

SCHILLER-GYMNASIUM HAMELN

12G SEMINARFACHARBEIT

MATHEMATIK/PHYSIK

---

# Umsetzung einer Künstlichen Intelligenz zur Erkennung handgeschriebener Zahlen

---

*Autor:*

Bui Anh Minh Leon Phan

*Tutoren:*

Hr. Fistler, Hr. Dr. Kajari

27. Januar 2023



# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>4</b>
<b>2</b>	<b>Künstliche Intelligenz</b>	<b>4</b>
2.1	Überwachtes Lernen . . . . .	5
<b>3</b>	<b>Voraussetzung für mehrschichtiges Lernen</b>	<b>5</b>
3.1	Datensatz . . . . .	6
3.2	Künstliches neuronales Netzwerk . . . . .	6
<b>4</b>	<b>Konvolutionales neuronales Netzwerk</b>	<b>7</b>
4.1	Klassifikationsebene . . . . .	8
4.1.1	Aktivierungsfunktion . . . . .	10
4.2	Feature Extraction . . . . .	11
4.2.1	2D Konvolution . . . . .	12
4.2.2	Max-Pooling . . . . .	13
4.2.3	Übergang von Feature Extraction zu Klassifikationsebene . . . . .	13
4.3	Verlustfunktion . . . . .	13
4.4	Optimierung . . . . .	14
4.4.1	Backpropagation . . . . .	14
<b>5</b>	<b>Umsetzung</b>	<b>15</b>
5.1	Werkzeuge . . . . .	16
5.2	Trainingsphase . . . . .	16
5.3	Auswertung . . . . .	16
5.3.1	Trainingsdaten . . . . .	16
5.3.2	Validationsdaten . . . . .	16
5.3.3	Testdaten . . . . .	16
<b>6</b>	<b>Schlussfolgerung</b>	<b>16</b>

## Abbildungsverzeichnis

1	Unterkategorien der Künstliche Intelligenz . . . . .	4
2	Ablauf eines überwachteten Lernen . . . . .	5
3	Handgeschriebene Zahl 5 . . . . .	6
4	Künstliches Neuronales Netzwerk . . . . .	7
5	Verbindungen zwischen den Neuronen . . . . .	8
6	Lineare Regression und nicht lineare Regression . . . . .	9
7	Verschiedene Extraktionsmöglichkeiten . . . . .	11
8	Einstellen der Lernrate . . . . .	15

## **Abkürzungs- & Akronymverzeichnis**

**KI** Künstliche Intelligenz

**CNN** Faltendes Neuronales Netzwerk

**FNN** Vorwärtsgerichtetes Neuronales Netzwerk

**ReLU** Rectified Linear Unit

**Backpropagation** Fehlerzurückführung

**Cross Entropy** Kreuzentropie

**Overfitting** Überanpassung

**Underfitting** Unteranpassung

**Accuracy** Genauigkeit

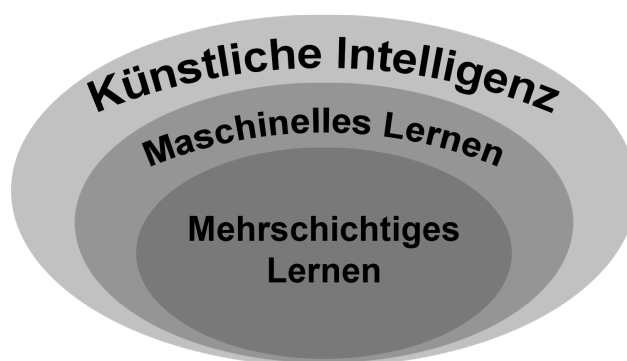
**Cross Correlation** Kreuzkorrelation

# 1 Einleitung

(Seite 5) In der heutigen Zeit spielt Künstliche Intelligenz (KI) eine sehr wichtige Rolle in unserer Gesellschaft. Es gibt viele Anwendungsbereiche für KI, sei es im Bereich Verkehr, Gesundheit, Werbung oder auch Cybersicherheit. [1] KI wird zum unverzichtbaren Mittel der Zukunft. So muss der Forschungsstand zum Bereich KI besonders weit sein, da die Fehlerberechnungen auf das Minimum beschränkt werden müssen. Nun ist es aber so, dass die jetzigen KIs nur auf einzelne Themen spezialisiert werden. Eine KI, die in mehreren Themengebieten professionell funktionieren kann, ist heutzutage noch nicht möglich. [2] Außerdem hinterfragt keiner, wie eine KI aufgebaut ist und wie dieser funktioniert. Zum Schluss schreiben

## 2 Künstliche Intelligenz

Der Begriff Künstliche Intelligenz (KI) ist sehr schwer definierbar, denn es mangelt an der Definition des Begriffs Intelligenz. Allgemein kann die KI definiert werden als der Versuch, bestimmte Entscheidungsstrukturen von Menschen nachzubilden mithilfe von Algorithmen. Manche Experten behaupten aber auch, dass eine KI nicht gleich eine KI ist, sondern sie in zwei Bereiche unterteilt werden kann. Dabei wird die KI zwischen einer starken KI und einer schwachen KI unterschieden. Schwache KIs sind nur in bestimmten Teilbereichen in der Lage an die menschliche Intelligenz heranzukommen. Doch sie ist nicht anpassungsfähig. Das bedeutet, dass eine schwache KI, die zum Beispiel auf Schach spezialisiert wird, nur die Intelligenz im Bereich Schach besitzt. Selbstständiges fahren ist mit der KI nicht möglich. Dafür muss eine andere KI spezialisiert werden, damit sie selbstständig fahren kann. Dagegen kann eine starke KI sich an derzeitigen Situationen anpassen. Diese nennen sich auch „Superintelligenz“. Solche starke KIs sind heutzutage noch nicht umsetzbar und es wird immer noch daran geforscht. Die schwache KI, die in der Facharbeit speziell zur Erkennung von handgeschriebenen Zahlen programmiert wird, können in 2 Teilbereichen unterteilt werden. Eine Unterkategorie der KI wäre der



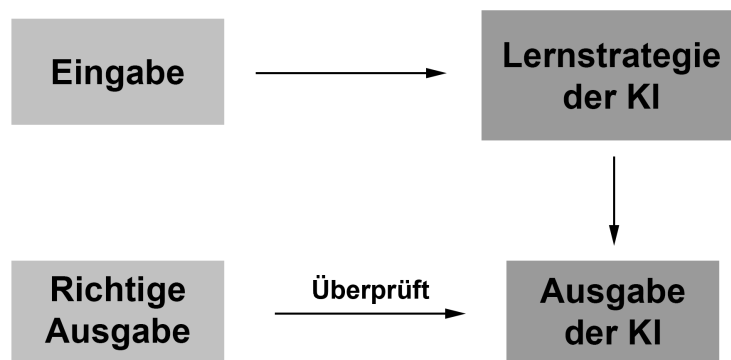
**Abbildung 1:** Die KI besitzt 2 Unterkategorien, Maschinelles Lernen und Mehrschichtiges Lernen.

Bereich Maschinelles Lernen. Maschinelles Lernen versucht mithilfe von Algorithmen mit

einer großen Anzahl von Daten ein Muster zu erkennen. Dazu werden die aus Daten gewonnen Erkenntnisse für Problemlösung verwendet oder der Algorithmus wird auf die Daten zugeschnitten und verallgemeinert. Der zweite Teilbereich wäre mehrschichtiges Lernen, das sehr an maschinelles Lernen ähnelt. Der Unterschied liegt aber an der Art und Weise, wie sie mit den Daten arbeitet. Mehrschichtiges Lernen versucht das menschliche Gehirn zu imitieren durch ein künstliches Neuronales Netzwerk. Durch das Imitieren sind die Strukturen vom mehrschichtigen Lernen deutlich komplexer als die vom maschinellen Lernen. Deswegen sind sehr große Datensätzen von Vorteil für mehrschichtiges Lernen, da deutlich mehr Merkmale der Daten erkannt werden. Darauf basiert sich die KI in der Facharbeit.

## 2.1 Überwachtes Lernen

Überwachtes Lernen ist eine Methode, wie die KI mit Daten lernen kann. Dazu hat die Datei 2 wichtige Merkmale, nämlich die Eingabe und die Ausgabe. Die Eingabe wird an die KI zum Lernen geschickt und dabei gibt die KI ein Ergebnis aus. Mit der Ausgabe der Datei wird nun überprüft, ob die Antwort, die die KI ausgegeben hat, übereinstimmt. Falls sie nicht übereinstimmt, lernt die KI aus den Fehlern und verbessert sich dadurch.



**Abbildung 2:** Zuerst wird die Eingabe verwendet, um zu lernen. Dafür verwendet die KI eine Lernstrategie. Nachdem etwas ausgegeben wird, werden die Ausgaben verglichen.

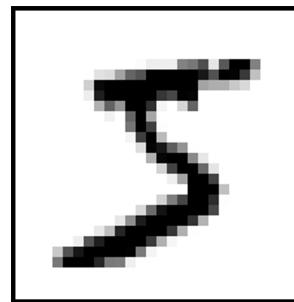
Überwachtes Lernen ist eine angemessene Methode für das Problem der Facharbeit, da die Antwort, die die KI ausgibt, übereinstimmen muss. Dadurch wird die Fehlerquote der KI möglichst kleingehalten, sodass eine genaue Erkennung der handgeschrieben Zahlen gelingen kann. Dabei ist anzumerken, dass die KI zwei Arten von Lösungen angeben kann. In dem Fall wird die sogenannte Klassifikation als Lösungsart akzeptiert. Jede Zahl wird als eine eigene Klasse zugeordnet. Die KI gibt aus, um welche Klasse es sich handelt.

## 3 Voraussetzung für mehrschichtiges Lernen

Irgendetwas schreiben für die Voraussetzung

### 3.1 Datensatz

Datensätze sind für die KIs das wichtigste Werkzeug für das Lernen. Denn die Qualität des Datensatzes bestimmt die Lerneffizienz der KI. Je schlechter die Qualität ist, desto schlechter lernt auch die KI. Wichtige Merkmale für Qualität ist die Auflösung, die Anzahl der Daten und die Variabilität. Für mehrschichtiges Lernen wird ein großer Datensatz gebraucht. Dabei wird für die Problemlösung der MNIST Datensatz von Yann LeCun gewählt. Sie besitzt insgesamt 70000 Bilder mit handgeschriebenen Zahlen, die die Werte zwischen 0 und 9 besitzt. Jedes Bild hat ein  $28 \times 28$  Pixel Format und ist farblos. Sie wird nur mit der Helligkeit dargestellt zwischen 0 und 255. Außerdem hat jedes Bild einen Wert, um welche Zahl es sich handelt, damit später geprüft werden kann, ob die KI die Zahl richtig erkannt hat oder ob sie falsch liegt. Der Datensatz ist gut zum Lernen, denn sie



Ausgabe: 5

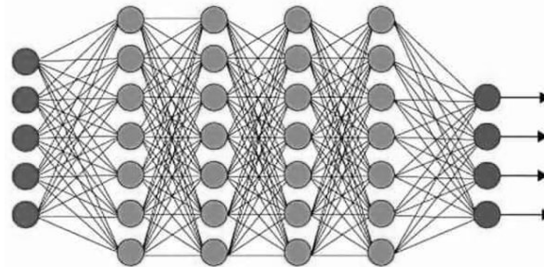
**Abbildung 3:** Handgeschriebene Zahl als Abbildung in  $28 \times 28$  Pixeln dargestellt, dabei ist die richtige Ausgabe die Zahl 5.

besitzt keine Fehlwerte oder verzerrende Bilder, die die KI beim Lernen stören kann. Auch ist die Auflösung in Ordnung und 70000 Bilder sind für kleine Aufgaben angemessen.

### 3.2 Künstliches neuronales Netzwerk

Im menschlichen Gehirn befinden sich großen Mengen von Neuronen, die miteinander interagieren. Das Neuron besteht aus drei relevante Komponente. Der erste Komponent besteht aus Empfängern, die Signale von anderen verbundenen Neuronen empfängt. Sie werden auch Dendriten genannt. Ein weiterer Komponent wären die Synapsen. Die Aufgabe ist es sogenannte Neurotransmitter an anderen verbundenen Neuronen weiterzuleiten, falls das Signal ankommen sollte. Damit das Signal ankommt, braucht es den dritten Komponent, zwar den Axon. Sie wird als Leitungsbahn verwendet, um Signale transportieren zu können und besteht aus einer lange dünne Röhre. Diese Röhre kann das Signal zu den Synapsen weiterleiten, unter Voraussetzung, dass der Schwellenwert erreicht ist. Fall er nicht erreicht werden sollte, wird das Signal abgebrochen. Der Schwellenwert wird erreicht, wenn genug Neurotransmitter vorhanden sind WAS ABSOLUT EINFACH DARGESTELLT IST UND EIGENTLICH VERBESSERT WERDEN MUSS, ABER ICH FRAGE ZUR SICHERHEIT NOCHMAL NACH. Sobald das Signal ankommt,

werden die Neurotransmitter ausgeschüttet und die Dendriten der anderen verbunden. Neuronen nehmen sie auf. Dadurch ändert sich die Struktur des Neurons und es kommt wieder zur Entscheidung, ob das Signal weitergeleitet werden soll oder abgebrochen wird. Es entsteht eine Kettenreaktion. Da das menschliche Gehirn imitiert werden soll, werden künstliche Neuronen erschaffen, die einen Wert von den vorherigen künstlichen Neuronen verarbeitet wurden. Durch die ganzen Verbindungen zwischen den Neuronen wird das auch als Netzwerk bezeichnet. Viele Eigenschaften der Neuronen werden mitgenommen, wie zum Beispiel die Dendriten, die Synapsen, sowie der Schwellenwert, in der Informatik auch Bias genannt. Der Bias spielt im Kapitel Klassifikationsebene eine Rolle. Unterschied ist aber, dass die künstlichen Neuronen deren Struktur nicht ändern können, dafür können ihnen aber Werte gegeben werden. Das bedeutet, dass das künstliche Neuron negative Werte annehmen kann, was bei den Neuronen nicht der Fall ist, da der Axon nur das Signal weiterleiten oder abbrechen kann. Sie können auch mit 0 und 1 bezeichnet werden. Das Netzwerk kann unterschiedliche Anordnungen von künstlichen Neuronen besitzen.



**Abbildung 4:** Künstliches Neuronales Netzwerk mit insgesamt 37 künstliche Neuronen in 6 Schichten verteilt.

Die geeignetste Anordnung nennt sich das konvolutionales neuronales Netzwerk (CNN), denn sie wird häufig für Bilderkennung verwendet. Grund dafür ist die Extrahierung der Merkmale des Bildes. Denn bevor das Bild an die künstlichen Neuronen weitergeleitet werden, werden unnötige Merkmale mithilfe von Konvolution aus dem Bild entfernt, sodass der Lernprozess der KI auf das Wichtigste begrenzt wird. Welche Merkmale eines Bildes gemeint ist, wird im Kapitel Feature Extraction verdeutlicht.

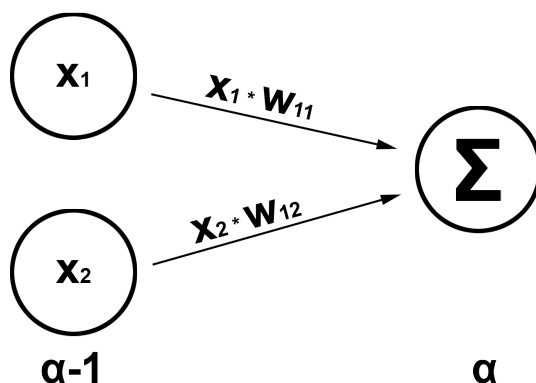
## 4 Konvolutionales neuronales Netzwerk

Das CNN kann in zwei Bereichen gegliedert werden. Im ersten Bereich werden Merkmale des Bildes extrahiert, wie zum Beispiel Ecken, Kurven, etc. Diese nennt sich Feature Extraction. Die Extrahierung wird durch 2D Convolution berechnet. Im zweiten Bereich werden die wichtigen Merkmalen aus der Feature Extraction in die sogenannte Klassifikationsebene weitergeleitet. Dort fängt der Lernprozess der KI an, die dann am Ende des Netzwerkes eine Klasse ausgibt.



## 4.1 Klassifikationsebene

In der Klassifikationsebene befindet sich das künstliches neuronales Netzwerk, die eine ähnliche Struktur wie bei Abbildung 4 hat. Diese Struktur nennt sich auch Vorwärtsgerichtetes Neuronales Netzwerk (FNN) und ist in mehreren Schichten aufgebaut. In Abbildung 4 wäre die Anzahl der Schichten  $A = 6$ . Die erste Schicht nennt sich Eingabeschicht, denn von dort kommen die Daten in die Neuronen rein, während bei der letzte Schicht die KI ein Ergebnis ausgibt. Diese Schicht heißt Ausgabeschicht. Die Schichten dazwischen werden ausgeblendete Schichten genannt. Die ausgeblendete Schichten sind wichtige Teile des Netzwerkes, denn durch diese werden Merkmale gesucht und analysiert. Das Netzwerk läuft also von links nach rechts durch. Das gibt nochmal eine deutlich stärkere Lerntiefe für die KI. Die einzelne Neuronen können einen Wert besitzen, sie wird als  $x_i^{(\alpha, \beta)}$  definiert, wobei  $\alpha \in \{1, \dots, A\}$  gilt. Der Index  $i$  beschreibt um welches Neuron in der Schicht  $\alpha$  es sich handelt. Der Wert  $\beta$  wird später verwendet. Für die Einheitlichkeit der Formeln steht in dem Fall  $\beta$  in der Variable. In der Klassifikationsebene bleibt  $\beta = 1$ . Dieser Wert lässt sich aus den ganzen Verbindungen der vorherigen Schicht berechnen, indem sie wie in Abbildung 5 addiert werden. Dabei werden die einzelne Werte  $x_j^{(\alpha-1, 1)}$  mit einem Gewicht  $w_{i,j}^{(\alpha, 1)}$  multipliziert, womit  $j \in \{1, \dots, N_{\alpha-1}\}$  gilt.  $N_\alpha$  beschreibt die Anzahl der Neuronen auf der Schicht  $\alpha$ . Allgemein gilt dann für die Berechnung



**Abbildung 5:** Die Verbindungen zwischen den Neuronen werden mit den Parameter  $w$  erweitert.

$$\bar{x}_i^{(\alpha, 1)} := \sum_{j=1}^{N_{\alpha-1}} w_{i,j}^{(\alpha, 1)} x_j^{(\alpha-1, 1)} + b_i^{(\alpha, 1)} \quad i \in \{1, \dots, N_\alpha\}, j \in \{1, \dots, N_{\alpha-1}\}. \quad (4.1)$$

Das  $b_i^{(\alpha, 1)}$  in der Gleichung steht für Bias und ist ein weiterer Parameter zum Einstellen wie das Gewicht. Beide spielen hier eine wichtige Rolle für das Weiterleiten der Neuronen. Das Gewicht entscheidet, wie wichtig der Wert vom Neuron ist. Je höher der Wert ist, desto entscheidender ist das Neuron. Wichtig ist anzumerken, dass die Gewichte und die Biases verstellbare Parameter sind, die für das Lernen der KI eine wichtige Rolle spielt. Damit die KI lernen kann, müssen die Parameter  $w$  und  $b$  optimal gesetzt werden, sodass die KI richtige Voraussagen treffen kann, um welche Zahl es sich handelt. Dafür wird die Fehlerrückführung verwendet. Erstmal werden nur die partielle Ableitungen berechnet.

Warum sie gebraucht werden wird im Kapitel Backpropagation genauer erläutert. So kann in der Gleichung 4.1 partiell nach drei Variablen der Funktion  $C(\vec{y}, \vec{x}^{(A,1)})$  abgeleitet werden:  $x_j^{(\alpha-1,1)}$ ,  $w_{i,j}^{(\alpha,1)}$  und  $b_i^{(\alpha,1)}$ . Die Funktion  $C(\vec{y}, \vec{x}^{(A,1)})$  wird in Kapitel Verlustfunktion genauer erläutert. Sie kommt in dem Fall direkt vor, da die partiellen Ableitungen davor für die einzelnen Kapiteln schon vorgeschrieben sind.??? Folgendes ergibt sich:

5

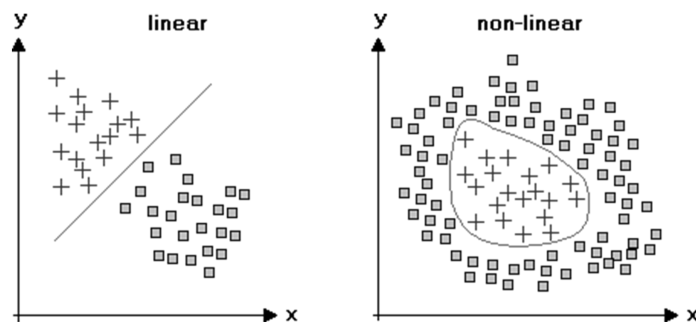
$$\frac{\partial C}{\partial b_i^{(\alpha,1)}} = \frac{\partial C}{\partial \bar{x}_i^{(\alpha,1)}} \quad (4.2)$$

$$\frac{\partial C}{\partial w_{i,j}^{(\alpha,1)}} = \frac{\partial C}{\partial \bar{x}_i^{(\alpha,1)}} x_j^{(\alpha-1,1)} \quad (4.3)$$

$$\frac{\partial C}{\partial x_j^{(\alpha-1,1)}} = \sum_{i=1}^{N_\alpha} \frac{\partial C}{\partial \bar{x}_i^{(\alpha,1)}} w_{i,j}^{(\alpha,1)} \quad (4.4)$$

Eine weitere Sache, die hinzugefügt werden muss, ist die Aktivierungsfunktion, denn sie gibt das Netzwerk eine Komplexität. Ohne diese Komplexität wäre das Netzwerk linear, was Nachteile mit sich bringt für die KI. In Abbildung 6 ist die Regression bei einer nicht lineare Struktur deutlich besser als dies einer lineare Struktur. Welche Aktivierungsfunktionen

10



**Abbildung 6:** Im ersten Graph ist sie gut mit der lineare Regressionen lösbar. Beim zweiten Graph ist es unmöglich eine genaue Regression zu bekommen mit einer lineare Regression. Dafür wird eine nicht lineare Regression verwendet.

verwendet werden kommt in Lektion AKTIVIERUNGSFUNKTION dran. So ergibt sich

$$x_i^{(\alpha,1)} := f^{(\alpha,1)}(\bar{x}_i^{(\alpha,1)}) = f^{(\alpha,1)}\left(\sum_{j=1}^{N_\alpha} w_{i,j}^{(\alpha,1)} x_j^{(\alpha-1,1)} + b_i^{(\alpha,1)}\right). \quad (4.5)$$

Nun kann die Schicht auch als einen Vektor angesehen werden, wobei die Neuronen die Elemente vom Vektor sind. Folglich kann das Netzwerk in Abbildung 4 in einem Matrixblock

umgeschrieben werden. So definiert sich

$$\vec{x}^{(\alpha,1)} := \begin{bmatrix} x_1^{(\alpha,1)} \\ x_2^{(\alpha,1)} \\ \vdots \\ x_{N_\alpha}^{(\alpha,1)} \end{bmatrix}. \quad (4.6)$$

Sobald die letzte Schicht erreicht ist, berechnet die Verlustfunktion Fehlermaße des Netzwerkes und gibt Aussage darüber, wie gut unsere KI ist. Mehr dazu im Kapitel Verlustfunktion.

#### 4.1.1 Aktivierungsfunktion

- 5 Mithilfe der Aktivierungsfunktion kann die KI eine nicht lineare Regression durchführen, denn aus der Gleichung 4.6 wird deutlich, dass alle Operationen linear sind, wenn nur der Parameter der Funktion  $f^{(\alpha,1)}(x)$  betrachtet wird. Für eine nicht lineare Regression sollte die Aktivierungsfunktion differenzierbar sein. Dafür gibt es verschiedene Formen von Aktivierungsfunktionen, die in der Praxis angewendet werden. Eine der meist verwendeten
- 10 Aktivierungsfunktionen ist der Rectified linear unit (ReLU):

$$f^{(\alpha,1)}(\bar{x}_i^{(\alpha,1)}) = \begin{cases} \bar{x}_i^{(\alpha,1)} & \text{falls } \bar{x}_i^{(\alpha,1)} > 0 \\ 0 & \text{falls } \bar{x}_i^{(\alpha,1)} \leq 0 \end{cases} \quad (4.7)$$

Die partielle Ableitung nach  $\bar{x}_i^{(\alpha,1)}$  zur Funktion  $C(\vec{y}, \vec{x}^{(A,1)})$  ist:

$$\frac{\partial C}{\partial \bar{x}_i^{(\alpha,1)}} = \begin{cases} \frac{\partial C}{\partial x_i^{(\alpha,1)}} & \text{falls } \bar{x}_i^{(\alpha,1)} > 0 \\ 0 & \text{falls } \bar{x}_i^{(\alpha,1)} \leq 0 \end{cases} \quad (4.8)$$

- Gründe für die Nutzung ist die Effizienz für linear gebaute Neuronale Netzwerke, sowie die Berechnung der Werte. Außerdem kann durch ReLU Neuronen dauerhaft deaktiviert werden, da alle Werte, die  $\leq 0$  sind, zu 0 gesetzt werden. Die Ableitung Eine weitere
- 15 Aktivierungsfunktion, die für die Facharbeit verwendet wird ist die Softmax Funktion:

$$f^{(\alpha,1)}(\bar{x}_i^{(\alpha,1)}) = \frac{e^{\bar{x}_i^{(\alpha,1)}}}{\sum_{j=1}^{N_\alpha} e^{\bar{x}_j^{(\alpha,1)}}} \quad (4.9)$$





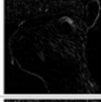
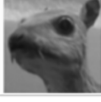

Die partielle Ableitung nach  $\bar{x}_i^{(\alpha,1)}$  zur Funktion  $C(\vec{y}, \vec{x}^{(A,1)})$  ist:

$$\frac{\partial C}{\partial \bar{x}_k^{(\alpha,1)}} = \frac{\partial C}{\partial x_i^{(\alpha,1)}} (\delta_{i,k} x_i^{(\alpha,1)} - x_i^{(\alpha,1)} x_k^{(\alpha,1)}) \quad (4.10)$$

- Meistens wird die Aktivierungsfunktion an der letzte Schicht verwendet, denn der Zweck dieser Aktivierungsfunktion ist es mithilfe des Vektors  $\vec{x}^{(A)}$  in einen neuen Vektor gleicher Dimension zu erstellen, deren Elemente zusammenaddiert 1 ergibt. Der Wertebereich der Funktion liegt bei  $[0; 1]$ . Dies kann dann in einer Wahrscheinlichkeitsverteilung dargestellt werden. Dies erleichtert die Dokumentierung der Ergebnisse der KI und eine Rückschlussführung, warum die KI wahrscheinlich diese Zahl nimmt, ist möglich.
- 20

## 4.2 Feature Extraction

Die Feature Extraction besteht aus mehreren Schichten von Operationen mit dem das eingebene Bild verarbeitet wird. Eine Hauptoperation ist der 2D Konvolution, denn diese Operation ist dafür zuständig, dass wichtige Merkmale extrahiert werden. Dafür werden verschiedene Filter angewendet, um verschiedene Arten von Merkmalen des Bilder zu extrahieren. Weitere wichtige Operationen ist der Max-Pooling und der Flattening. Auf-

Operation	Filter	Convolved Image			
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$		Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$		Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$		Gaussian blur (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$				

**Abbildung 7:** Hier sind verschiedene Extrahierungen zu erkennen mit verschiedenen Aufgaben. Das erste Bild mit der Operation Identity ist das Originalbild.

gabe von Max-Pooling ist es unnötige Merkmale zu ignorieren, indem nur die hellsten Pixel rausgefiltert werden. Die Dimension des Bildes halbiert sich. Flattening ist für die Übertragung auf die Klassifikationseben zuständig, indem die Matrix des Bildes in einen Vektor geflacht wird, sodass die KI optimal lernen kann. Das Bild kann in einer  $28 \times 28$  Matrix dargestellt werden

$$X^{(1,1)} := (x_{ij}) \quad i \in \{1, \dots, 28\}, j \in \{1, \dots, 28\}. \quad (4.11)$$

Der erste Index beschreibt auf welcher Schicht die Variable sich befindet und der zweite Index beschreibt auf welches Bild sie drauf bezieht, denn in einer Schicht können mehrere Bilder vorkommen. Dies nennt sich Tiefe. Da am Anfang aber nur ein Bild eingegeben wird, gilt  $N_{\beta}^{(1)} = 1$ . Für die nächsten Schichten gilt im Allgemeinen

$$X^{(\alpha,\beta)} := (x_{ij}) \quad i \in \{1, \dots, m_x^{(\alpha,\beta)}\}, j \in \{1, \dots, n_x^{(\alpha,\beta)}\}. \quad (4.12)$$

Die Werte  $m_x^{(\alpha,\beta)}$  und  $n_x^{(\alpha,\beta)}$  werden basiert auf die Operation berechnet, wobei  $\alpha \in \{1, \dots, A\}$  und  $\beta \in \{1^{(\alpha)}, \dots, N_{\beta}^{(\alpha)}\}$ . Dabei ist die Anzahl der Tiefe von der jeweilige Schicht abhängig.

#### 4.2.1 2D Konvolution

Mithilfe von 2D Konvolution können wichtige Merkmale herausgefiltert werden. Damit sie aber herausgefiltert werden kann, wird eine neue Matrix definiert, die auch als Kernel genannt wird

$$K^{(\alpha,\beta)} := (k_{i,j})^{(\alpha,\beta)} \quad i \in \{1; 2; \dots; m_k^{(\alpha,\beta)}\}, j \in \{1; 2; \dots; n_k^{(\alpha,\beta)}\} \quad (4.13)$$

- 5 Der Kernel kann nach der Abbildung 7 verschiedene Formen von Matrizen annehmen, dadurch entsteht durch die 2D Konvolution eine neue Matrix, in dem Fall ein neues Bild. Diese neue Matrix gilt für die nächste Schicht, weshalb die neue Matrix der Gleichung 4.12 entspricht, wobei  $m_x \leftarrow m_x - m_k + 1$  und  $n_x \leftarrow n_x - n_k + 1$ . Da sowohl die Eingabe des Bildes, sowie der Kernel beide Matrizen sind, kann mit der 2D Konvolution gerechnet werden. Sie ist folgendermaßen definiert

$$a_{i,j} = (X * K)_{i,j} := \sum_{m'=1}^{m_k} \sum_{n'=1}^{n_k} x_{i-m'+1, j-n'+1} k_{m',n'}. \quad (4.14)$$

Eine weitere Operation, die sehr wichtig für die 2D Konvolution ist die Kreuzkorrelation (Cross Correlation)

$$a_{i,j} = (X \star K)_{i,j} := \sum_{m'=1}^{m_k} \sum_{n'=1}^{n_k} x_{i+m'-1, j+n'-1} k_{m',n'}. \quad (4.15)$$

- 15 Aus (4.14) wird deutlich, dass Cross Correlation nichts anderes ist als eine 2D Konvolution, wo nur der Kernel um  $180^\circ$  rotiert ist. Für die Operation können auch Biases, sowie Aktivierungsfunktionen angewandt werden. Dies ähnelt der Struktur der FCNN. Werden diese Eigenschaften hinzugefügt, folgt daraus für die 2D Convolution:

$$x_{i,j}^{(\alpha,\beta)} = f^{(\alpha)} \left( \sum_{\beta'=1}^{N_\beta^{(\alpha-1)}} \sum_{m'=1}^{m_k} \sum_{n'=1}^{n_k} x_{i+m'-1, j+n'-1}^{(\alpha-1,\beta')} k_{m',n'}^{(\alpha,\beta')} + b_{i,j}^{(\alpha,\beta)} \right) \quad (4.16)$$

Für die Fehlerzurückführung (Backpropagation) werden die partielle Ableitungen nach  $k_{i,j}^{(\alpha,\beta)}$ , nach  $b_{i,j}^{(\alpha,\beta)}$  und nach  $x_{i,j}^{(\alpha-1,\beta)}$  zur Funktion  $C$  gebraucht:

$$\frac{\partial C}{\partial k_{i,j}^{(\alpha,\beta)}} = \sum_{i'=1}^{m_x^{(\alpha,\beta)}} \sum_{j'=1}^{n_x^{(\alpha,\beta)}} \frac{\partial C}{\partial \bar{x}_{i',j'}^{(\alpha,\beta)}} x_{i+i'-1, j+j'-1}^{(\alpha-1,\beta)} = (X^{(\alpha-1,\beta)} \star \frac{\partial C}{\partial \bar{X}^{(\alpha,\beta)}})_{i,j} \quad (4.17)$$

$$\frac{\partial C}{\partial b_{i,j}^{(\alpha,\beta)}} = \frac{\partial C}{\partial \bar{x}_{i,j}^{(\alpha,\beta)}} \quad (4.18)$$

20

$$\frac{\partial C}{\partial x_{i,j}^{(\alpha-1,\beta)}} = \sum_{i'=1}^{m_x^{(\alpha,\beta)}} \sum_{j'=1}^{n_x^{(\alpha,\beta)}} \frac{\partial C}{\partial \bar{x}_{i',j'}^{(\alpha,\beta)}} k_{i-i'+1, j-j'+1}^{(\alpha,\beta)} = (K^{(\alpha,\beta)} * \frac{\partial C}{\partial \bar{X}^{(\alpha,\beta)}})_{i,j} \quad (4.19)$$

## 4.2.2 Max-Pooling

## 4.2.3 Übergang von Feature Extraction zu Klassifikationsebene

Nachdem die Merkmale der Bilder durch 2D Konvolution und Max-Pooling extrahiert wurden, muss diese Matrix zur Klassifikationsebene übertragen werden. Da aber die Klassifikationsebene nur einen Vektor als Eingabe annimmt, muss die Matrix abgeflacht werden. Mithilfe des Vektoroperators  $vec()$  kann die Ausgangsmatrix der Feature Extraction in einen Vektor umgewandelt werden:

$$\vec{x}^{(\alpha,\beta)} = vec_{n_x^{(\alpha,\beta)}, m_x^{(\alpha,\beta)}}(X^{(\alpha-1,\beta)T}) \quad (4.20)$$

Dabei gibt die Indizes die die Dimension der Matrix an, die abgeflacht werden soll. Für die Backpropagation muss der Vektor aber wieder in einer Matrix mit der gleichen Dimension umgeformt werden. Dafür existiert eine Umkehrfunktion des Vektoroperators  $vec^{-1}()$  die diese Berechnung durchführen kann:

$$X^{(\alpha-1,\beta)} = vec_{n_x^{(\alpha,\beta)}, m_x^{(\alpha,\beta)}}^{-1}(\vec{x}^{(\alpha,\beta)})^T \quad (4.21)$$

Dieser Übergang wird auch als Flattening benannt.

## 4.3 Verlustfunktion

Um auszuwerten, ob eine KI gute oder schlechte Vorhersagen trifft, werden Verlustfunktionen verwendet. Verlustfunktionen sind Funktionen, die Abweichungen der vorhergesagte Zahl  $\vec{x}^{(A,1)}$  mit der Ausgangszahl  $\vec{y}$  berechnet wird. Je größer die Abweichung ist, desto unpräziser trifft die KI Vorhersagen. Für die Optimierung bedeutet dies, dass der Wert der Verlustfunktion kleinstmöglich wird, sodass die KI präzise Vorhersagen treffen kann. Durch die Änderung der Parameter kann die Verlustfunktion minimiert werden, denn die Verlustfunktion sind von allen Parameter  $w, b$  und  $k$  in der Variable  $\vec{x}^{(A,1)}$  abhängig. Es gilt:

$$C(\vec{y}, \vec{x}^{(A,1)}) \equiv C(\vec{y}, \vec{w}), \vec{w} \in \text{AlleParameterwundk} \quad (4.22)$$

Doch wie die Parameter richtig gesetzt werden, kommt in Kapitel Optimierung. Für die Facharbeit wird die Kreuzentropie (Cross Entropy) verwendet. Sie ist folgendermaßen definiert:

$$C(\vec{y}, \vec{x}^{(A,1)}) = - \sum_{i=1}^{N_A} y_i \ln(x_i^{(A,1)}) \quad (4.23)$$

Die folgende partielle Ableitung nach  $x_i^{(A,1)}$  ist:

$$\frac{\partial C}{\partial x_i^{(A,1)}} = - \frac{y_i}{x_i^{(A,1)}} \quad (4.24)$$

Voraussetzung der Cross Entropy ist, dass  $\sum_{i=1}^{N_A} x_i^{(A,1)} = 1$  ergibt. Dies wird durch die Softmax Funktion in der Gleichung 4.9 erfüllt. Vorteil dieser Verlustfunktion ist zum einen, dass sie die Abweichungen präzise für die Wahrscheinlichkeitsverteilung berechnen kann.

Außerdem funktioniert sie gut für Klassifikationen, denn die KI der Facharbeit basiert sich auf einer Klassifizierung.

## 4.4 Optimierung

Mithilfe der Optimierung kann die Abweichung der Verlustfunktion verringert werden, indem die Parameter optimal gewählt werden. In der Schulmathematik wird die erste Ableitung der Verlustfunktion genommen und auf 0 gesetzt. Problem bei der Rechnung ist nun, dass in der Praxis die Anzahl der Parameter der KI sehr hoch sein kann. Es wird von mehrere tausende Paramater gesprochen. So ist die Berechnung der Parameter uneffizient und könnte für einen Rechner sehr lange dauern, bis sie perfekte Parameter für die Verlustfunktion findet. Deswegen werden auf Algorithmen zurückgegriffen, die diese Berechnungen deutlich zeitsparender durchführen können. Einer dieser Algorithmen nennt sich Gradientenverfahren. Das Gradientenverfahren besagt die höchste Steigung, die zurückgelegt werden kann, sie wird mit einen  $\nabla C$  definiert:

$$\nabla C = \begin{bmatrix} \frac{\partial C}{\partial w_1} \\ \frac{\partial C}{\partial w_2} \\ \dots \\ \frac{\partial C}{\partial w_n} \end{bmatrix} \quad (4.25)$$

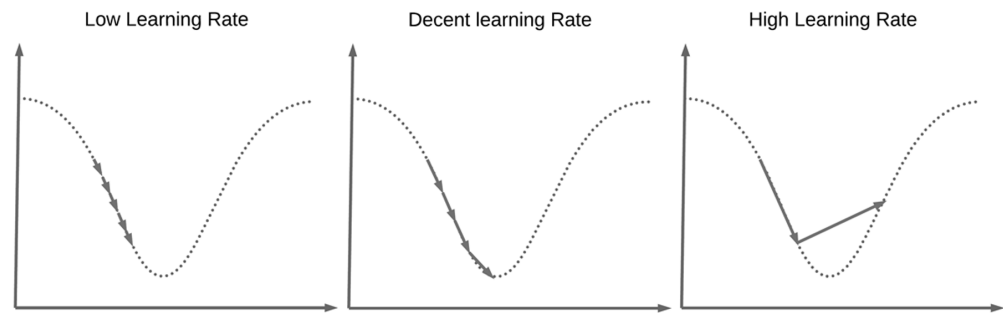
Da aber das Minimum der Funktion gesucht wird, wird die tiefste Steigung benötigt, indem  $\nabla C$  negativ wird, also  $-\nabla C$ . Aus der Gleichung 4.25 können die Elemente für die Annäherung der optimalen Parameter genutzt werden. So kann durch jede Iteration der Parameter aktualisiert werden:

$$w_i \leftarrow w_i - \Delta w_i \equiv w_i - \eta \frac{\partial C}{\partial w_i} \quad (4.26)$$

Hier wurde ein neuer Parameter  $\eta$  hinzugefügt, diese nennt sich Lernrate. Sie gibt an, wie schnell sie sich an das Minimum antasten soll. Es ist wichtig einen guten Wert für die Lernrate zu setzen, denn falls die Lernrate zu groß wird, kann sie vielleicht über das Minimum vorbeispringen. Wenn aber die Lernrate zu klein ist, dann dauert auch das Lernen deutlich länger, da die Optimierung langsamer am Minimum antastet. Trotz des Algorithmus kann die Berechnung immer noch nicht effizient genug sein, denn wenn als Beispiel der Datensatz von MNIST verwendet wird, dann müssen für alle 70000 Bilder die Gradienten berechnet werden. Diese Zeit kann erspart werden, indem nicht für alle Bilder den Gradienten berechnen werden müssen. Als Beispiel soll nur der Gradient berechnet werden bei jedes 100. Bild. So werden die Gradienten statt für 70000 Bilder, nur 700 Bilder berechnet. Die Antastung des Minimums ist nicht sehr präzise, aber dennoch aussagekräftig und zeitsparend. Die Optimierung nennt sich Stochastisches Gradientenverfahren.

### 4.4.1 Backpropagation

Um die Parameter zu aktualisieren muss der Wert  $\frac{\partial C}{\partial w_i}$  berechnet werden. Mithilfe der Backpropagation kann dieser Wert berechnet werden, indem rückwärts gegangen wird vom



**Abbildung 8:** Sehr gut erkennbar sind die Iterationen: Wenn die Lernrate richtig gewählt ist, dann erreicht sie das Minimum schneller.

FCNN. Als Beispiel wird ein FCNN dargestellt mit drei Schichten. Für die Verlustfunktion soll der Cross Entropy verwendet werden. Ziel ist es den Parameter  $w_{2,1}^{(3,1)}$  zu aktualisieren, weshalb  $\frac{\partial C}{\partial w_{2,3}^{(2,1)}}$  gesucht ist. ABBILDUNG DES NETZWERKES. Dafür wird angeschaut, wie die Verlustfunktion  $C$  von  $w_{2,1}^{(3,1)}$  abhängt. Da die Werte bei jeder Schicht weitergegeben wurde, kann die Kettenregel angewendet werden:

$$\frac{\partial C}{\partial w_{2,3}^{(2,1)}} = \frac{\partial C}{\partial x_2^{(3,1)}} \frac{\partial x_2^{(3,1)}}{\partial \bar{x}_2^{(3,1)}} \frac{\partial \bar{x}_2^{(3,1)}}{\partial x_2^{(2,1)}} \frac{\partial x_2^{(2,1)}}{\partial \bar{x}_2^{(2,1)}} \frac{\partial \bar{x}_2^{(2,1)}}{\partial w_{2,3}^{(2,1)}} \quad (4.27)$$

Dies kann abgekürzt werden mit:

$$\frac{\partial C}{\partial w_{2,3}^{(2,1)}} = \frac{\partial C}{\partial \bar{x}_2^{(2,1)}} \frac{\partial \bar{x}_2^{(2,1)}}{\partial w_{2,3}^{(2,1)}} \stackrel{(4.3)}{=} \frac{\partial C}{\partial \bar{x}_2^{(2,1)}} x_3^{(1,1)} \quad (4.28)$$

Die Gleichung 4.28 ist für die Umsetzung deutlich angenehmer, da der Wert  $\frac{\partial C}{\partial \bar{x}_2^{(2,1)}}$  von der nächsten Schicht schon berechnet wurde. Dies bedeutet, dass nur der Wert  $\frac{\partial \bar{x}_2^{(2,1)}}{\partial w_{2,3}^{(2,1)}}$  berechnet werden muss.

## 5 Umsetzung

Zuerst braucht es eine Struktur für das CNN. Dafür wird eine Klasse Layer erstellt, die als Objekt angesehen werden soll. Dabei soll die Layer folgende Eigenschaften besitzen: Eine Eingabe und eine Ausgabe. Außerdem wird eine neue Klasse DenseLayer erstellt, die diese Eigenschaften der Klasse Layer erben. Der DenseLayer soll die Schichten im FCNN darstellen und die Parameter  $w$  und  $b$  als Variablen annehmen können. Weitere Klassen, wie Schichten für die Feature Extraction oder Aktivierungsfunktion und Verlustfunktion



müssen erstellt werden. Vorteil an der Programmierung sind die Open Source Pakete, die das Internet anbietet. So müssen nicht die einzelne Elemente der Vektoren ausgerechnet werden, sondern kann alle Elemente des Vektors auf einmal berechnen. Auch muss der Lernprozess programmiert werden, indem das Netzwerk durchläuft. Die Anzahl der Durchläufe des Netzwerkes kann bestimmt werden. Diese nennt sich auch Epochen. Am Ende werden verschiedene Diagramme generiert, die für die Analyse der KI nützlich sein könnten. Mehr dazu im Kapitel Auswertung.

## 5.1 Werkzeuge

Für die Umsetzung wird die Programmiersprache Python verwendet, denn Python wird in der Praxis größtenteils für Data Science verwendet. Python wird dafür verwendet, weil sie sehr gute Open Source Pakete besitzt. Dabei spielen zwei Pakete eine wichtige Rolle für die Umsetzung: NumPy und Matplotlib. NumPy ist ein Paket für Vektor- und Matrizenberechnungen und berechnet die Werte effizienter als Python selbst. Außerdem bietet sie sehr viele Operationen an, die für die KI notwendig sind. Der Matplotlib ist für die Visualisierung der Bilder, sowie die Filter von der Feature Extraction zuständig. Sie hilft den Nutzer die KI besser zu verstehen und nachzuvollziehen.

## 5.2 Trainingsphase

- Daten müssen in 3 Sets aufgeteilt werden: Training, Test und Validation - Irgendwo muss noch Overfitting und Underfitting rein, das mit Validation vermieden wird

## 5.3 Auswertung

- Seite 14 - Verschiedene Diagramme werden ausgewertet - Z.B. Lernverlauf - Accuracy anschauen - Wert der Verlustfunktion im Verlauf anschauen

### 5.3.1 Trainingsdaten

### 5.3.2 Validationsdaten

### 5.3.3 Testdaten

## 6 Schlussfolgerung

- Seite 19 Paar Formeln, wobei gilt: Lineare Funktion zwischen zwei Neuronenschichten:

$$\vec{x}^{(\alpha)} = f^{(\alpha)}(W^{(\alpha)}\vec{x}^{(\alpha-1)} + \vec{b}^{(\alpha)})$$

## Literaturverzeichnis

- [1] Europäisches Parlament: Was ist künstliche Intelligenz und wie wird sie genutzt?. 2021. Stand: 24.11.2022. <https://www.europarl.europa.eu/news/de/headlines/society/20200827STO85804/was-ist-kunstliche-intelligenz-und-wie-wird-sie-genutzt> [1]
- [2] Uni Oldenburg: Schwache KI und Starke KI. 2008/2009. Stand: 24.11.2022. [http://www.informatik.uni-oldenburg.de/~iug08/ki/Grundlagen\\_Starke\\_KI\\_vs.\\_Schwache\\_KI.html](http://www.informatik.uni-oldenburg.de/~iug08/ki/Grundlagen_Starke_KI_vs._Schwache_KI.html) [1]
- [3] Yann LeCun, Corinna Cortes, Christopher J.C. Burges: THE MNIST DATABASE of handwritten digits. Stand: 24.11.2022. <http://yann.lecun.com/exdb/mnist/>
- [4] Bundesministerium für Wirtschaft und Energie: Zur Diskussion der Effekte Künstlicher Intelligenz in der wirtschaftswissenschaftlichen Literatur. 2018. Stand: 24.11.2022. [https://www.bmwk.de/Redaktion/DE/Downloads/Diskussionspapiere/20190205-diskussionspapier-effekte-kuenstlicher-intelligenz-in-der-wirtschaftswissenschaftlichen-literatur.pdf?\\_\\_blob=publicationFile&v=6](https://www.bmwk.de/Redaktion/DE/Downloads/Diskussionspapiere/20190205-diskussionspapier-effekte-kuenstlicher-intelligenz-in-der-wirtschaftswissenschaftlichen-literatur.pdf?__blob=publicationFile&v=6)
- [5] Michael Nielsen: Neural Network and Deep Learning. 2019. Stand: 24.11.2022. <http://neuralnetworksanddeeplearning.com/>
- [6] My Great Learning: Types of Neural Networks and Definition of Neural Network. 2022. Stand: 24.11.2022. <https://www.mygreatlearning.com/blog/types-of-neural-networks/>
- [7] Chi Nhan Nguyen, Oliver Zeigermann: Machine Learning kurz & gut O'Reillys Taschenbibliothek. 2. Auflage. 2021
- [8] Musstafa: Optimizers in Deep Learning. 2021. Stand 24.11.2022. <https://medium.com/mlearning-ai/optimizers-in-deep-learning-7bf81fed78a0>
- [9] Bui Anh Minh Leon Phan: handwrittenDigits. 2022. Stand: 24.11.2022. <https://github.com/xXChezyXx/handwrittenDigits>
- [10] Bundeswettbewerb KI: Künstliche Intelligenz. 2022. Stand: 24.11.2022. <https://www.bw-ki.de/>

## **Schülererklärung**

Hiermit erkläre ich, dass ich die vorliegende Seminarfacharbeit selbstständig angefertigt, keine anderen als die angegebenen Hilfsmittel benutzt und die Stellen der Seminarfacharbeit, die im Wortlaut oder im wesentlichen Inhalt aus anderen Werken entnommen wurden, mit genauer Quellenangabe kenntlich gemacht habe.