

# Neue Einsatzmöglichkeit von Hardwarebeschleunigern für nachhaltigere KI-Modelle: Entwicklung und Evaluation der Boltzmann Maschinen auf einem physikinspirierten Hardwarebeschleuniger

Bachelorarbeit

submitted on March 6, 2024

Fakultät Wirtschaft und Gesundheit

Studiengang Wirtschaftsinformatik

Kurs WWI2021F

von

SIMON SPITZER

Betreuer in der Ausbildungsstätte:

DHBW Stuttgart:

⟨ Hewlett Packard GmbH ⟩  
⟨ Dr. Fabian Böhm ⟩  
⟨ Research Scientist at Hewlett Packard Labs ⟩

⟨ Prof. Dr., Kai Holzweißig ⟩  
⟨ der/des wissenschaftlichen Betreuerin/Prüferin ⟩

Unterschrift der Betreuerin/des Betreuers

# Contents

<b>List of abbreviations</b>	<b>IV</b>
<b>List of figures</b>	<b>V</b>
<b>List of tables</b>	<b>VI</b>
<b>1 Einleitung</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Problem statement . . . . .	1
1.3 Objective . . . . .	2
1.4 Research method . . . . .	3
1.5 Aufbau der Arbeit . . . . .	4
<b>2 Aktueller Stand der Forschung und Praxis</b>	<b>5</b>
2.1 Ressourcenverbrauch bei KI-Modellen . . . . .	5
2.1.1 Ressourcenverbrauch bei KI-Modellen . . . . .	5
2.2 Neural Networks - Boltzmann Machines . . . . .	5
2.2.1 Energy-based models . . . . .	6
2.2.2 concept of Boltzmann Maschines . . . . .	8
2.2.3 Training of Restriced Boltzmann Machines . . . . .	10
2.2.4 Current Problems witht BMs and RBMs . . . . .	14
2.3 Hardwarebeschleuniger . . . . .	15
2.3.1 Aktuelle Ansätze im Bereich KI und weitere Lösungen . . . . .	15
2.3.2 ISING Maschine/ Physikinspirierter Hardwarebeschleuniger . . . . .	15
2.4 Memristor Hopfield Network . . . . .	15
2.4.1 Memristor . . . . .	15
2.4.2 Hopfield Network . . . . .	15
2.4.3 Crossbar . . . . .	17
2.4.4 Output Hopfield Network . . . . .	17
2.4.5 Noisy HNN . . . . .	17
<b>3 Zielspezifikation und Darlegung der Forschungsmethodik</b>	<b>18</b>
3.1 Zielspezifikation (genauer als in Einleitung, Metriken erwähnen, Erfolg meiner Methode bewerten, Welcher Teil der Forschungsfrage wird beantwortet?) . . . . .	18
3.2 Design Science Research . . . . .	18
3.3 Zielsetzung(ohne gneaue Metriken nennen, generell halten) . . . . .	18
3.4 Laborexperiment für die Umsetzung . . . . .	18
<b>4 Implementierung/Laborexperiment der Simulator Pipeline</b>	<b>19</b>
4.1 Zielsetzung und Forschungsmethodik . . . . .	19
4.2 Aufbau der Simulator Pipeline . . . . .	19
4.3 KI-Bibliothek Scikit-Learn . . . . .	19
<b>5 Evaluation der BM auf dem physikinspiriertem Hardwarebeschleuniger</b>	<b>20</b>
5.1 Zielsetzung und Forschungsmethodik . . . . .	20
5.1.1 Prediction Accuracy . . . . .	20

5.1.2	Troughput (Samples/Sec) . . . . .	20
5.1.3	Energieverbrauch (Energy/Operation) . . . . .	20
5.2	Vergleichen mit anderen Hardwarebeschleuniger, FPGA, GPU oder CPU aus der Literatur . . . . .	20
<b>6</b>	<b>Kritische Reflexion und Ausblick</b>	<b>21</b>
6.1	Evaluation der Erkenntnisse in Bezug auf die Zielsetzung der Arbeit . . . . .	21
6.2	Kritische Reflexion der Ergebnisse und Methodik . . . . .	21
6.3	Zielsetzung(ohne gneaue Metriken nennen, generell halten) . . . . .	21
6.4	Ergebnisextraction für Theorie und Praxis (evtl. mit 6.4 Zusammenlegen) . . . . .	21
6.5	Ausblick . . . . .	21
	<b>Appendix</b>	<b>22</b>
	<b>List of references</b>	<b>24</b>

# List of abbreviations

Ein Abkürzungsverzeichnis ist optional. Das Paket `acronym` kann weit mehr, als hier gezeigt.<sup>1</sup> Beachten Sie allerdings, dass Sie die Einträge selbst in sortierter Reihenfolge angeben müssen.

<b>BM</b>	Boltzmann Maschine
<b>RBM</b>	Restriced Boltzmann Maschine
<b>DNN</b>	Deep Neural Network
<b>EBM</b>	Energy Based Model
<b>MCMC</b>	Markov chain Monte Carlo
<b>DNN</b>	Deep Neural Networks

**Ergänzende Bemerkung:** Eine im Text verwendete Abkürzung sollte bei ihrer ersten Verwendung erklärt werden. Falls Sie sich nicht selbst darum kümmern möchten, kann das das Paket `acronym` übernehmen und auch automatisch Links zum Abkürzungsverzeichnis hinzufügen. Dazu ist an allen Stellen, an denen die Abkürzung vorkommt, `\ac{ITIL}` zu schreiben.

Das Ergebnis sieht wie folgt aus:

- erstmalige Verwendung von `\ac{ITIL}` ergibt: **ITIL!** (**ITIL!**),
- weitere Verwendung von `\ac{ITIL}` ergibt: **ITIL!**

Wo benötigt, kann man mit dem Befehl `\acl{ITIL}` wieder die Langfassung ausgeben lassen: **ITIL!**.

Falls man die Abkürzungen durchgängig so handhabt, kann man durch Paket-Optionen (in `_dhbw_preamble.tex`) erreichen, dass im Abkürzungsverzeichnis nur die tatsächlich verwendeten Quellen aufgeführt werden (Option: `printonlyused`) und zu jedem Eintrag die Seite der ersten Verwendung angegeben wird (Option: `withpage`).

---

<sup>1</sup>siehe <http://ctan.org/pkg/acronym>

## List of Figures

1	Figure of a simplified energy landscape . . . . .	7
2	figure of a general Boltzmann Machine . . . . .	8
3	Figure of a Restriced Boltzmann Maschine (RBM) . . . . .	10
4	Figure of a logistic sigmoid function RBM . . . . .	11
5	Figure of the exponential flip probability function . . . . .	13
6	Figure of a hopfield network . . . . .	16
7	Mal wieder das DHBW-Logo. . . . .	23

## List of Tables

# 1 Einleitung

## 1.1 Motivation

In the research and development of generative AI models, the computing speed and energy efficiency are increasingly becoming the focus<sup>2</sup> The authors of Open AI confirm, that the growth rate of machine learning models surpassed the growth of efficiency within computerchips. The required computing power of the models double each 3-4 months but the power of computerchips, after Moore's Law double only every 2 years.<sup>3</sup> Focusing on current problems like rising energy consumption of datacenters and the associated greenhouse gas emissions, the search for more efficient solutions is essential for the future. Worldwide energy consumption of datacenters increase yearly around 20-40%, which means that in 2022 about 1,3% of the total global energy consumption and about 1% of energy-related global greenhouse gas emissions was caused by them.<sup>4</sup> However, it is not clear how large the AI share contributed to the total numbers.

## 1.2 Problem statement

One approach that is already well known is the use of AI accelerators based on ASICs (application-specific integrated circuits) - i.e. circuits that are used application specific, such as Google TPUs (Tensor Processing Unit).<sup>5</sup> This is useful because the usage of multimodels for discriminating tasks compared to task specific models are more energy intense.<sup>6</sup> One promising alternative concept in research is the usage on physics-inspired hardware accelerators, that are primarily used for optimization problems because of their ability to solve problems faster and more efficient than GPUs.<sup>7</sup> A scalable physics-inspired hardware accelerator (also called Ising-machine), that surpasses the power of existing standard digital computers, could have a large influence on practical applications for a variety of optimization problems.<sup>8</sup>

Such physics-inspired hardware accelerator offer, due to their special calculation method, potential for efficient processing of computationally intensive tasks. Specifically, the acceleration in contrast to digital computers is achieved by calculating the computationally intense tasks with analog signals. On top of that the implementation on dedicated hardware offers the possibility to exploit the parallelization of digital hardware accelerators and analog computation.<sup>9</sup>

---

<sup>2</sup>Vgl. Luccioni/Jernite/Strubell 2023, p. 1

<sup>3</sup>Vgl. Dario Amodei/Danny Hernandez 2024, p. 1

<sup>4</sup>Vgl. Hintemann/Hinterholzer 2022, p. 1

<sup>5</sup>Vgl. Wittpahl 2019, p. 39

<sup>6</sup>Vgl. Luccioni/Jernite/Strubell 2023, p. 5

<sup>7</sup>Vgl. Mohseni/McMahon/Byrnes 2022, p. 1

<sup>8</sup>Vgl. Mohseni/McMahon/Byrnes 2022, p. 1

<sup>9</sup>Vgl. Mohseni/McMahon/Byrnes 2022, p. 4

Interesting enough, despite their different applications, the energy function of the hardware accelerator that is used in Ising-machines shows big parallels to those used in Boltzmann Maschine (BM), therefore it can be suggested that Ising-machines could work well for AI.<sup>10</sup> Ising-machines strive to minimize their energy, which is defined by the pairwise interaction of binary variables (Spins).<sup>11</sup> In contrast, BMs are energy-based neuronal networks that are used for classification tasks by allocating a scalar energy for each configuration of variables. Minimizing the total network energy is therefore equal with the solution of a optimization problem.<sup>12</sup> Current problems with BMs are the high complexity and high requirements for the all-to-all communication between the processing units, which causes the implementation on conventional digital computers to be inefficient, but also an inherently slow convergence in certain processes such as simulated annealing.<sup>13</sup> These challenges complicate the training and the usage of BMs, especially for large data volumes and complex optimization tasks.<sup>14</sup> Nevertheless the similarities of both models implicate, that Ising-machines could be able to execute this specific AI-model with higher energy efficiency and with higher computing speed. Currently there are only few concepts that exist on how to achieve a implementation of a BM within on a Ising-machine. The paper of the authors Mahdi, Nazm, Bojnordi and Ipek is a promising approach, However it could not be shown how a implementation on a real accelerator chip could function.

With the given background, the following central research question for this thesis arises:

1. Can Boltzmann Machines be efficiently implemented on physics-inspired Hardware accelerators by analog noise injection?
  - What is the accuracy of the AI-model on the hardware accelerator?
    - Metric: Prediction accuracy and negative Likelihood
  - Comparison between other hardware accelerators, FPGA, GPU, or CPU within the literature in terms of energy efficiency and computing speed.
    - Metrics: Throughput (Samples/Sec), Energy usage (Energy/Operation)

It is therefore necessary to test whether this generative AI model is compatible with Ising machines and whether this solution is efficient or not.

### 1.3 Objective

The primary objective of this bachelor thesis is the research and extension of a existing physics-inspired hardware accelerator (Ising machine) for the implementation and evaluation of BMs as an energy based AI-model. The aim is to answer the posed research question. In addition to

---

<sup>10</sup>Vgl. Cai et al. 2019, p. 10

<sup>11</sup>Vgl. Wang/Roychowdhury 2017, p. 1

<sup>12</sup>Vgl. Nazm Bojnordi/Ipek 2016, p. 2

<sup>13</sup>Vgl. Nazm Bojnordi/Ipek 2016, p. 1

<sup>14</sup>Vgl. Nazm Bojnordi/Ipek 2016, p. 2



that, it would be beneficial if rules for the influence of hyperparameters could be established since there is no data available for this new method.

To initially accomplish this objective it is necessary to establish a simulator pipeline to the hardware accelerator that translates BM on top of it. The simulator pipeline consists of an existing machine learning library and an existing hardware accelerator that are connected to each other. With the simulator pipeline it needs to be shown that it is possible for the hardware accelerator to realize BMs.

Within the simulator pipeline, the activation probabilities of individual neurons are measured on the simulated hardware. If this process proves successful, it is then expanded to simulate a complete neuronal network. The final step is that the hardware accelerator can be used for training and inference and is comparable to conventional machine learning libraries. This phase includes a careful adjustment and possibly extension of the existing accelerator to be compliant with the specific requirements of BMs.

If the simulator pipeline can be validated a workload consisting of a standard data set to recognize handwritten digits will be tested. The prediction accuracy, throughput (samples/sec) and the energy consumption (energy/operation) of the Boltzmann machines on the Ising hardware accelerator will be investigated as metrics in order to thereby answer the second part of the posed research questions.

## 1.4 Research method

Design Science Research

1. **Problemorientierung:** DSR fokussiert auf die Lösung praktischer Probleme, wie die Forschung zur Steigerung der Effizienz und Rechengeschwindigkeit in KI-Modellen.
2. **Artefakt Entwicklung:** Zentral in DSR ist die Entwicklung innovativer Artefakte. Die Arbeit zielt darauf ab, ein solches Artefakt in Form des physikinspirierten Hardwarebeschleunigers weiterzuentwickeln und für KI-Modelle einzusetzen.
3. **Iterative Evaluation:** Durch die iterative Vorgehensweise in DSR kann die Ausarbeitung der Lösung fortlaufend verbessert und angepasst werden, was für die Entwicklung und Optimierung von KI-Systemen entscheidend ist (ebenfalls das Konzept).
4. **Beitrag zur Wissensbasis und Praxisrelevanz:** DSR unterstützt die Generierung neuer Erkenntnisse und stellt sicher, dass Forschungsergebnisse sowohl theoretisch fundiert als auch praktisch anwendbar sind, was mit den Zielen Ihres Projekts im Einklang steht. Untermethodik könnte hierbei eine Simulation sein. Variabel, je nach Verlauf der Forschung.

## **1.5 Aufbau der Arbeit**

## 2 Aktueller Stand der Forschung und Praxis

### 2.1 Ressourcenverbrauch bei KI-Modellen

#### 2.1.1 Ressourcenverbrauch bei KI-Modellen

substantial challenges in high consumption of computational, memory, energy, and financial resources, especially in environments with limited resource capabilities<sup>15</sup>

**Nachhaltigkeit**

**Stromverbrauch**

**Rechenleistung begrenzt, KI-Modelle wachsen schneller als verfügbare Leistung**

### 2.2 Neural Networks - Boltzmann Machines

In recent years, artificial neural networks have first revolutionised computer vision but also other fields, such as natural language processing, controlling and planning (playing games: e.g., Atari and Go), and navigational tasks (finding the shortest path on a map).<sup>16</sup> Nowadays, neural networks have reached nearly every field of science and are a crucial part of various real world applications.<sup>17</sup> Particularly in the last two years, artificial intelligence has also garnered widespread interest from the public, especially regarding chatbots like ChatGPT and Google Bard.<sup>18</sup> The most important feature of a neural network-based system that are inspired by our brain, is that they can learn and adapt to data.<sup>19</sup>

Internally, neural networks are computational models that consist of many simple processing units, called neurons that work together in parallel within interconnected layers.<sup>20</sup> They consist out of a network architecture, which describes the layout and how the neurons are wired. Secondly, they have a optimization function which specifies the goals pursued in the learning process.<sup>21</sup> Lastly, there is a training algorithm that varies all of the hyperparameters, like connection strengths between neurons, training iterations, the learning rate, etc..<sup>22</sup> When these interconnected layers are stacked on top of each other the network is called deep.<sup>23</sup> Currently,

---

<sup>15</sup>Vgl. Bai et al. 2024, pp. 1–2

<sup>16</sup>Vgl. Cichy/Kaiser 2019, p. 305

<sup>17</sup>Vgl. Gawlikowski et al. 2023, p. 1513

<sup>18</sup>Vgl. Singh/Kumar/Mehra 2023, pp. 1–2

<sup>19</sup>Vgl. Cichy/Kaiser 2019, p. 305

<sup>20</sup>Vgl. Cichy/Kaiser 2019, p. 305

<sup>21</sup>Vgl. Durstewitz/Koppe/Meyer-Lindenberg 2019, p. 1583

<sup>22</sup>Vgl. Durstewitz/Koppe/Meyer-Lindenberg 2019, p. 1583

<sup>23</sup>Vgl. Cichy/Kaiser 2019, p. 305

Deep Neural Networks (DNN)s can have up to 1200 interconnected layers that equal to more than 16 million neurons inside the network.<sup>24</sup> In general, deep learning methods can be seen as subset of machine learning methods and are today's fundament of artificial intelligence.<sup>25</sup> For example some regression tasks within computer vision in DNN include object detection, medical image registration, head- and body-pose estimation, age estimation and visual tracking.<sup>26</sup> Such models can make use of the availability of the recent results in the field of neural networks and deep learning, leading to highly performing, yet very large neural networks with millions of parameters.<sup>27</sup> As a result, such models often have a negative effect on the environment in terms of unnecessary energy consumption and a limitation to their deployment on low-resource devices because they are excessively oversized and redundant.<sup>28</sup>

### 2.2.1 Energy-based models

An Energy Based Model (EBM) is a type of statistical model where the likelihood of a particular state is determined by an energy function.<sup>29</sup> Since 1982, those statistical neural network models have been continuously emerging in the machine learning field when J.J. Hopfield introduced the Hopfield Network.<sup>30</sup> Current developments include their use in reinforcement learning, potential replacements for discriminators in generative adversarial networks and for quantum EBMs.<sup>31</sup> In addition to that, Open AI showed that EBMs are useful models across a wide variety of tasks like achieving state-of-the-art out-of-distribution classification and continual online class learning to name a few.<sup>32</sup> The underlying idea behind EBMs is to establish a probabilistic physical system that is able to learn and memorize patterns but most importantly generalize it.<sup>33</sup> Especially, it involves learning an energy function  $E_{\theta}(x) \in \mathbb{R}$  and assigning the low energy to observed data  $x_i$  and high energy to other values  $x$ .<sup>34</sup>

---

<sup>24</sup>Vgl. Mall et al. 2023, p. 2

<sup>25</sup>Vgl. Durstewitz/Koppe/Meyer-Lindenberg 2019, p. 1583

<sup>26</sup>Vgl. Gustafsson et al. 2020, pp. 325–326

<sup>27</sup>Vgl. Marinó et al. 2023, p. 152

<sup>28</sup>Vgl. Marinó et al. 2023, p. 152

<sup>29</sup>Vgl. Huembeli et al. 2022, p. 2

<sup>30</sup>Vgl. Hopfield 1982

<sup>31</sup>Vgl. Verdon et al. 2019, p. 1; Vgl. Du/Lin/Mordatch 2021, p. 1

<sup>32</sup>Vgl. Du/Mordatch 2020, pp. 1–2

<sup>33</sup>Vgl. Huembeli et al. 2022, p. 2

<sup>34</sup>Vgl. Gustafsson et al. 2020, p. 330

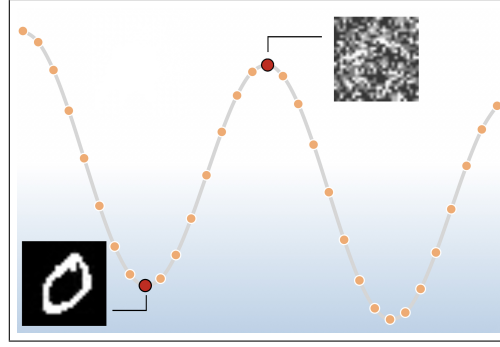


Abb. 1: Figure of a simplified energy landscape

In this figure 1 a simplified energy landscape is shown where the local minima correspond to states that encode an MNIST digit.<sup>35</sup> It is visible that observed data settles in the local minimum of the energy landscape, in this case a clear 0. On the other hand close to the local maxima of the energy landscape the 0 is only barely recognizable and therefore got a higher energy value assigned to it. The assumption of the underlying distribution function  $P(x)$  is equal to the solution of the optimization problem:

$$P(x) = \frac{1}{Z} \exp\left(-\frac{E(x)}{T}\right), \quad (2.1)$$

where  $Z$  is given by the partition function to ensure that the density function normalizes to a total probability of 1 and  $T$  is interpreted as the temperature.<sup>36</sup> As a result the behavior of a EBM is determined by 2.1. The aim of the training is to match the real data  $P_{\text{data}}$  as closely as possible with the internal model  $P_{\text{model}}$ . A practical method to achieve this goal is to use the KL divergence. KL divergence is a mathematical equation that helps to measure how close the predictions are by comparing the model's learned distribution to the true distribution of the data:

$$G = \sum_x P^+(x) \ln\left(\frac{P^+(x)}{P^-(x)}\right) \quad (2.2)$$

Here,  $P^+(x)$  is the probability when the states are determined by a data input from the environment, while  $P^-(x)$  represents the internal network running freely, also referred to as “dreaming”.<sup>37</sup> To optimise the KL divergence, in this case  $G$ , the energy is adjusted, whereby data is assigned to low energy states (according to 2.1) and the training data receives high energy and therefore high probabilities.<sup>38</sup> To complete the section the “partition function”,  $Z$ , used in 2.1 is given by summing over all possible pairs of visible and hidden vectors:

$$Z = \sum_x \exp\left(-\frac{E(x)}{T}\right) \quad (2.3)$$

---

<sup>35</sup>Vgl. Huembeli et al. 2022, p. 6

<sup>36</sup>Vgl. Huembeli et al. 2022, pp. 2–3

<sup>37</sup>Vgl. Ackley/Hinton, G. E./Sejnowski, T. J. 1985, pp. 154–155

<sup>38</sup>Vgl. Zhai et al. 2016, pp. 2–3

As a side note it is worth mentioning that using the maximum likelihood estimator for  $Z$  is intractable due to the requirement of summing over all possible states, which leads to an exponential increase in the number of states for larger systems.<sup>39</sup>

### 2.2.2 concept of Boltzmann Machines

A BM is a specific symmetrical EBM consisting of binary neurons  $\{0, 1\}$ .<sup>40</sup> The neurons of the network can be split into two functional groups, a set of visible neurons and a set of hidden neurons.<sup>41</sup> Therefore, the BM is a two-layer model with a visible layer (“v”) and a hidden layer (“h”).<sup>42</sup> The visible layer is the interface between the network and the environment. It receives data inputs during training and sets the state of a neuron to either  $\{0, 1\}$  which represents activated or not activated. On the other hand, the hidden units are not connected to the environment and can be used to “explain” underlying constraints in the internal model of input vectors and they cannot be represented by pairwise constraints.<sup>43</sup> The connection between the individual neurons is referred to as bidirectional, as each neuron communicates with each other in both directions.<sup>44</sup>

As early as 1985, one of the founding fathers of artificial intelligence, “Geoffrey Hinton”, was aware that an BM is able to learn its underlying features by looking at data from a domain and developing a generative internal model.<sup>45</sup> In the next step, it is possible to generate examples with the same probability distribution as the input data examples shown. In the following figure 2, a general BM is depicted, where the upper layer embodies a vector of stochastic binary ‘hidden’ features, while the lower layer embodies a vector of stochastic binary ‘visible’ variables.<sup>46</sup>

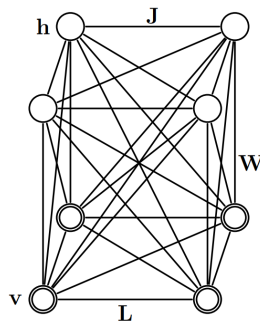


Abb. 2: figure of a general Boltzmann Machine

---

<sup>39</sup>Vgl. Zhai et al. 2016, pp. 2–3

<sup>40</sup>Vgl. Amari/Kurata/Nagaoka 1992, p. 260

<sup>41</sup>Vgl. Ackley/Hinton, G. E./Sejnowski, T. J. 1985, p. 154

<sup>42</sup>Vgl. Salakhutdinov/Hinton, G. 2009, p. 448

<sup>43</sup>Vgl. Ackley/Hinton, G. E./Sejnowski, T. J. 1985, p. 154

<sup>44</sup>Vgl. Ackley/Hinton, G. E./Sejnowski, T. J. 1985, p. 149

<sup>45</sup>Vgl. Ackley/Hinton, G. E./Sejnowski, T. J. 1985, p. 148

<sup>46</sup>Vgl. Salakhutdinov/Hinton, G. 2009, p. 449

The model contains a set of visible units  $v \in \{0, 1\}$ , and a set of hidden units  $h \in \{0, 1\}$  (see Fig. 1). The energy function of the BM with the states  $\{v, h\}$  is defined as:

$$E(v, h; \theta) = -\frac{1}{2}v^T L v - \frac{1}{2}h^T J h - v^T W h, \quad (2.4)$$

where  $\theta = \{W, L, J\}$  are the model parameters.<sup>47</sup>  $W, L, J$  represent visible-to-hidden, visible-to-visible and hidden-to-hidden weights. The individual neurons can be made to try to minimize the global energy by setting the right assumptions.<sup>48</sup> Entering a particular input to the machine, the system will find the minimum energy configuration that can illustrate the input.<sup>49</sup> A simple method to find a local energy minimum is to switch into whichever of the two states of a neuron hold the lower energy given the current state of the other neurons.<sup>50</sup> The exact reason for this is the following: “If all the connection strengths are symmetrical, which is typically the case for constraint satisfaction problems, each unit can compute its effect on the total energy from information that is locally available.”<sup>51</sup> By inserting the function 2.4 into the earlier introduced KL-divergence 2.2 and doing gradient descend the following learning rule to update the weights and biases results<sup>52</sup>:

$$\Delta w_{ij} = \epsilon(\langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{model}}) \quad (2.5)$$

The network can now update the weights “W” that exist between the neurons through the training rule based on the observations that served as input and modified by the learning rate  $\epsilon$ .<sup>53</sup>

Performing exact maximum likelihood learning in this model is intractable because exact computation of the data predictions and the model predictions takes a time that is exponential in the number of hidden units.<sup>54</sup> When the number of hidden units is large compared to the number of visible units it is impossible to achieve a perfect model because of the totally connected network and the resulting  $2^n$  possibilities.<sup>55</sup> This leads back to the briefly mentioned constraint of equation 2.3, that is needed to calculate an activation probability of a neuron, which is required to update a weight in the training process shown in 2.5.

A specific example to demonstrate why it is intractable to calculate a activation of a BM is the following. A fictional BM has 80 visible nodes and 120 hidden nodes and therefore the possibilities of states of neurons are  $2^{200}$ , which is  $1.61 \times 10^{60}$ . To put this in perspective the total atoms that

---

<sup>47</sup> Vgl. Salakhutdinov/Hinton, G. 2009, p. 448

<sup>48</sup> Vgl. Ackley/Hinton, G. E./Sejnowski, T. J. 1985, p. 150

<sup>49</sup> Vgl. Ackley/Hinton, G. E./Sejnowski, T. J. 1985, p. 150

<sup>50</sup> Vgl. Fahlman/Hinton, G./Sejnowski, T. 1983, p. 110

<sup>51</sup> Fahlman/Hinton, G./Sejnowski, T. 1983, p. 110

<sup>52</sup> Vgl. Hinton, G. E. 2012b, p. 5

<sup>53</sup> Vgl. Barra et al. 2012, pp. 1–2

<sup>54</sup> Vgl. Salakhutdinov/Hinton, G. 2009, p. 449

<sup>55</sup> Vgl. Ackley/Hinton, G. E./Sejnowski, T. J. 1985, p. 154

exist on earth are only estimated to be around  $1.33 \times 10^{50}$ .<sup>56</sup> That means even if it would be possible to store one information per atom it would just not be enough.

### 2.2.3 Training of Restricted Boltzmann Machines

As a solution for the training problem Hinton and Sejnowski proposed Gibbs sampling as an algorithm to approximate both expectations.<sup>57</sup> Furthermore, the intralayer connections of the model got removed and the result is the so called RBM. To transform an BM into a RBM the diagonal elements  $L$  and  $J$  introduced earlier, are set to 0 and as a result the well-known model of a RBM establishes shown in fig.2.<sup>58</sup>

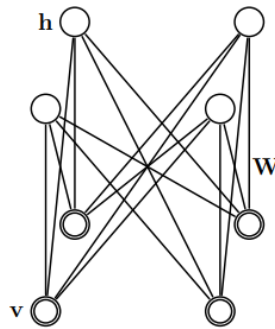


Abb. 3: Figure of a RBM

What can be recognized that no more visible-to-visible and hidden-to-hidden connections can be found in the model. The configuration of the visible and hidden units ( $v, h$ ) therefore has also an updated energy function (Hopfield, 1982) given by:

$$E(v, h) = - \sum_{i \in \text{visible}} a_i v_i - \sum_{j \in \text{hidden}} b_j h_j - \sum_{i,j} v_i h_j w_{ij}, \quad (2.6)$$

where  $v_i, h_j$  are the binary states of a visible unit  $i$  and hidden unit  $j$ ,  $a_i, b_j$  are their biases and  $w_{ij}$  is the weight between them.<sup>59</sup> Despite, compared to the fully connected BM, the RBM is less complex but the advantages of training surpasses the loss in expressivity.<sup>60</sup> The RBM has recently been drawing attention in the machine learning community because of its adaption and extension for various tasks such as representational learning, document modeling, image recognition and for serving as foundational components for deep networks including Deep Boltzmann Machines, Deep Belief Networks and hybrid models with CNNs.<sup>61</sup> The training of the model can be split up into the following steps:

---

<sup>56</sup>Vgl. Helmenstine 2022, p. 478-480; Vgl. Schlamming 2014, p. 1

<sup>57</sup>Vgl. Ackley/Hinton, G. E./Sejnowski, T. J. 1985, pp. 158–165

<sup>58</sup>Vgl. Salakhutdinov/Hinton, G. 2009, p. 449

<sup>59</sup>Vgl. Hinton, G. E. 2012a, pp. 3–4

<sup>60</sup>Vgl. Huembeli et al. 2022, p. 4

<sup>61</sup>Vgl. Zhang et al. 2018, p. 1186



### 1. Forward Pass (positive phase)

During the forward pass using the Gibbs Sampling method, the visible units are set to a completely random state. Next up the hidden units are computed. The computation of the hidden units involves calculating their activation probabilities and performing an actual sampling with their calculated activation probabilities. With the RBM it is now easy to get an analytical calculated unbiased sample of  $(\mathbf{v}_i \mathbf{h}_j)_{data}$ .<sup>62</sup> Given an input data out of the training images,  $v$ , the binary state,  $j$ , of each hidden unit,  $h_j$ , is set to 1 with following probability:

$$p(h_j = 1|\mathbf{v}) = \sigma(b_j + \sum_i v_i w_{ij}), \quad (2.7)$$

where  $\sigma(x)$  is the logistic sigmoid function with an unbiased sample. The sigmoid function is defined as  $\sigma(x) = \frac{1}{1+\exp(-x)}$  and shown in figure 4:

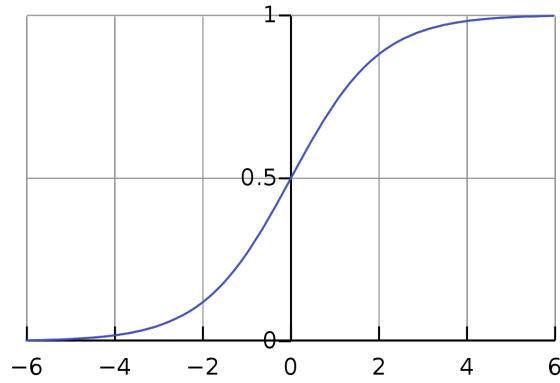


Abb. 4: Figure of a logistic sigmoid function RBM

The result is a set of probabilities that reflect how likely it is for each hidden unit to be on or off given the input data.<sup>63</sup> The sampling part of the positive phase uses the just calculated activation probability of each hidden unit and performs a random experiment with it. Afterwards, the hidden unit is either activated or not activated and the training process continues with the new state of the hidden units (activated or not activated).

### 2. Reconstruction (negative phase)

In this phase, the sampled hidden states are used to reconstruct the visible units. This is essentially a prediction of the input, which is calculated with following probability:<sup>64</sup>

$$p(v_i = 1|\mathbf{h}) = \sigma(a_i + \sum_j h_j w_{ij}) \quad (2.8)$$

---

<sup>62</sup>Vgl. Hinton, G. E. 2012b, p. 5

<sup>63</sup>Vgl. Huembeli et al. 2022, p. 6

<sup>64</sup>Vgl. Hinton, G. E. 2012b, p. 6

The sampling part of the negative phase uses the just calculated activation probability of each visible unit and performs a random experiment, like in the positive phase. Now, the result is a prediction of the input in the visible nodes. Afterwards, a half forward pass is made to calculate the activation probability of a hidden unit again based on the activated or not activated visible units.

### 3. Updating the weights

Meanwhile, all the requirements to update the weights are satisfied and can be used within the equation 2.5. The delta that results is summed to the current weight and therefore the internal model gets closer to predicting the observed data. Therefore, one training iteration consisting out of 1 Forward Pass, 1 Reconstruction and 0.5 Forward Pass again is accomplished. Repeating this training steps  $N$  times for a suitable chosen  $N$  the model learns better, since more steps of alternating Gibbs sampling were performed.<sup>65</sup>

### Markov-Chain-Monte-Carlo-Verfahren

**Contrastive Divergence:** Contrastive divergence is a special Gibbs Sampling training method developed by Geoffrey Hinton for the efficient training of Boltzmann Machines, especially RBMs.<sup>66</sup> In traditional Gibbs sampling would have to generate a long chain of samples, until independent samples are obtained from the observed data distribution of the model.<sup>67</sup> The samples are needed for each iteration of the gradient ascent on the log-likelihood resulting in large computational costs.<sup>68</sup> To solve this issue contrastive divergence minimizes an approximation of the Kullback-Leibler divergence between the empirical distribution of the training data and the distribution generated by the model.<sup>69</sup> They way to achieve this is by initializing the Markov chain with the samples from the data distributon.<sup>70</sup> The outcome has been shown to heavily increase the training time while only adding a small bias.<sup>71</sup> What this means is initializing the visible units with a real data input for example a MNIST sample and starting the proposed steps with the underlying states. Often the process can be stopped after only sampling a very small number of steps (often only one).<sup>72</sup>

**Metropolis-Hastings:** The Metropolis-Hastings algorithm, often only called Metropolis algorithm, is a technique out of Markov chain Monte Carlo (MCMC) class techniques.<sup>73</sup> The Metropolis-Hastings method was invented by Metropolis et al. in 1953 when he noticed, that for an intractable distribution with too many states it can be seen as a limiting distribution of

---

<sup>65</sup>Vgl. Huembeli et al. 2022, p. 6

<sup>66</sup>Vgl. Hinton, G. E. 2012b, pp. 4–5

<sup>67</sup>Vgl. Huembeli et al. 2022, pp. 5–6

<sup>68</sup>Vgl. Upadhyaya/Sastry 2019, pp. 7–8

<sup>69</sup>Vgl. Mocanu et al. 2016, p. 246

<sup>70</sup>Vgl. Upadhyaya/Sastry 2019, pp. 7–8

<sup>71</sup>Vgl. Larochelle/Bengio 2008, p. 537

<sup>72</sup>Vgl. Larochelle/Mandel, et al. 2012, p. 646

<sup>73</sup>Vgl. Patrón et al. 2024, p. 1

Markov chains.<sup>74</sup> The goal of the technique is to create a sequence of correlated steps from a random walk that, after enough iterations, makes it possible to sample a desired target probability distribution.<sup>75</sup> The intractable distribution to handle with the Metropolis-Hastings technique in the case of RBMs is equation 2.3. An Interpretation of the method can be expressed as: "A visitor to a museum that is forced by a general blackout to watch a painting with a small torch. Due to the narrow beam of the torch, the person cannot get a global view of the painting but can proceed along this painting until all parts have been seen."<sup>76</sup> The version already adjusted for RBMs incorporates the following functionality of the Metropolis technique:

First, select a random or given configuration  $x_{\text{old}}$  of a RBM that holds the states of all visible and hidden neurons.<sup>77</sup> Secondly, the energy of the configuration, noted as  $E_{\text{old}}$ , must be calculated using Equation 2.6, as previously introduced. Subsequently, this energy value is stored. Thirdly, the configuration gets updated by picking one random neuron and changing the state of it from 0 to 1 or vice versa.<sup>78</sup> This new configuration is stored as  $x_{\text{new}}$ . Following that the energy of the new configuration  $E_{\text{new}}$  is calculated and stored. Now the two energy values are compared and if  $E_{\text{new}} < E_{\text{old}}$  the new configuration will be accepted and  $x_{\text{old}} = x_{\text{new}}$ .<sup>79</sup> If  $E_{\text{new}} > E_{\text{old}}$  then there are some extra steps to be followed:

The flip probability is calculated as  $p = \exp\left(-\frac{E_{\text{new}} - E_{\text{old}}}{kT}\right)$ .  $kT$  is interpreted as the temperature in the network and with higher temperature it increases the activation probability leading to an faster exploration through the landscape but with less details.<sup>80</sup> For RBMs  $kT$  is assumed to be 1. (FOOTCITE von Fabian fehlt). In the following figure 5 the resulting probability function is shown.

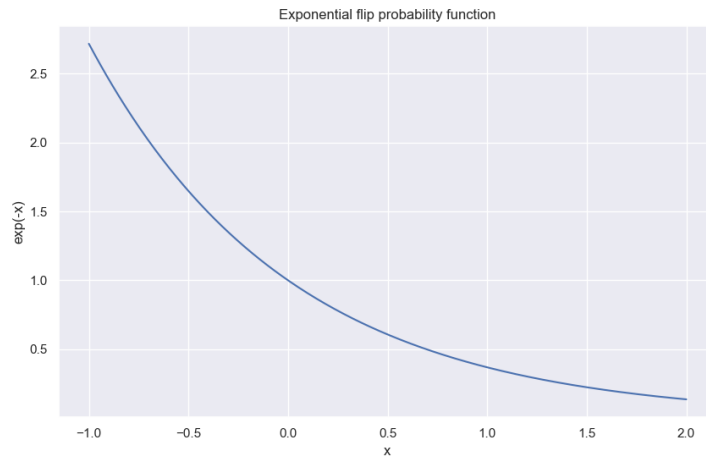


Abb. 5: Figure of the exponential flip probability function

---

<sup>74</sup>Vgl. Metropolis et al. 1953, pp. 1087–1092

<sup>75</sup>Vgl. Patrón et al. 2024, p. 1

<sup>76</sup>Vgl. Robert 2016, p. 2

<sup>77</sup>Vgl. Beichl/Sullivan 2000, p. 65

<sup>78</sup>Vgl. Rosenthal 2009, p. 1

<sup>79</sup>Vgl. Patrón et al. 2024, pp. 1–2

<sup>80</sup>Vgl. Li et al. 2016, pp. 1–9

In the next step a uniform random number  $r$  between 0 and 1 is generated. After generating  $r$  the configuration will be accepted if  $r \leq p$  (i.e.,  $x_{\text{old}} = x_{\text{new}}$ ).<sup>81</sup> Otherwise, a rejection takes place if  $r > P$  (i.e.,  $x_{\text{old}} = x_{\text{old}}$ ).

Finally, the configuration  $x_{\text{old}}$  can be stored and the process repeats beginning from step 2 on.<sup>82</sup> After repeating enough times the activation probability for each neuron can be calculated by summing over all samples ( $x_1 + x_2 + x_3 + \dots$ ) and the result is divided by the total number of samples.

### 2.2.4 Current Problems with BMs and RBMs

One general problem that occurs in the learning process of a BM is that it is both time-consuming and difficult.<sup>83</sup> This is because sampling from an undirected graphical model is not straightforward and therefore RBMs can make use of MCMC proposed methods like Contrastive Divergence and Metropolis Hastings.<sup>84</sup> In addition to that, the selection of hyperparameters can be difficult since for the training of a practical model a large hyper-parameter space needs to be explored.<sup>85</sup> Especially finding the right size of the hidden layer, the learning rate and number of training iterations but also the method for calculating activation probabilities (Contrastive Divergence, Metropolis Hastings, etc.) can be seen as art. Furthermore, training can become unstable due to the system's low temperature, which impacts the training negatively.<sup>86</sup> A lower temperature reduces the system's possibility to explore the energy landscape thoroughly, leading to the false selection of local minima instead of finding the global minimum.

---

<sup>81</sup>Vgl. Patrón et al. 2024, pp. 2–3

<sup>82</sup>Vgl. Patrón et al. 2024, p. 17

<sup>83</sup>Vgl. Fischer/Igel 2012, pp. 1–2

<sup>84</sup>Vgl. Fischer/Igel 2012, p. 2

<sup>85</sup>Vgl. Larochelle/Bengio 2008, p. 536

<sup>86</sup>Vgl. Huembeli et al. 2022, pp. 3–4

## 2.3 Hardwarebeschleuniger

### 2.3.1 Aktuelle Ansätze im Bereich KI und weitere Lösungen

Asics

Quantencomputing

### 2.3.2 ISING Maschine/ Physikinspirierter Hardwarebeschleuniger

Konzept (mit Energiefunktion), Probleme der Digitalrechner bzw. Unterschied zu Digitalrechner

Aktuelle Anwendung

Potentielle Einsatzgebiete für KI-Modelle

Parallelen Energiefunktion BM und ISING Maschine

## 2.4 Memristor Hopfield Network

### 2.4.1 Memristor

### 2.4.2 Hopfield Network

A Hopfield Network is an EBM and belongs to the field of recurrent neural networks.<sup>87</sup> The structure of the network consists of only one single layer with binary valued neurons inside.<sup>88</sup> Therefore, the neurons state can either be  $\{1, 0\}$  or  $\{1, -1\}$ . The connections between the neurons are symmetrical, which means that the weights of the connections are the same in either direction.<sup>89</sup> Initially, the primary applications of this type of network were to serve as storage for associative patterns and to facilitate pattern retrieval.<sup>90</sup> In practice given a query pattern a Hopfield Network can retrieve a pattern that is most similar or an average of similar patterns.<sup>91</sup> In this paper the Hopfield Network's update function interests us because it possibly could be used to sample the intractable training of a RBM mentioned earlier. Surprisingly, since Hopfield networks were introduced by J.J Hopfield in 1982 the storage capacity got increased

---

<sup>87</sup>Vgl. Dramsch 2020, p. 35

<sup>88</sup>Vgl. Ahad/Qadir/Ahsan 2016, p. 7

<sup>89</sup>Vgl. Ahad/Qadir/Ahsan 2016, p. 7

<sup>90</sup>Vgl. Ramsauer et al. 2021, p. 2

<sup>91</sup>Vgl. Ramsauer et al. 2021, p. 2

over time but the fundamentals stayed the same.<sup>92</sup> In following figure 6 an example of a Hopfield Network can be seen.<sup>93</sup>

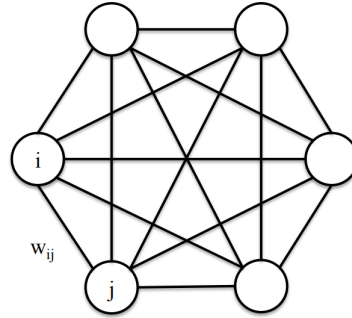


Abb. 6: Figure of a hopfield network

The exemplary network has 6 neurons and bidirectional weights  $W_{ij}$  between the neurons. In addition to that, a Hopfield network has no input or output layer.<sup>94</sup> The main goal is to find the values for each neuron in the network given a specific input that minimizes the total energy of the system.<sup>95</sup> The minimum energy is then equal to the state where the network is able to perform as a memory item.<sup>96</sup> This energy state can be calculated with the following energy equation<sup>97</sup>:

$$E = -\frac{1}{2} \sum_{i \neq j} T_{ij} V_i V_j. \quad (2.9)$$

This energy function invented by Hopfield has big similarities with a BM when comparing to the equation 2.4. This is one of the reasons why the execution on the Memristor Hopfield Network could work out. When comparing a Hopfield Network, they seek to achieve the effect of changing node activation on the overall energy of the network but BMs replace this with the probability of a certain node being activated on the network energy.<sup>98</sup> The second important reason to research the hopfield networks is for their updating process. Approximately, the acitivity rule for each neuron is to update its state as if it were a single neuron with the threshold activation function.<sup>99</sup>

$$s_i \leftarrow \begin{cases} +1 & \text{if } \sum_j w_{ij} s_j + b \geq \theta_i, \\ -1 & \text{otherwise.} \end{cases}$$

The state of the neuron will be updated to 1 if the sum over all weights multiplied with the states  $\{1, -1\}$  added to a bias  $b$  is greater than the threshold  $\theta_i$ . In the case of our accelerator the threshold is set to 0 but in theory can be used as an hyperparameter.

<sup>92</sup>Vgl. Hopfield 1982, p. 2554-2558; Vgl. Ramsauer et al. 2021, p. 2

<sup>93</sup>Vgl. Yao/Gripon/Rabbat 2013, pp. 1-2

<sup>94</sup>Vgl. Yao/Gripon/Rabbat 2013, p. 3

<sup>95</sup>Vgl. Ahad/Qadir/Ahsan 2016, p. 7

<sup>96</sup>Vgl. Ahad/Qadir/Ahsan 2016, p. 7

<sup>97</sup>Vgl. Hopfield 1982, p. 2556

<sup>98</sup>Vgl. Ahad/Qadir/Ahsan 2016, p. 7

<sup>99</sup>Vgl. MacKay 2003, p. 506

Since every neuron's output is an input to all the other neurons the order of the updates need to be specified.<sup>100</sup> There is the possibility to update all neurons synchronous or asynchronous. There is no study that shows what update method leads to better results. Therefore, this paper follows the asynchronous option and ensures to do enough iterations, so that every neuron has at least updated once before moving on. In addition to that, the idea of the updating method of the accelerator is slightly different. The idea behind this is to inject noise into the system so that the acitivation function could work together with the activation function that a RBM needs to perform. In detail the idea is to add a normal gaussian distribution  $g(x)$  on top of the activation function.<sup>101</sup> As a result the new statistical updating function looks like the following:

$$s_i \leftarrow \begin{cases} +1 & \text{if } \sum_j w_{ij} + b + g(x) \geq \theta_i, \\ -1 & \text{otherwise.} \end{cases}$$

Now the system could potentially be used to update the states of the neurons within a RBM. Since the success of this method is not guaranteed or tested in literature yet the practical part first needs to validate if this concept is feasible.

### 2.4.3 Crossbar

### 2.4.4 Output Hopfield Network

### 2.4.5 Noisy HNN

---

<sup>100</sup>Vgl. MacKay 2003, p. 506

<sup>101</sup>Vgl. Böhm et al. 2022, pp. 4–5

### 3 Zielspezifikation und Darlegung der Forschungsmethodik

3.1 Zielspezifikation (genauer als in Einleitung, Metriken erwähnen, Erfolg meiner Methode bewerten, Welcher Teil der Forschungsfrage wird beantwortet?)

3.2 Design Science Research

3.3 Zielsetzung(ohne genaue Metriken nennen, generell halten)

3.4 Laborexperiment für die Umsetzung



## 4 Implementierung/Laborexperiment der Simulator Pipeline

Hopfield Netzwerk aktivierungsfunktion der Updating methode

-> Konzeptionell Art des Updates mit keiner Temperatur wie bei MCMC Unterschied von MCMC zu Hopfield Netzwerk -> Zufällige Konfiguration und minimale Energie finden. Jedoch hat ein Hopfield Netzwerk keine Temperatur

-> Starte zufällige Konfiguration -> Wähle ein Neuron aus und Berechne Summe und addiere mit Bias, -> Update wenn threshold überschritten 1 und dann auf 0 -> Speichern der neuen Konfiguration -> Starte iteration von gespeicherter Konfiguration -> Am Ende habe ich 10000 Vektoren (Die Konfigurationen) -> V1 Neuron wurde so und so oft aktiviert und ich muss average über das neuron und habe dadurch die Aktivierungswarscheinlichkeit.

-Aktivierungsfunktion einfügen (Binary Step und verfleich zu sigmoid von Abb.4)

-Testen der Aktivierungsfunktion, wenn ich ein Neuron trainiere und dann Mitteln - Von vornerein auf Netzwerk Basis arbeiten mit mehreren Neuron, jedoch für 1 Neuron testen

### 4.1 Zielsetzung und Forschungsmethodik

### 4.2 Aufbau der Simulator Pipeline

### 4.3 KI-Bibliothek Scikit-Learn

## 5 Evaluation der BM auf dem physikinspiriertem Hardwarebeschleuniger

### 5.1 Zielsetzung und Forschungsmethodik

#### 5.1.1 Prediction Accuracy

#### 5.1.2 Troughput (Samples/Sec)

#### 5.1.3 Energieverbrauch (Energy/Operation)

### 5.2 Vergleichen mit anderen Hardwarebeschleuniger, FPGA, GPU oder CPU aus der Literatur

## 6 Kritische Reflexion und Ausblick

6.1 Evaluation der Erkenntnisse in Bezug auf die Zielsetzung der Arbeit

6.2 Kritische Reflexion der Ergebnisse und Methodik

6.3 Zielsetzung(ohne genaue Metriken nennen, generell halten)

6.4 Ergebnisextraktion für Theorie und Praxis (evtl. mit 6.4 Zusammenlegen)

6.5 Ausblick

# Appendix

## List of appendices

Anhang 1	So funktioniert's . . . . .	23
Anhang 1/1	Wieder mal eine Abbildung . . . . .	23

## Appendix 1: So funktioniert's

Um den Anforderungen der Zitierrichtlinien nachzukommen, wird das Paket `tocloft` verwendet. Jeder Anhang wird mit dem (neu definierten) Befehl `\anhang{Bezeichnung}` begonnen, der insbesondere dafür sorgt, dass ein Eintrag im Anhangsverzeichnis erzeugt wird. Manchmal ist es wünschenswert, auch einen Anhang noch weiter zu unterteilen. Hierfür wurde der Befehl `\anhangteil{Bezeichnung}` definiert.

In Anhang 1/1 finden Sie eine bekannte Abbildung und etwas Source Code in ??.

### Anhang 1/1: Wieder mal eine Abbildung



Abb. 7: Mal wieder das DHBW-Logo.

## List of references

- Ackley, D. H./Hinton, G. E./Sejnowski, T. J. (1985): A Learning Algorithm for Boltzmann Machines. In: *Cognitive Science* 9.1, pp. 147–169. ISSN: 0364-0213. DOI: 10.1016/S0364-0213(85)80012-4. URL: <https://www.sciencedirect.com/science/article/pii/S0364021385800124> (retrieval: 02/16/2024).
- Ahad, N./Qadir, J./Ahsan, N. (2016): Neural Networks in Wireless Networks: Techniques, Applications and Guidelines. In: *Journal of Network and Computer Applications* 68, pp. 1–27. ISSN: 1084-8045. DOI: 10.1016/j.jnca.2016.04.006. URL: <https://www.sciencedirect.com/science/article/pii/S1084804516300492> (retrieval: 02/28/2024).
- Amari, S./Kurata, K./Nagaoka, H. (1992): Information Geometry of Boltzmann Machines. In: *IEEE Transactions on Neural Networks* 3.2, pp. 260–271. ISSN: 1941-0093. DOI: 10.1109/72.125867. URL: <https://ieeexplore.ieee.org/abstract/document/125867> (retrieval: 02/16/2024).
- Bai, G./Chai, Z./Ling, C./Wang, S./Lu, J./Zhang, N./Shi, T./Yu, Z./Zhu, M./Zhang, Y./Yang, C./Cheng, Y./Zhao, L. (2024): Beyond Efficiency: A Systematic Survey of Resource-Efficient Large Language Models. DOI: 10.48550/arXiv.2401.00625. arXiv: 2401.00625 [cs]. URL: <http://arxiv.org/abs/2401.00625> (retrieval: 02/23/2024). preprint.
- Barra, A./Bernacchia, A./Santucci, E./Contucci, P. (2012): On the Equivalence of Hopfield Networks and Boltzmann Machines. In: *Neural Networks* 34, pp. 1–9. ISSN: 0893-6080. DOI: 10.1016/j.neunet.2012.06.003. URL: <https://www.sciencedirect.com/science/article/pii/S0893608012001608> (retrieval: 02/16/2024).
- Beichl, I./Sullivan, F. (2000): The Metropolis Algorithm. In: *Computing in Science & Engineering* 2.1, pp. 65–69. ISSN: 1558-366X. DOI: 10.1109/5992.814660. URL: <https://ieeexplore.ieee.org/document/814660> (retrieval: 02/27/2024).
- Böhm, F./Alonso-Urquijo, D./Verschaffelt, G./Van der Sande, G. (2022): Noise-Injected Analog Ising Machines Enable Ultrafast Statistical Sampling and Machine Learning. In: *Nature Communications* 13.1 (1), p. 5847. ISSN: 2041-1723. DOI: 10.1038/s41467-022-33441-3. URL: <https://www.nature.com/articles/s41467-022-33441-3> (retrieval: 02/15/2024).
- Cai, F./Kumar, S./Van Vaerenbergh, T./Liu, R./Li, C./Yu, S./Xia, Q./Yang, J. J./Beausoleil, R./Lu, W./Strachan, J. P. (2019): Harnessing Intrinsic Noise in Memristor Hopfield Neural Networks for Combinatorial Optimization. DOI: 10.48550/arXiv.1903.11194. arXiv: 1903.11194 [cs]. URL: <http://arxiv.org/abs/1903.11194> (retrieval: 02/15/2024). preprint.
- Cichy, R. M./Kaiser, D. (2019): Deep Neural Networks as Scientific Models. In: *Trends in Cognitive Sciences* 23.4, pp. 305–317. ISSN: 1364-6613, 1879-307X. DOI: 10.1016/j.tics.2019.01.009. pmid: 30795896. URL: [https://www.cell.com/trends/cognitive-sciences/abstract/S1364-6613\(19\)30034-8](https://www.cell.com/trends/cognitive-sciences/abstract/S1364-6613(19)30034-8) (retrieval: 02/23/2024).
- Dario Amodei/Danny Hernandez (2024): AI and Compute. URL: <https://openai.com/research/ai-and-compute> (retrieval: 02/15/2024).

- Dramsch, J. S. (2020):** ‘Chapter One - 70 Years of Machine Learning in Geoscience in Review’. In: *Advances in Geophysics*. Ed. by Ben Moseley/Lion Krischer. Vol. 61. Machine Learning in Geosciences. Elsevier, pp. 1–55. DOI: 10.1016/bs.agph.2020.08.002. URL: <https://www.sciencedirect.com/science/article/pii/S0065268720300054> (retrieval: 02/28/2024).
- Du, Y./Lin, T./Mordatch, I. (2021):** Model Based Planning with Energy Based Models. DOI: 10.48550/arXiv.1909.06878. arXiv: 1909.06878 [cs, stat]. URL: <http://arxiv.org/abs/1909.06878> (retrieval: 02/19/2024). preprint.
- Du, Y./Mordatch, I. (2020):** Implicit Generation and Generalization in Energy-Based Models. DOI: 10.48550/arXiv.1903.08689. arXiv: 1903.08689 [cs, stat]. URL: <http://arxiv.org/abs/1903.08689> (retrieval: 02/23/2024). preprint.
- Durstewitz, D./Koppe, G./Meyer-Lindenberg, A. (2019):** Deep Neural Networks in Psychiatry. In: *Molecular Psychiatry* 24.11 (11), pp. 1583–1598. ISSN: 1476-5578. DOI: 10.1038/s41380-019-0365-9. URL: <https://www.nature.com/articles/s41380-019-0365-9> (retrieval: 02/23/2024).
- Fahlman, S./Hinton, G./Sejnowski, T. (1983):** Massively Parallel Architectures for AI: NETL, Thistle, and Boltzmann Machines., p. 113. 109 pp.
- Fischer, A./Igel, C. (2012):** An Introduction to Restricted Boltzmann Machines. In: *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Ed. by Luis Alvarez/Marta Mejail/Luis Gomez/Julio Jacobo. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 14–36. ISBN: 978-3-642-33275-3. DOI: 10.1007/978-3-642-33275-3\_2.
- Gawlikowski, J./Tassi, C. R. N./Ali, M./Lee, J./Humt, M./Feng, J./Kruspe, A./Triebel, R./Jung, P./Roscher, R./Shahzad, M./Yang, W./Bamler, R./Zhu, X. X. (2023):** A Survey of Uncertainty in Deep Neural Networks. In: *Artificial Intelligence Review* 56.1, pp. 1513–1589. ISSN: 1573-7462. DOI: 10.1007/s10462-023-10562-9. URL: <https://doi.org/10.1007/s10462-023-10562-9> (retrieval: 02/23/2024).
- Gustafsson, F. K./Danelljan, M./Bhat, G./Schön, T. B. (2020):** Energy-Based Models for Deep Probabilistic Regression. In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi/Horst Bischof/Thomas Brox/Jan-Michael Frahm. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 325–343. ISBN: 978-3-030-58565-5. DOI: 10.1007/978-3-030-58565-5\_20.
- Helmenstine, A. (2022):** How Many Atoms Are in the World? Science Notes and Projects. URL: <https://sciencenotes.org/how-many-atoms-are-in-the-world/> (retrieval: 02/21/2024).
- Hintemann, R./Hinterholzer, S. (2022):** Data Centers 2021: Data Center Boom in Germany Continues - Cloud Computing Drives the Growth of the Data Center Industry and Its Energy Consumption. DOI: 10.13140/RG.2.2.31826.43207.
- Hinton, G. E. (2012a):** ‘A Practical Guide to Training Restricted Boltzmann Machines’. In: *Neural Networks: Tricks of the Trade: Second Edition*. Ed. by Grégoire Montavon/Geneviève B. Orr/Klaus-Robert Müller. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 599–619. ISBN: 978-3-642-35289-8. DOI: 10.1007/978-3-642-35289-8\_32. URL: [https://doi.org/10.1007/978-3-642-35289-8\\_32](https://doi.org/10.1007/978-3-642-35289-8_32) (retrieval: 02/15/2024).

- Hinton, G. E. (2012b):** ‘A Practical Guide to Training Restricted Boltzmann Machines’. In: *Neural Networks: Tricks of the Trade*. Ed. by Grégoire Montavon/Geneviève B. Orr/Klaus-Robert Müller. Vol. 7700. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 599–619. ISBN: 978-3-642-35288-1 978-3-642-35289-8. DOI: 10.1007/978-3-642-35289-8\_32. URL: [http://link.springer.com/10.1007/978-3-642-35289-8\\_32](http://link.springer.com/10.1007/978-3-642-35289-8_32) (retrieval: 02/15/2024).
- Hopfield, J. J. (1982):** Neural Networks and Physical Systems with Emergent Collective Computational Abilities. In: *Proceedings of the National Academy of Sciences* 79.8, pp. 2554–2558. DOI: 10.1073/pnas.79.8.2554. URL: <https://www.pnas.org/doi/10.1073/pnas.79.8.2554> (retrieval: 02/19/2024).
- Huembeli, P./Arrazola, J. M./Killoran, N./Mohseni, M./Witteck, P. (2022):** The Physics of Energy-Based Models. In: *Quantum Machine Intelligence* 4.1, p. 1. ISSN: 2524-4914. DOI: 10.1007/s42484-021-00057-7. URL: <https://doi.org/10.1007/s42484-021-00057-7> (retrieval: 02/19/2024).
- Larochelle, H./Bengio, Y. (2008):** Classification Using Discriminative Restricted Boltzmann Machines. In: *Proceedings of the 25th International Conference on Machine Learning*. ICML ’08. New York, NY, USA: Association for Computing Machinery, pp. 536–543. ISBN: 978-1-60558-205-4. DOI: 10.1145/1390156.1390224. URL: <https://dl.acm.org/doi/10.1145/1390156.1390224> (retrieval: 02/22/2024).
- Larochelle, H./Mandel, M./Pascanu, R./Bengio, Y. (2012):** Learning Algorithms for the Classification Restricted Boltzmann Machine. In: *The Journal of Machine Learning Research* 13, pp. 643–669.
- Li, G./Deng, L./Xu, Y./Wen, C./Wang, W./Pei, J./Shi, L. (2016):** Temperature Based Restricted Boltzmann Machines. In: *Scientific Reports* 6.1 (1), p. 19133. ISSN: 2045-2322. DOI: 10.1038/srep19133. URL: <https://www.nature.com/articles/srep19133> (retrieval: 02/27/2024).
- Luccioni, A. S./Jernite, Y./Strubell, E. (2023):** Power Hungry Processing: Watts Driving the Cost of AI Deployment? DOI: 10.48550/arXiv.2311.16863. arXiv: 2311.16863 [cs]. URL: <http://arxiv.org/abs/2311.16863> (retrieval: 02/15/2024). preprint.
- MacKay, D. J. C. (2003):** Information Theory, Inference and Learning Algorithms. Cambridge University Press. 694 pp. ISBN: 978-0-521-64298-9. Google Books: AKuMj4PN EMC.
- Mall, P. K./Singh, P. K./Srivastav, S./Narayan, V./Paprzycki, M./Jaworska, T./Ganzha, M. (2023):** A Comprehensive Review of Deep Neural Networks for Medical Image Processing: Recent Developments and Future Opportunities. In: *Healthcare Analytics* 4, p. 100216. ISSN: 2772-4425. DOI: 10.1016/j.health.2023.100216. URL: <https://www.sciencedirect.com/science/article/pii/S2772442523000837> (retrieval: 02/23/2024).
- Marinó, G. C./Petrini, A./Malchiodi, D./Frasca, M. (2023):** Deep Neural Networks Compression: A Comparative Survey and Choice Recommendations. In: *Neurocomputing* 520, pp. 152–170. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2022.11.072. URL: <https://www.sciencedirect.com/science/article/pii/S0925231222014643> (retrieval: 02/23/2024).
- Metropolis, N./Rosenbluth, A. W./Rosenbluth, M. N./Teller, A. H./Teller, E. (1953):** Equation of State Calculations by Fast Computing Machines. In: *The Journal of Chemi-*



- cal Physics* 21.6, pp. 1087–1092. ISSN: 0021-9606. DOI: 10.1063/1.1699114. URL: <https://doi.org/10.1063/1.1699114> (retrieval: 02/27/2024).
- Mocanu, D. C./Mocanu, E./Nguyen, P. H./Gibescu, M./Liotta, A. (2016):** A Topological Insight into Restricted Boltzmann Machines. In: *Machine Learning* 104.2, pp. 243–270. ISSN: 1573-0565. DOI: 10.1007/s10994-016-5570-z. URL: <https://doi.org/10.1007/s10994-016-5570-z> (retrieval: 02/22/2024).
- Mohseni, N./McMahon, P. L./Byrnes, T. (2022):** Ising Machines as Hardware Solvers of Combinatorial Optimization Problems. DOI: 10.48550/arXiv.2204.00276. arXiv: 2204.00276 [physics, physics:quant-ph]. URL: <http://arxiv.org/abs/2204.00276> (retrieval: 02/15/2024). preprint.
- Nazm Bojnordi, M./Ipek, E. (2016):** Memristive Boltzmann Machine: A Hardware Accelerator for Combinatorial Optimization and Deep Learning, p. 13. 1 p. DOI: 10.1109/HPCA.2016.7446049.
- Patrón, A./Chepelianskii, A. D./Prados, A./Trizac, E. (2024):** On the Optimal Relaxation Rate for the Metropolis Algorithm in One Dimension. DOI: 10.48550/arXiv.2402.11267. arXiv: 2402.11267 [cond-mat, physics:math-ph]. URL: <http://arxiv.org/abs/2402.11267> (retrieval: 02/27/2024). preprint.
- Ramsauer, H./Schäfl, B./Lehner, J./Seidl, P./Widrich, M./Adler, T./Gruber, L./Holzleitner, M./Pavlović, M./Sandve, G. K./Greiff, V./Kreil, D./Kopp, M./Klambauer, G./Brandstetter, J./Hochreiter, S. (2021):** Hopfield Networks Is All You Need. DOI: 10.48550/arXiv.2008.02217. arXiv: 2008.02217 [cs, stat]. URL: <http://arxiv.org/abs/2008.02217> (retrieval: 02/28/2024). preprint.
- Robert, C. P. (2016):** The Metropolis-Hastings Algorithm. DOI: 10.48550/arXiv.1504.01896. arXiv: 1504.01896 [stat]. URL: <http://arxiv.org/abs/1504.01896> (retrieval: 02/27/2024). preprint.
- Rosenthal, S. (2009):** Optimal Proposal Distributions and Adaptive MCMC. In: *Handbook of Markov Chain Monte Carlo*.
- Salakhutdinov, R./Hinton, G. (2009):** Deep Boltzmann Machines. In: *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*. Artificial Intelligence and Statistics. PMLR, pp. 448–455. URL: <https://proceedings.mlr.press/v5/salakhutdinov09a.html> (retrieval: 02/16/2024).
- Schlamming, S. (2014):** A Cool Way to Measure Big G. In: *Nature* 510.7506 (7506), pp. 478–480. ISSN: 1476-4687. DOI: 10.1038/nature13507. URL: <https://www.nature.com/articles/nature13507> (retrieval: 02/21/2024).
- Singh, S. K./Kumar, S./Mehra, P. S. (2023):** Chat GPT & Google Bard AI: A Review. In: *2023 International Conference on IoT, Communication and Automation Technology (ICICAT)*. 2023 International Conference on IoT, Communication and Automation Technology (ICICAT), pp. 1–6. DOI: 10.1109/ICICAT57735.2023.10263706. URL: [https://ieeexplore.ieee.org/abstract/document/10263706?casa\\_token=JMHwBzQgxnwAAAAA:700nfYs5ECetZhuq8D\\_F3QXyua1Xu65rLoa\\_Ywve3mch00UAeSs0yVjhWCUvDuBpMX83NAbpUpM](https://ieeexplore.ieee.org/abstract/document/10263706?casa_token=JMHwBzQgxnwAAAAA:700nfYs5ECetZhuq8D_F3QXyua1Xu65rLoa_Ywve3mch00UAeSs0yVjhWCUvDuBpMX83NAbpUpM) (retrieval: 02/23/2024).

- Upadhyay, V./Sastry, P. (2019):** An Overview of Restricted Boltzmann Machines. In: *Journal of the Indian Institute of Science* 99. DOI: 10.1007/s41745-019-0102-z.
- Verdon, G./Marks, J./Nanda, S./Leichenauer, S./Hidary, J. (2019):** Quantum Hamiltonian-Based Models and the Variational Quantum Thermalizer Algorithm. arXiv: 1910.02071 [quant-ph]. URL: <http://arxiv.org/abs/1910.02071> (retrieval: 02/19/2024). preprint.
- Wang, T./Roychowdhury, J. (2017):** Oscillator-Based Ising Machine. DOI: 10.48550/arXiv.1709.08102. arXiv: 1709.08102 [physics]. URL: <http://arxiv.org/abs/1709.08102> (retrieval: 02/15/2024). preprint.
- Wittpahl, V., ed. (2019):** Künstliche Intelligenz: Technologie | Anwendung | Gesellschaft. Berlin, Heidelberg: Springer. ISBN: 978-3-662-58041-7 978-3-662-58042-4. DOI: 10.1007/978-3-662-58042-4. URL: <http://link.springer.com/10.1007/978-3-662-58042-4> (retrieval: 02/15/2024).
- Yao, Z./Gripon, V./Rabbat, M. (2013):** A Massively Parallel Associative Memory Based on Sparse Neural Networks. In.
- Zhai, S./Cheng, Y./Lu, W./Zhang, Z. (2016):** Deep Structured Energy Based Models for Anomaly Detection. In: *Proceedings of The 33rd International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, pp. 1100–1109. URL: <https://proceedings.mlr.press/v48/zhai16.html> (retrieval: 02/19/2024).
- Zhang, N./Ding, S./Zhang, J./Xue, Y. (2018):** An Overview on Restricted Boltzmann Machines. In: *Neurocomputing* 275, pp. 1186–1199. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2017.09.065. URL: <https://www.sciencedirect.com/science/article/pii/S0925231217315849> (retrieval: 02/15/2024).

# Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit mit dem Thema: *Mein Titel* selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

(Ort, Datum)

(Unterschrift)