Hopfield Network for Particle Recognition of the Pixel Detector at Belle II Experiment

JUSTUS-LIEBIG-UNIVERSITÄT GIESSEN

Irina Heinz

II. Physikalisches Institut

February 21, 2020

Contents

Physical Motivation

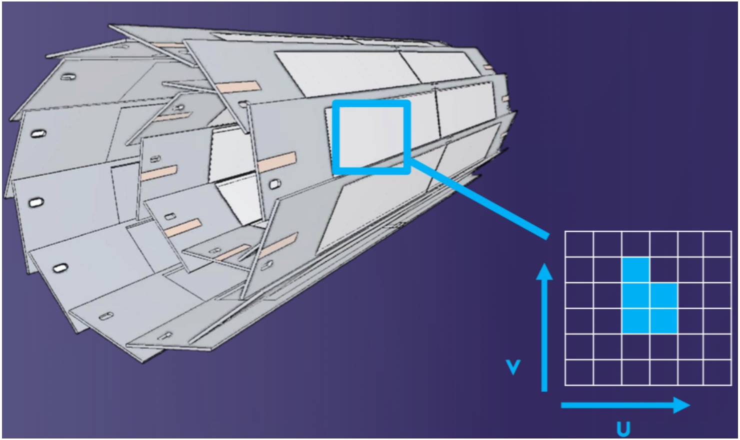


Figure: S. Käs, K. Dort, J. S. Lange: Poster on Multiparameter Analysis of the Belle II Pixeldetector's Data

Hopfield Network

- all N nodes connected to each other
- output = input layer
- bipolar {-1, 1} or binary {0, 1}
- connection between nodes defined by symmetric weigths (N^2), determined by saved states

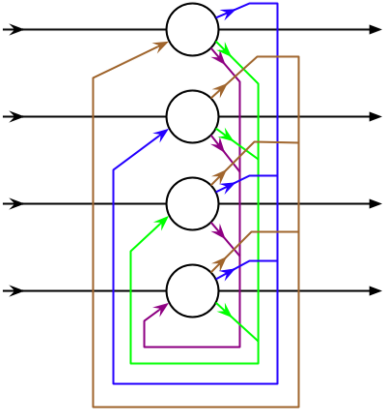
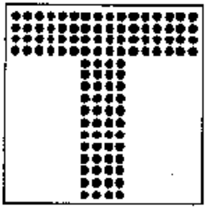


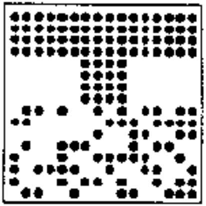
Figure: https://de.wikipedia.org/wiki/Hopfield-Netz#/media/Datei:Hopfield-net-vector.svg

Hopfield Network

- It can be proven that the system reaches a stable end state
- 3 types of end state:
  - saved state
  - reversed saved state
  - spurious state



Original ‘T’



half of image corrupted by noise

Figure: http://web.cs.ucla.edu/ rosen/161/notes/hopfield.html

Hebb's Learning Rule

If p is the number of pattern to store:

- bipolar: $x_i, x_j \in \{-1, 1\}$

$$w_{ij} = \sum_{k=1}^{p} x_i^k x_j^k \quad (1)$$

- binary: $x_i, x_j \in \{0, 1\}$

$$w_{ij} = \sum_{k=1}^{p} (2x_i^k - 1)(2x_j^k - 1) \quad (2)$$

- $\forall\ i = j \Rightarrow w_{ij} = 0$

Storkey's Learning Rule

- recursive definition of weight matrix
- bipolar: $x_i, x_j \in \{-1, 1\}$

$$w_{ij}^{k} = w_{ij}^{k-1} + x_i^k x_j^k - x_i^k h_{ji}^k - x_j^k h_{ij}^k \tag{3}$$

where the local field is defined as

$$h_{ij}^{k} = \sum_{m=1:m\neq j}^{N} w_{im}^{k-1} x_m^k \tag{4}$$

- $\forall\ i = j \Rightarrow w_{ij} = 0$

Updating Nodes

- synchronous and asynchronous updating of nodes
- for each node i:
  1. calculate weighted sum (bipolar):

$$V_i = \sum_{j=1}^{N} w_{ij} x_i x_j \qquad (5)$$

1	1	1
0	1	0
0	1	0

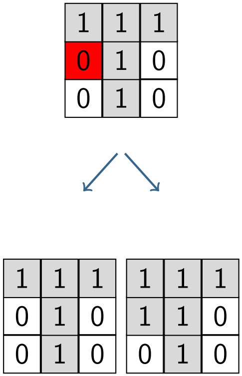
Updating Nodes

- synchronous and asynchronous updating of nodes
- for each node i:
  1. calculate weighted sum (bipolar):

$$V_i = \sum_{j=1}^{N} w_{ij} x_i x_j \qquad (5)$$

  2. compare to given threshold ε (usually 0):

$$V_i > \varepsilon \Rightarrow x_i = 1$$
$$V_i < \varepsilon \Rightarrow x_i = -1 \ \text{(binary: 0)}$$



Energy

- analogy in physics: Ising model (spin interaction in a lattice)

$$E = -\sum_{\langle i,j \rangle}^{N} J_{ij} s_i s_j - \mu \sum_{i=1}^{N} h_i s_i \qquad (6)$$

J: spin-spin interaction, s_i: z-component of spin,

μ: atomic magnetic moment, h_i: external magnetic field

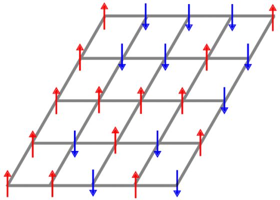


Figure 1: https://www.researchgate.net/figure/Schematic-representation-of-a-configuration-of-the-2D-Ising-model-on-a-square-lattice_fig2_321920877

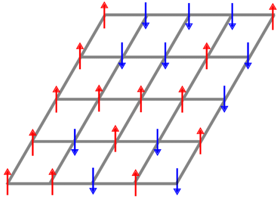
Energy

- analogy in physics: Ising model (spin interaction in a lattice)

$$E = -\sum_{\langle i,j \rangle}^{N} J_{ij} s_i s_j - \mu \sum_{i=1}^{N} h_i s_i \quad (6)$$

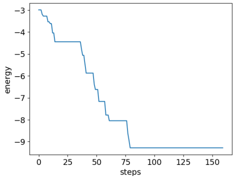
J: spin-spin interaction, s_i: z-component of spin,

μ: atomic magnetic moment, h_i: external magnetic field



- define the energy function:

$$E = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} w_{ij} x_i x_j + \sum_{i=1}^{N} \theta_i x_i \quad (7)$$

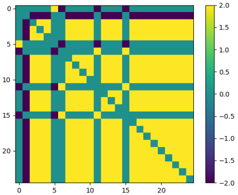


- every bit flip is due to energy minimization (local minimum)

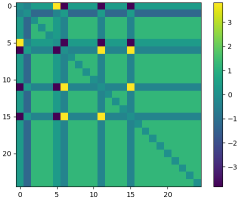
Figure 1: https://www.researchgate.net/figure/Schematic-representation-of-a-configuration-of-the-2D-Ising-model-on-a-square-lattice_fig2_321920877

Capacity of a Hopfield Network

- weight matrix after storing patterns

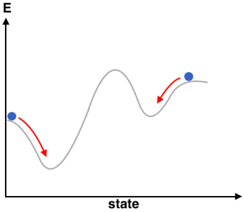


(a) Hebb learning



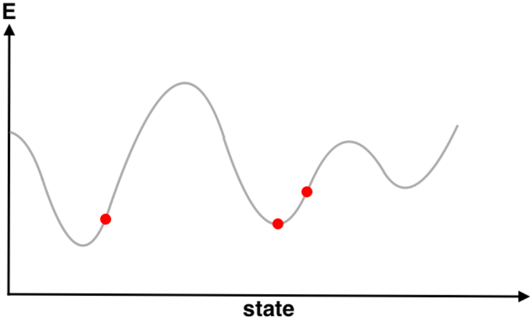
(b) Storkey learning

- pattern recognition = finding local minimum
- storage capacity depends on weight matrix i.e. stored pattern
- rule of thumb: $p/N < 0.14$ (for Hebbian learning)



"Post Training"

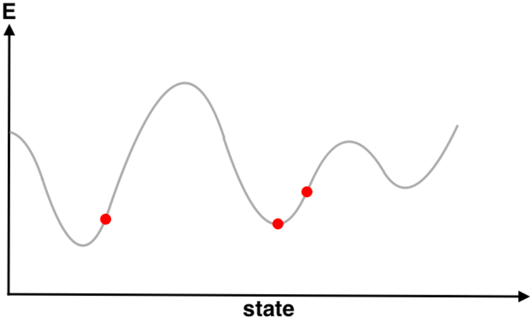
- complex trained network with several "real" and spurious minima



"Post Training"

- complex trained network with several "real" and spurious minima
- emphasize target pattern:

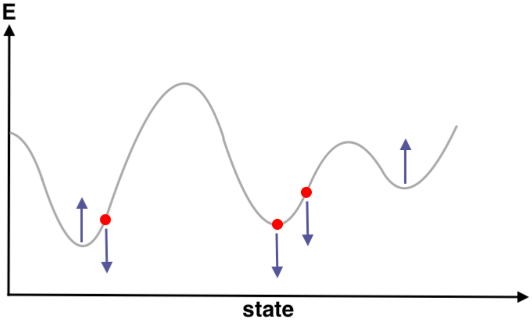
$$W = W + \mu \left(\sum_{x \in X_p} x_i x_j - \sum_{x \notin X_p} x_i x_j \right) \tag{8}$$



"Post Training"

- lift only spurious states X_s, such that:

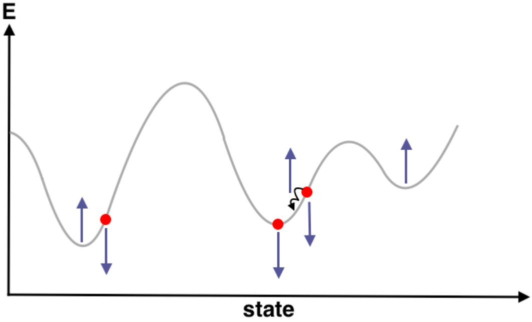
$$W = W + \mu \left(\sum_{x \in X_p} x_i x_j - \sum_{x \in X_s} x_i x_j \right) \tag{9}$$



"Post Training"

- start at target pattern and calculate a few steps
- if not end state $\Rightarrow$ lift energy at this state ($\in X_{fs}$)

$$W = W + \mu \left(\sum_{x \in X_p} x_i x_j - \sum_{x \in X_{fs}} x_i x_j \right) \tag{10}$$



Activation Function

- How can we determine the probability of bit flipping?
- How can we put additional information in our network?

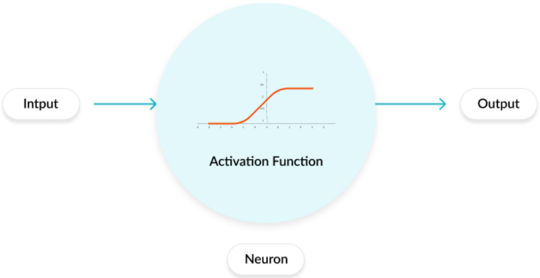


Figure: https://missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right/

PXD Data

- 5 $\times$ 5 matrices of signal pixels containing charge information
- absolut position of pattern in detector

0	0	0	0	0
0	149	30	0	0
0	67	7	0	0
0	10	0	0	0
0	0	0	0	0

→

0	0	0	0	0
0	1	1	0	0
0	1	1	0	0
0	1	0	0	0
0	0	0	0	0

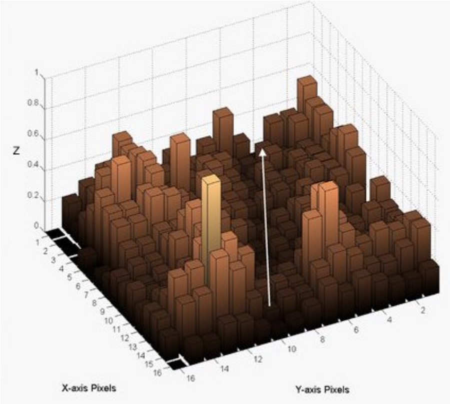
Data Properties

- data properties are:
  - total number of signal pixels
  - x length, y length
  - total charge
  - maximal charge
  - position in detector / angle

0	0	0	0	0
0	149	30	0	0
0	67	7	0	0
0	10	0	0	0
0	0	0	0	0

- idea: feed network with information via activation function

3D Pattern



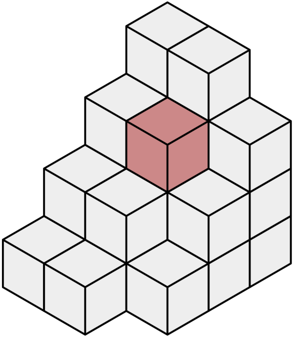


Figure 1: https://www.researchgate.net/figure/Left-2D-and-Right-3D-pixel-intensity-map-for-the-ERD2004-CZT-detector-eV2-as-found_fig80_236194566

Figure 2: https://en.wikipedia.org/wiki/Voxel

Analysis of PXD Data

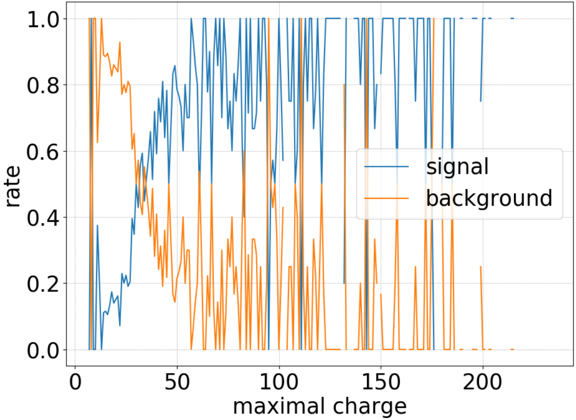


Figure: Probability for signal of given maximal charged pixel

Analysis of PXD Data

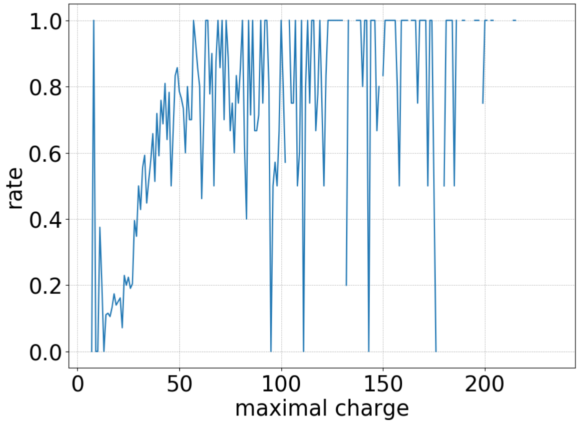


Figure: Probability for signal of given maximal charged pixel

Analysis of PXD Data

$$P(q_{max}) = \frac{1}{2}\left(1 - \frac{1}{1+\exp(-a \cdot q_{max} + b)}\right)$$

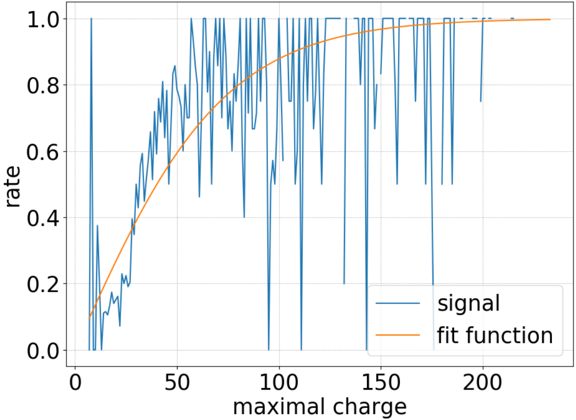
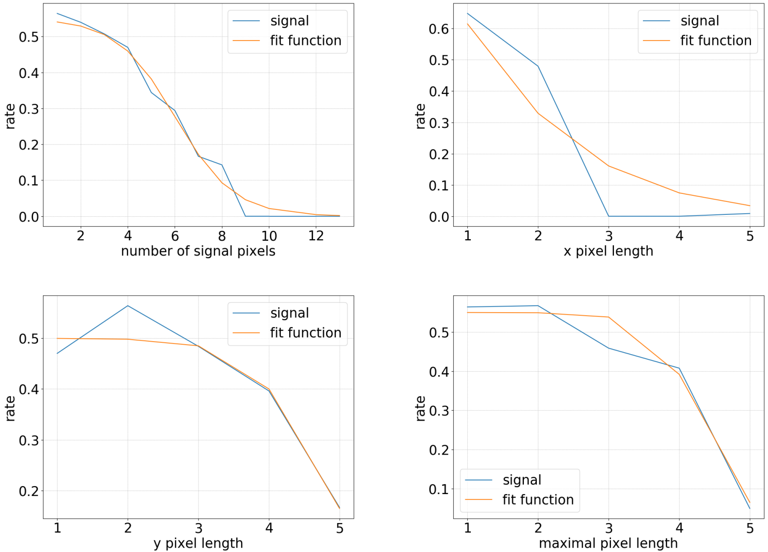


Figure: Probability for signal of given maximal charged pixel

Analysis of PXD Data



Analysis of PXD Data

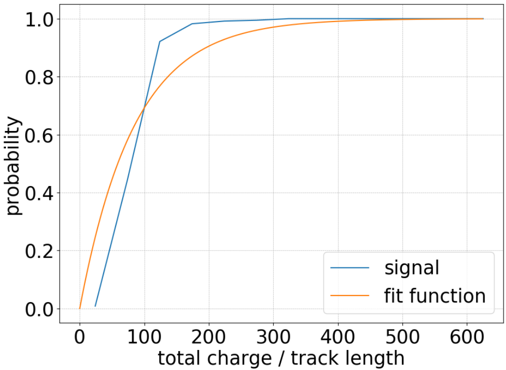
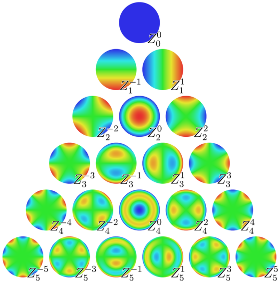


Figure: Probability for signal of given realtion between total charge and track length

Zernike Moments

- Zernike polynomials Z_{nm} form a complete orthogonal set of functions on the unit disk
- Zernike moment of order n and repetition m:

$$A_{nm} = \frac{n+1}{\pi} \sum_{x} \sum_{y} f(x,y) Z^{*}_{nm} \tag{11}$$



where $n - |m|$ is even,

$$Z_{nm}(x,y) = Z_{nm}(\rho,\theta) = R_{nm}(\rho)e^{im\theta}$$

and R_{nm} describes radial part

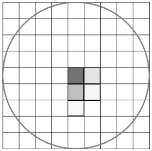


Figure: https://en.wikipedia.org/wiki/Zernike_polynomials#/media/File:Zernike_polynomials2.png

Activation Function

- How can we use activatoin function f for our problem?

$$\begin{pmatrix} w_{11}^{signal} & \cdots & w_{n1}^{signal} \\ \vdots & \ddots & \vdots \\ w_{1n}^{signal} & \cdots & w_{nn}^{signal} \end{pmatrix} \qquad \begin{pmatrix} w_{11}^{background} & \cdots & w_{n1}^{background} \\ \vdots & \ddots & \vdots \\ w_{1n}^{background} & \cdots & w_{nn}^{background} \end{pmatrix}$$

$$f \searrow \quad \swarrow 1-f$$

$$\begin{pmatrix} w_{11} & \cdots & w_{n1} \\ \vdots & \ddots & \vdots \\ w_{1n} & \cdots & w_{nn} \end{pmatrix}$$

Performance

- tests with only simulation data (50:50)

3D	local prop.	global prop.	Zernike mom.	accuracy
✓	×	×	×	63.0 %
✓	✓	×	×	76.5 %
✓	✓	✓	×	95.7 %
✓	✓	✓	✓	97.7 %

- real background data: 95.6 %
- *Remark:* Storkey rule and "post training" didn't make an essential improvement in our case (also pattern dependent)

Performance

- find comparable quantities:
  efficiency:

$$\nu_{signal} = \frac{\text{number of correctly identified signals}}{\text{total number of signals}}$$

  background rejection:

$$\nu_{background} = \frac{\text{number of correctly identified background}}{\text{total number of background}}$$

- simulation data:

$$\nu_{signal} = 96.83\,\%$$
$$\nu_{background} = 98.49\,\%$$

- real background data:

$$\nu_{background} = 95.60\,\%$$

Conclusion and Outlook

- Hopfield network works for binary data
- capacity depends on stored pattern and pattern size
- optimization by using activation function (data properties and Zernike moments)
- for our data: combination of neural network and data analysis
- Next step:
  - analyse real data
  - try higher Zernike moments
  - variate parameters in activation function
  - try 3-dimensional Storkey learning and "post training"

Thank You for Your attention!

http://www.vias.org/science_cartoons/neural_network.html