Detecting composite orders in layered models via machine learning

Giacomo Bighin

Yerevan, Armenia, May 13th, 2022

Quantum Magnetism and Statistical Mechanics of Lattice Models

Introduction: coupled interacting systems

A system consisting of smaller subsystems, connected via a tunable coupling.

- Josephson junction between coupled superconductors or Bose-Einstein condensates.
- The droplet phase in bosonic mixtures.
- Lattice spin models: magnetic spin ladders and layered two-dimensional systems.

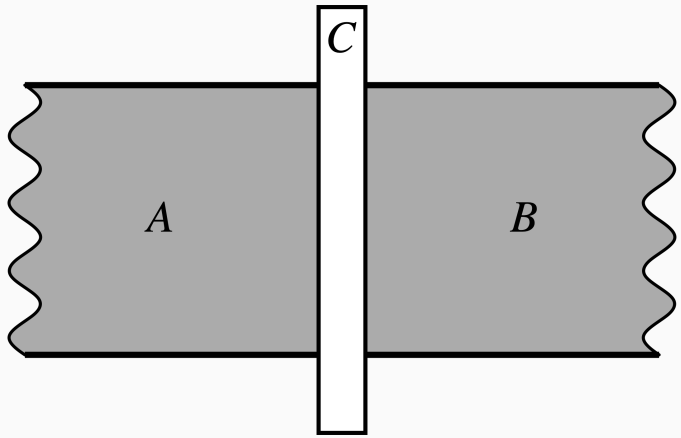


Image from: Wikimedia Commons

Are there new phases? How can we find them, if we don't know *a priori* the order parameter driving the transition?

Introduction: coupled interacting systems

A system consisting of smaller subsystems, connected via a tunable coupling.

- Josephson junction between coupled superconductors or Bose-Einstein condensates.
- The droplet phase in bosonic mixtures.
- Lattice spin models: magnetic spin ladders and layered two-dimensional systems.

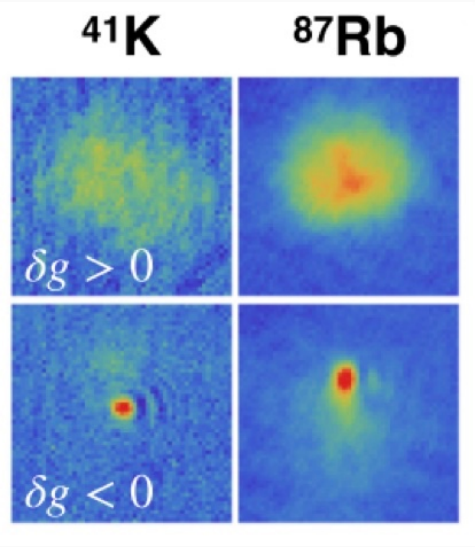


Image from: C. D'Errico *et al.*, Phys. Rev. Research **1**, 033155 (2019).

Are there new phases? How can we find them, if we don't know *a priori* the order parameter driving the transition?

Introduction: coupled interacting systems

A system consisting of smaller subsystems, connected via a tunable coupling.

- Josephson junction between coupled superconductors or Bose-Einstein condensates.
- The droplet phase in bosonic mixtures.
- Lattice spin models: magnetic spin ladders and layered two-dimensional systems.

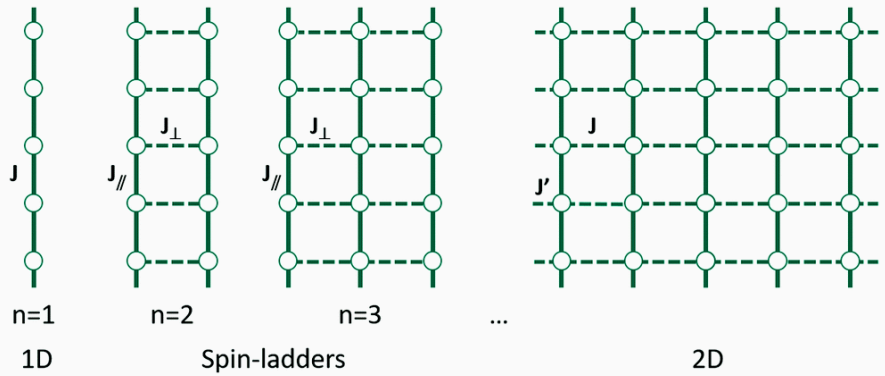


Image from: J. Mater. Chem. C **9**, 10573 (2021).

Are there new phases? How can we find them, if we don't know *a priori* the order parameter driving the transition?

Introduction: coupled interacting systems

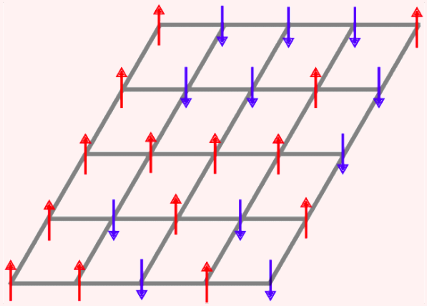
A system consisting of smaller subsystems, connected via a tunable coupling.

- Jose
  cou
  Bos
- The
  mix
- Latt
  spir
  two

Are the … der
parame…

This talk: I will demonstrate a machine-learning approach that can answer the question for classical, two-dimensional, layered square-lattice spin models.

SCAN ME



Based on: W. Rządkowski, N. Defenu, S. Chiacchiera, A. Trombettoni, GB, New J. Phys. **22**, 093026 (2020).

Image from: S. Wald, PhD thesis (2017).

Convolutional neural networks: a power tool for classification problems

Fully connected neural networks do not work very well for image classification. Inspired by the visual cortex in human brain, convolutional neural network (CNN) use a locally connected part, followed by a fully connected one.

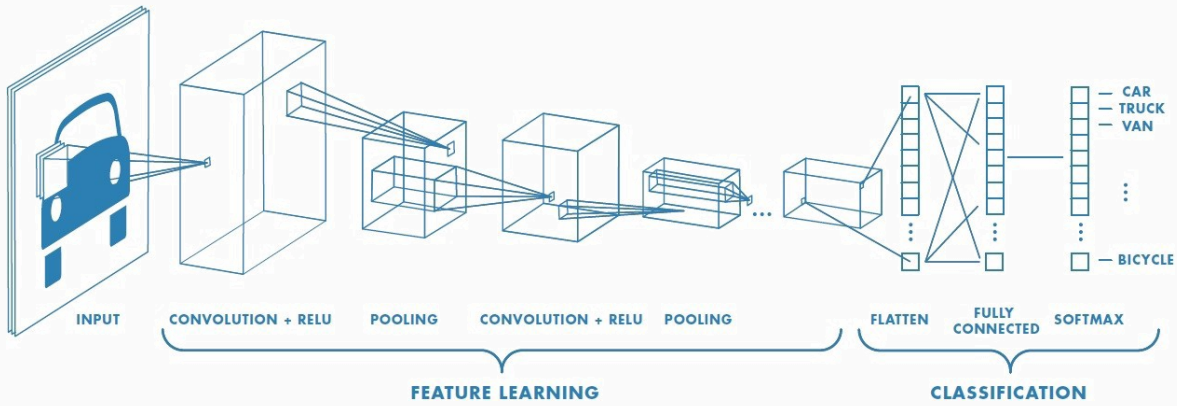


Image from: Sumit Saha, Towards Data Science

The convolution operation aims at extracting the higher level features from the image (i.e. edges, crossings, etc...). Example: MNIST.

It exploits local spatial correlations, while also making the process translationally invariant.

Very flexible at identifying patterns! Incidentally, this also makes CNNs good for the game of go!

Image from: Wikimedia Commons

Convolutional neural networks: the convolution operation

The locally connected part of a CNN (usually) consists in the repeated application of the convolution and pooling operation.

Convolution operation: in the example on the right we have a 5×5 image (green), convoluted with a 3×3 kernel (yellow), resulting in a 3×3 feature map.

The kernel entries are parameters to be optimized during the training of the network.

The convolution operation:

- Aims at turning the image into a feature map of high level features.
- Subsequent convolutional layers compose higher level features, reconstructing whole objects.

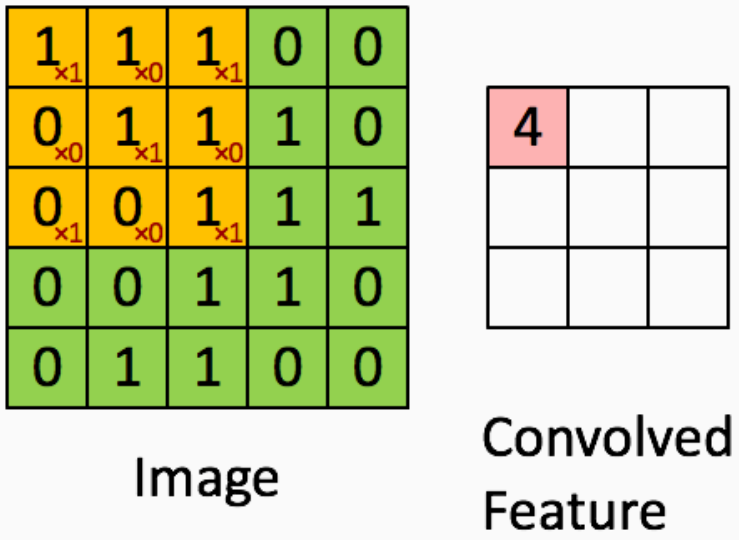


Image credit: Sumit Saha, Towards Data Science

Convolutional neural networks: the convolution operation

The locally connected part of a CNN (usually) consists in the repeated application of the convolution and pooling operation.

Convolution operation: in the example on the right we have a 5×5 image (green), convoluted with a 3×3 kernel (yellow), resulting in a 3×3 feature map.

The kernel entries are parameters to be optimized during the training of the network.

The convolution operation:

- Aims at turning the image into a feature map of high level features.
- Subsequent convolutional layers compose higher level features, reconstructing whole objects.

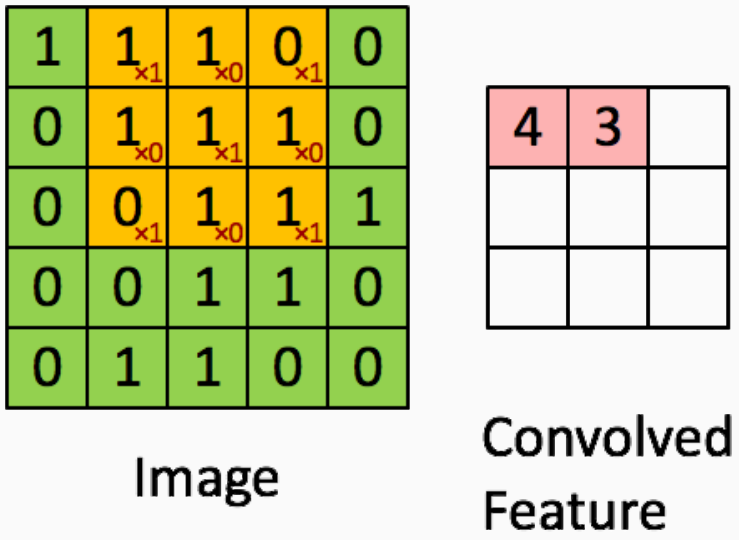


Image credit: Sumit Saha, Towards Data Science

Convolutional neural networks: the convolution operation

The locally connected part of a CNN (usually) consists in the repeated application of the convolution and pooling operation.

Convolution operation: in the example on the right we have a 5×5 image (green), convoluted with a 3×3 kernel (yellow), resulting in a 3×3 feature map.

The kernel entries are parameters to be optimized during the training of the network.

The convolution operation:

- Aims at turning the image into a feature map of high level features.
- Subsequent convolutional layers compose higher level features, reconstructing whole objects.

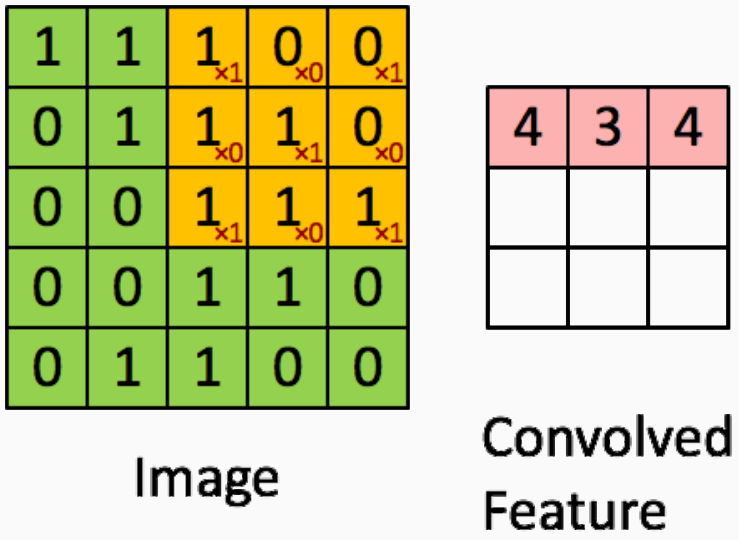


Image credit: Sumit Saha, Towards Data Science

Convolutional neural networks: the convolution operation

The locally connected part of a CNN (usually) consists in the repeated application of the convolution and pooling operation.

Convolution operation: in the example on the right we have a 5×5 image (green), convoluted with a 3×3 kernel (yellow), resulting in a 3×3 feature map.

The kernel entries are parameters to be optimized during the training of the network.

The convolution operation:

- Aims at turning the image into a feature map of high level features.
- Subsequent convolutional layers compose higher level features, reconstructing whole objects.

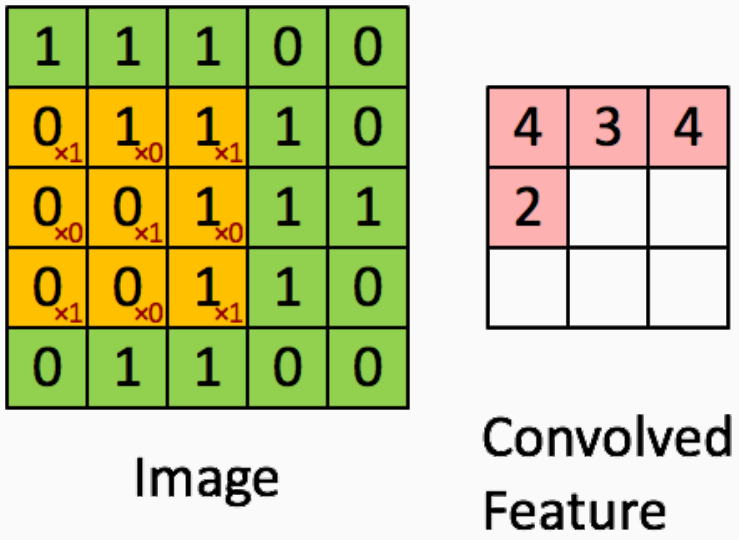


Image credit: Sumit Saha, Towards Data Science

Convolutional neural networks: the convolution operation

The locally connected part of a CNN (usually) consists in the repeated application of the convolution and pooling operation.

Convolution operation: in the example on the right we have a 5×5 image (green), convoluted with a 3×3 kernel (yellow), resulting in a 3×3 feature map.

The kernel entries are parameters to be optimized during the training of the network.

The convolution operation:

- Aims at turning the image into a feature map of high level features.
- Subsequent convolutional layers compose higher level features, reconstructing whole objects.

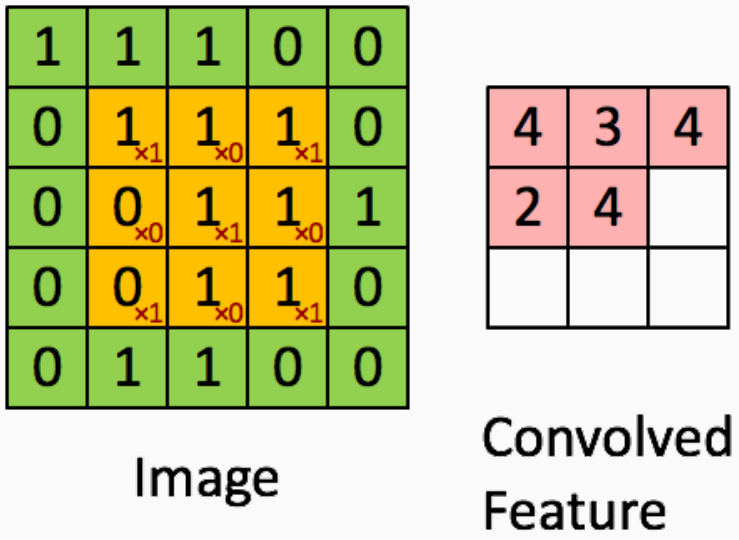


Image credit: Sumit Saha, Towards Data Science

Convolutional neural networks: the convolution operation

The locally connected part of a CNN (usually) consists in the repeated application of the convolution and pooling operation.

Convolution operation: in the example on the right we have a 5×5 image (green), convoluted with a 3×3 kernel (yellow), resulting in a 3×3 feature map.

The kernel entries are parameters to be optimized during the training of the network.

The convolution operation:

- Aims at turning the image into a feature map of high level features.
- Subsequent convolutional layers compose higher level features, reconstructing whole objects.

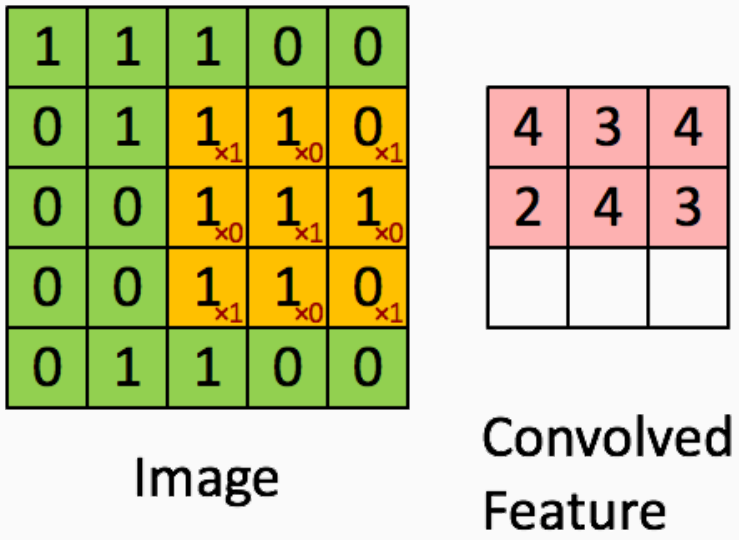


Image credit: Sumit Saha, Towards Data Science

Convolutional neural networks: the convolution operation

The locally connected part of a CNN (usually) consists in the repeated application of the convolution and pooling operation.

Convolution operation: in the example on the right we have a 5×5 image (green), convoluted with a 3×3 kernel (yellow), resulting in a 3×3 feature map.

The kernel entries are parameters to be optimized during the training of the network.

The convolution operation:

- Aims at turning the image into a feature map of high level features.
- Subsequent convolutional layers compose higher level features, reconstructing whole objects.

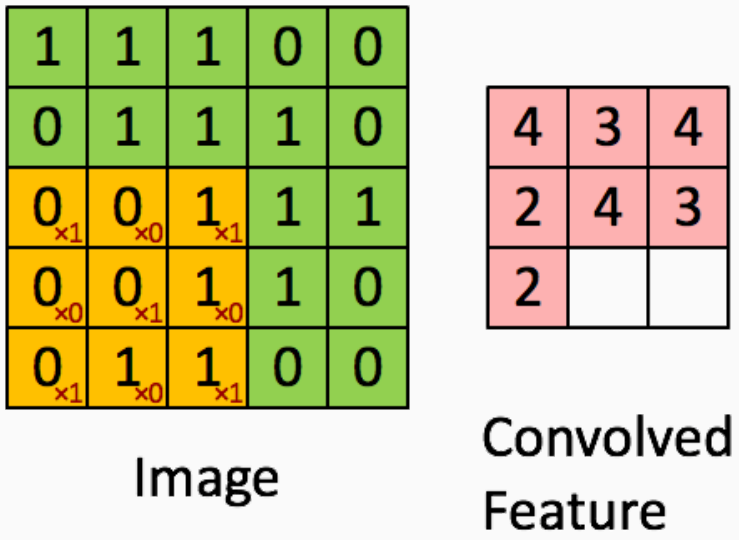


Image credit: Sumit Saha, Towards Data Science

Convolutional neural networks: the convolution operation

The locally connected part of a CNN (usually) consists in the repeated application of the convolution and pooling operation.

Convolution operation: in the example on the right we have a 5×5 image (green), convoluted with a 3×3 kernel (yellow), resulting in a 3×3 feature map.

The kernel entries are parameters to be optimized during the training of the network.

The convolution operation:

- Aims at turning the image into a feature map of high level features.
- Subsequent convolutional layers compose higher level features, reconstructing whole objects.

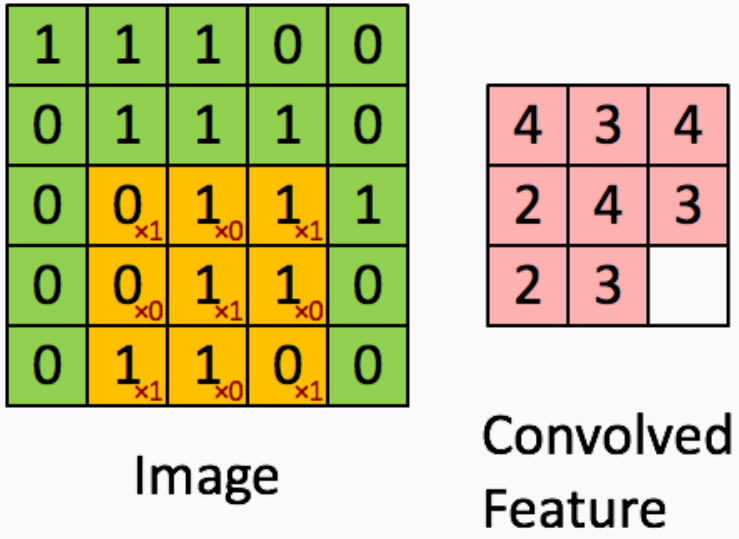


Image credit: Sumit Saha, Towards Data Science

Convolutional neural networks: the convolution operation

The locally connected part of a CNN (usually) consists in the repeated application of the convolution and pooling operation.

Convolution operation: in the example on the right we have a 5×5 image (green), convoluted with a 3×3 kernel (yellow), resulting in a 3×3 feature map.

The kernel entries are parameters to be optimized during the training of the network.

The convolution operation:

- Aims at turning the image into a feature map of high level features.
- Subsequent convolutional layers compose higher level features, reconstructing whole objects.

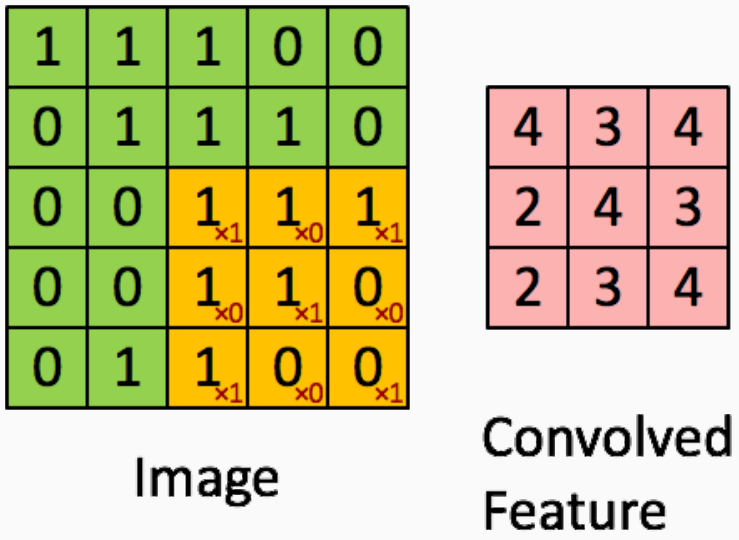


Image credit: Sumit Saha, Towards Data Science

Convolutional neural networks: the convolution operation

The locally connected part of a CNN (usually) consists in the repeated application of the convolution and pooling operation.

Convolution op
on the right we
(green), convol
(yellow), result
map.

The kernel ent
be optimized d
network.

The convolutio

- Aims at tu
feature ma
- Subseque
compose higher level features,
reconstructing whole objects.

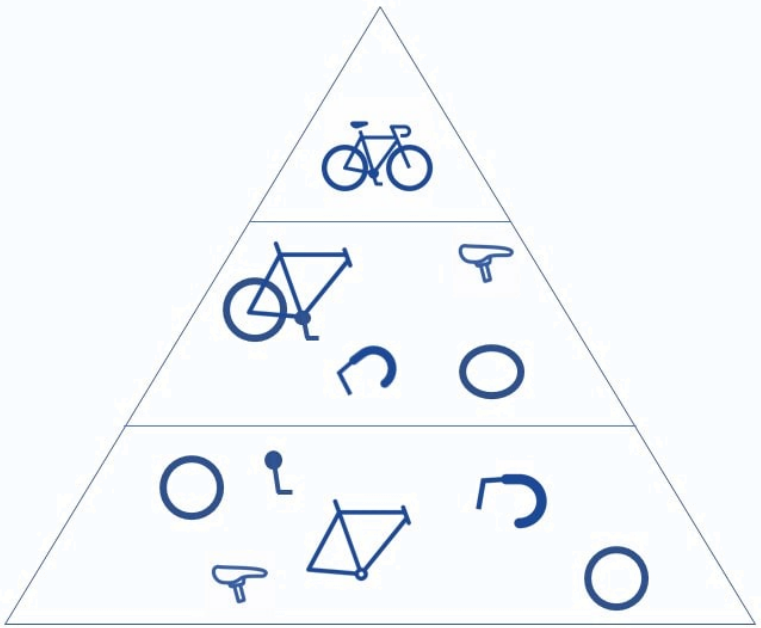


Image credit: IBM

4	3	4
2	4	3
2	3	4

Convolved Feature

Image credit: Sumit Saha, Towards Data Science

Convolutional neural networks: the pooling operation

Pooling operation: here we have a 5×5 feature map (orange), pooled via a 3×3 kernel, resulting in a 3×3 ouput (cyan).

No kernel here, just an aggregation function applied to values (maximum, average, etc...). Hence, no parameters are involved.

The pooling operation aims at:

- Reducing the size of the data.
- Keeping only the most relevant features (which are translationally invariant).
- Suppressing the noise.

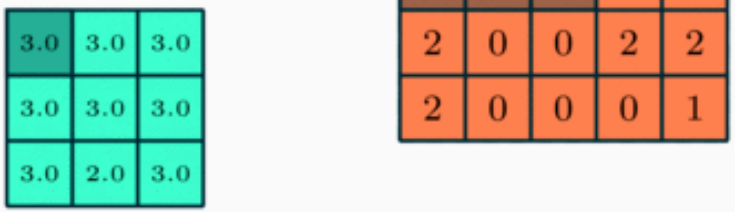


Image credit: Sumit Saha, Towards Data Science

The last pooling operation is usually followed by one or more fully connected layers for classification.

Convolutional neural networks: the pooling operation

Pooling operation: here we have a 5×5 feature map (orange), pooled via a 3×3 kernel, resulting in a 3×3 ouput (cyan).

No kernel here, just an aggregation function applied to values (maximum, average, etc...). Hence, no parameters are involved.

The pooling operation aims at:

- Reducing the size of the data.
- Keeping only the most relevant features (which are translationally invariant).
- Suppressing the noise.

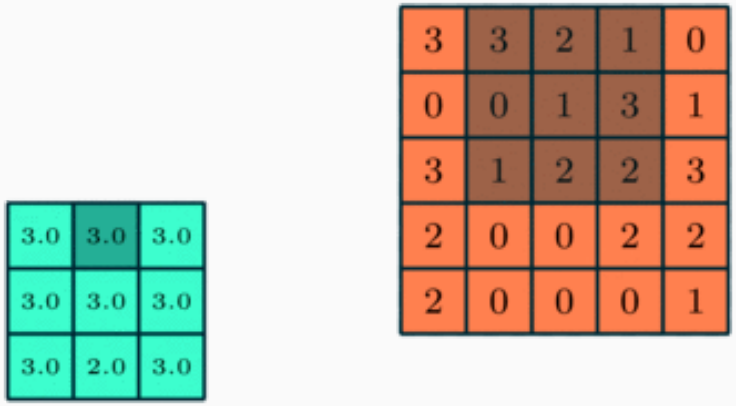


Image credit: Sumit Saha, Towards Data Science

The last pooling operation is usually followed by one or more fully connected layers for classification.

Convolutional neural networks: the pooling operation

Pooling operation: here we have a 5×5 feature map (orange), pooled via a 3×3 kernel, resulting in a 3×3 ouput (cyan).

No kernel here, just an aggregation function applied to values (maximum, average, etc...). Hence, no parameters are involved.

The pooling operation aims at:

- Reducing the size of the data.
- Keeping only the most relevant features (which are translationally invariant).
- Suppressing the noise.

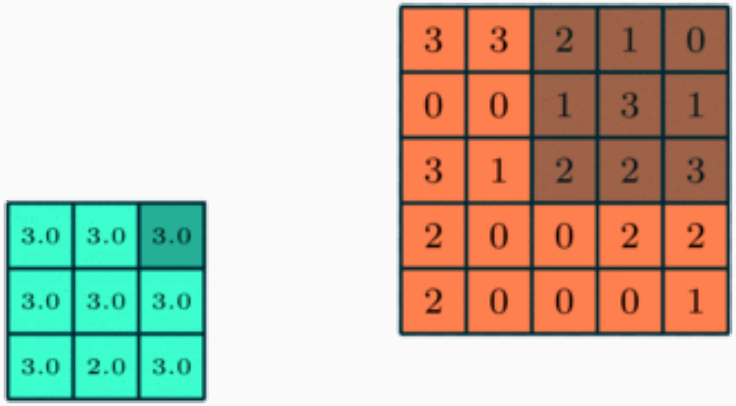


Image credit: Sumit Saha, Towards Data Science

The last pooling operation is usually followed by one or more fully connected layers for classification.

Convolutional neural networks: the pooling operation

Pooling operation: here we have a 5×5 feature map (orange), pooled via a 3×3 kernel, resulting in a 3×3 ouput (cyan).

No kernel here, just an aggregation function applied to values (maximum, average, etc...). Hence, no parameters are involved.

The pooling operation aims at:

- Reducing the size of the data.
- Keeping only the most relevant features (which are translationally invariant).
- Suppressing the noise.

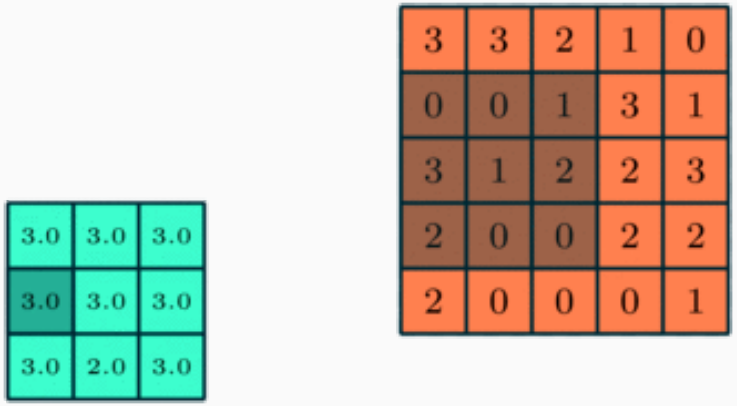


Image credit: Sumit Saha, Towards Data Science

The last pooling operation is usually followed by one or more fully connected layers for classification.

Convolutional neural networks: the pooling operation

Pooling operation: here we have a 5×5 feature map (orange), pooled via a 3×3 kernel, resulting in a 3×3 ouput (cyan).

No kernel here, just an aggregation function applied to values (maximum, average, etc...). Hence, no parameters are involved.

The pooling operation aims at:

- Reducing the size of the data.
- Keeping only the most relevant features (which are translationally invariant).
- Suppressing the noise.

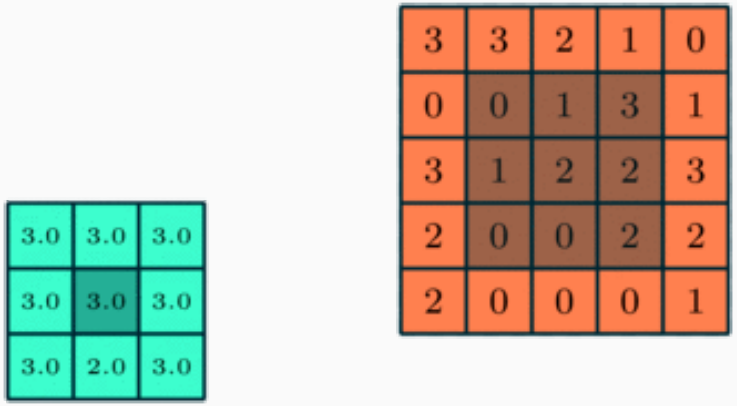


Image credit: Sumit Saha, Towards Data Science

The last pooling operation is usually followed by one or more fully connected layers for classification.

Convolutional neural networks: the pooling operation

Pooling operation: here we have a 5×5 feature map (orange), pooled via a 3×3 kernel, resulting in a 3×3 ouput (cyan).

No kernel here, just an aggregation function applied to values (maximum, average, etc...). Hence, no parameters are involved.

The pooling operation aims at:

- Reducing the size of the data.
- Keeping only the most relevant features (which are translationally invariant).
- Suppressing the noise.

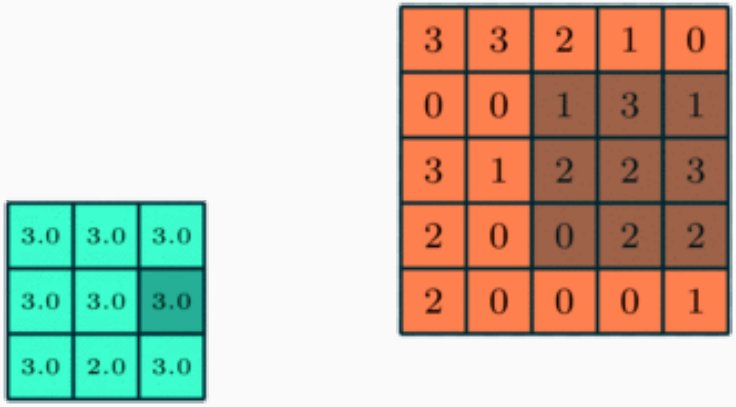


Image credit: Sumit Saha, Towards Data Science

The last pooling operation is usually followed by one or more fully connected layers for classification.

Convolutional neural networks: the pooling operation

Pooling operation: here we have a 5×5 feature map (orange), pooled via a 3×3 kernel, resulting in a 3×3 ouput (cyan).

No kernel here, just an aggregation function applied to values (maximum, average, etc...). Hence, no parameters are involved.

The pooling operation aims at:

- Reducing the size of the data.
- Keeping only the most relevant features (which are translationally invariant).
- Suppressing the noise.

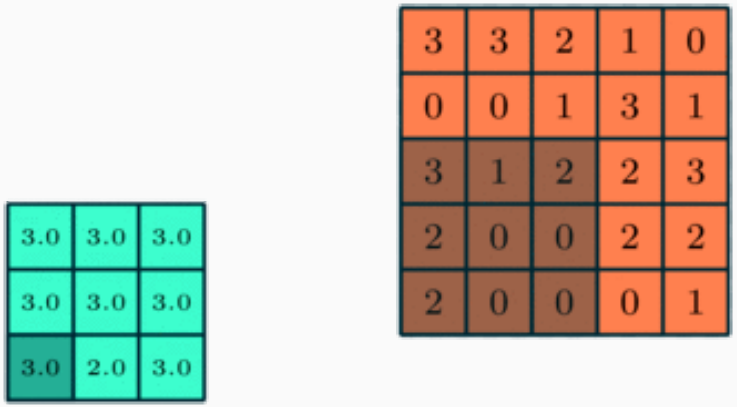


Image credit: Sumit Saha, Towards Data Science

The last pooling operation is usually followed by one or more fully connected layers for classification.

Convolutional neural networks: the pooling operation

Pooling operation: here we have a 5×5 feature map (orange), pooled via a 3×3 kernel, resulting in a 3×3 ouput (cyan).

No kernel here, just an aggregation function applied to values (maximum, average, etc...). Hence, no parameters are involved.

The pooling operation aims at:

- Reducing the size of the data.
- Keeping only the most relevant features (which are translationally invariant).
- Suppressing the noise.

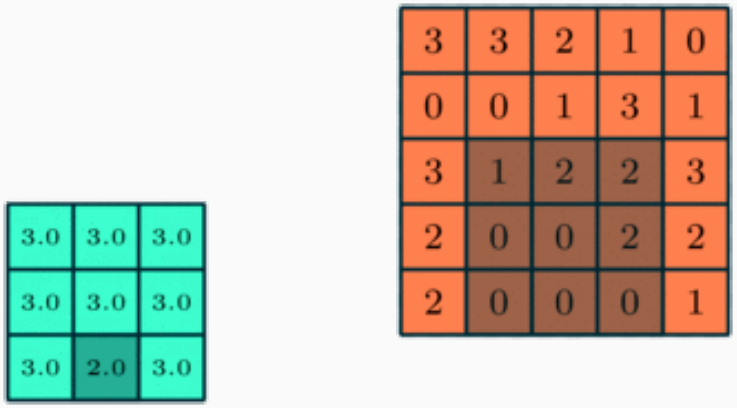


Image credit: Sumit Saha, Towards Data Science

The last pooling operation is usually followed by one or more fully connected layers for classification.

Convolutional neural networks: the pooling operation

Pooling operation: here we have a 5×5 feature map (orange), pooled via a 3×3 kernel, resulting in a 3×3 ouput (cyan).

No kernel here, just an aggregation function applied to values (maximum, average, etc...). Hence, no parameters are involved.

The pooling operation aims at:

- Reducing the size of the data.
- Keeping only the most relevant features (which are translationally invariant).
- Suppressing the noise.

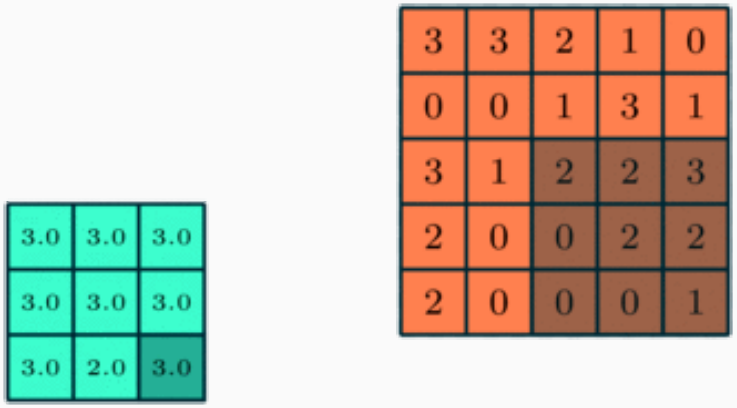


Image credit: Sumit Saha, Towards Data Science

The last pooling operation is usually followed by one or more fully connected layers for classification.

Convolutional neural networks: applications

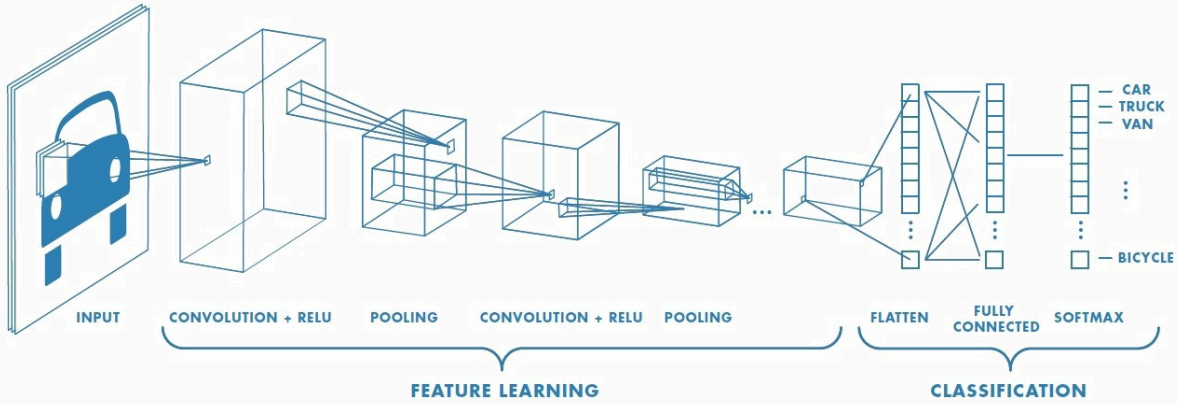


Image from: Sumit Saha, Towards Data Science

- Classification of images and videos.
- Natural language processing.
- Financial time series.
- Medical imaging.
- ...

Applications in physics (among many others): locating a phase transition? Does it work also for composite or hidden order parameters?

Convolutions neural networks for multi-layer spin models: the problem

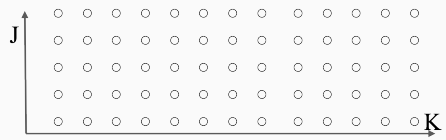
We consider a spin system whose Hamiltonian is defined by two parameters, J and K. For consistency let us consider a classical, two-dimensional Ising bilayer

$$H_{\text{bilayer}} = -J\sum_{\langle ij\rangle}\sigma_i\sigma_j - J\sum_{\langle ij\rangle}\tau_i\tau_j - K\sum_i \sigma_i\tau_i$$

where $\sigma_i, \tau_i = \pm 1$ are Ising variables on a two-dimensional square lattice.

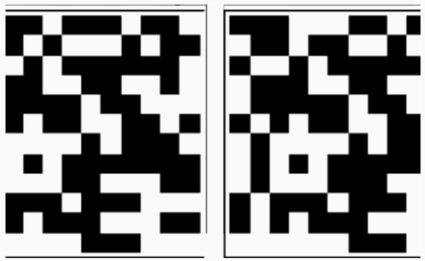
We generate a large number (600) of uncorrelated Monte Carlo snapshots of the model at equilibrium on a $32 \times 32 \times 2$ lattice, having fixed βJ and βK.

Two-dimensional (discretized) phase diagram



J: **intra**-layer coupling
K: **inter**-layer coupling

For each (J, K) point, the configuration of the snapshots are encoded into binary samples of shape $32 \times 32 \times 2$.

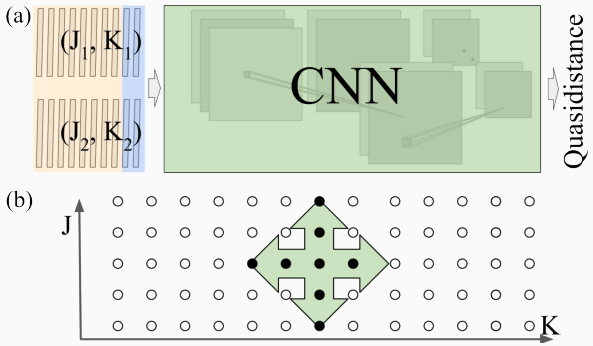


Assign a phase to each point:

- Not knowing the number of phases.
- Not knowing the properties of each phase.

Convolutions neural networks for multi-layer spin models: the algorithm (1/2)

1. For each (J, K) point divide the data into 80% training & 20% validation.
2. For each pair of nearby points (J_1, K_1), (J_2, K_2) attempt training the CNN.
3. Calculate quasidistance between (J_1, K_1) and (J_2, K_2) based on the training outcome. Training fails: same phase. Training succeeds: different phases.



More precisely: we quantify the classification accuracy on the validation set as the fraction φ of correctly labeled examples from the validation set. Learning from confusion! We introduce the following quasidistance between the two phase diagram points (J_1, K_1) and (J_2, K_2):

$$d((J_1, K_1), (J_2, K_2)) = 2(\varphi - 0.5)\Theta(\varphi - 0.5) \ ,$$

where $\Theta(x)$ is the Heavyside step function. Perfect discrimination $\varphi = 1$ (signaling different phases) corresponds to $d = 1$, while perfect confusion $\varphi = 0.5$ (signaling the same phase) corresponds to $d = 0$.

Convolutions neural networks for multi-layer spin models: the algorithm (2/2)

4. Based on the quasidistances, calculate and plot the lattice Laplacian.

More in detail: we make use of the quasi-distances to construct a field $u(J, K)$ defined through its finite-difference lattice gradient

$$\nabla u(J,K) = \begin{pmatrix} (u(J+\Delta J, K) - u(J,K))/\Delta J \\ (u(J, K+\Delta K) - u(J,K))/\Delta K \end{pmatrix} \equiv \begin{pmatrix} d((J+\Delta J, K),(J,K))/\Delta J \\ d((J, K+\Delta K),(J,K))/\Delta K \end{pmatrix}$$

Clearly, ∇u will be constant in phase diagram regions belonging to the same phase, since we expect that the difficulty of telling first neighbors apart should be uniformly quite high. On the other hand, we expect the value of ∇u to abruptly change in the vicinity of a phase transition. It is then natural to introduce the finite-difference lattice Laplacian

$$\nabla^2 u(J,K) \approx \frac{1}{(\Delta J)^2} \sum_{i=0}^{n} (-1)^i \binom{n}{i} u(J + (n/2 - i)\Delta J, K) + (\text{same for K})$$

with the $n = 2$, $n = 3$ and $n = 4$ cases corresponding to a 5-point, 9-point or 13-point stencil, respectively.

Convolutions neural networks for multi-layer spin models: the algorithm (2/2)

4. Based on the quasidistances, calculate and plot the lattice Laplacian.

More in detail: we make use of the quasi-distances to construct a field $u(J$

∇u ΔJ
ΔK

Cle e
pha t
sho e
of n
nat

Incidentally (or not?): a connection with **quantum fidelity**!

Define $|\Psi_0(\lambda)\rangle$ as the ground state of a system, whose Hamiltonian depends on a parameter λ, and consider the quantum fidelity defined as

$$|\langle\Psi_0(\lambda)|\Psi_0(\lambda+\delta\lambda)\rangle|^2 \approx 1 - \frac{(\delta\lambda)^2}{2}\chi_F$$

and the fidelity susceptibility χ_F diverges at a phase transition.

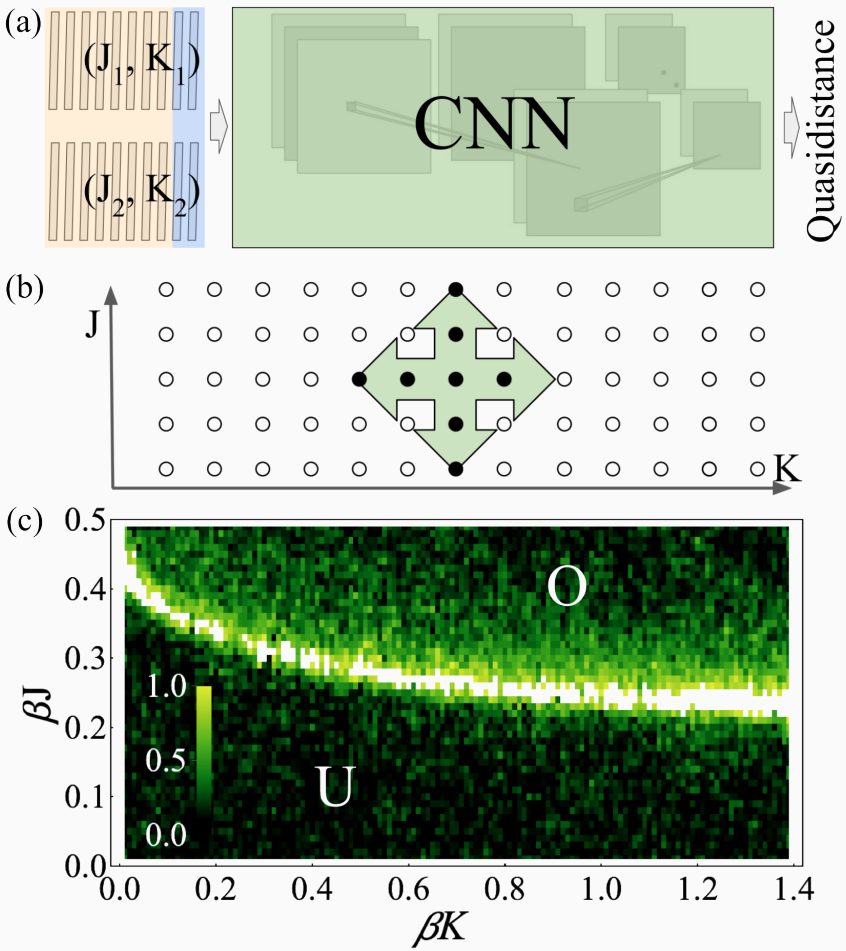
For a review of this method, see: S.-J. Gu, Int. J. Mod. Phys. B **24**, 4371 (2010).

$$\nabla^2 u(J,K) \approx \frac{1}{(\Delta J)^2}\sum_{i=0}(-1)^i \binom{n}{i} u(J+(n/2-i)\Delta J), K) + (\text{same for K})$$

with the $n=2$, $n=3$ and $n=4$ cases corresponding to a 5-point, 9-point or 13-point stencil, respectively.

Ising bilayer: phase diagram

- Finally, the reconstructed phase diagram is immediately visualized from the plot of the lattice Laplacian $\nabla^2 u$ as a function of βJ and βK.
- Expected behaviour for the classical Ising bilayer: ordered and unordered phases.
- Limiting values in agreement with the classical analytical result by Onsager $(\beta J)_c = \log\left(1+\sqrt{2}\right)/2 \approx 0.44$.



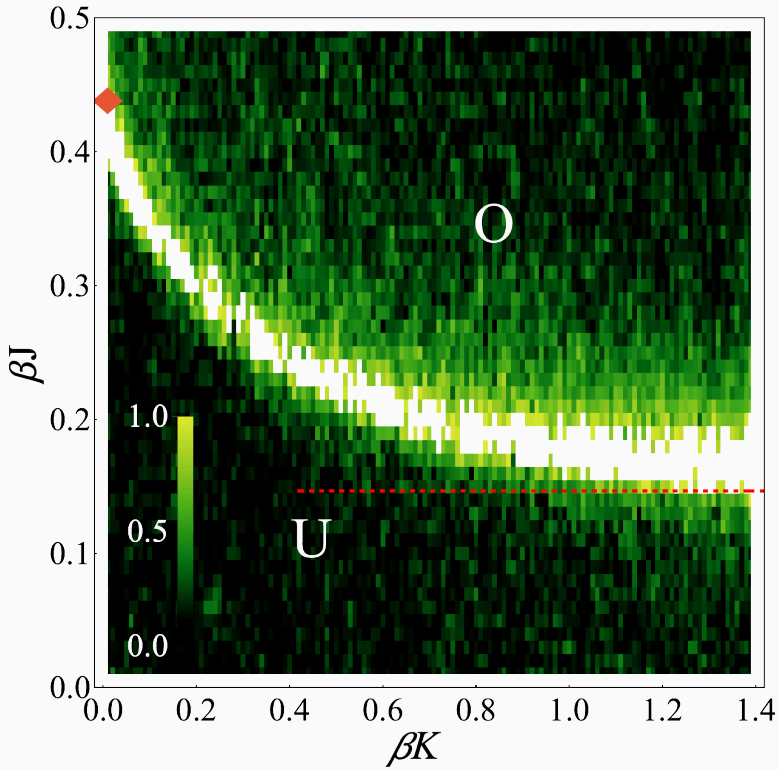
Ising trilayer: phase diagram

Next, we consider a trilayer system, whose Hamiltonian is:

$$H_{\text{trilayer}} = -J\sum_{\langle ij\rangle}\sigma_i\sigma_j - J\sum_{\langle ij\rangle}\tau_i\tau_j - J\sum_{\langle ij\rangle}\upsilon_i\upsilon_j - K\sum_i\sigma_i\tau_i - K\sum_i\tau_i\upsilon_i\ ,$$

and we apply the same procedure to characterize its phase diagram.

- In the uncoupled limit ($\beta K \to 0$) one sees $(\beta J)_c \approx 0.44$ in good agreement with the famous analytical result by Onsager $(\beta J)_c = \log\left(1+\sqrt{2}\right)/2$.
- The strong-interlayer-coupling critical temperature is $(\beta J)_c'' = (\beta J)_c/3$.
- We see no evidence of other phases, besides the 'usual' ordered an unordered phases.

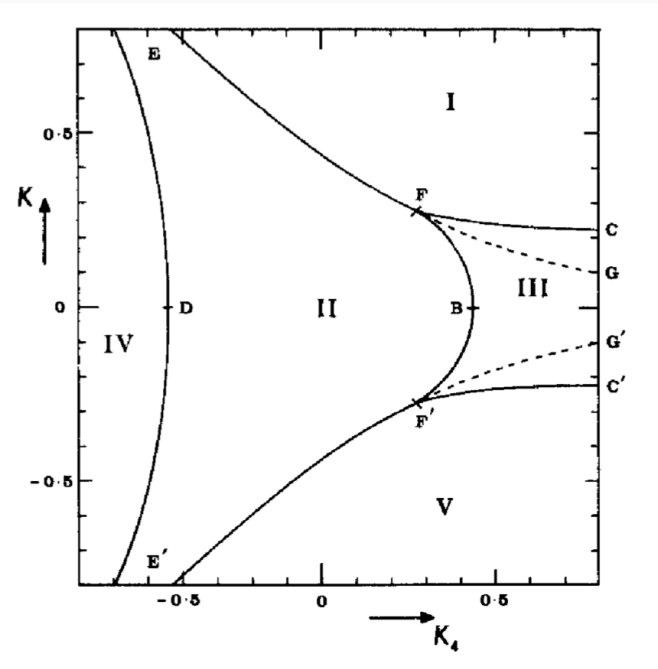


Ashkin-Teller model: an exactly solvable model with a composite order

The Ashkin-Teller model: a composite order parameter by construction.

$$H_{\mathrm{AT}} = -J\sum_{\langle ij\rangle}\sigma_i\sigma_j - J\sum_{\langle ij\rangle}\tau_i\tau_j - K\sum_{\langle ij\rangle}\sigma_i\sigma_j\tau_i\tau_j$$

The Ashkin-Teller model features a rich phase diagram, and remarkably in two dimensions can be studied analytically[1]. Here we consider the case of ferromagnetic couplings, $J, K \geq 0$, that already sports three different phases:



- An ordered phase, denoted by I, characterized by $\langle\sigma\rangle \neq 0 \neq \langle\tau\rangle$.
- A disordered phase, II, characterized by $\langle\sigma\rangle = \langle\tau\rangle = 0$.
- Remarkably, one also finds the peculiar phase III in which the single spins σ and τ are disordered, whereas a composite order parameter given by their product is ferromagnetically ordered, i.e. $\langle\sigma\tau\rangle \neq 0$.

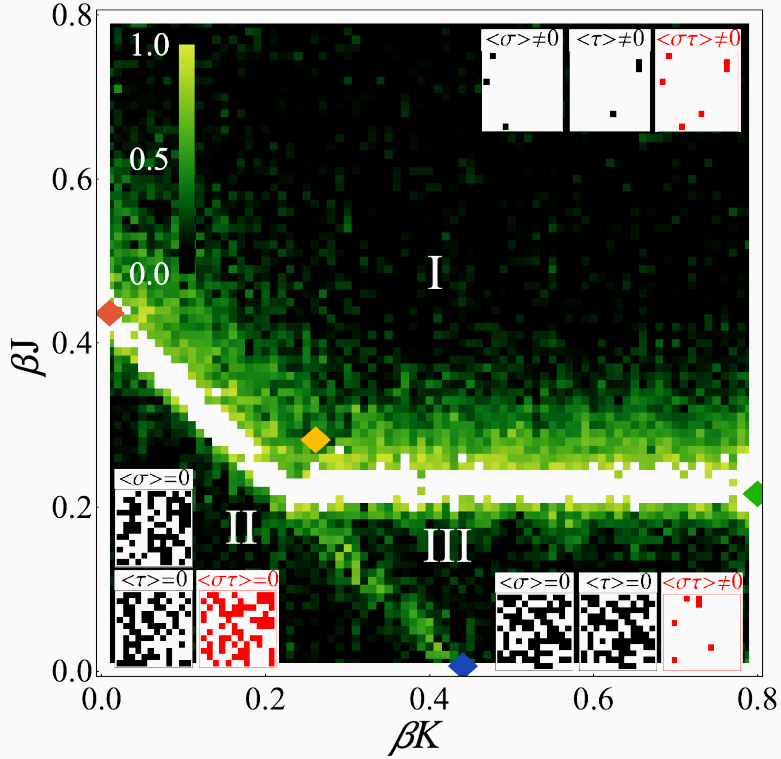
[1] R. J. Baxter, "Exactly Solved Models in Statistical Mechanics", Dover book on physics (Dover Publications, 2007).

Ashkin-Teller model: phase diagram

The previous investigation of Ising-like models makes us confident we can identify phase I and phase II, however it is not *a priori* clear that phase III can be correctly identified.

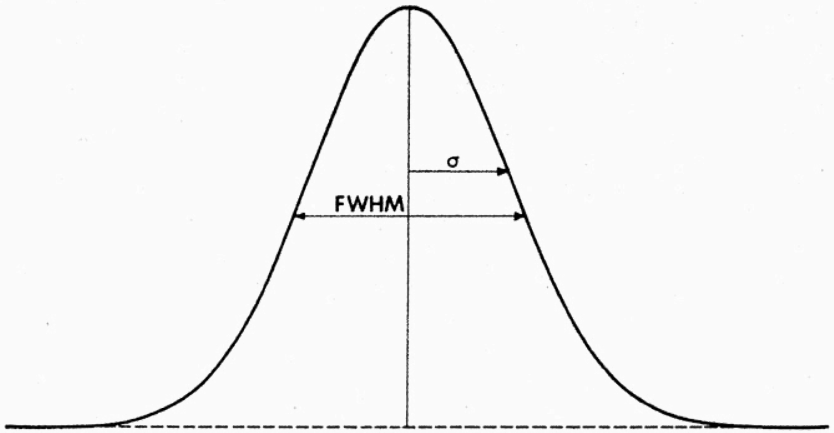
MC snapshots show disordered spins both in phase II and in phase III, the transition being determined by the $\sigma\tau$ composite variable, that we do not directly feed to the CNN. **In order to learn the existence of the II-III phase transition the CNN must learn to reconstruct the composite order parameter**.

We can compare the phase diagram we obtain with some exact results, marked by diamonds of different colors.



Robustness of the approach: finite size effects

We characterize each phase transition through the full width at half maximum of the peak in the reconstructed phase diagram (ML) or in the magnetic susceptibility (raw MC data). ML vs. standard approach.



$$\text{FWHM} = 2\sqrt{2\ln 2}\sigma \approx 2.35\sigma$$

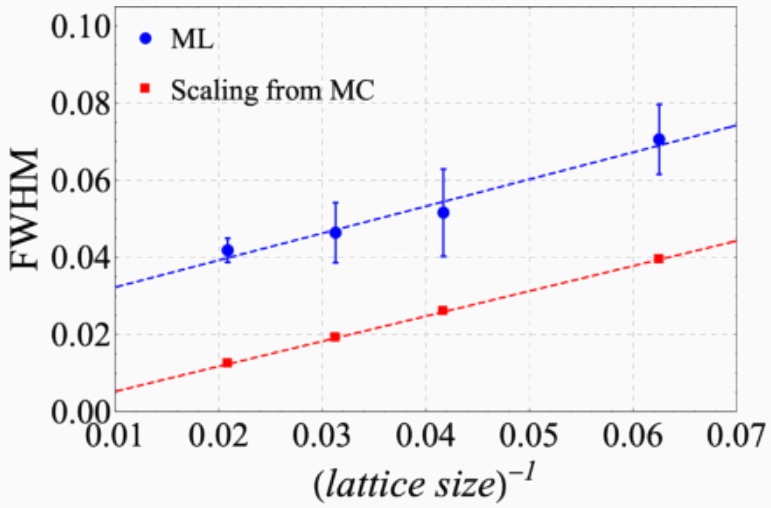


Image credit: W. R. Leo, *"Statistics and the treatment of experimental data"*, Springer-Verlag, 1992.

Robustness of the approach: signal to noise ratio

Our approach is able to obtain a phase diagram of quality high enough to visually identify different phases.

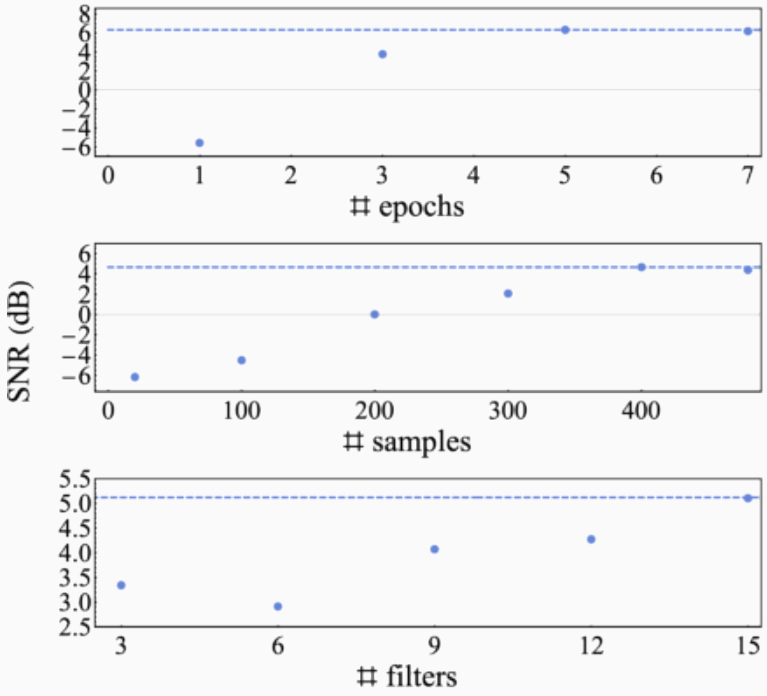
We now characterize our method by quantifying signal to noise ratio (SNR) and studying its behavior when essential (hyper)-parameters are changed. We define the SNR as

$$\text{SNR} \equiv \log_{10}\left(\frac{\frac{1}{N}\sum_i (x_i - \nu)^2}{\nu^2}\right),$$

x_i being the values of the $\nabla^2 u$ field, the summation extending over a region containing N values, ν being the 'noise', i.e. the average value of $\nabla^2 u$ in a subset of the region far away from a phase transition.

We consider:

- # of training epochs.
- # of samples.
- # of convolutional filter.



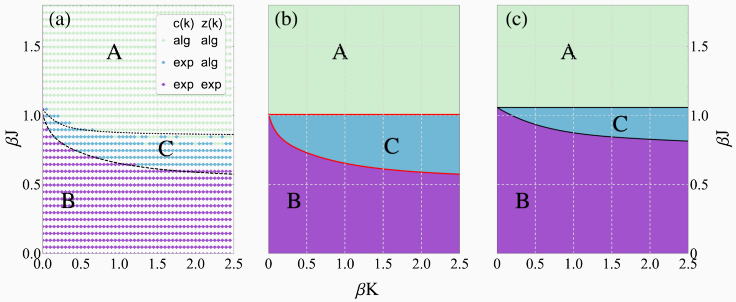
Outlook: beyond Ising-like models, other layered systems

Classical bilayer XY model: BKT-paired phase.

$$\mathcal{H} = -J\sum_{\langle ij\rangle}\cos\left(\phi_i-\phi_j\right) - J\sum_{\langle ij\rangle}\cos\left(\psi_i-\psi_j\right) - K\sum_{i}\cos\left(\phi_i-\psi_i\right)$$

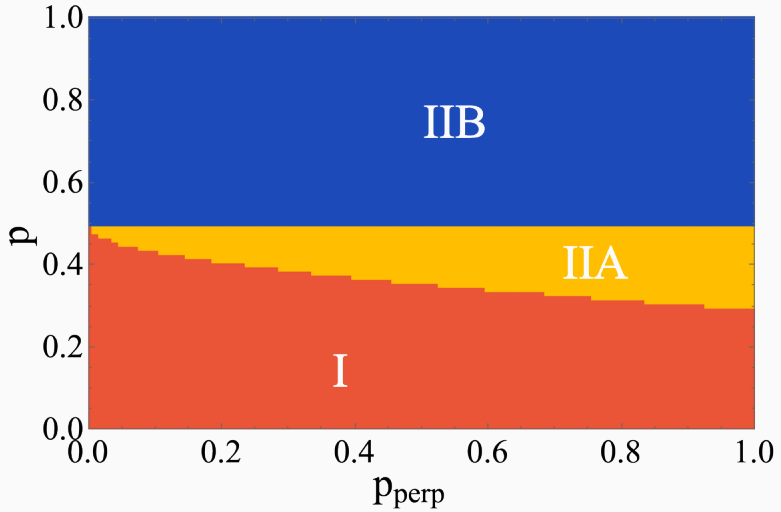
$$c_{\uparrow}(k) = \sum_{|i-j|=k}\exp(\mathrm{i}\phi_i - \mathrm{i}\phi_j)$$

$$z(k) = \sum_{|i-j|=k}\exp(\mathrm{i}\phi_i + \mathrm{i}\psi_i - \mathrm{i}\phi_j - \mathrm{i}\psi_j)$$



GB, N. Defenu, I. Nándori, L. Salasnich, A. Trombettoni, Phys. Rev. Lett. **123**, 100601 (2019).

Bond percolation on two-dimensional multilayers: p is the activation probability for intra-layer bonds, while p_{perp} is the activation probability for inter-layer bonds.



GB, A. Trombettoni, N. Defenu, *"Phase diagram of multilayer percolation"*, in preparation.

Conclusions and future perspectives

- Our work demonstrates that ML approaches are able to learn the order parameter driving a phase transition in layered models, also when this parameter is not immediately apparent from the snapshots without preprocessing.
- This is directly possible due to the convolutional filters which are, without any a priori knowledge, capable of learning even involved algebraic operations that uncover the order parameters from the data.
- This paves the way for the use of ML approaches to investigate the properties of systems of increasing complexity and to characterizing phases of matter described by multiple, possibly non-local order parameters.
- Our results pave the way for fully automated study of phase diagrams of more general and complicated spin systems.
- Open questions: explainable artificial intelligence (XAI), fidelity.

W. Rządkowski, N. Defenu, S. Chiacchiera, A. Trombettoni, GB, New J. Phys. **22**, 093026 (2020).

GB, N. Defenu, I. Nándori, L. Salasnich, A. Trombettoni, Phys. Rev. Lett. **123**, 100601 (2019).

GB, A. Trombettoni, N. Defenu, *"Phase diagram of multilayer percolation"*, in preparation.

Conclusions and future perspectives

- Our work demonstrates that ML approaches are able to learn the order parameter driving a phase transition in layered models, also when this parameter is not immediately apparent from the snapshots without pre
- Thi ıt any op
- Thi rties of s er des
- Ou mo
- Op

In collaboration with:

Nicolò Defenu – ETH, Zürich

Wojciech Rządkowski – Google, Zürich

Silvia Chiacchiera – Daresbury, UK

Andrea Trombettoni – Trieste

W. Rządkow

GB, N. Defe

GB, A. Trombettoni, N. Defenu, *"Phase diagram of multilayer percolation"*, in preparation.

Thank you for your attention.

STRUCTURES
CLUSTER OF
EXCELLENCE

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

This work is supported by the DFG (German Research Foundation) under Germany's Excellence Strategy — the Heidelberg STRUCTURES Excellence Cluster.

These slides at http://bigh.in

Backup slide # 1

Backup slide # 2

Backup slide # 3