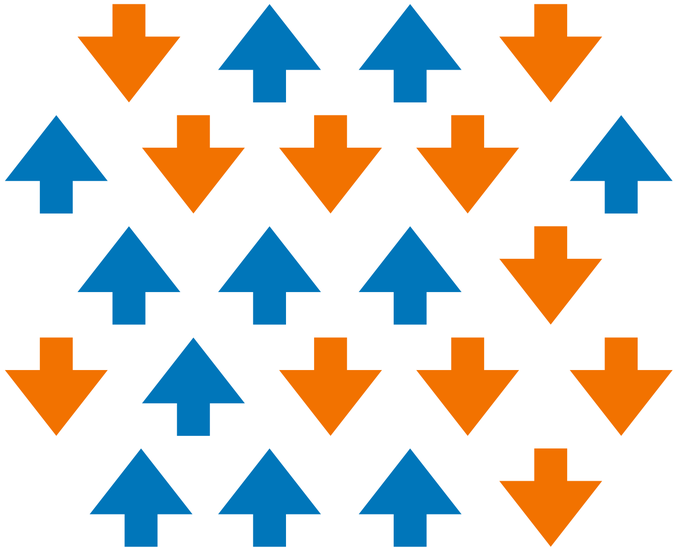


IGT QUIZ

Which of these has the lowest energy?

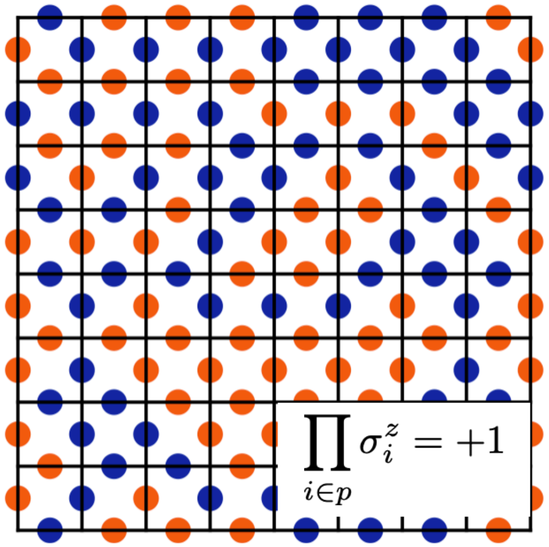


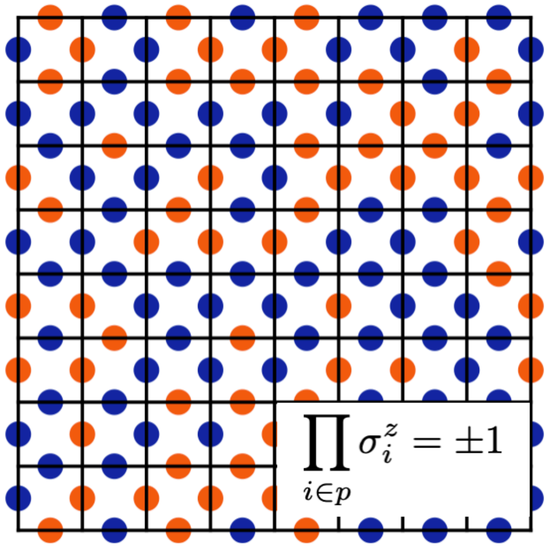
A

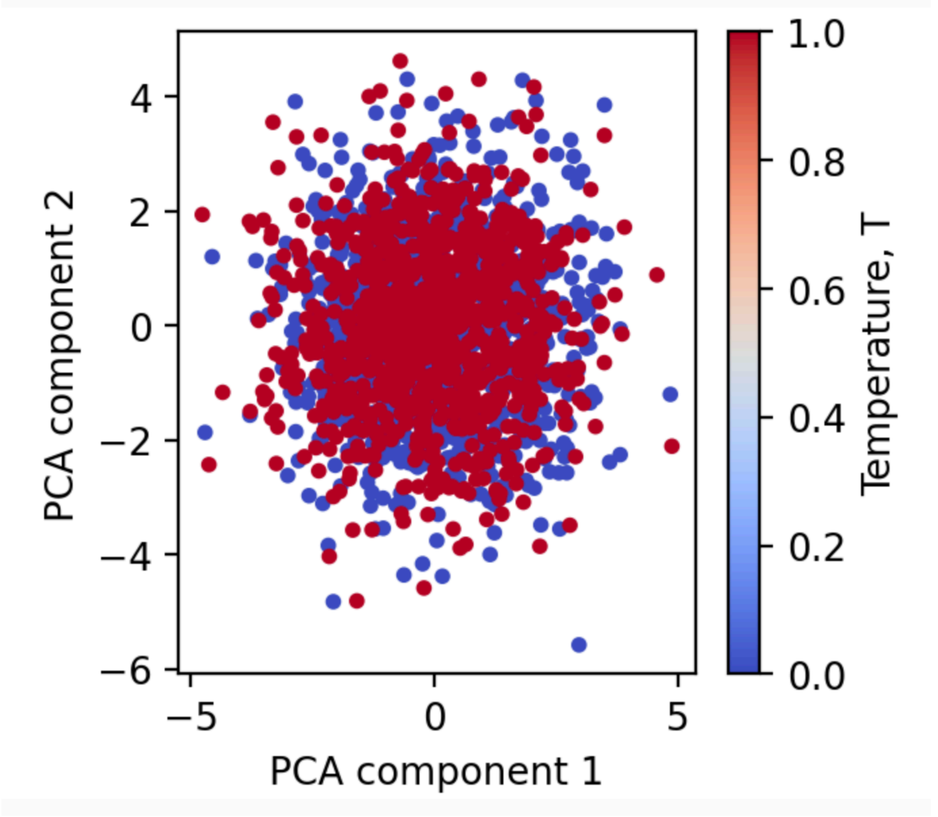


B

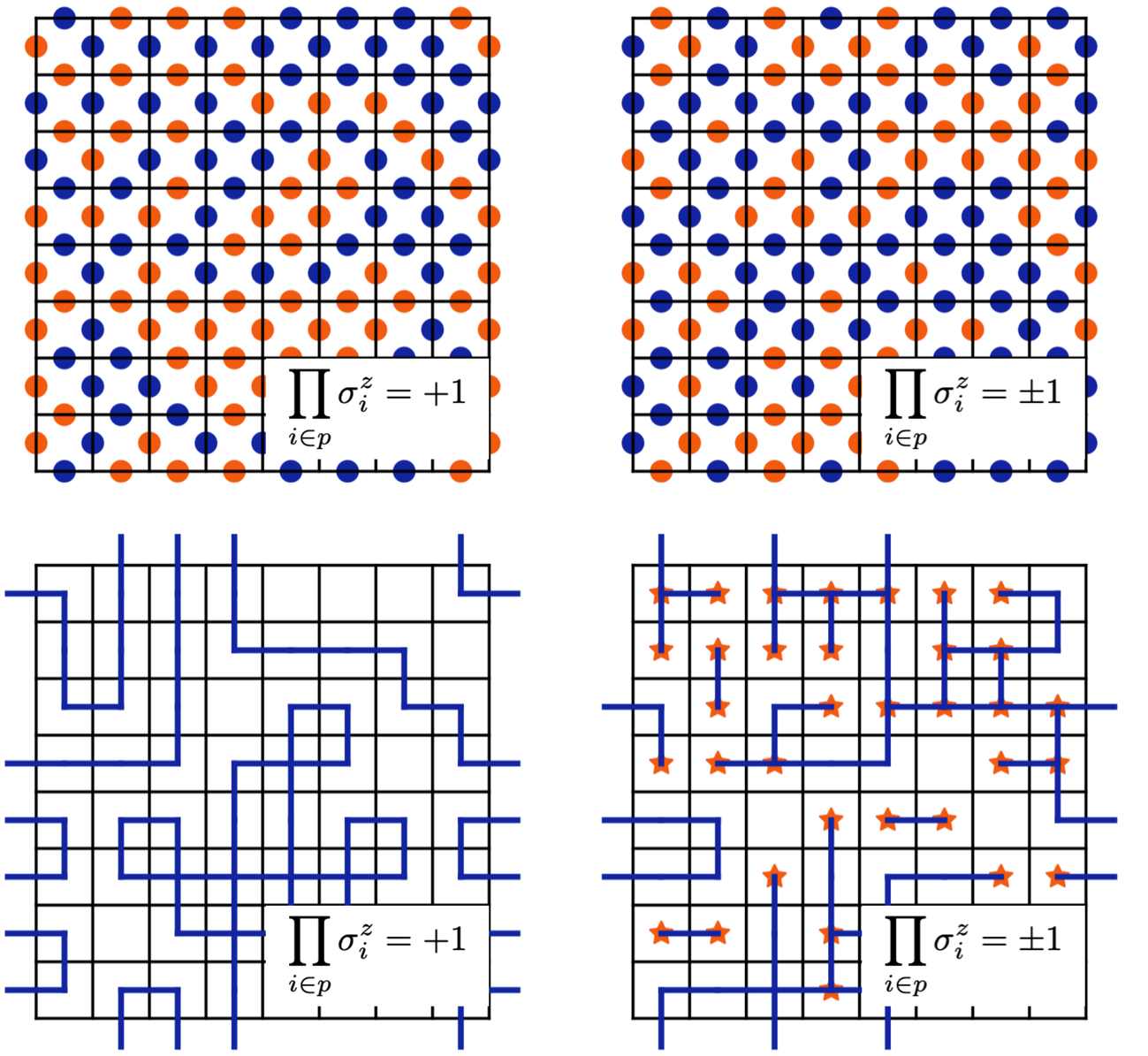
$$H_{IGT} = -J \sum_{p} \prod_{i \in p} \sigma_i^z$$

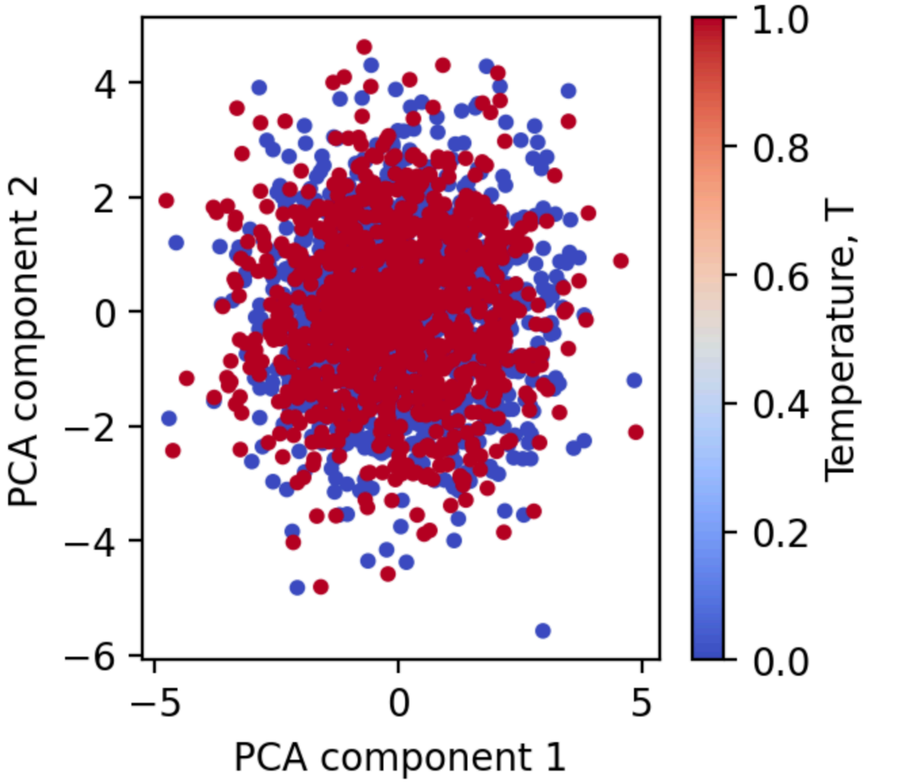




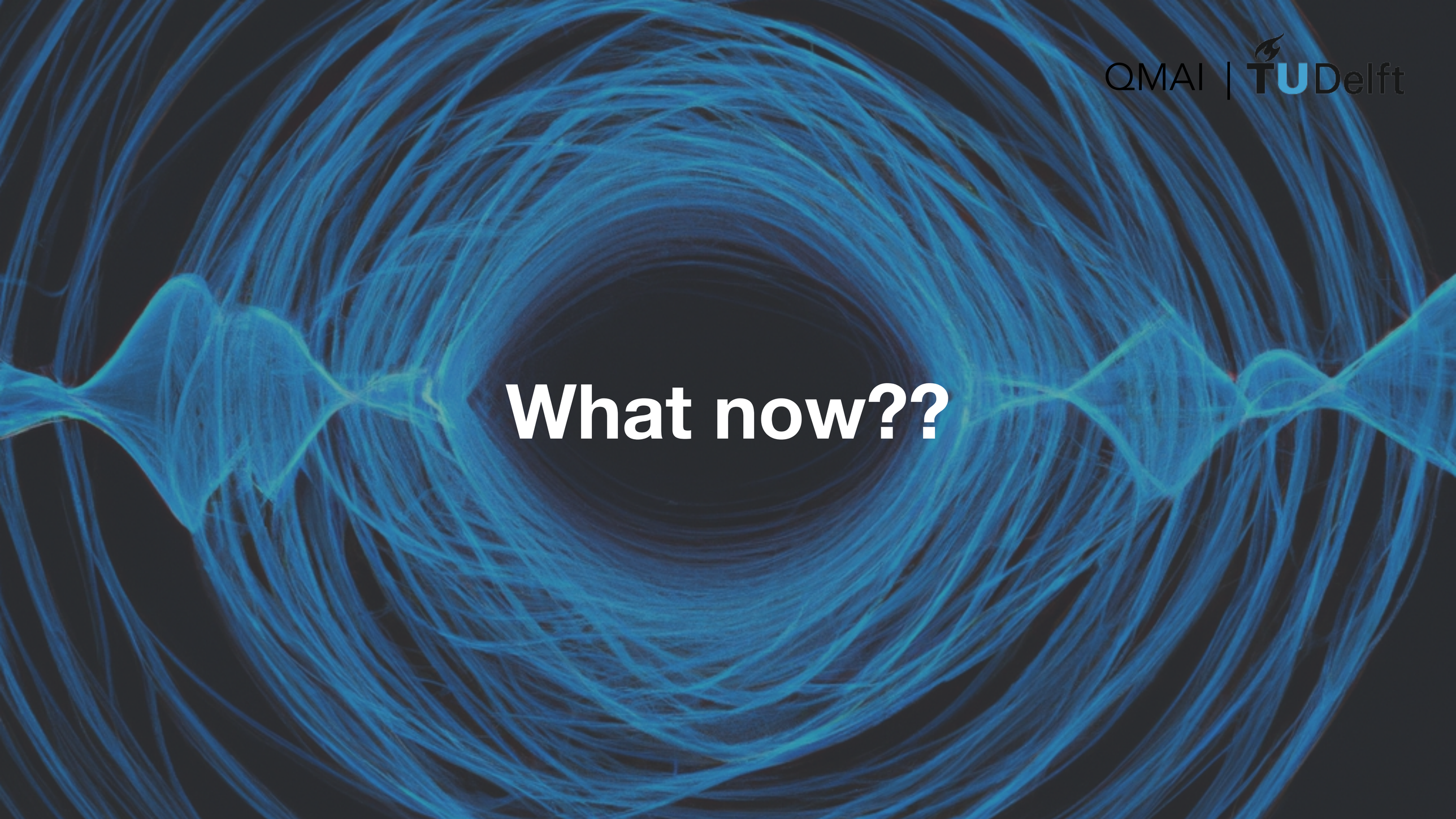


Greplova, E., Valenti, A., Boschung, G., Schäfer, F., Lörch, N., & Huber, S. D. (2020). Unsupervised identification of topological phase transitions using predictive models. *New Journal of Physics*, *22*(4), 045003.





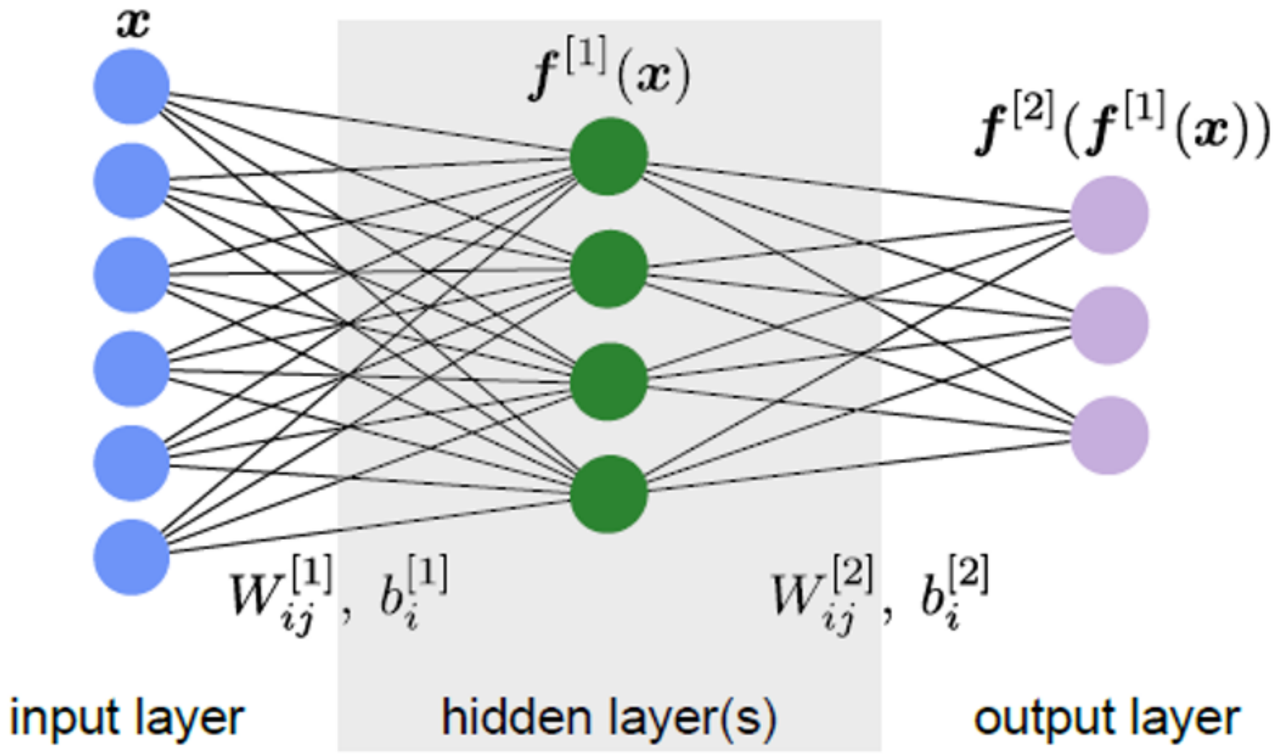
Greplova, E., Valenti, A., Boschung, G., Schäfer, F., Lörch, N., & Huber, S. D. (2020). Unsupervised identification of topological phase transitions using predictive models. *New Journal of Physics*, *22*(4), 045003.


QMAI | TUDelft
What now??

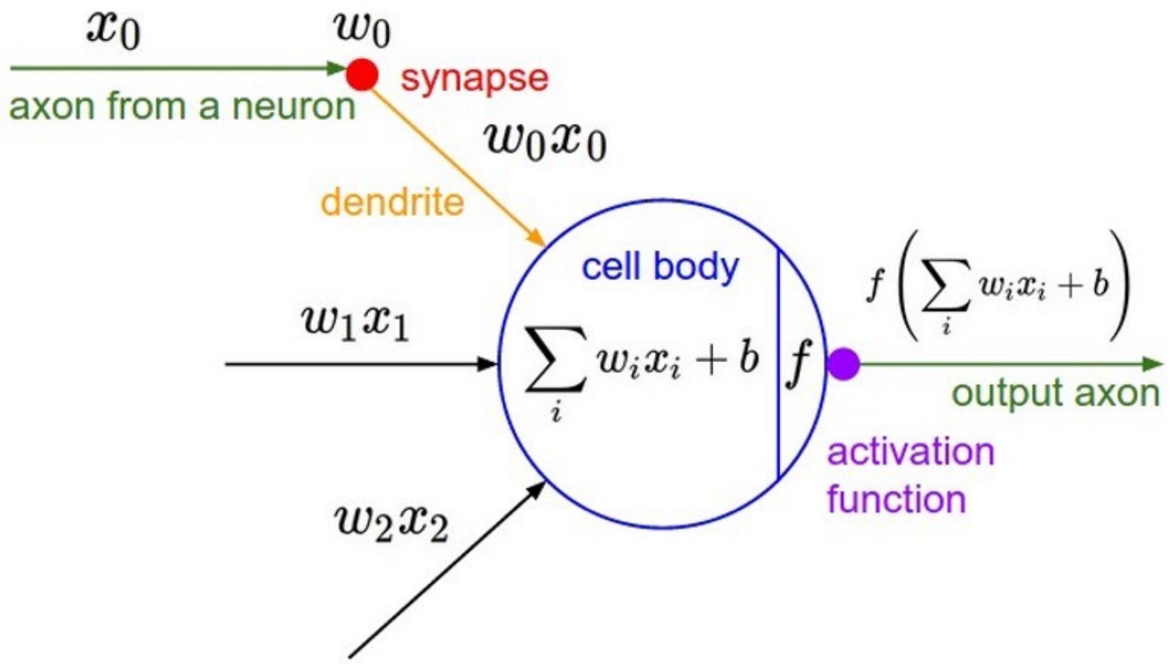
Machine Learning Primer: Neural Network Introduction

$$\frac{dL(f^{[2]})}{dW_{ij}^{[1]}} = \frac{dL}{df^{[2]}} \frac{df^{[2]}}{df^{[1]}} \frac{df^{[1]}}{dW^{[1]}}$$

Neural Networks 101: Supervised learning

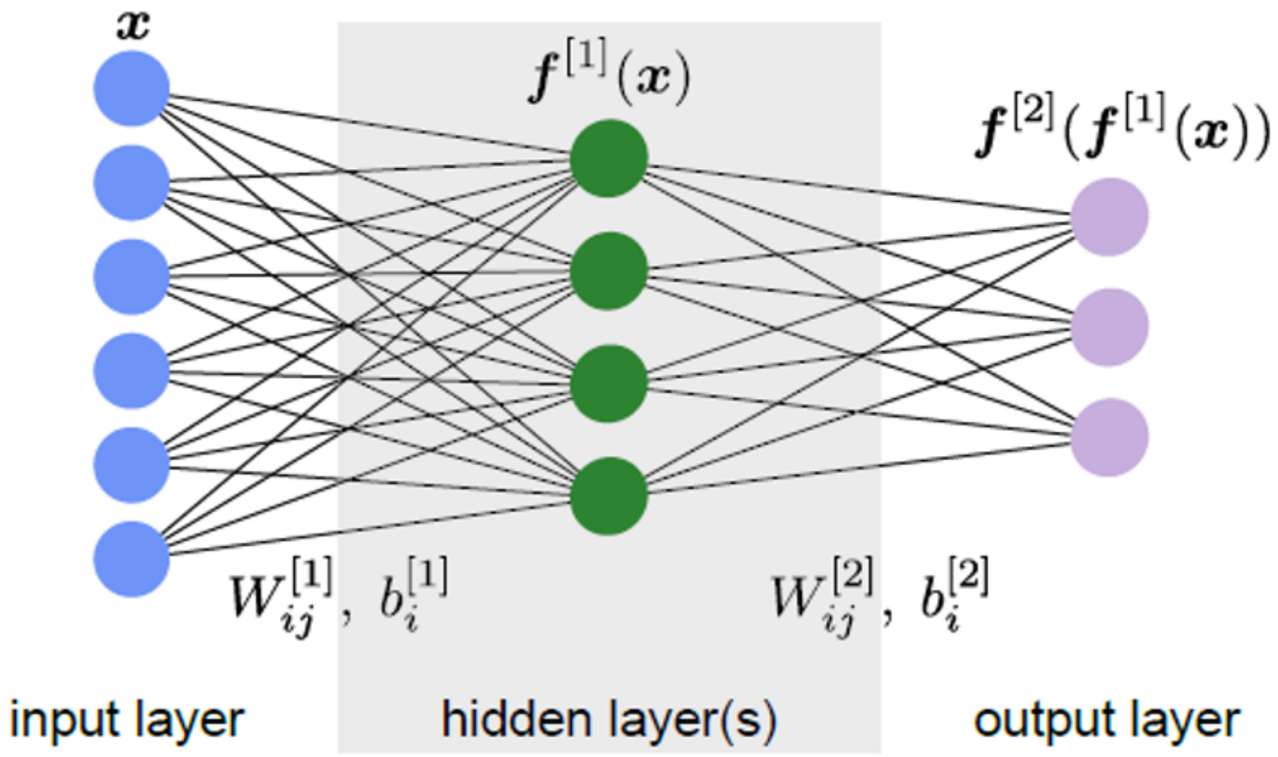


What happens in a neuron?

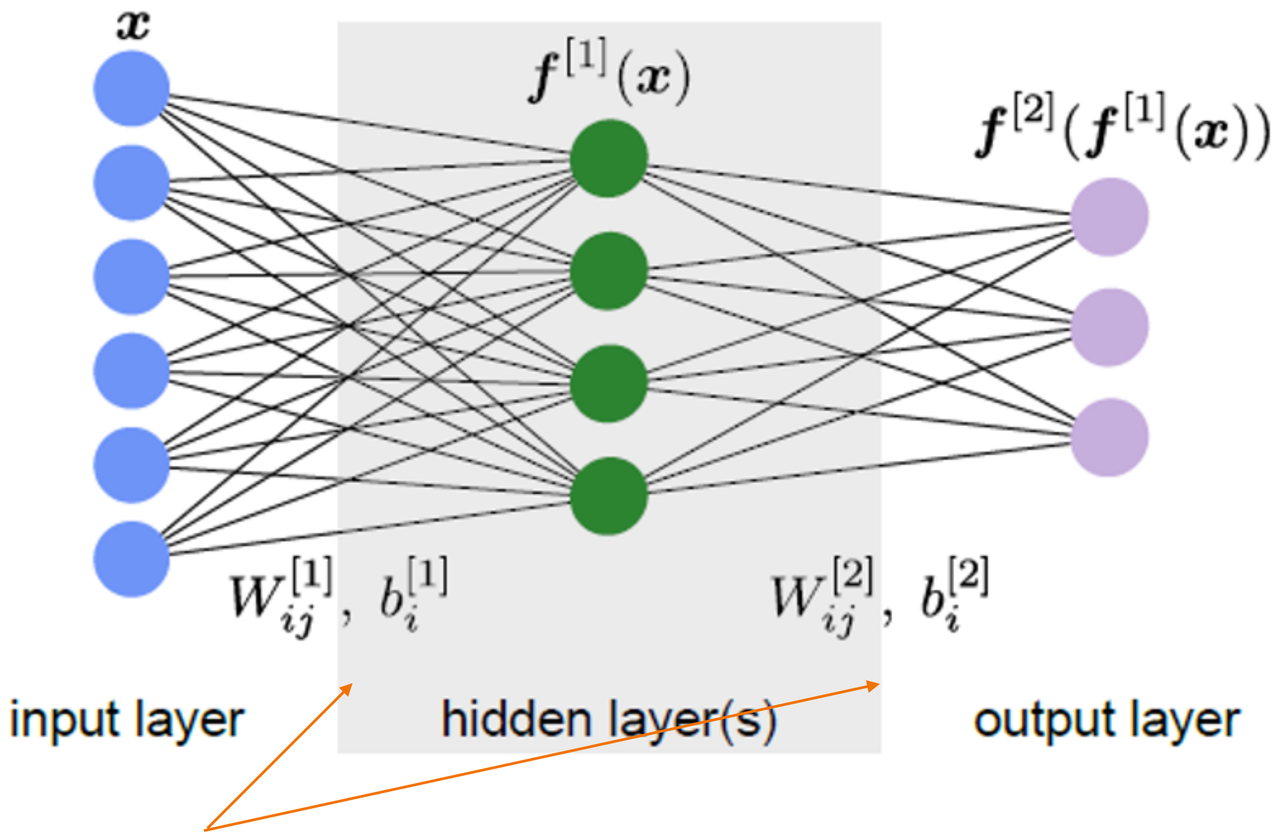


From https://cs231n.github.io

Putting neurons in the networks

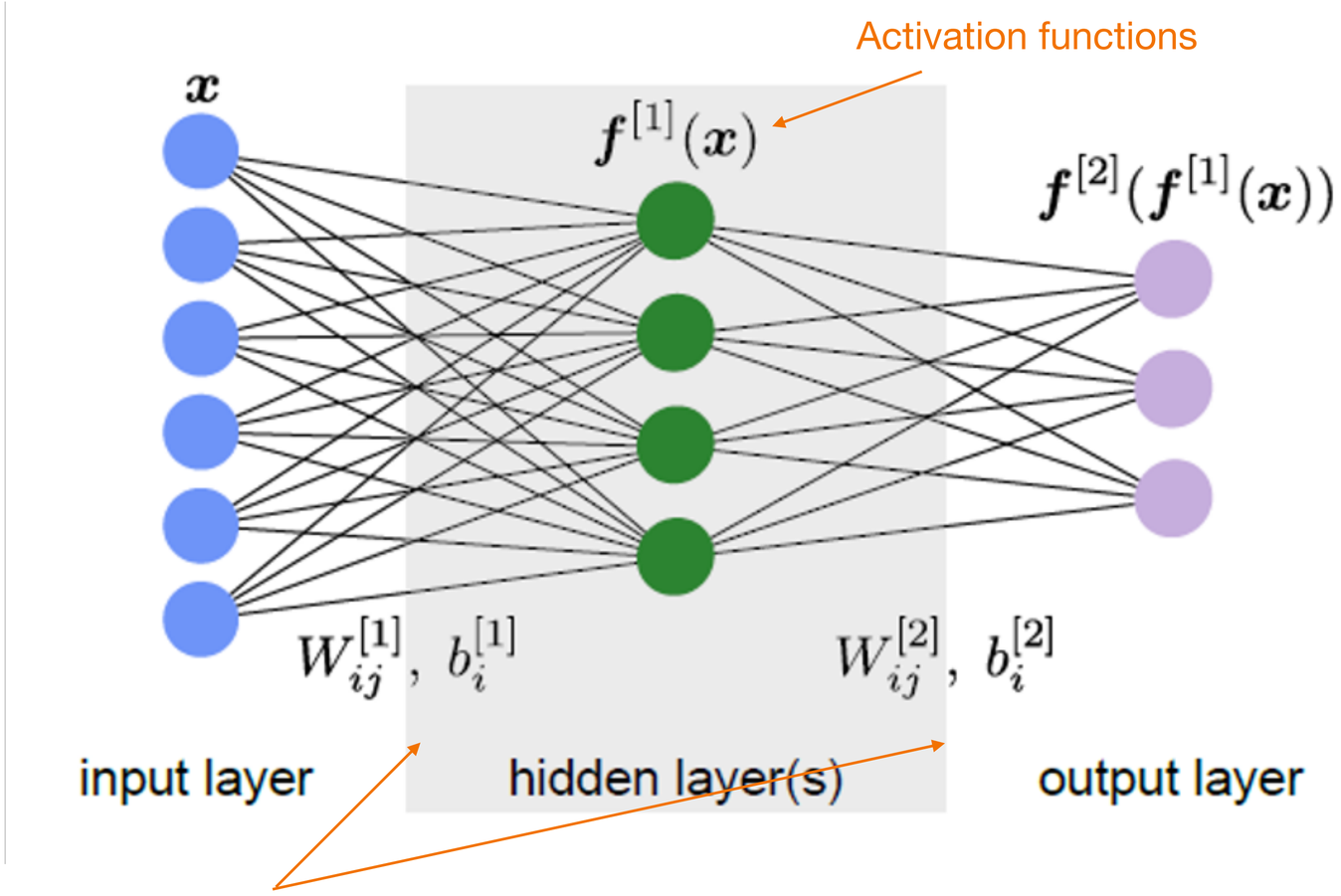


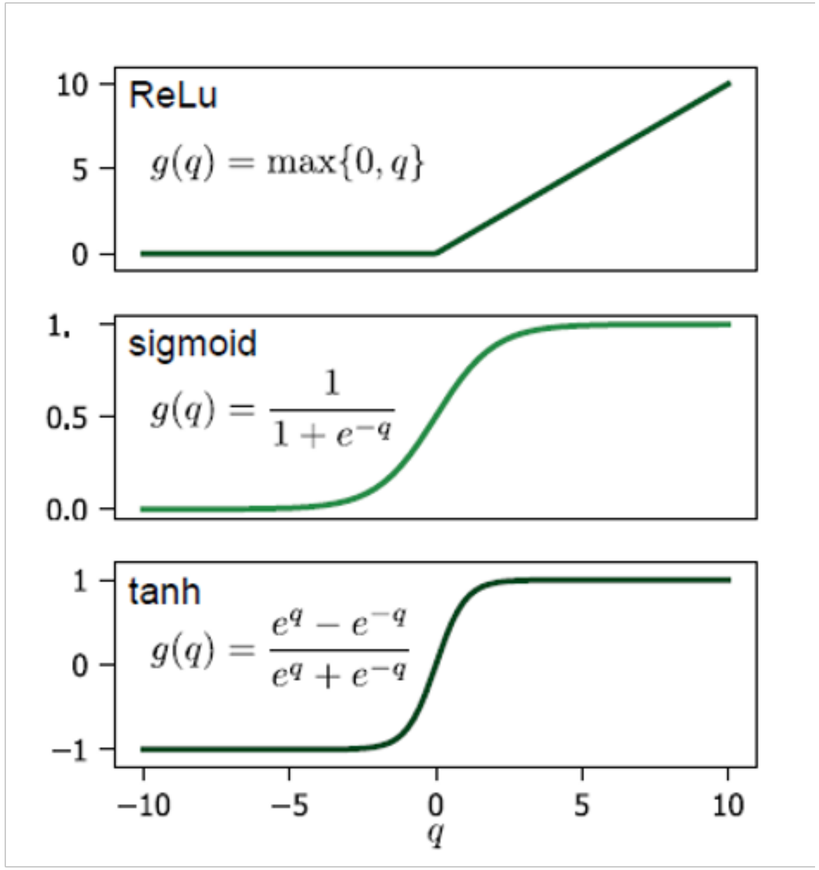
Putting neurons in the networks



Weights (can be represented as matrices)

Putting neurons in the networks





How do we train neural network?

- Define the loss function L(x) [x = our data] that we will minimise during training
- Calculate the derivative of L wrt weights and biases

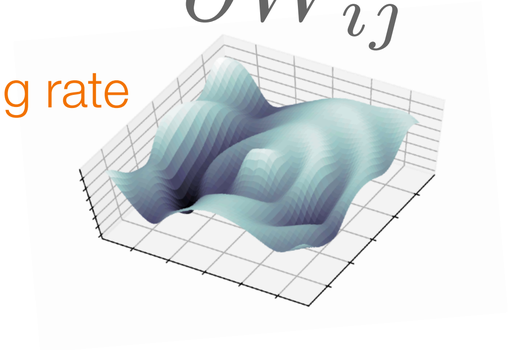
$$W_{ij} \rightarrow W_{ij} - \epsilon \frac{\partial L}{\partial W_{ij}}$$

How do we train neural network?

- Define the loss function L(x) [x = our data] that we will minimise during training
- Calculate the derivative of L wrt weights and biases

$$W_{ij} \rightarrow W_{ij} - \epsilon \frac{\partial L}{\partial W_{ij}}$$

learning rate



How do we calculate the derivative?

- CHAIN RULE REMINDER:

How do we calculate the derivative?

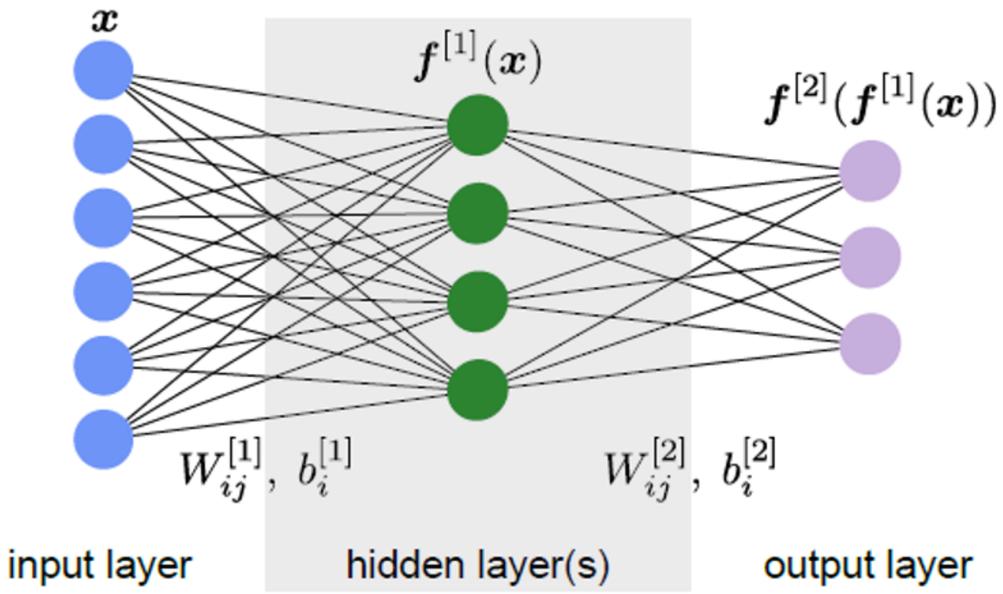
- CHAIN RULE REMINDER:

$$f(x) = g(h(x)) => \frac{df}{dx} = \frac{dg}{dh} * \frac{dh}{dx}$$

How do we calculate the derivative?

- CHAIN RULE REMINDER: $$f(x) = g(h(x)) => \frac{df}{dx} = \frac{dg}{dh} * \frac{dh}{dx}$$

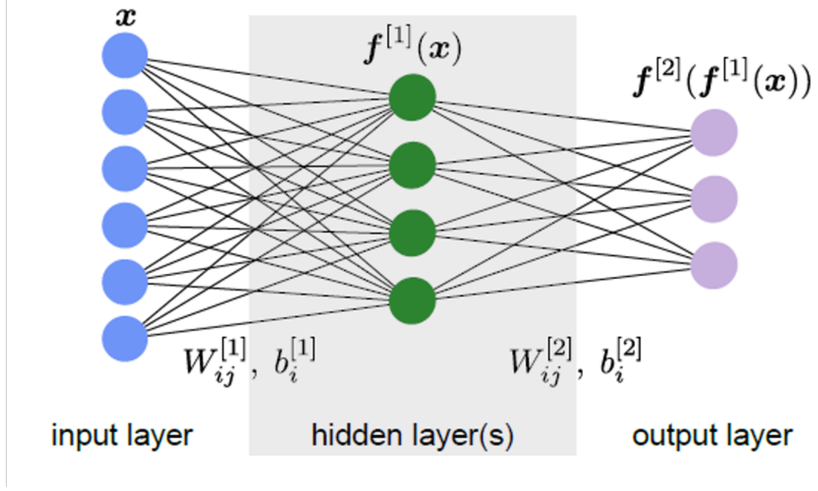
- APPLY TO NEURAL NET:



How do we calculate the derivative?

$$\frac{dL(f^{[2]})}{dW_{ij}^{[1]}} = \frac{dL}{df^{[2]}} \frac{df^{[2]}}{df^{[1]}} \frac{df^{[1]}}{dW^{[1]}}$$

- APPLY TO NEURAL NET:

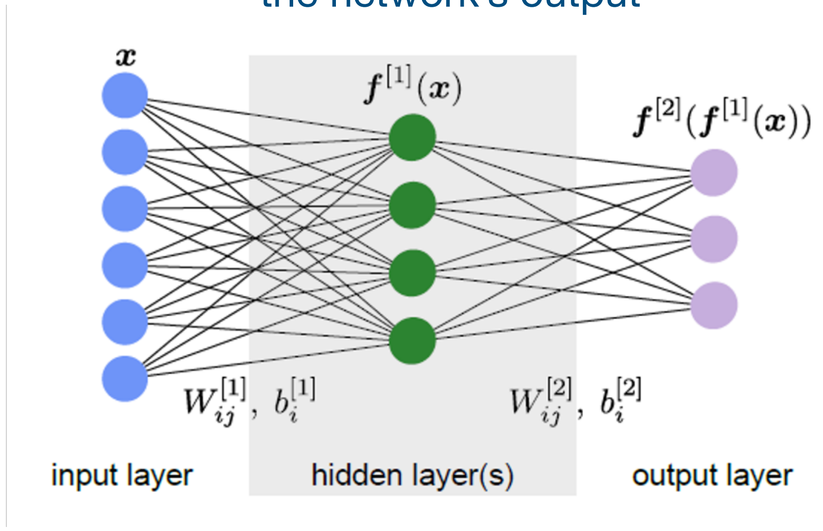


How do we calculate the derivative?

$$\frac{dL(f^{[2]})}{dW_{ij}^{[1]}} = \frac{dL}{df^{[2]}} \frac{df^{[2]}}{df^{[1]}} \frac{df^{[1]}}{dW^{[1]}}$$

First calculate this using the network's output

- APPLY TO NEURAL NET:



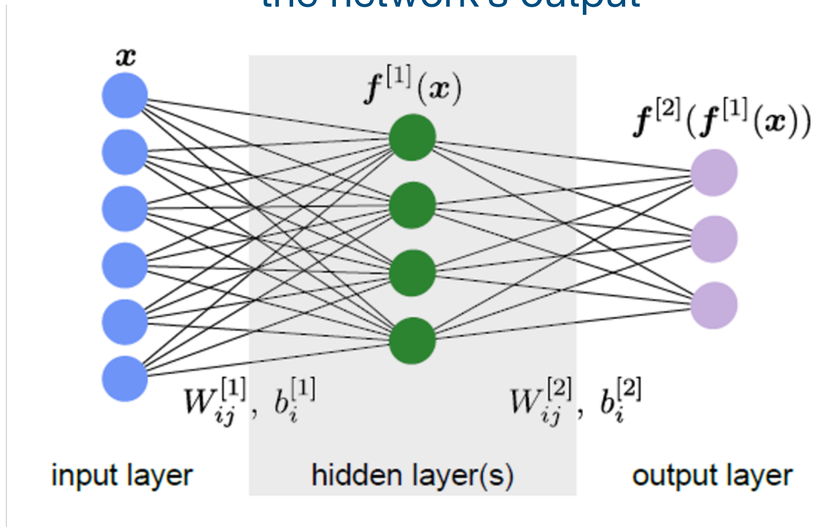
How do we calculate the derivative?

Calculate this using estimate of f[2] from previous step

$$\frac{dL(f^{[2]})}{dW_{ij}^{[1]}} = \frac{dL}{df^{[2]}} \frac{df^{[2]}}{df^{[1]}} \frac{df^{[1]}}{dW^{[1]}}$$

First calculate this using the network's output

- APPLY TO NEURAL NET:



How do we calculate the derivative?

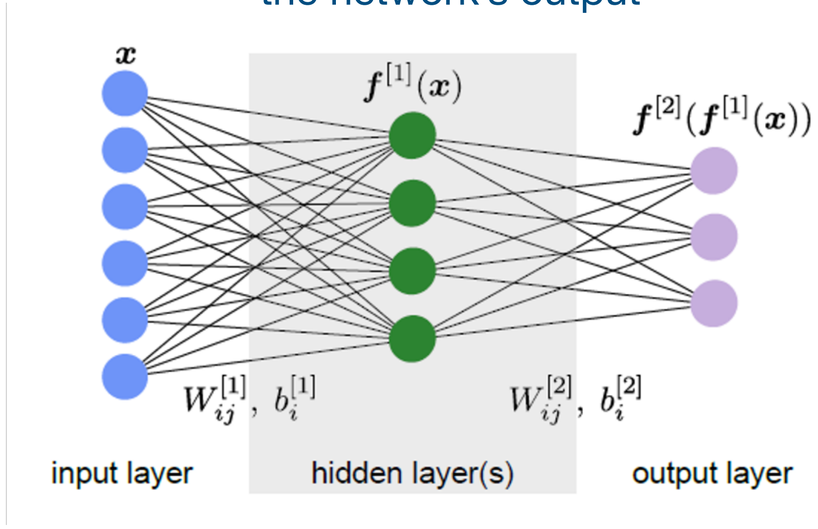
Calculate this using estimate of f[2] from previous step

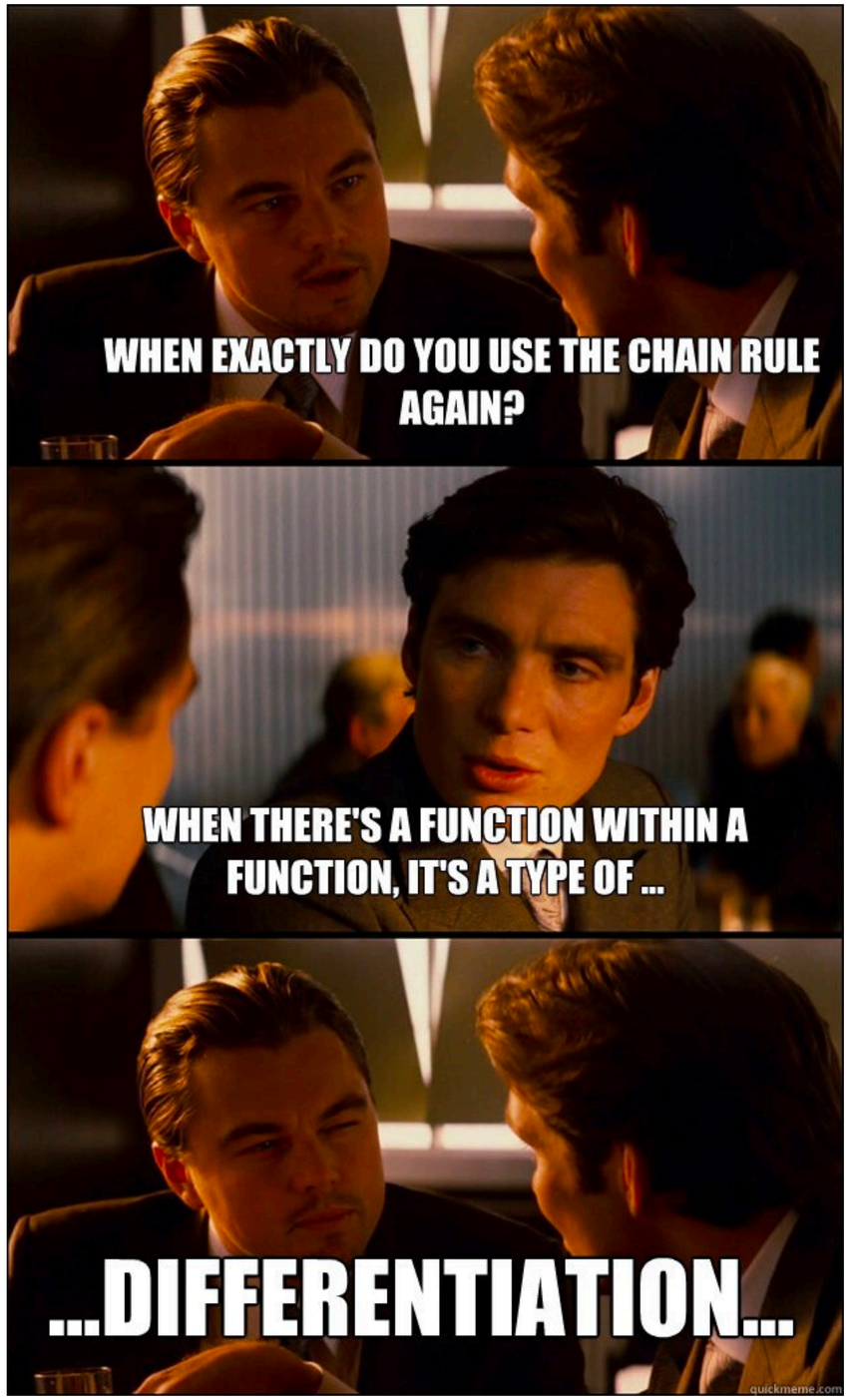
$$\frac{dL(f^{[2]})}{dW_{ij}^{[1]}} = \frac{dL}{df^{[2]}} \frac{df^{[2]}}{df^{[1]}} \frac{df^{[1]}}{dW^{[1]}}$$

First calculate this using the network's output

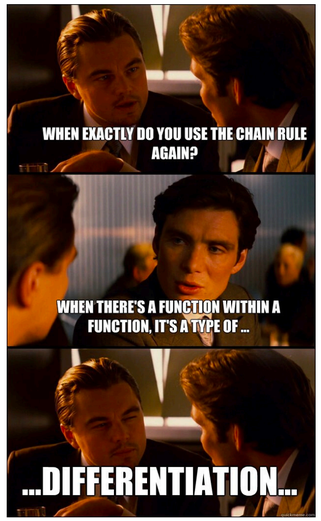
..and so on: we calculate each derivative from the back

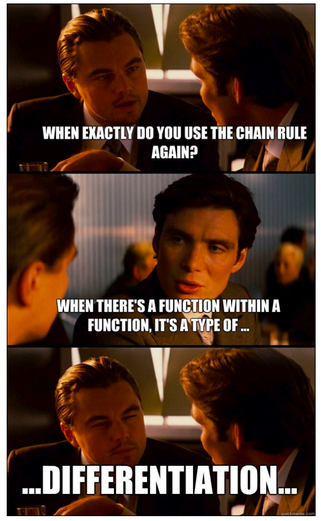
- APPLY TO NEURAL NET:

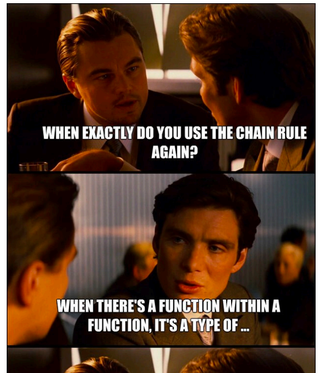




Backpropagation algorithm

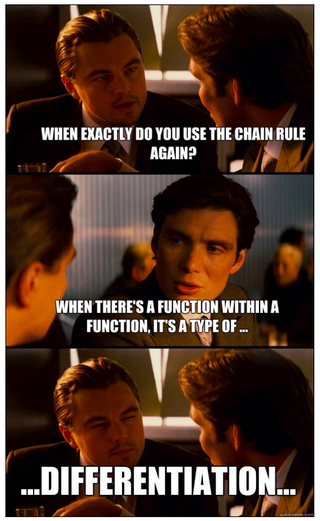


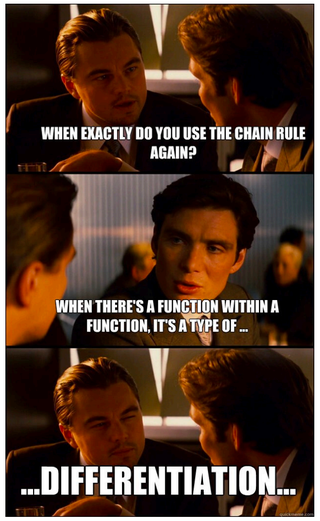


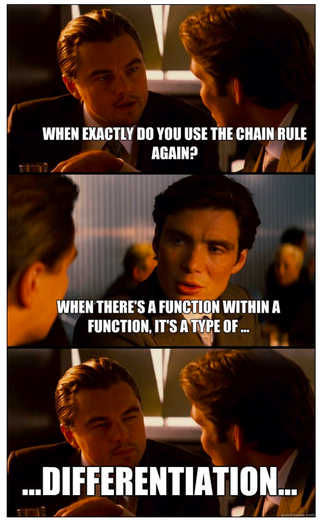


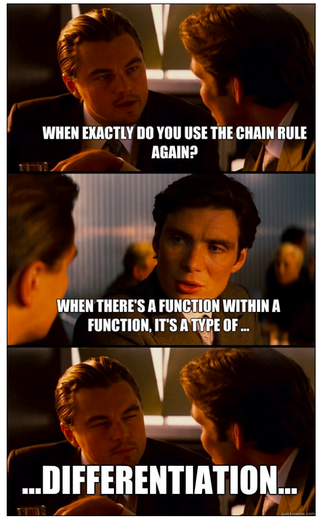
Backpropagation algorithm

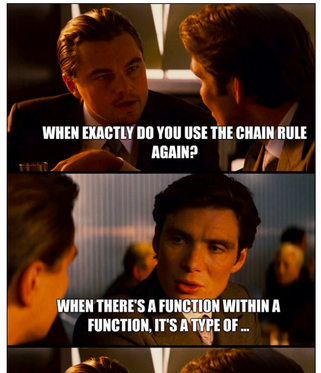
1. Calculate forward propagation of the network
2. Calculate the backward phase
   - a: Estimate the error in the final layer
   - b: Propagate that error into the previous layer
   - c: Evaluate the derivative of each parameter in the network
3. Combine all partial gradients into the final gradient
4. Update the weights using the calculated gradients to minimise the loss





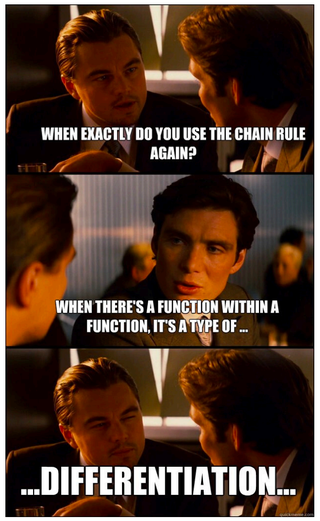


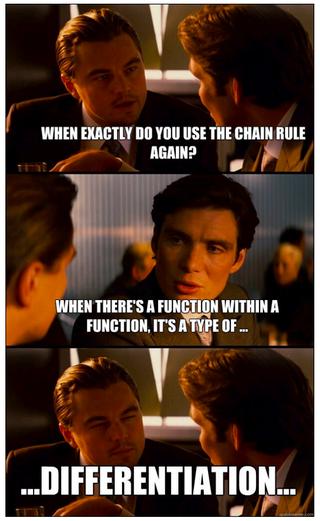


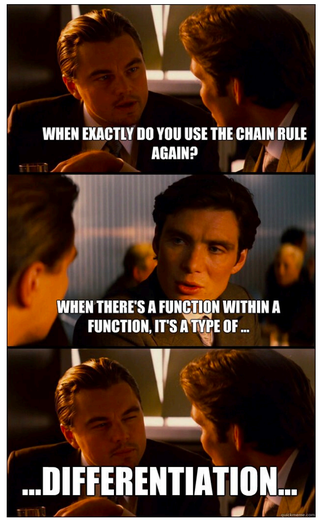


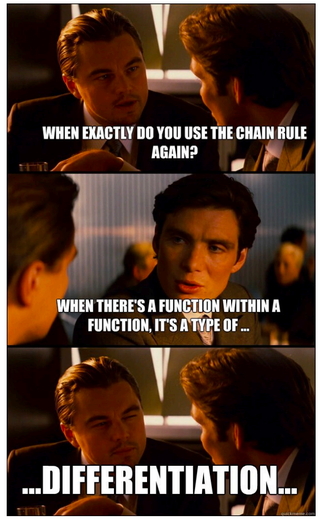
Backpropagation algorithm

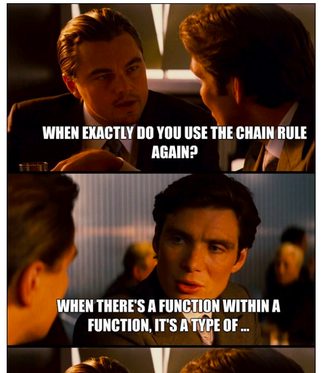
1. Calculate forward propagation of the network
2. Calculate the backward phase
   - a: Estimate the error in the final layer
   - b: Propagate that error into the previous layer
   - c: Evaluate the derivative of each parameter in the network
3. Combine all partial gradients into the final gradient
4. Update the weights using the calculated gradients to minimise the loss





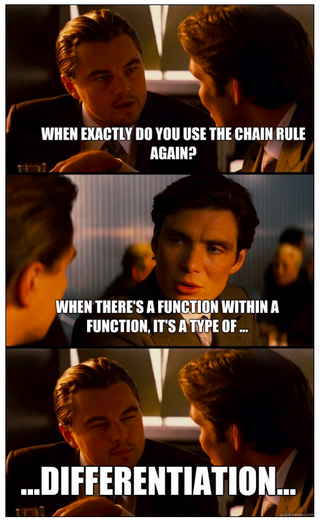


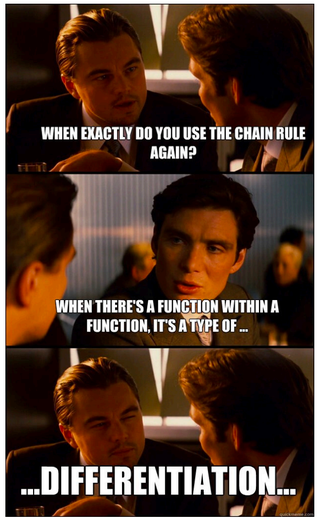


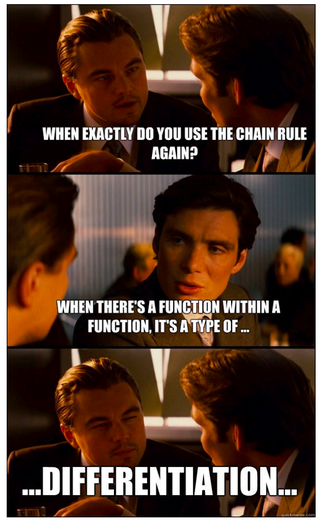


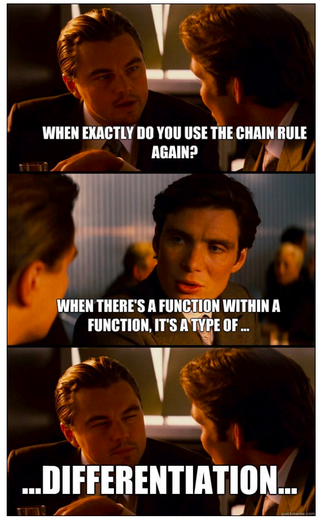
Backpropagation algorithm

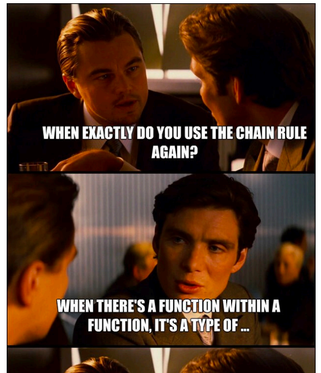
1. Calculate forward propagation of the network
2. Calculate the backward phase
   - a: Estimate the error in the final layer
   - b: Propagate that error into the previous layer
   - c: Evaluate the derivative of each parameter in the network
3. Combine all partial gradients into the final gradient
4. Update the weights using the calculated gradients to minimise the loss





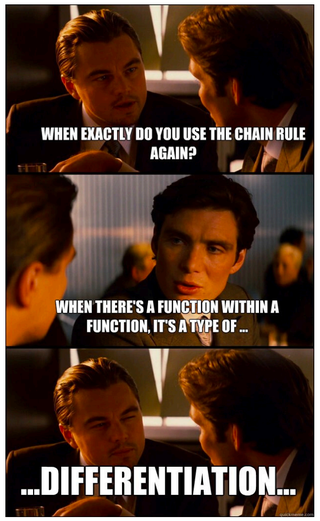


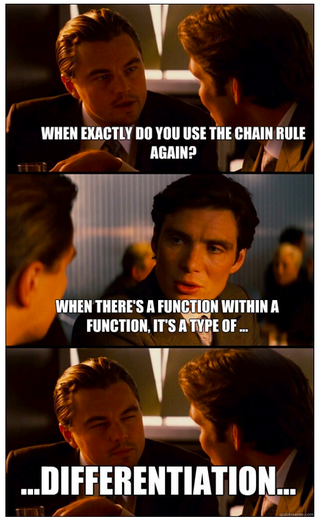




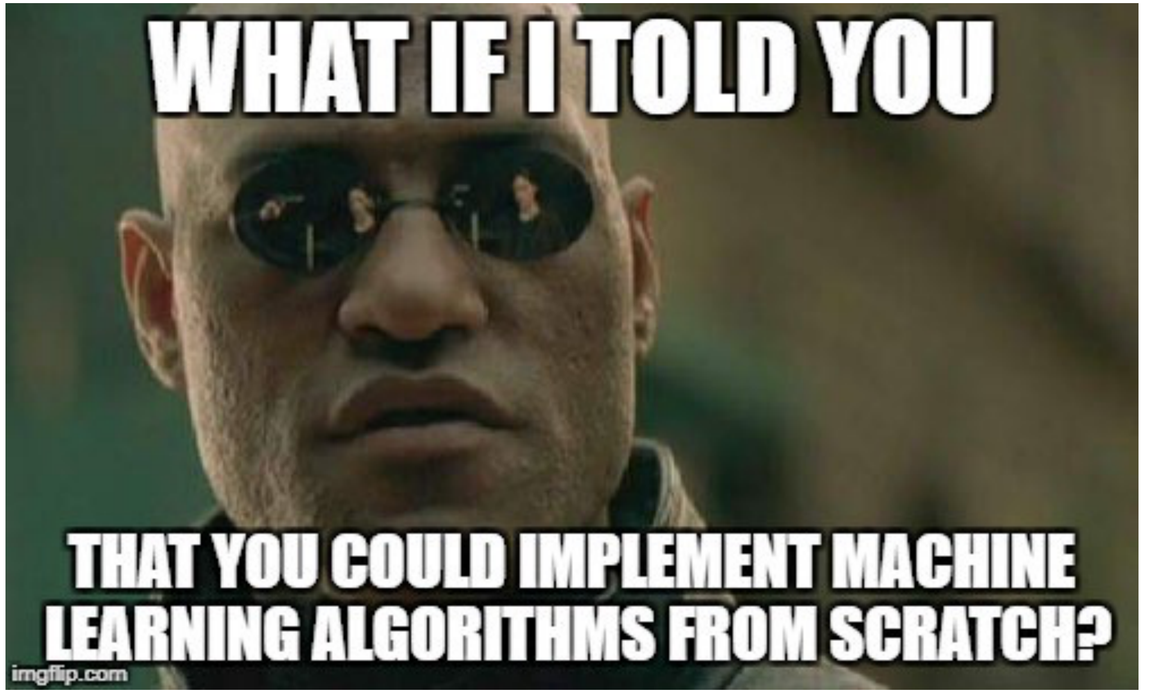
Backpropagation algorithm

1. Calculate forward propagation of the network
2. Calculate the backward phase
   - a: Estimate the error in the final layer
   - b: Propagate that error into the previous layer
   - c: Evaluate the derivative of each parameter in the network
3. Combine all partial gradients into the final gradient
4. Update the weights using the calculated gradients to minimise the loss





BREAK


WHAT IF I TOLD YOU
THAT YOU COULD IMPLEMENT MACHINE LEARNING ALGORITHMS FROM SCRATCH?
imgflip.com

Setting up the neural network

Classification: Does a picture belong into the class “A" or class “B”?

Build a network with two outputs that tell you the probability from which movie franchise your character is from.

Output and loss function

- LAST LAYER ACTIVATION FUNCTION:

  (normalises the output and maps it on the probability distribution)

$$f(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

- LOSS FUNCTION:

  (calculate the difference between predicted and correct outcome during training)

$$L = -\sum_x p(x) \log q(x)$$

Classification: a mini example

Label: p(x)

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

Network output: q(x)

$$\begin{bmatrix} q\ (\text{class A}) \\ q\ (\text{class B}) \end{bmatrix}$$

Loss: $-1 \log(q(\text{class A})) - 0 \log(q(\text{class B}))$ $\qquad -0 \log(q(\text{class A})) - 1 \log(q(\text{class B}))$

Classification: a mini example

Label: p(x)

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

Network output: q(x)

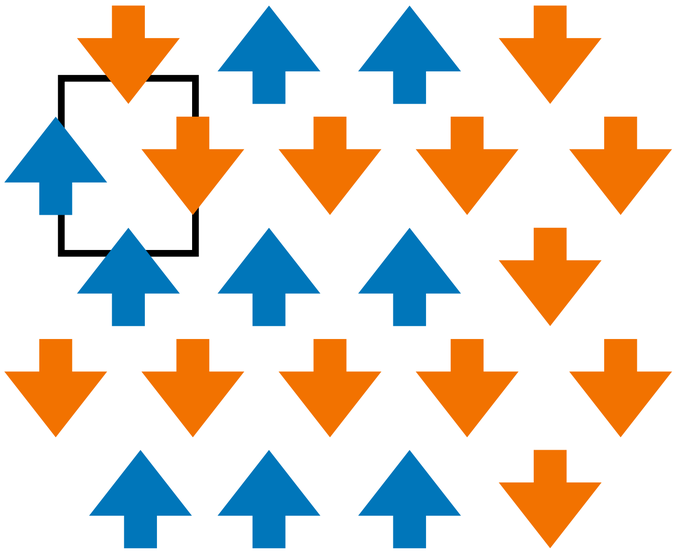
$$\begin{bmatrix} q\ (\text{class A}) \\ q\ (\text{class B}) \end{bmatrix}$$

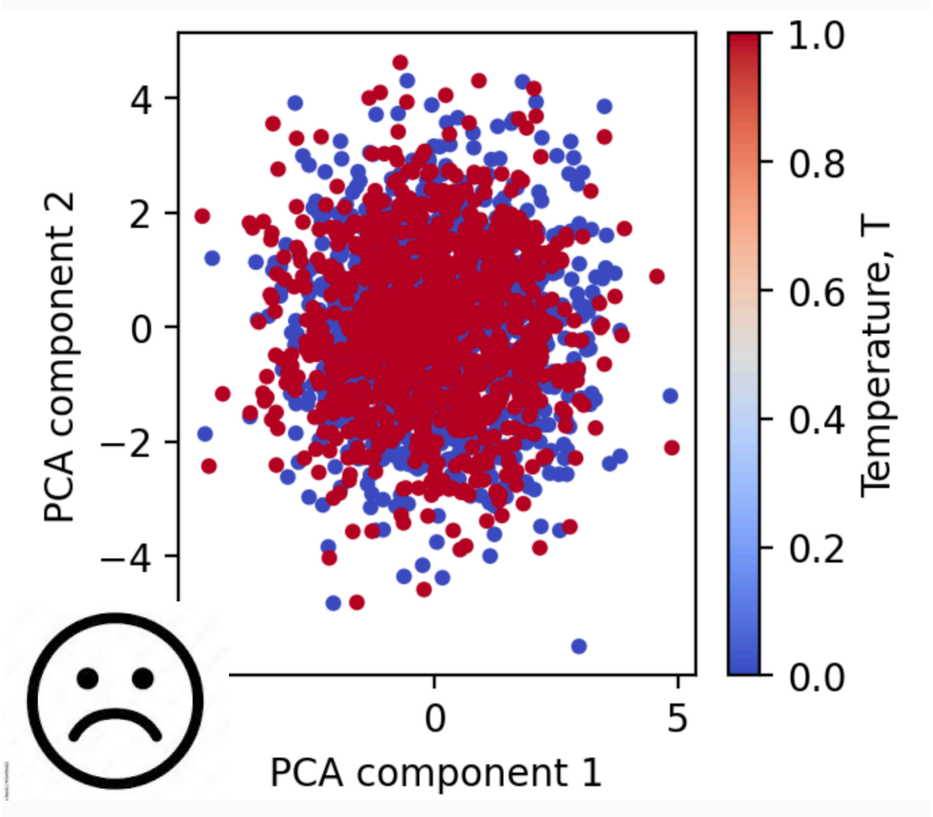
Loss: $-1 \log(q(\text{class A})) - 0 \log(q(\text{class B}))$ $\qquad -0 \log(q(\text{class A})) - 1 \log(q(\text{class B}))$

!the loss is minimal if q(x) matches the labels!

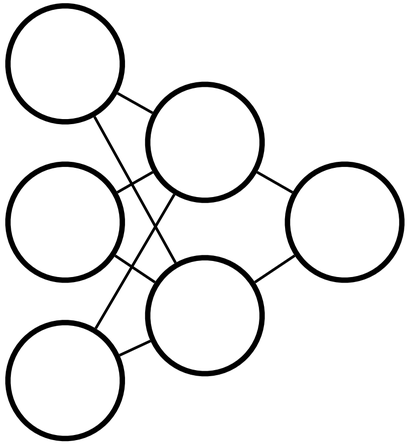
Before the ML detour

$$H_{IGT} = -J \sum_{p} \prod_{i \in p} \sigma_i^z$$

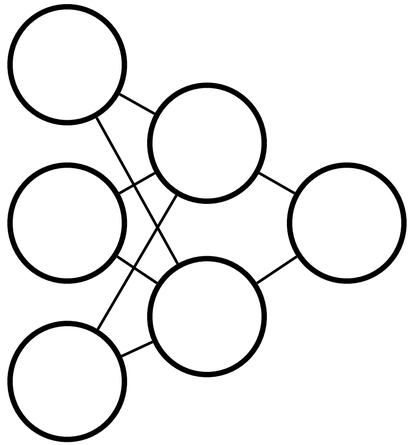




AI Image Classification

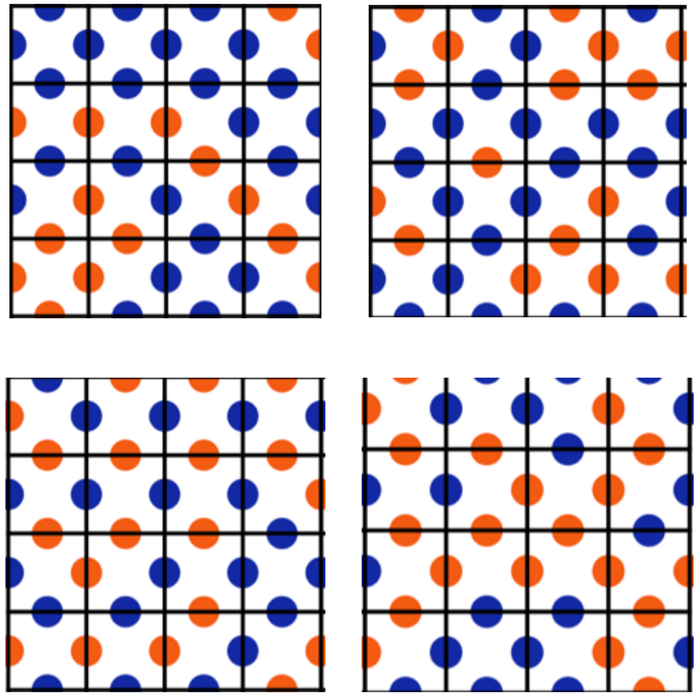


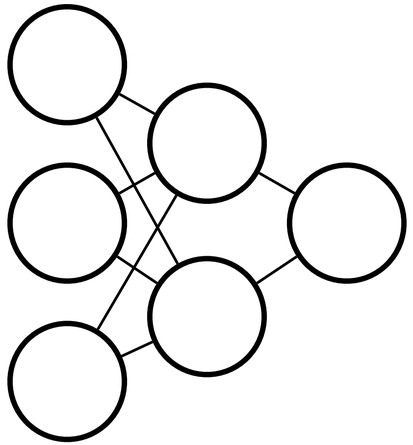
AI Image Classification



QMAI | TU Delft

AI Image Classification

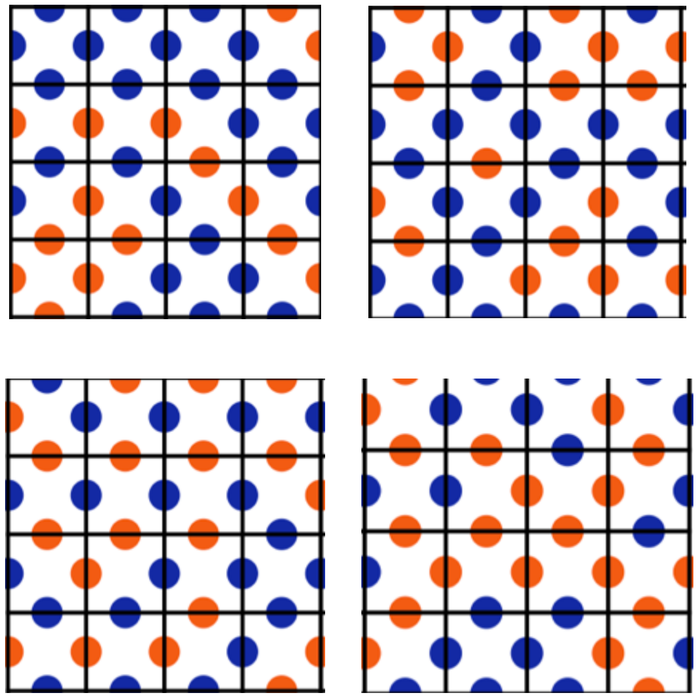


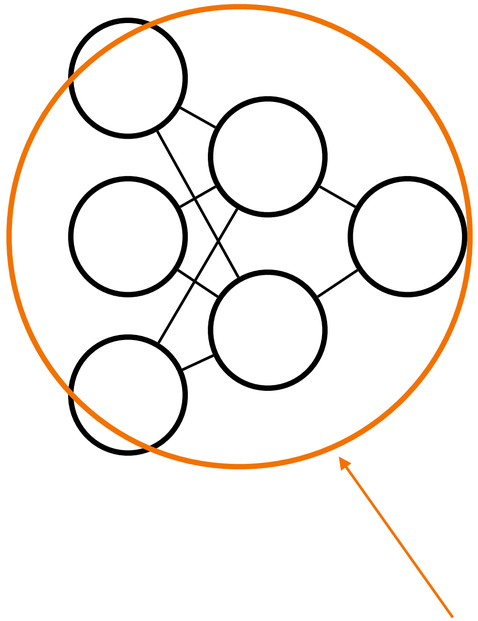


Carrasquilla, Juan, and Roger G. Melko. "Machine learning phases of matter." *Nature Physics* 13.5 (2017): 431-434.

QMAI | TU Delft

AI Image Classification





MACHINE LEARNING MODEL

Wetzel, Sebastian J. "Unsupervised learning of phase transitions: From principal component analysis to variational autoencoders." *Physical Review E* 96.2 (2017): 022140.

Van Nieuwenburg, Evert PL, Ye-Hua Liu, and Sebastian D. Huber. "Learning phase transitions by confusion." *Nature Physics* 13.5 (2017): 435-439.

Carrasquilla, Juan, and Roger G. Melko. "Machine learning phases of matter." *Nature Physics* 13.5 (2017): 431-434.

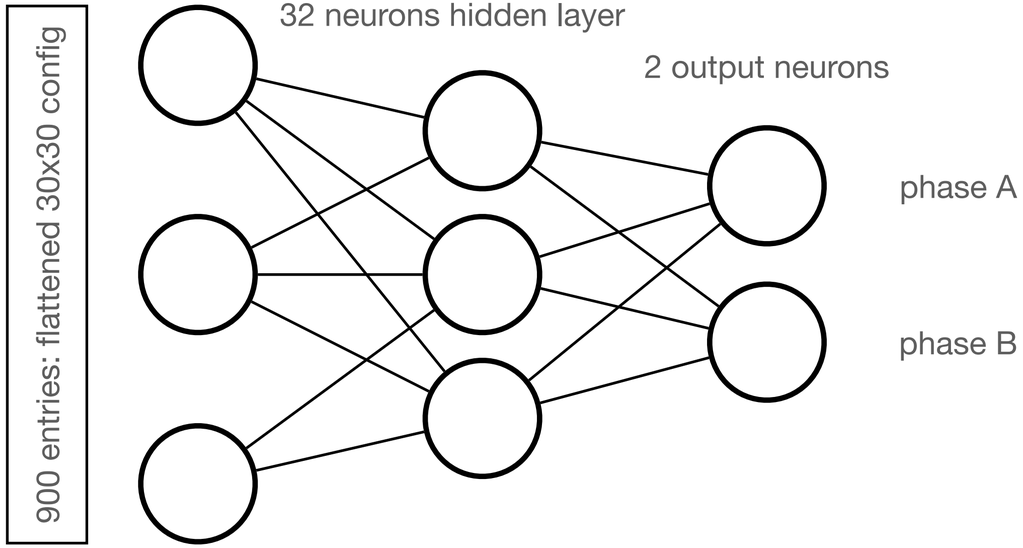
Greplova, E., Valenti, A., Boschung, G., Schäfer, F., Lörch, N., & Huber, S. D. (2020). Unsupervised identification of topological phase transitions using predictive models. *New Journal of Physics*, *22*(4), 045003.

PyTorch

QMAI | TU Delft

Coding NNs in PyTorch

```
class NeuralNetwork(nn.Module):
    def __init__(self):
        super(NeuralNetwork,self).__init__()
        self.flatten = nn.Flatten()
        self.linear_relu_stack = nn.Sequential(
            nn.Linear(900,32),
            nn.ReLU(),
            nn.Linear(32,2),
        )
```

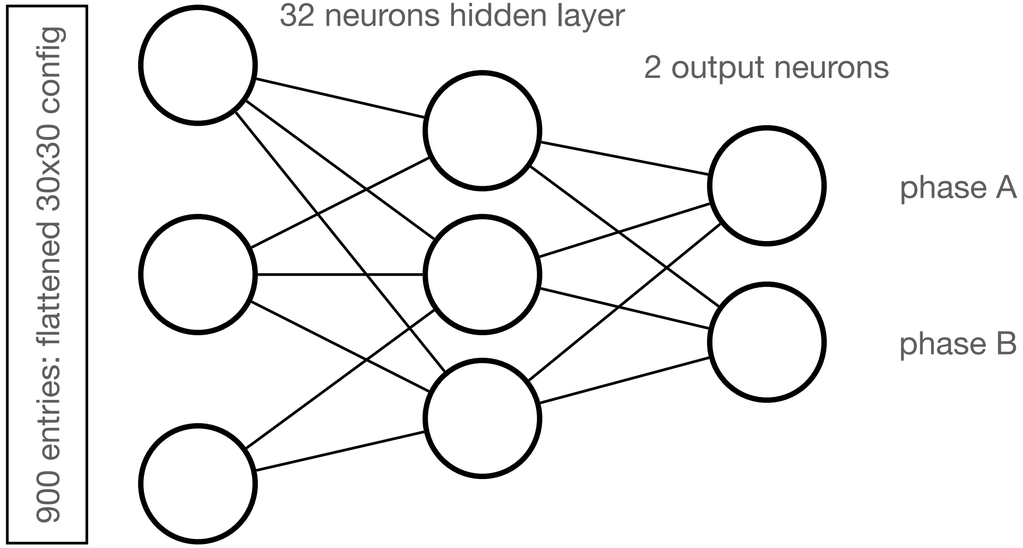


PyTorch

QMAI | TU Delft

Coding NNs in PyTorch

```
class NeuralNetwork(nn.Module):
    def __init__(self):
        super(NeuralNetwork,self).__init__()
        self.flatten = nn.Flatten()
        self.linear_relu_stack = nn.Sequential(
            nn.Linear(900,32),
            nn.ReLU(),
            nn.Linear(32,2),
        )
```



Go to: https://www.eliskagreplova.com/ai-for-physicists-ap3751 (Week 2)

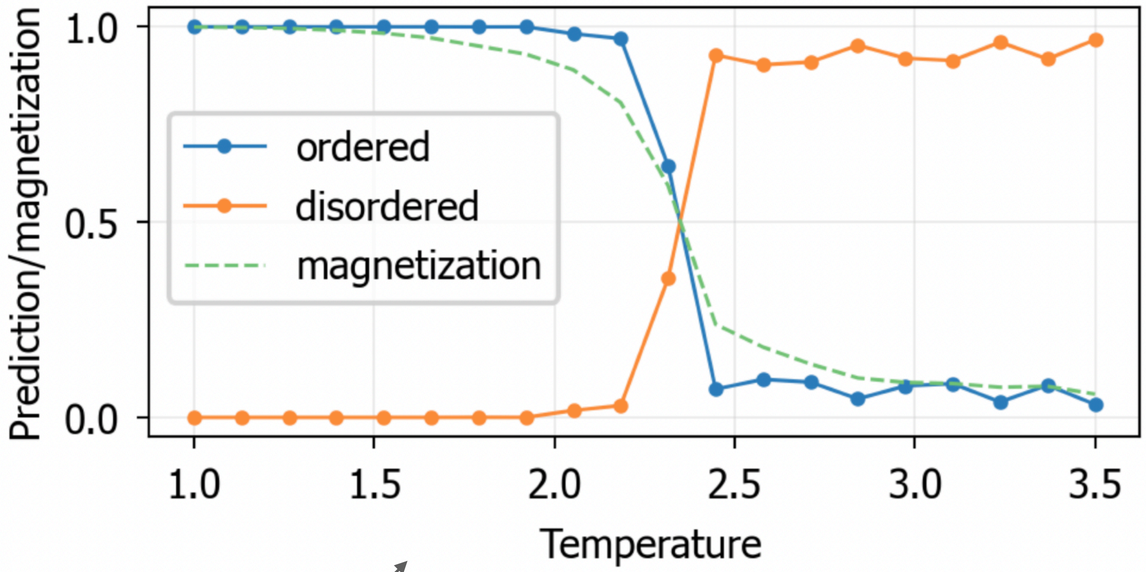
Experiment with neurons, layers, batches

…and get better accuracy than us!

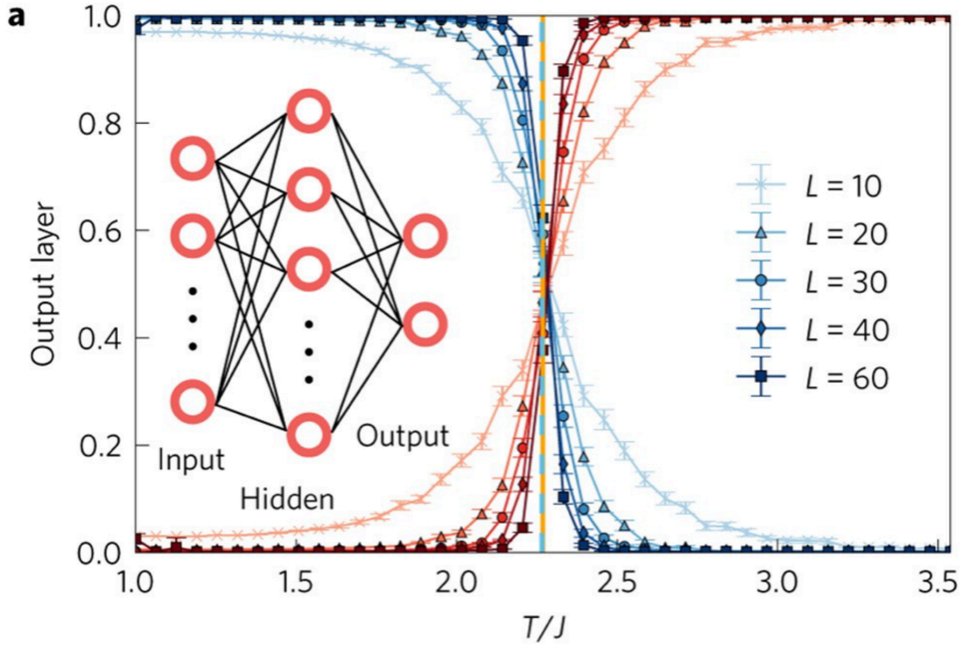
PyTorch

QMAI | TU Delft

Coding NNs in PyTorch



You will make this plot!



Carrasquilla&Melko, Nature Physics **13**, 431–434(2017)

https://www.eliskagreplova.com/ai-for-physicists-ap3751

QMAI | TU Delft

eliskagreplova.com

Neural nets: More sophisticated methods

Learning phases of matter with local order parameters with supervised learning

Unsupervised learning of topological phases of matter

Neural nets: More sophisticated methods

Previously we have seen:

(1) clustering works elegantly for simple problems, does not generalise well for hard ones

(2) supervised learning works great for simple and hard problems BUT if we had to label it first are we learning something new?

Learning by confusion

STEP 1: make a guess of phase transition temperature T_c

STEP 2: train a classification model assuming your guess in step 1 in a correct temperature

STEP 3: train many models as many guess of T_c as you like

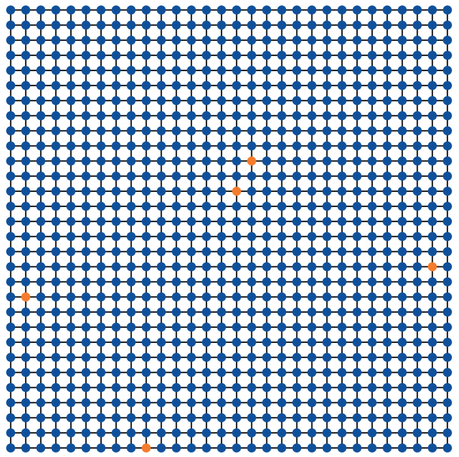
STEP 4: look at the accuracy of your model as a function of T_c

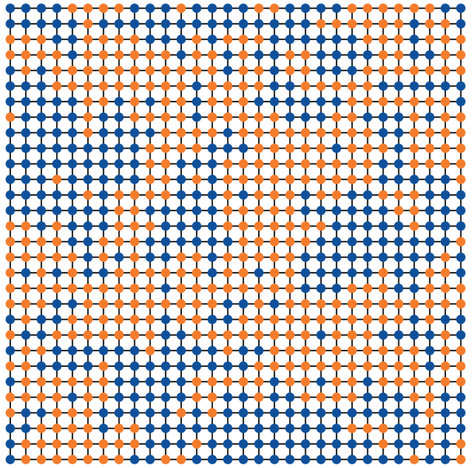
EPL Van Nieuwenburg, YH Liu, SD Huber - Nature Physics, 2017

QMAI | TU Delft

Learning by confusion

Step 1 : Just guess transition temperature



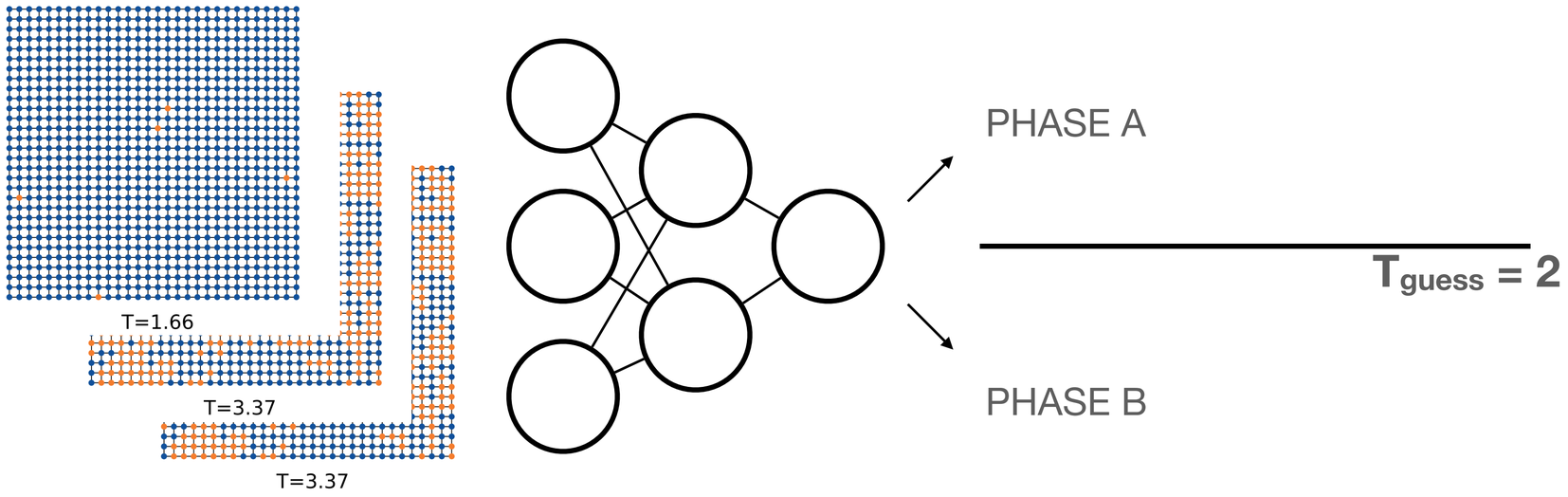


$T_{guess} = 2$

EPL Van Nieuwenburg, YH Liu, SD Huber - Nature Physics, 2017

Learning by confusion

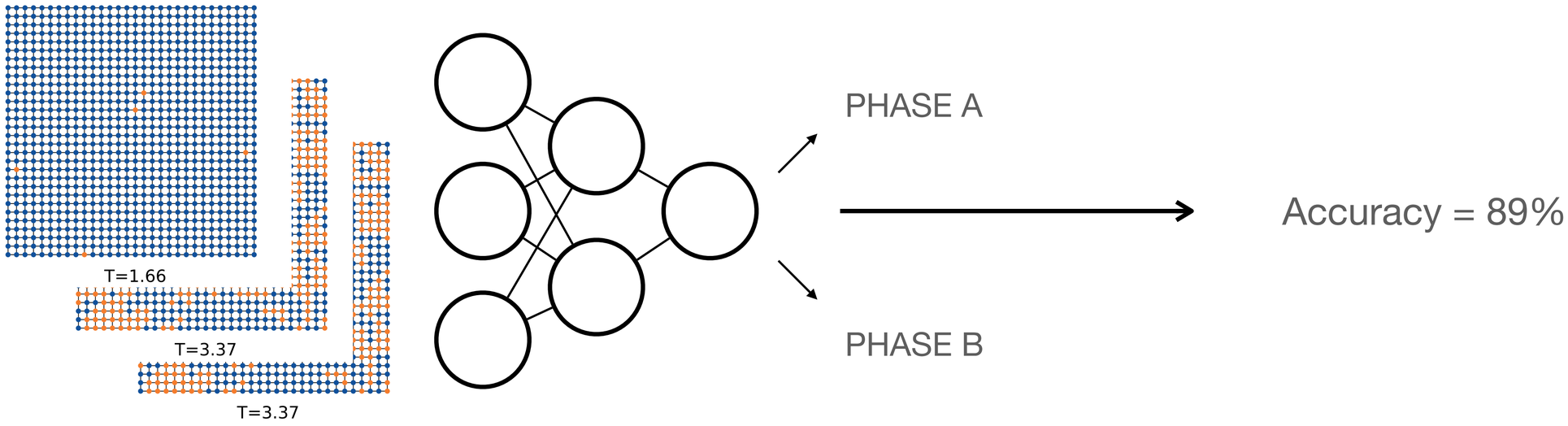
Step 2: Train supervised NN assuming T_{guess}



EPL Van Nieuwenburg, YH Liu, SD Huber - Nature Physics, 2017

Learning by confusion

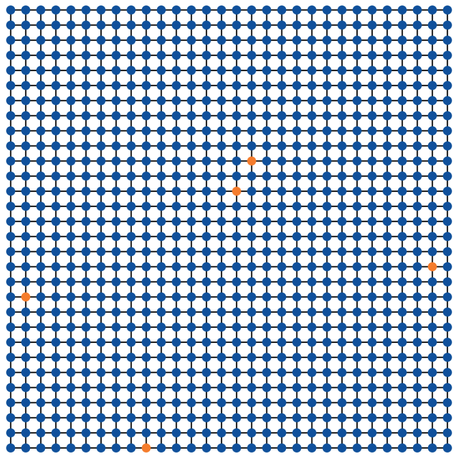
Step 3: Save accuracy

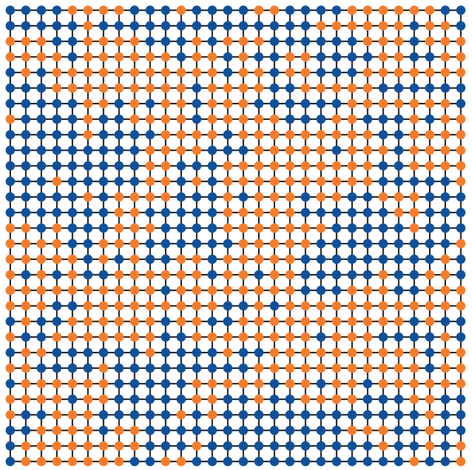


EPL Van Nieuwenburg, YH Liu, SD Huber - Nature Physics, 2017

Learning by confusion

Step 4: Make a new T_{guess} and repeat Steps 2-4



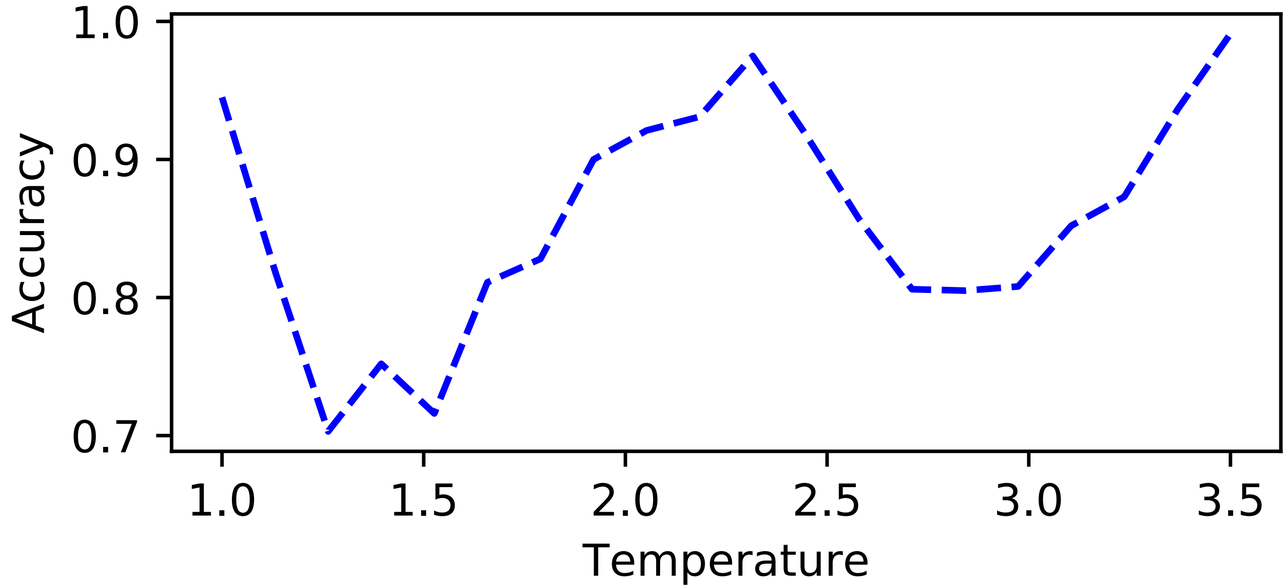


T_{guess} = 2 + epsilon

EPL Van Nieuwenburg, YH Liu, SD Huber - Nature Physics, 2017

Learning by confusion

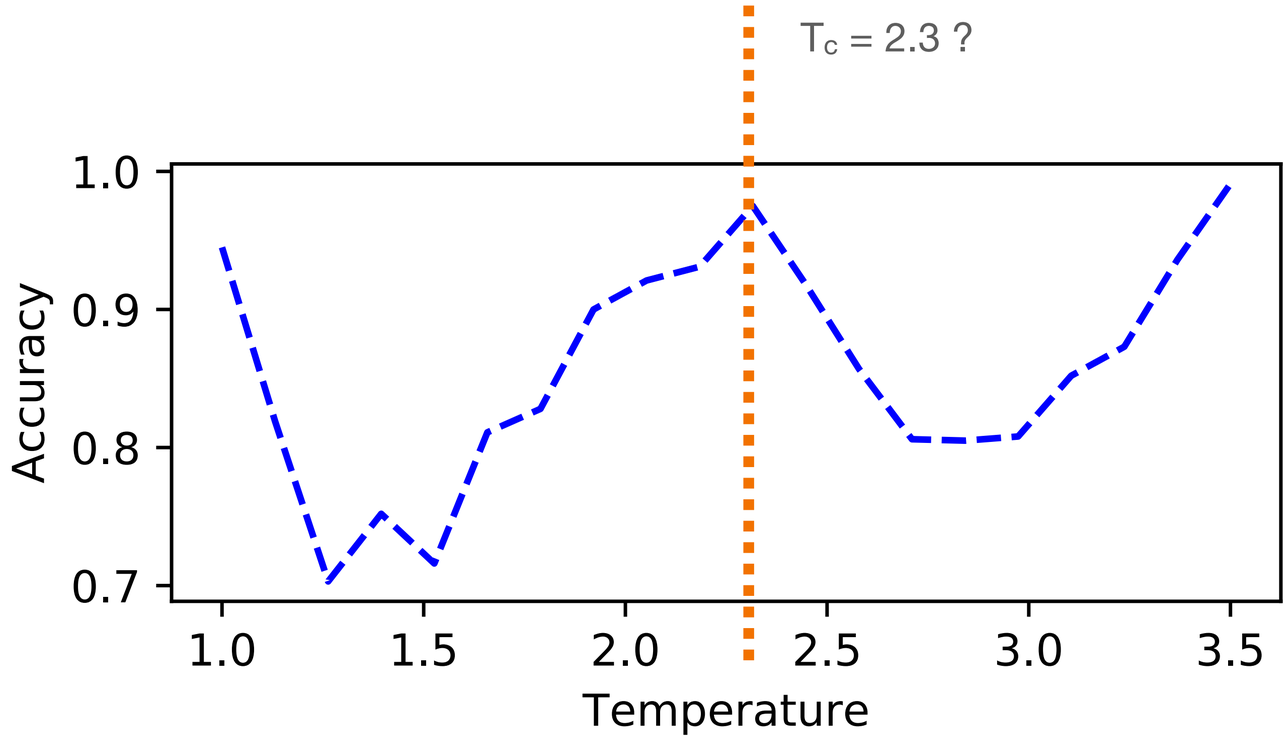
Step 5: Plot accuracies as a function of the transition temperature



EPL Van Nieuwenburg, YH Liu, SD Huber - Nature Physics, 2017

Learning by confusion

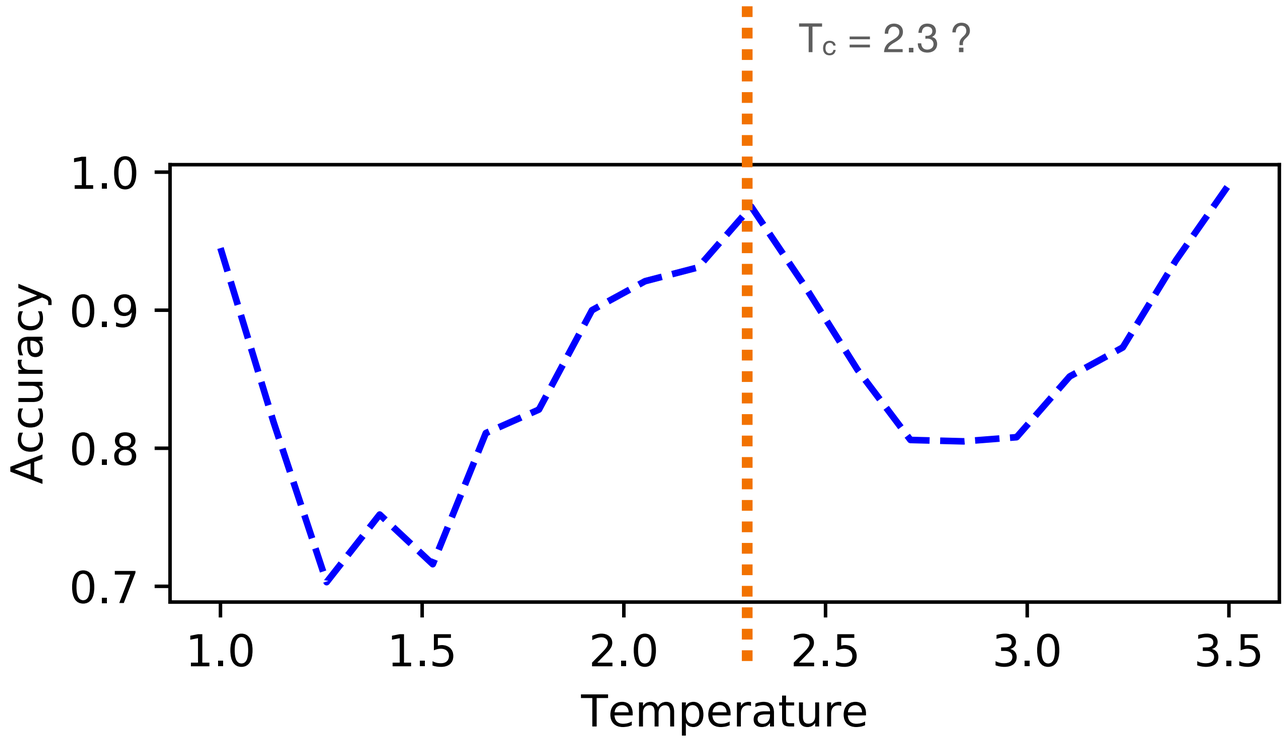
Step 5: Plot accuracies as a function of the transition temperature



EPL Van Nieuwenburg, YH Liu, SD Huber - Nature Physics, 2017

Learning by confusion

Step 5: Plot accuracies as a function of the transition temperature



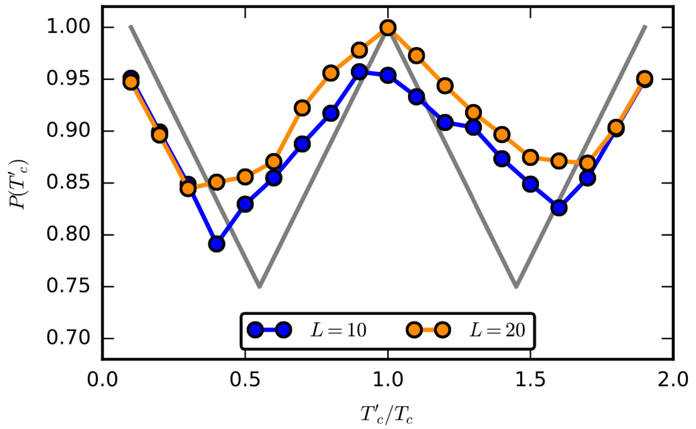
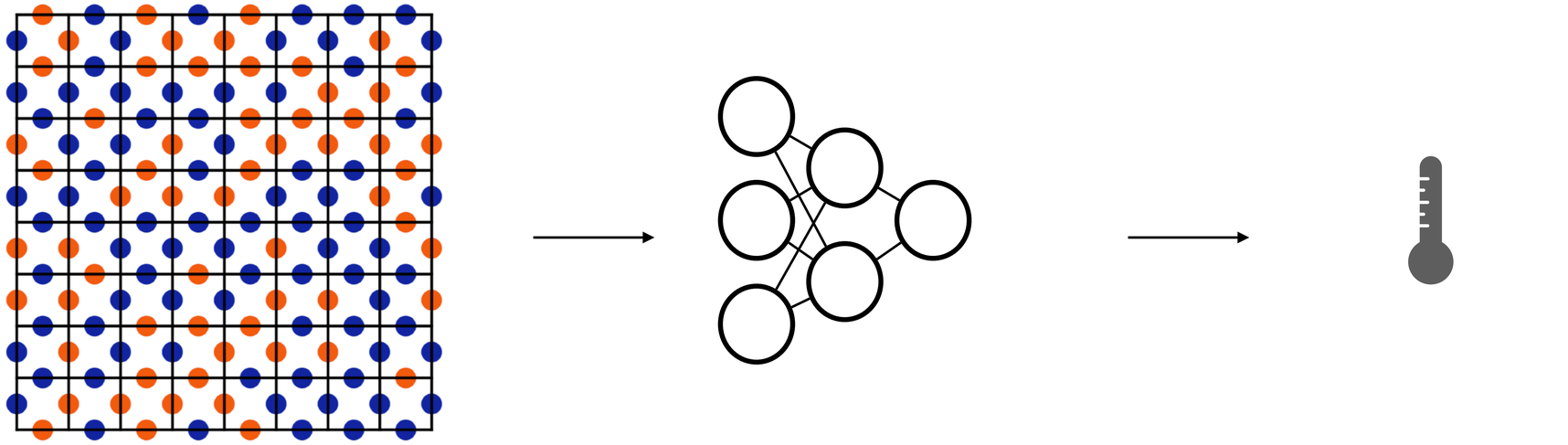


FIG. 2. Position of the middle peak in the universal W-shape deviates from $T_c' = T_c$ for $L = 10$ due to the finite-size effect. Here $k_B T_c \approx 2.27J$ is the exact transition temperature in the thermodynamic limit. For $L = 20$ the middle peak is located exactly at $T_c' = T_c$. (Parameters for training: batch size $N_b = 100$, learning rate $\alpha = 0.02$ and regularization $l_2 = 0.005$.)

EPL Van Nieuwenburg, YH Liu, SD Huber - Nature Physics, 2017

Unsupervised learning with a predictive model

Let's try to predict NOT the phase but parameter we have access to



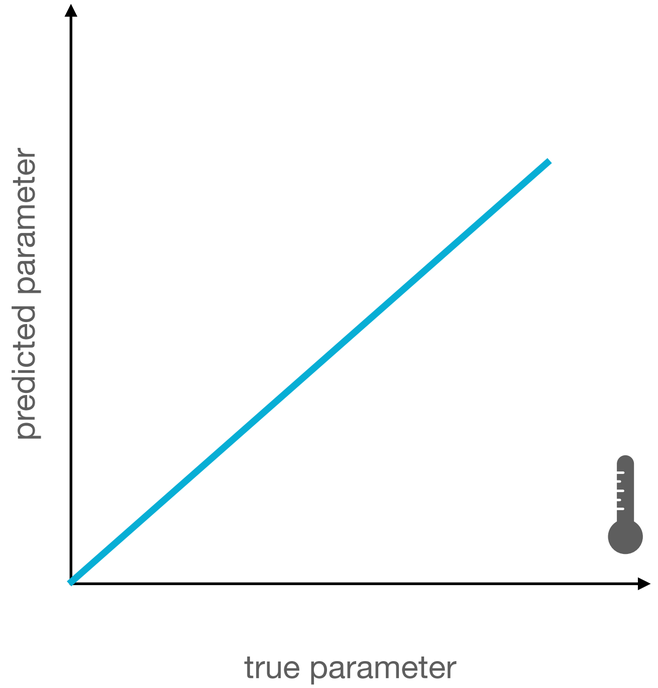
measurement

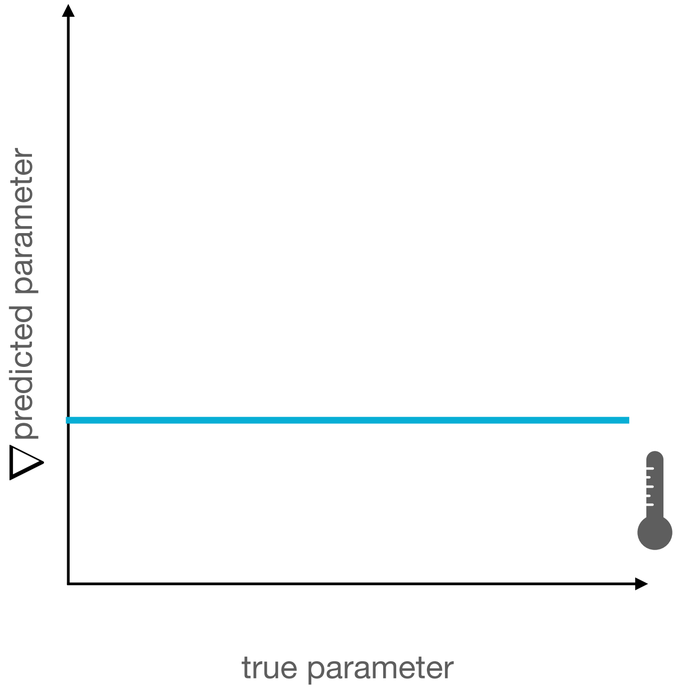
tuning parameter β

EG, A. Valenti, G. Boschung, F. Schäfer, N. Lörch, S. Huber, New J. Phys. 22 045003 (2020)
F. Schäfer, N. Lörch, Phys. Rev. E 99, 062107 (2019)

QMAI | TU Delft

Unsupervised learning:
Can we discover new phases just from data?

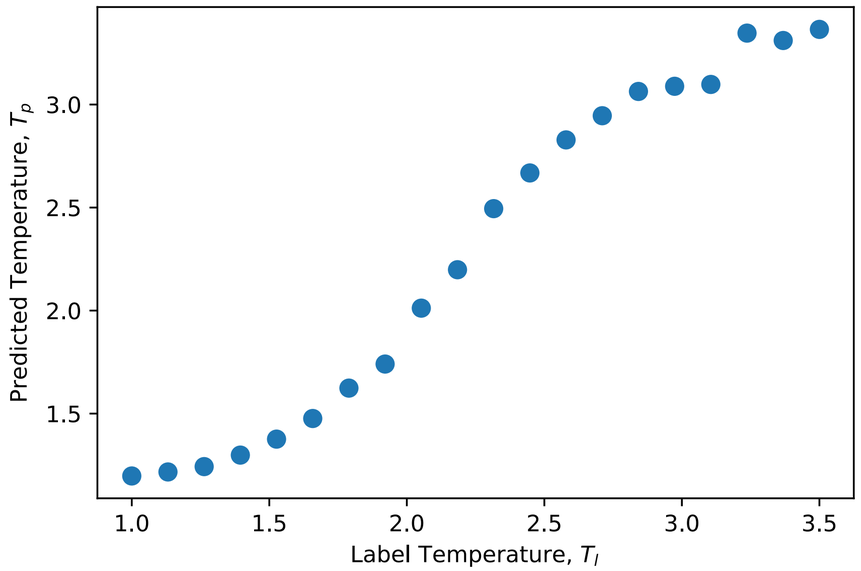


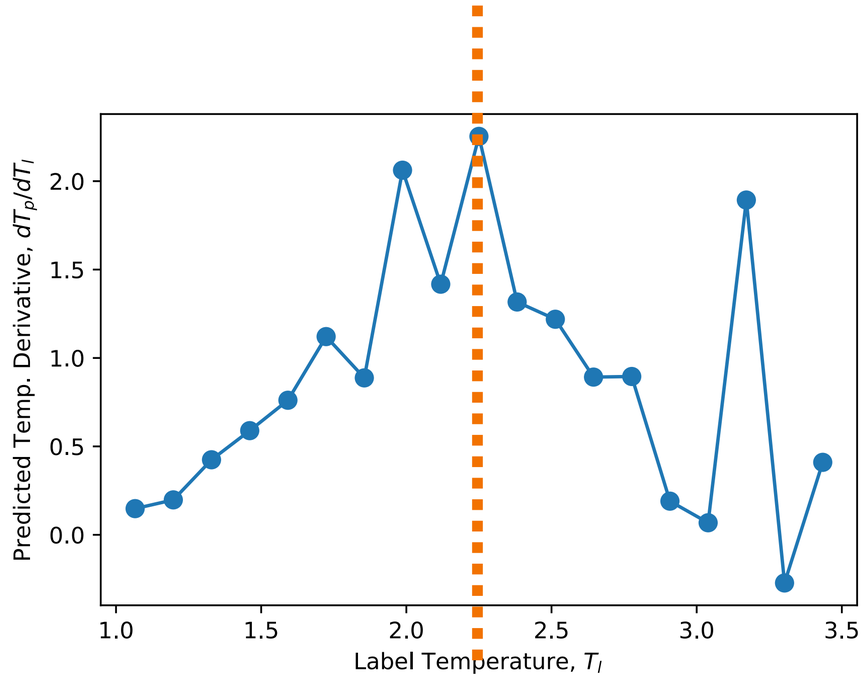


EG, A. Valenti, G. Boschung, F. Schäfer, N. Lörch, S. Huber, New J. Phys. 22 045003 (2020)
F. Schäfer, N. Lörch, Phys. Rev. E 99, 062107 (2019)

QMAI | TU Delft

Unsupervised learning:
Can we discover new phases just from data?





EG, A. Valenti, G. Boschung, F. Schäfer, N. Lörch, S. Huber, New J. Phys. 22 045003 (2020)
F. Schäfer, N. Lörch, Phys. Rev. E 99, 062107 (2019)

Unsupervised learning:
Can we discover new phases just from data?

STEP 1: train your network to predict the parameter you know: temperature of the sample

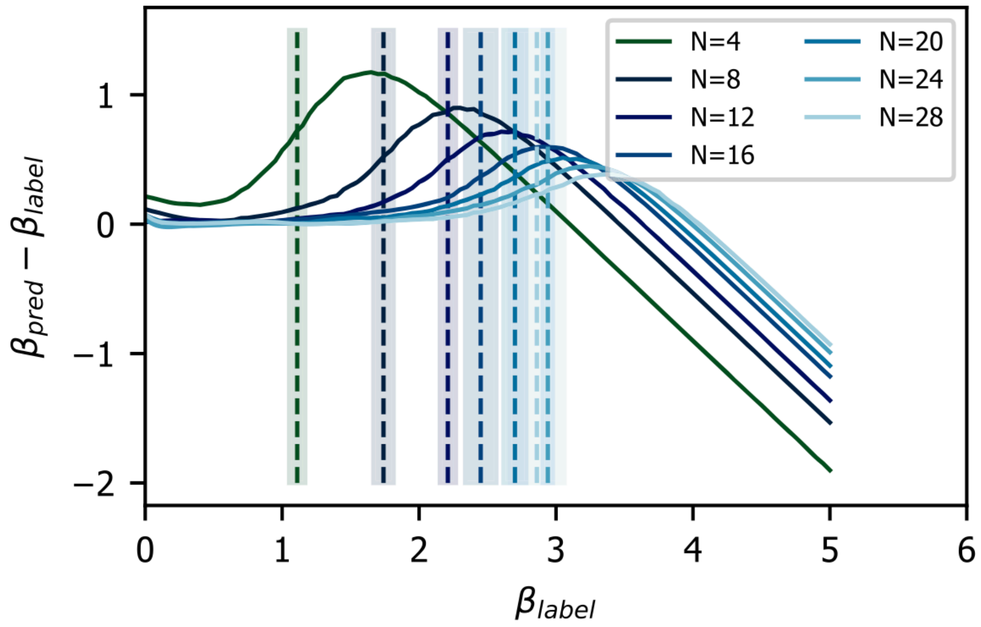
STEP 2: for the validation data plot correct temperature vs predicted temperature

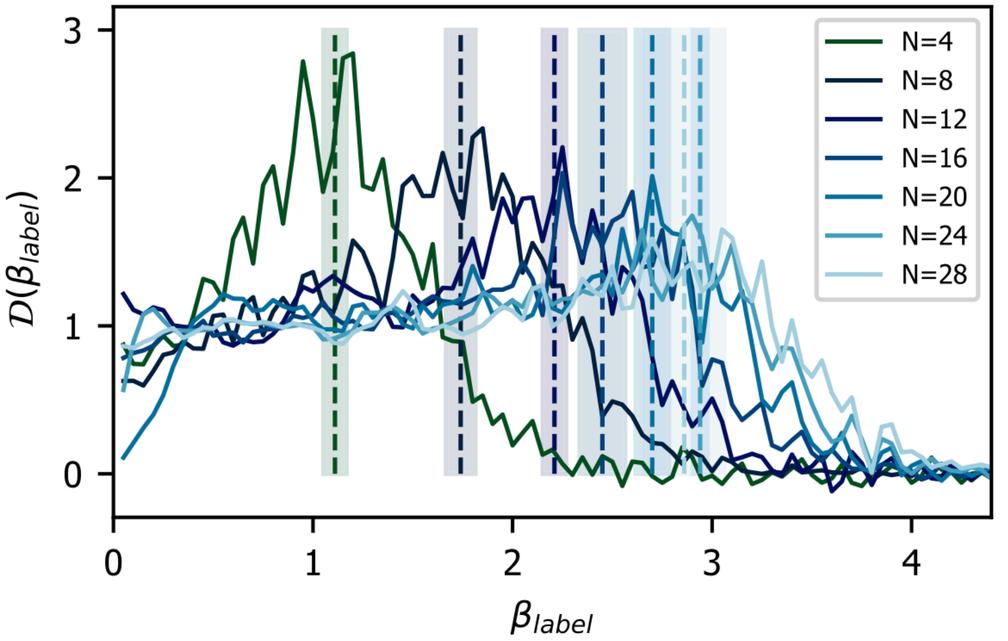
STEP 3: take numerical derivative of STEP 2

STEP 4: what is the temperature for which the derivative of the prediction shows the biggest change?

QMAI | TU Delft

Unsupervised learning: IGT





EG, A. Valenti, G. Boschung, F. Schäfer, N. Lörch, S. Huber, New J. Phys. 22 045003 (2020)

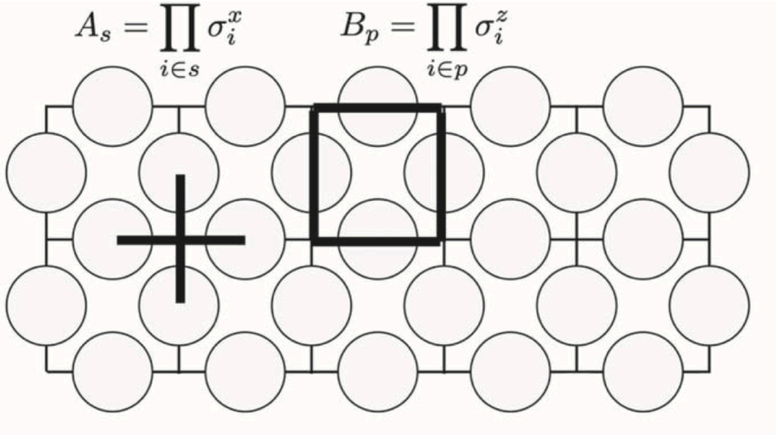
F. Schäfer, N. Lörch, Phys. Rev. E 99, 062107 (2019)

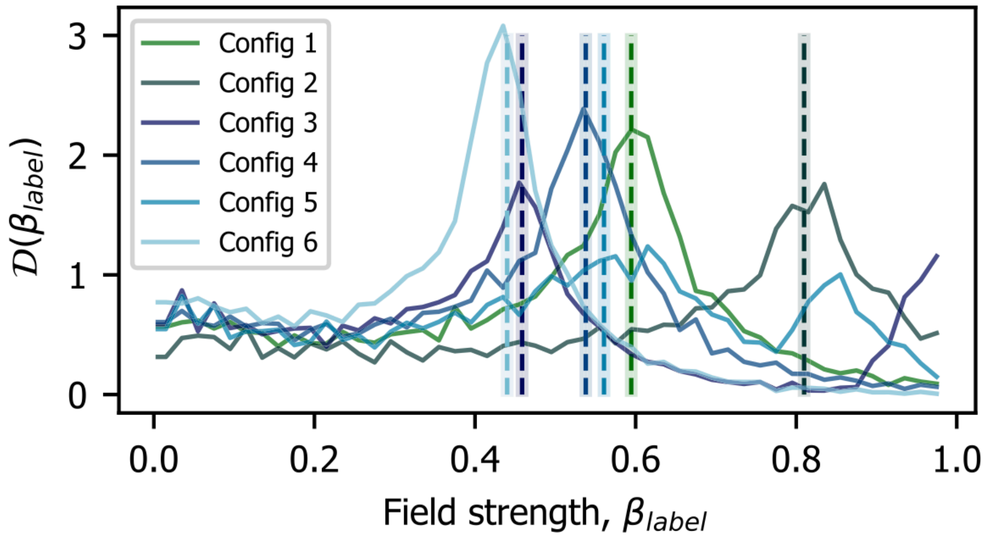
QMAI | TU Delft

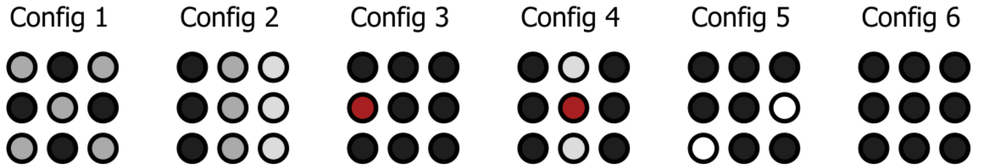
Unsupervised learning

For funky quantum topological phase transitions it works too!

TORIC CODE







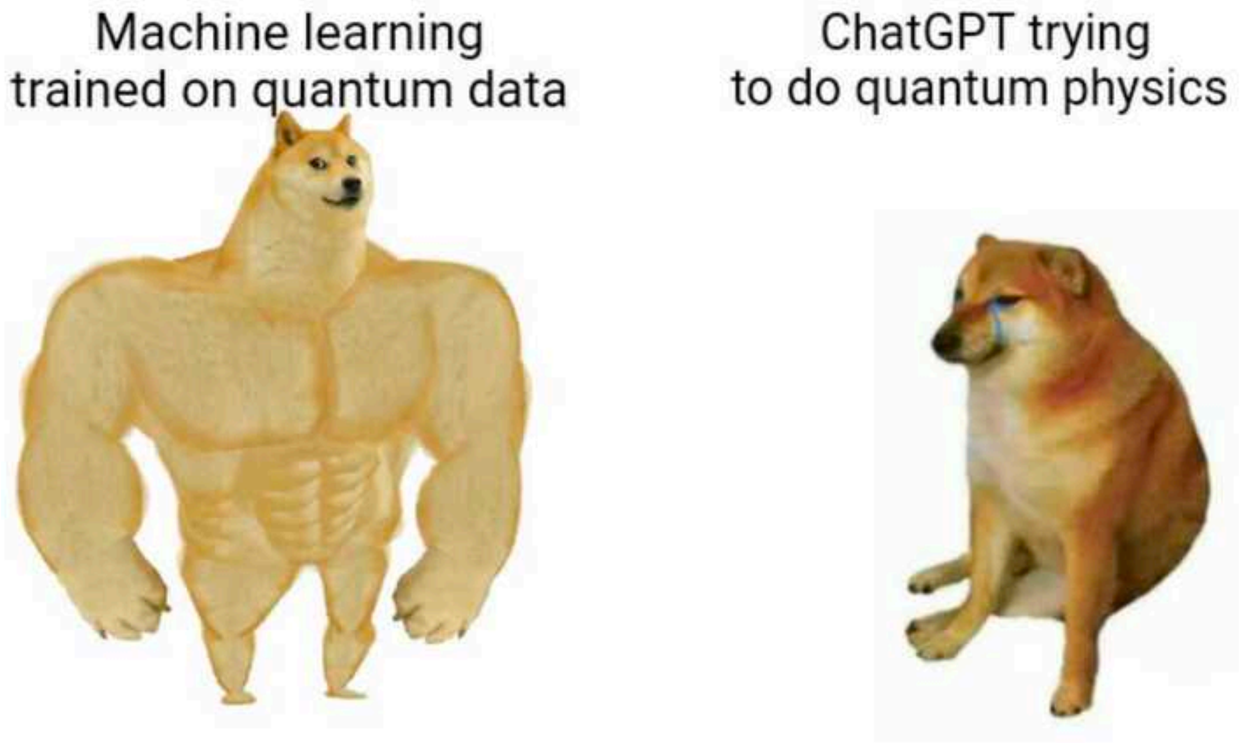
EG, A. Valenti, G. Boschung, F. Schäfer, N. Lörch, S. Huber, New J. Phys. 22 045003 (2020)

F. Schäfer, N. Lörch, Phys. Rev. E 99, 062107 (2019)

Wrapping-up

QMAI | TU Delft

- We can IDENTIFY MANY pretty complex physics phenomena using ML
- INTERPRETING them right the hard part
- Powerful tool for FEATURE IDENTIFICATION even without complex theory building when used right



TOMORROW WE LOOK AT SOME CONTEMPORARY STATE OF ML QUANTUM APPLICATIONS!

TU Delft Delft University of Technology

QMAI

QuTech

KAVLI INSTITUTE of Nanoscience Delft