

Arbeit zur Erlangung des akademischen Grades
Bachelor of Science

Lösungen inverser Probleme

Entfaltung von Energiespektren mit neuronalen Netzen in DSEA

Samuel Haefs
geboren in Herdecke

2022

Lehrstuhl für Experimentelle Physik V
Fakultät Physik
Technische Universität Dortmund

Erstgutachter: Prof. Dr. Dr. Rhode
Zweitgutachter: Prof. Dr. Albrecht
Abgabedatum: 30. August 2022

Kurzfassung

Physiker und Physikerinnen werden ständig mit inversen Problemen konfrontiert. Über die Zeit wurden verschiedene Methoden zur Lösung entwickelt. Der **Dortmund Spectrum Estimation Algorithm** (DSEA) fasst das zugrundeliegende Problem als Klassifikationsaufgabe auf. Dabei können verschiedene Klassifikationsalgorithmen verwendet werden. Der Einsatz von neuronalen Netzen als Klassifizierer in DSEA ist ein neuer Ansatz und wird auf IceCube-Neutrino Simulationen untersucht. Neuronale Netze in DSEA bieten die Möglichkeit den Trainingsprozess weiter fortzuführen, statt einen Klassifizierer in jeder Iteration neu zuinitialisieren. Systematisch werden die Hyperparameter des neuronalen Netzes und von DSEA untersucht. Ein häufiges Problem neuronaler Netze ist die Überanpassung an die Trainingsdaten. Es wird geprüft, ob die Modelle eine unabhängige Vorhersage treffen.

Abstract

Physicists are frequently confronted with inverse problems. Over time, various methods have been developed to solve them. The **Dortmund Spectrum Estimation Algorithm** (DSEA) treats the fundamental problem as a classification task. Various classification algorithms can be used. The use of neural networks as classifier in DSEA is a new approach and is explored on IceCube neutrino simulations. Neural networks in DSEA offer the possibility to continue the training process, instead of reinitializing the classifier in each iteration. Systematically, the hyperparameters of the neural network and DSEA are investigated. A common problem of neural networks is overfitting to the training data. The models are tested for unbiased predictions.

Inhaltsverzeichnis

1	Einleitung	1
2	Physikalische Grundlagen	2
2.1	IceCube-Neutrino-Observatorium	2
2.2	Neutrino-Astronomie	3
3	Entfaltung	5
3.1	Inverse Probleme	5
3.2	Lösungen mit DSEA	6
3.3	Neuronale Netze als Klassifikationsalgorithmus	7
4	Verwendete Methoden	9
4.1	Chi-Quadrat-Distanz	9
4.2	Bootstrapping: Abschätzung der Unsicherheiten	9
5	Testen der Hyperparameter in DSEA und neuronalen Netzen	11
5.1	Datensatz	11
5.2	Entfaltung als Klassifikationsproblem	12
5.3	Entfaltung mit DSEA	17
5.4	Abhängigkeit der Entfaltung vom Trainingsspektrum	23
6	Fazit und Ausblick	25
A	Monte-Carlo Sample	26
B	Entfaltung als Klassifikationsproblem	27
C	Entfaltung mit DSEA	31
D	Abhängigkeit der Entfaltung vom Trainingsspektrum	41
	Abbildungsverzeichnis	44
	Literatur	45
	Danksagung	48

1 Einleitung

Die Astroteilchenphysik untersucht aller Art Teilchen, die von extraterrestrischen Quellen ausgesandt wurden. Ziel ist die Bestimmung der Position und der Eigenschaften von Himmelskörpern über die ankommenden Botenteilchen. Die Botenteilchen setzen sich aus geladenen Teilchen (kosmische Strahlung), Photonen γ und Neutrinos ν zusammen.[8]

IceCube ist ein Detektor am geographischen Südpol und ist dafür ausgelegt, hoch-energetische Neutrinos nachzuweisen. Zum einen ist der Ort der Neutrinoquelle von großer Interesse. Aktive Galaxiekern, schwarze Löcher oder auch Neutronensterne können mögliche Quellen der Neutrinos sein und sind für Physiker*innen von großem Interesse. Eine weitere wichtige Größe stellt das Energiespektrum der Neutrinos dar. Die Neutrinoenergie ist nicht direkt messbar. Stattdessen werden die Zerfallsprodukte der Neutrinos (Elektronen, Myonen, Tau-Leptonen) gemessen. Das zu lösende Problem ist die Rekonstruktion der Neutrinoenergie aus den gemessenen Größen. Es handelt sich um ein inverses Problem, welches grundsätzlich schlecht konditioniert ist. Die Rekonstruktion der Energie wird als Entfaltung bezeichnet und benötigt spezielle Ansätze.

Der **D**ortmund **S**pectrum **E**stimation **A**lgorithm (DSEA) [20] betrachtet das Problem als Klassifikationsproblem. Unter Verwendung von Methoden des maschinellen Lernens kann eine Entfaltung mit DSEA durchgeführt werden.

In dieser Arbeit wird zum ersten mal die Verwendung von neuronalen Netzen als Klassifizierer in DSEA untersucht. Dies bietet neue Möglichkeiten. Die bisher verwendeten Klassifizierer wurden in jeder DSEA-Iteration neu initialisiert. Neuronale Netze basieren auf einer Verlustfunktion. Es ist daher möglich den Trainingsprozess eines Modells fortzuführen, statt in jeder DSEA-Iteration von neuem zu beginnen. Die Struktur eines neuronalen Netzes kann an das zugrundeliegende Problem angepasst werden. Es gibt eine große Auswahl an Ebenen, Verlustfunktionen und Aktivierungsfunktionen. Durch Anpassung der Hyperparameter kann die Lösung optimiert werden.

2 Physikalische Grundlagen

In diesem Kapitel wird der physikalische Hintergrund zu dieser Arbeit behandelt. Zuerst wird auf den Aufbau des IceCube-Neutrino-Observatoriums eingegangen und dabei die Funktionsweise des Detektors erläutert. Anschließend werden kurz wichtige Aspekte der Neutrinophysik aufgeführt.

2.1 IceCube-Neutrino-Observatorium

IceCube ist ein Hochenergie-Neutrino-Observatorium am geographischen Südpol. Der Detektor befindet sich in einer Tiefe von 1450 m bis 2450 m im antarktischen Eis und deckt ein Volumen von 1 km^3 ab.[8] An 86 Strings sind in regelmäßigen Abständen DOMs (Digital Optical Module) angebracht. Der Detektoraufbau ist schematisch in der Abbildung 2.1 dargestellt.

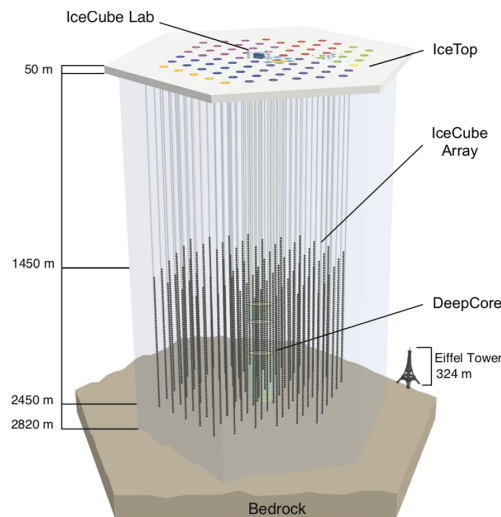


Abbildung 2.1: Schematische Darstellung des sich im antarktischen Eis befindenden IceCube-Detektors und des IceTop Arrays. IceCube liegt 1450 m bis 2450 m unter dem Eis und beinhaltet den Teildetektor DeepCore. Jeder Punkt stellt einen DOM und jeder Kreis eine IceTop-Station auf der Eisoberfläche dar [2].

Ein großer Teil des DOMs besteht aus einem Photomultiplier der Licht im Wellenlängenbereich 300 nm bis 650 nm registriert. Hochenergetische Neutrinos wechselwirken mit dem Eis und zerfallen in Leptonen (Elektron, Myon, Tau-Lepton). Aufgrund der Energieerhaltung besitzen die Leptonen eine hohe kinetische Energie. Sie bewegen sich mit einer Geschwindigkeit v die größer als die Lichtgeschwindigkeit im Medium c_m ist durch das Eis. Dabei wird kegelförmig Cherenkov-Strahlung[11] im Winkel θ zur Teilchenbahn nach

$$\cos(\theta) = \frac{c_m}{v}$$

emittiert. Treffen die Photonen auf die Photomultiplier wird eine Ladung erzeugt, diese wird verstärkt und digitalisiert. Das Signal wird von den DOMs an die nahliegende Amundsen-Scott Südpolstation übertragen.

Die durch das Licht erzeugte Signatur im Detektor, gibt Kenntniss über die Energie und Herkunft des Neutrinos. IceCube ist darauf ausgelegt Neutrinos mit Energien im TeV-Bereich[2] zu detektieren.

2.2 Neutrino-Astronomie

Neutrinos sind aufgrund ihrer Eigenschaften optimale Botenteilchen. Sie sind neutral geladen und wechselwirken daher nicht mit elektromagnetischen Feldern. Aufgrund der sehr geringen Masse sind auch gravitative Effekte nicht von Bedeutung.

Die geringe Wechselwirkungswahrscheinlichkeit der Neutrinos hat den Vorteil, dass diese unabgelenkt von weitentfernten kosmischen Quellen auf der Erde ankommen. Sie tragen Informationen der ursprünglichen Quelle in sich. Zum einen lässt die Neutrinoenergie Rückschlüsse auf die Art der Quelle zu. Zum anderen kann über die Rekonstruktion der NeutrinoBahn die Quelle lokalisiert werden.

Aus dem gleichen Grund gestaltet sich die Detektion der Neutrinos als große Herausforderung. Um Neutrinos zu detektieren werden große Detektorvolumen benötigt. IceCube deckt den Hochenergiebereich ab, wobei bis heute auch die höchsten Energien ($\gtrsim 100$ PeV) verborgen bleiben.

Der Neutrinofluss der energiereichsten Neutrinos ist so gering, dass zum Nachweis ein Detektor notwendig ist, der 1 bis 3 Größenordnungen größer als IceCube ist. Wie in Abbildung 2.2 zu erkennen ist, handelt es sich bei den in IceCube detektierten Neutrinos um atmosphärische Neutrinos und Neutrinos aus aktiven Galaxiekernen (AGN).

Atmosphärische Neutrinos entstehen bei der Kollision hochenergetischer Teilchen mit der Erdatmosphäre. Kosmische Strahlung besteht überwiegend aus geladenen Kernen. Diese wechselwirken mit den Atomkernen in der Atmosphäre und zerfallen über die schwache Wechselwirkung. Es wird ein Neutrino frei.

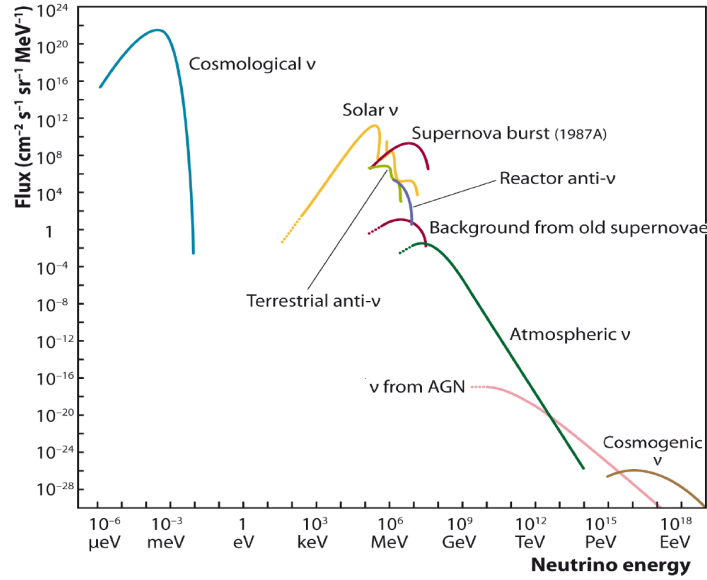


Abbildung 2.2: Neutrinofluss in Abhängigkeit der Energie für verschiedene Neutrinoquellen [22].

Ein Ziel von IceCube ist die Gewinnung neuer Erkenntnisse über die Quellen kosmischer Strahlung, in denen auch hochenergetische Neutrinos erzeugt werden.

Es existieren drei Generationen (Flavor) von Neutrinos, das Elektron-Neutrino ν_e , das Myon-Neutrino ν_μ und das Tau-Neutrino ν_τ . Aus Modellen, die verschiedene Neutrinoquellen betrachten geht hervor, dass Neutrinos im Verhältnis $\nu_e:\nu_\mu:\nu_\tau = 1:2:0$ entstehen. Auf der Erde werden alle Neutrinos gleichermaßen gemessen: $\nu_e:\nu_\mu:\nu_\tau = 1:1:1$. Grund sind die Neutrino-Oszillationen[3, 25], die einen Wechsel zwischen Generationen ohne weitere Wechselwirkung erlauben.

Die drei Neutrinoarten erzeugen im Detektor unterschiedliche Signaturen und ermöglichen eine Bestimmung des Flavors. Eine **Kaskade**[1] wird beobachtet, wenn bei der Wechselwirkung eines ν_e ein e^\pm entsteht. Aufgrund von Bremsstrahlungs- und Paarproduktionsprozessen nimmt die Energie ab. Am Ort der Wechselwirkung befindet sich das Energiemaximum.

Reagiert ein ν_μ mit dem Eis, so entsteht als sekundäres Teilchen ein Myon μ . Dieses propagiert durch das Eis und verliert Energie in Form von Ionisation, Paarerzeugung, Bremsstrahlung und photonuklearen Wechselwirkungen. Die Signatur entspricht einer **Spur** mit abnehmender Energie.

Der **Double Bang** („Doppel-Knall“) wird durch ein ν_τ hervorgerufen. Bei der Wechselwirkung entsteht ein Tauon τ und es wird eine Kaskade erzeugt. Der Zerfall des Tauons, nach der mittleren Lebensdauer von $(290,3 \pm 0,5) \cdot 10^{-15} \text{ s}$ [26] führt zu einer weiteren Kaskade. So ein Ereignis wäre ein klarer Nachweis für ein ν_τ .

3 Entfaltung

Vorerst wird das zugrundeliegende Problem in Abschnitt 3.1 aufgeführt und die Entfaltung diskutiert. Im Anschluss wird in Abschnitt 3.2 auf den Lösungsansatz mit DSEA eingegangen und der Algorithmus erläutert. Das letzte Kapitel Abschnitt 3.3 befasst sich mit neuronalen Netzen und wie sie als Klassifikationsalgorithmus verwendet werden können.

3.1 Inverse Probleme

Inverse Probleme treten in der Physik überall da auf, wo die gesuchte physikalische Größe nicht *direkt* messbar ist. Die Neutrinoenergie ist nur *indirekt* messbar. Ziel ist die Rekonstruktion der gesuchten Größe, also der Neutrinoenergie aus der Messgröße. In IceCube wird das Cherenkov-Licht der Zerfallsprodukte gemessen, siehe dazu Abschnitt 2.1.

Mathematisch ist das Problem über die Fredholm-Integralgleichung[23]

$$\int_{\Omega} A(x, y) f(x) dx + b(y) = g(y) \quad (3.1)$$

definiert. Die Verteilung des wahren Parameters x wird durch $f(x)$ und die Verteilung der Messdaten durch $g(x)$ dargestellt. Der gesamte Messprozess und die ablaufenden physikalischen Prozesse werden durch die Antwortfunktion $A(x, y)$ beschrieben. Üblicherweise wird der Messprozess durch einen Untergrund $b(y)$ beeinflusst. Dieser ist nicht von Interesse und kann mit Signal-Untergrund-Trennverfahren separiert werden.

Im diskreten Fall werden die kontinuierlichen Verteilungen $f(x), g(y)$ zu **Vektoren** und $A(x, y)$ zu einer Antwort**matrix**. Ohne Untergrund vereinfacht sich die Gleichung zu

$$g_i = \sum_{j=1}^n A_{ij} f_j. \quad (3.2)$$

Die Rekonstruktion der Größe f wird als Entfaltung bezeichnet. Der naive Ansatz über Bildung der Inverse von A führt häufig zu unbrauchbaren Lösungen. Kleine anfängliche Störungen verstärken sich und man erhält eine stark oszillierende Lösung. Das Problem ist schlecht konditioniert und benötigt spezielle Ansätze.

3.2 Lösungen mit DSEA

Der **D**ortmund **S**pectrum **E**stimation **A**lgorithm (DSEA)[20, 5, 15] betrachtet die Entfaltung als Klassifikationsproblem. Vorab wird die Zielvariable $f(x)$ in n Intervalle unterteilt:

$$f(x) \rightarrow \vec{f}: \quad f_j = \int_{y_{j-1}}^{y_j} f(x) dx, \quad j \in [1, n] \quad (3.3)$$

Jedes Intervall bzw. jeder Bin repräsentiert den in Gleichung 3.3 definierten Energiebereich. Die Bins stellen in DSEA die Zielklassen für eine Klassifikation dar.

DSEA ist ein iteratives Verfahren. Im ersten Schritt wird ein Klassifikationsalgorithmus auf dem Datensatz (X, W, F) trainiert. Dabei steht X für die Features, die mit den Probengewichten W gewichtet werden und F für das Label. DSEA geht von Daten mit gleichverteilten Klassen aus, d.h. die Gewichte W werden dementsprechend initialisiert. Dies hat den Vorteil, dass kein Prior angenommen werden muss und somit das Modell unabhängig von der Klassenverteilung im Trainingsdatensatz ist.

Voraussetzung an den Klassifizierer ist, dass dieser eine Konfidenz c_{ij} ausgibt, die die Zugehörigkeit des i -ten Events zum j -ten Bin beschreibt. Die Konfidenz wird als Wahrscheinlichkeit interpretiert.

Anschließend wird im 2. Schritt mit dem trainierten Klassifizierer/Modell eine Vorhersage für die zu entfaltenden Daten getroffen. Über die für jedes Event i und Bin j erhaltende Konfidenz wird das Spektrum

$$\hat{f}_j = \sum_{i=1}^n c_{ij} \quad (3.4)$$

rekonstruiert, wobei n die Anzahl der Datenpunkte angibt. Die Konfidenzen werden für jeden Bin j über alle Events aufsummiert und man erhält eine Vorhersage für den j -ten Bin des zu entfaltenden Spektrums \hat{f}_j .

Im letzten (3.) Schritt werden die Probengewichte w_i

$$w_i = \frac{f_{l_i}}{n} \quad (3.5)$$

über das rekonstruierte Spektrum aktualisiert, wobei l_i dem wahren Label des i -ten Events entspricht. Die Gewichte entsprechen dem normierten Spektrum.

Die erste DSEA-Iteration ist hiermit vollendet. Anschließend wird wieder Schritt 1 ausgeführt und der Klassifizierer mit angepassten Probengewichten trainiert. Es werden so viele Iterationen ausgeführt bis Konvergenz eintritt.

3.3 Neuronale Netze als Klassifikationsalgorithmus

Grundsätzlich kann jeder Klassifizierer verwendet werden, der für jede Zielklasse eine Konfidenz c_{ij} ausgibt. Beliebte Klassifikationsalgorithmen in DSEA sind Naive Bayes oder auch der Random Forest. In dieser Arbeit werden neuronale Netze (NN) als Klassifizier in DSEA untersucht. Das NN in Abbildung 3.1 beinhaltet

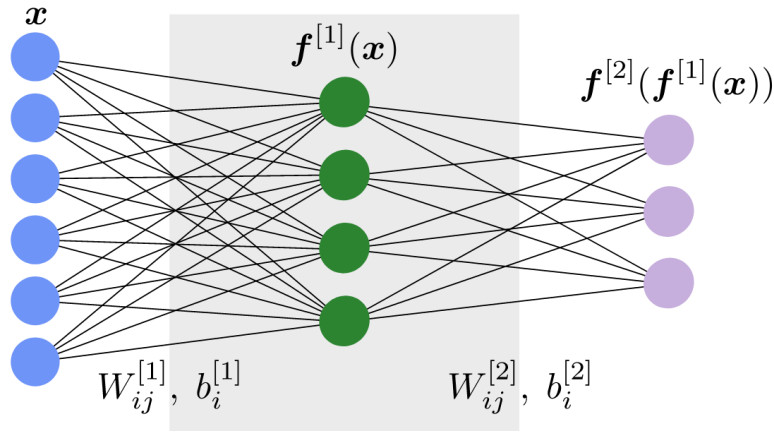


Abbildung 3.1: Darstellung der Struktur eines einfachen neuronalen Netzes. Die blauen Kreise stellen die Neuronen der Eingansebene, die Grünen der tiefen Ebene und die Blauen der Ausgangsebene dar. Jedes Neuron besitzt zwei Parameter, eine Gewichtsmatrix \mathbf{W}_{ij} und einen Biasvektor \vec{b}_i . Das Resultat für jede Ebene wird durch $f(x)$ beschrieben [17].

drei Ebenen, wobei jede Ebene eine bestimmte Anzahl an Neuronen enthält. Die Eingangsebene (Blau) besteht aus so vielen Neuronen, wie Features verwendet werden. Jedes Neuron erhält also als Eingang ein Feature. Es können beliebig viele tiefe Ebenen (Grün) verwendet werden. Die Verbindungen zwischen zwei Ebenen wird durch die Gewichtsmatrix \mathbf{W} , den Biasvektor \vec{b} und einer Aktivierungsfunktion $g(\vec{x})$ nach

$$g(f(\vec{x})) = g(\mathbf{W}\vec{x} + \vec{b})$$

beschrieben, wobei \vec{x} den Ausgang aus der vorherigen Ebene beschreibt. Es gibt eine große Auswahl an Aktivierungsfunktionen[18]. Große NN mit vielen Parametern haben den Nachteil sich stark an die Daten anzupassen (Bias). Bei dem in dieser Arbeit verwendeten NN handelt es sich um ein tiefes neuronales Netz mit etwa 60 000 Parametern. Die Struktur ist in Abbildung 3.2 graphisch dargestellt. Für quantitative Angaben der Ein- und Ausgangsdimensionen siehe Tabelle B.1. Der einfache Aufbau soll zu einem geringen Bias im Modell führen.

3 Entfaltung

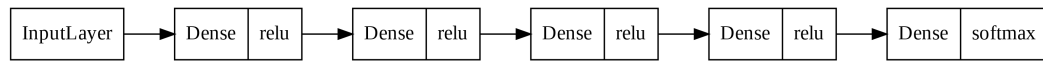


Abbildung 3.2: Der Aufbau des NN schematisch dargestellt. Nach der Eingangsebene folgen mehrere tiefe Ebenen mit der Aktivierungsfunktion ReLU. Die Verbindungen der letzten Ebene werden durch die Softmax-Funktion definiert.

In den tiefen Ebenen wird die Aktivierungsfunktion *ReLU* (**R**ectified **L**inear **U**nit)

$$g_{\text{ReLU}}(x) = \begin{cases} x & \text{if } x \geq 0, \\ 0 & \text{otherwise} \end{cases}$$

genutzt. Ist der Eingang x größer 0, so wird dieser unverändert weitergegeben, sonst 0.

Die Ausgangsebene enthält genauso viele Neuronen, wie es Zielklassen gibt. In dieser Ebene wird die Aktivierungsfunktion Softmax verwendet. Wie die Definition

$$g_{\text{Softmax}}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}.$$

zeigt, ist die Summe über alle Klassen für jedes Event gleich eins. Sie erfüllt damit die Vorgabe an DSEA, dass für jede Zielklasse eine Konfidenz c_{ij} ausgegeben wird. Die Kostenfunktion *kategorische Kreuzentropie*[7] kann für ein Klassifikationsproblem verwendet werden. Ziel ist die Minimierung der Kosten in Abhängigkeit der Modellparameter. Die Minimierung der Kreuzentropie ist äquivalent zu einer Maximierung der Log-Likelihood-Funktion.

Mit dem Optimierer *ADAM* (Adaptive Moment Estimation)[10] wird die Kostenfunktion minimiert. Der Algorithmus ist eine Erweiterung des Gradientenverfahrens und beachtet zusätzlich zu der Steigung das Moment. Bildlich gesehen entspricht dies einer Kugel, die in einer hügeligen Landschaft Richtung globalem Minimum rollt und dabei lokale Minima überwindet. Die Lernrate stellt ein Hyperparameter des Optimierers dar und ist ein Maß für die Schrittweite entlang des fallenden Gradienten.

Das Modell wird auf einem Teil der Daten trainiert und evaluiert. Dazu wird der Datensatz in mehrere Unter-Datensätze mit einer spezifizierten Größe (*Batch Size*)[4] unterteilt. Das Modell wird auf einem Batch trainiert und die Kostenfunktion ausgewertet. Wird dies für alle Batches durchgeführt, so wurde jeder Datenpunkt im Training einmal berücksichtigt. Durch das häufige Anpassen der Parameter konvergiert die Kostenfunktion schneller.

Das Modell wird mehrfach auf den Trainingsdaten trainiert. Die Anzahl der Trainingsprozesse wird durch die *Epochenzahl*[4] angegeben.

4 Verwendete Methoden

In diesem Kapitel werden die verwendeten Methoden vorgestellt. Abschnitt 4.1 behandelt die χ^2 -Distanz als Abstandsmaß zweier diskreter Verteilungen. In Abschnitt 4.2 wird die Berechnung der Unsicherheiten mithilfe von Bootstrapping erläutert.

4.1 Chi-Quadrat-Distanz

Die χ^2 -Distanz[13] ist ein gewichtetes Abstandsmaß und dient zur Überprüfung von Konvergenz. Der Abstand zwischen der wahren Verteilung f und der rekonstruierten Verteilung \hat{f} wird über die Abstände der einzelnen Bins j nach

$$\chi^2 = \frac{1}{2} \sum_{j=1}^n \frac{(\hat{f}_j - f_j)^2}{\hat{f}_j + f_j} \quad (4.1)$$

bestimmt. Ist die χ^2 -Distanz kleiner oder gleich der χ^2 -Distanz in der vorherigen DSEA-Iteration, kann das Verfahren abgebrochen werden.

4.2 Bootstrapping: Abschätzung der Unsicherheiten

Bootstrapping[12] ist eine Methode zur Bestimmung der Unsicherheiten von Wahrscheinlichkeitsmodellen. Sie ist sinnvoll, wenn normalerweise nur ein kleiner Teil des Trainingsdatensatzes entfaltet wird.

Die Methode beruht auf dem „Resampling“, also einer Veränderung des ursprünglichen Datensatzes durch Mehrfachzählung und Nichtbeachtung von Events.

In **Schritt 1** wird der Datensatz durch Ersetzung von Events variiert. Dieser Datensatz wird in **Schritt 2** mit DSEA (siehe Abschnitt 3.2) entfaltet. In **Schritt 3** wird das entfaltete Spektrum gespeichert. Die Schritte 1-3 werden n -mal wiederholt.

Anschließend kann die Verteilung jedes einzelnen Energie-Bins untersucht werden. Zur Bestimmung des Mittelwerts kann der Median

$$\tilde{x} = \begin{cases} x_{m+1} & ,\text{für ungerades } n = 2m+1 \\ \frac{1}{2}(x_m + x_{m+1}) & ,\text{für gerades } n = 2m \end{cases} \quad (4.2)$$

für jeden Energie-Bin berechnet werden, wobei n die Anzahl der Entfaltungen angibt. Die zugehörigen Unsicherheiten können allgemein über das untere und obere Quantil angegeben werden. Das untere Quantil beinhaltet 16 % und das obere Quantil 84 % der Datenpunkte. Zwischen den Quantilen liegen also 68 % der Messwerte. Handelt es sich um normalverteilte Daten, so sind die Quantile äquivalent zu der Standardabweichung und der Median ist gleich dem Mittelwert.

5 Testen der Hyperparameter in DSEA und neuronalen Netzen

In diesem Kapitel werden die Ergebnisse präsentiert und diskutiert. Vorerst wird der verwendete Datensatz charakterisiert und die Schritte der Daten-Vorverarbeitung beschrieben. In Abschnitt 5.2 wird die Entfaltung über eine Klassifikation mit einem NN analysiert. Ebenfalls wird bereits das Spektrum über die Aufsummierung der Konfidenzen rekonstruiert. Hinzu kommt die iterative Anpassung der Gewichte, in Abschnitt 5.3. Es werden neuronale Netze in DSEA systematisch über eine Gittersuche analysiert. Anschließend wird der Trainingsprozess, die entfalteten Spektren und Korrelationen betrachtet. Zum Schluss, wird in Abschnitt 5.4 die Abhängigkeit der Modelle vom Trainingsspektrum untersucht.

5.1 Datensatz

Der Monte-Carlo Datensatz 11374 [6] beinhaltet Neutrinos mit einem simulierten Spektrum E^{-2} . Es werden nur sog. „upgoing“ Neutrinos betrachtet. Das sind Neutrinos die vor der Detektion die Erde durchquert haben. Es wurde bereits eine Eventauswahl durchgeführt. Dazu wird der Zenitwinkel mit einem Fehler von 5°

Tabelle 5.1: Statistische Kennzahlen zu der Energie des primären Neutrinos. Links für die ursprünglichen MC-Daten (Rohdaten) und im direkten Vergleich, der abgeschnittene Datensatz mit Energien ≤ 100 TeV (rechts).

	Rohdaten	Daten nach Schnitt
Anzahl Events	$1,334 \cdot 10^7$	$1,326 \cdot 10^7$
Mittlere Energie μ	6,3 TeV	2,2 TeV
Standardabweichung σ	246 TeV	7 TeV
Minimum	100 GeV	100 GeV
Maximum	10^8 GeV	10^5 GeV
25 %-Quantil	271 GeV	270 GeV
50 %-Quantil	547 GeV	543 GeV
75 %-Quantil	1,42 TeV	1,39 TeV

rekonstruiert und Events von 86° bis 180° beibehalten. Dies hat den Vorteil, dass keine Signaturen von Myonen beachtet werden müssen. Die Myonen wechselwirken mit der Erde und können den Detektor nicht erreichen. Anders als für Myonen, stellt die Erde für Neutrinos kein Hindernis dar. Aufgrund des geringen Wirkungsquerschnitts erreichen die meisten Neutrinos den Detektor. Der Datensatz beinhaltet 13 Millionen Events. Die Energie des primären Neutrinos stellt die Zielvariable des Klassifikationsproblems dar. In Tabelle 5.1 sind wichtige Informationen über die Lage und Streuung der Neutrinoenergie aufgeführt. Auffällig ist, dass der größte Teil der Events im niederenergetischen Bereich liegt. Aufgrund der geringen Statistik im Hochenergiebereich werden nur Events unter 100 TeV berücksichtigt. Die untere Energiegrenze bleibt unverändert und beträgt 100 GeV. Mit einer logarithmischen Skala wird die Energie in 10 Klassen aufgeteilt. Das logarithmische Binning kompensiert zum Teil die wenigen Events im Hochenergiebereich. Wird mit realen Daten gearbeitet, so können an dieser Stelle Under- und Overflow-Bins eingeführt werden. Diese decken den für die Analyse nicht-relevanten Bereich ab. 90 % der Events werden zum Trainieren und die restlichen Events zur Evaluation des Modells verwendet. Die 12 Features (siehe Tabelle A.1) werden vor der Entfaltung normalisiert

$$z = \frac{x - \mu}{\sigma}$$

um gleiche Abstände zwischen den Features zu gewährleisten. Dies ist ein notwendiger Schritt zur Optimierung des Trainingsprozesses [24].

5.2 Entfaltung als Klassifikationsproblem

Die Betrachtung der Entfaltung als Klassifikationsproblem ist eine der grundlegenden Ideen von DSEA. In diesem Kapitel soll vorerst die Entfaltung über eine klassische Klassifikation diskutiert werden. Die Abbildung 5.1 zeigt exemplarisch die Vorhersage eines einzelnen Events. Die Zugehörigkeit des Events zum j -ten Energie-Bin wird durch die Konfidenz c_j angegeben. Ein Datenpunkt wird nach dem Prinzip *maximum a posteriori* der Klasse mit der **maximalen Konfidenz** zugeordnet [21]:

$$\hat{y} = \arg \max_j \{c_j\}, \quad j = [0, \dots, n - 1]$$

Die Anzahl der Energie-Bins ist über n definiert. Das NN wird auf den Trainingsdaten mit 50 Epochen trainiert. In Tabelle B.2 sind die verwendeten Hyperparameter des NN aufgeführt. 500 000 Events der Evaluationsdaten werden entfaltet und analysiert. Ein Event wird jeweils der maximal vorhergesagten Klasse zugeordnet. Die Kostenfunktion, sowie die Metriken werden in jeder Epoche im Trainingsprozess für die Trainings- und Evaluationsdaten ausgewertet. Der Verlauf der kategorischen

Kreuzentropie und der X^2 -Distanz ist in Abbildung 5.2 dargestellt.

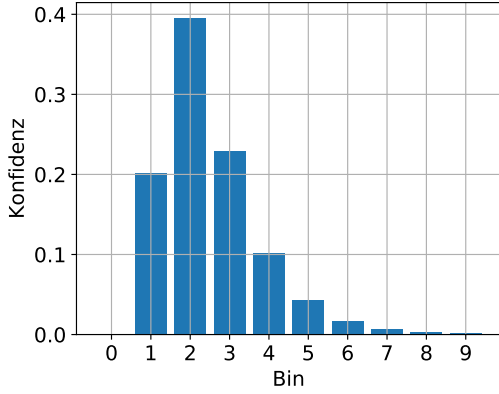
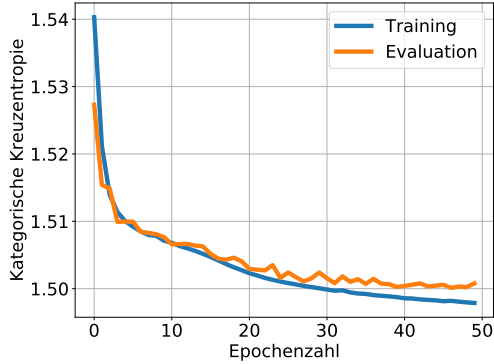


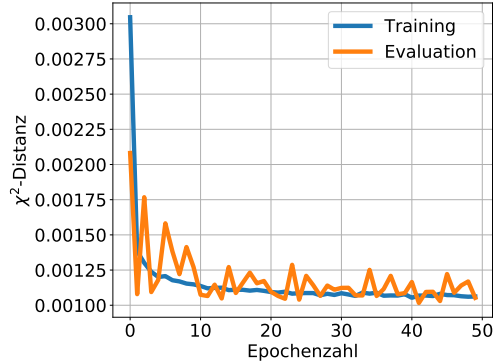
Abbildung 5.1: Visuelle Darstellung der Ausgabe des NN für ein Event mit der wahren Energieklasse 2. Die Zugehörigkeit des Events zu Bin j ist durch die Konfidenz c_j definiert.

Die X^2 -Distanz, als Abstandsmaß zwischen dem wahren und vorhergesagten Spektrum konvergiert. Ebenso verläuft der Verlust der Evaluationsdaten gegen ein Minimum. Der Anstieg des Trainingsverlusts deutet auf ein leichtes Overtraining („Übertrainieren“) hin. Eine Fortsetzung des Trainings ist daher nicht sinnvoll. Das resultierende Spektrum in Abbildung 5.3 weist starke Abweichungen von der wahren Verteilung auf. Bins im hochenergetischen Bereich werden stark unterschätzt, hingegen werden die Bins bis etwa 1 TeV überschätzt. Aufgrund der geringen Accuracy von $\sim 39\%$ kommt es zu vielen Fehlklassifikationen. Siehe dazu den Verlauf der Accuracy im Trainingsprozess

Abbildung B.1. Die Korrelation zu ähnlichen Energien bzw. benachbarten Bins wird hier nicht berücksichtigt.



(a) Kostenfunktion: Kategorische Kreuzentropie



(b) Metrik: X^2 -Distanz

Abbildung 5.2: Verlauf des Trainingsprozess des NN. Zum einen ist die Kostenfunktion *kategorische Kreuzentropie* und zum anderen die *Chi-Quadrat-Distanz* als Metrik in Abhängigkeit der Epochenzahl aufgeführt.

Die zweite grundlegende Idee die DSEA definiert, ist die **Wahrscheinlichkeitsin-**

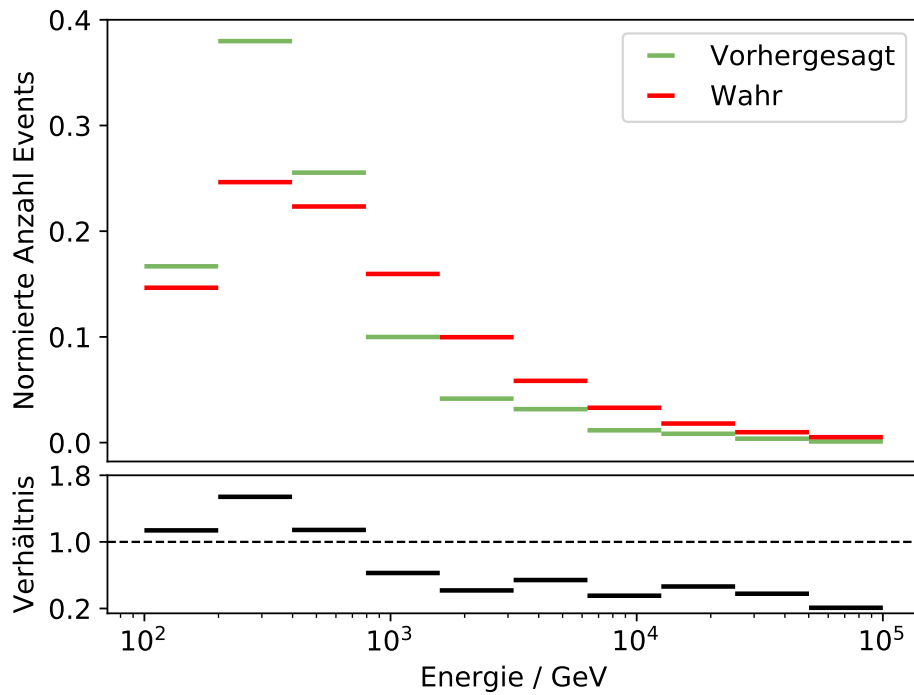


Abbildung 5.3: Das vorhergesagte und wahre Spektrum für ein NN, wenn ein Event der maximal vorhergesagten Klasse zugeordnet wird. Der untere Teil zeigt das Verhältnis zwischen wahrem und vorhergesagtem Spektrum und ist ein Maß für die relative Abweichung. Energien bis etwa 1 TeV werden systematisch überschätzt, wobei die hochenergetischen Events unterschätzt werden.

terpretation der Konfidenzen c_{ij} . Anstatt die Klasse mit der größten Konfidenz zu wählen, werden die Konfidenzen für jede Zielklasse aufsummiert (Gleichung 3.4). Die Abbildung 5.4 zeigt das resultierende Spektrum mit zugehöriger Abweichung. Die Vorhersage stimmt hier weitgehend mit dem wahren Spektrum überein, wie

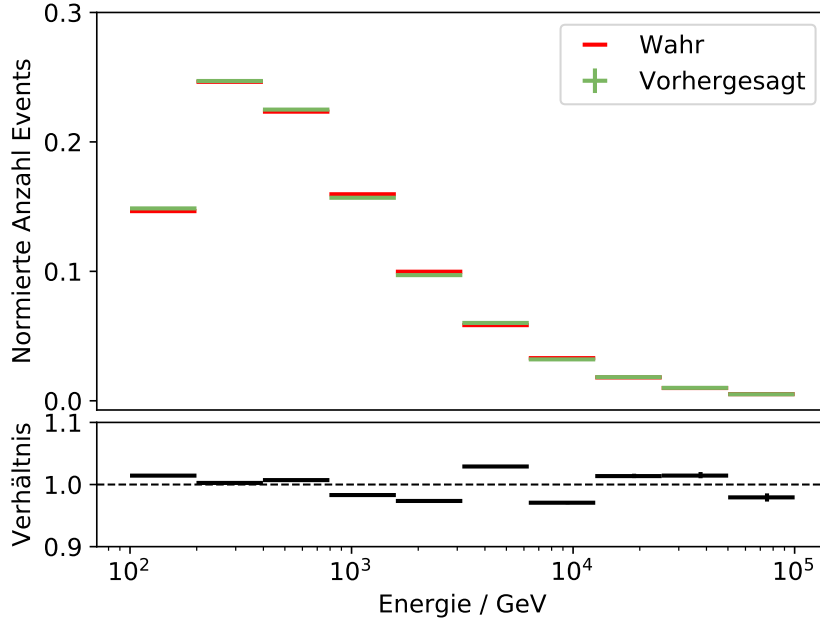


Abbildung 5.4: Vorhergesagtes und wahres Spektrum für ein NN, wenn die Konfidenzen für jeden Bin aufsummiert werden. Der untere Teil zeigt das Verhältnis zwischen wahrem und vorhergesagtem Spektrum und dient als Maß für die relative Abweichung. Bestimmung der Unsicherheiten über das Bootstrapping-Verfahren, siehe Abschnitt 4.2.

auch der Verhältnis-Plot zeigt. Größere Abweichungen sind in den Hochenergie-Bins zu beobachten. Dort treten auch die für eine Entfaltung typischen Oszillationen auf.

Wie die Vorhersage einzelner Events (Abbildung B.3) andeutet, wird durch die in Abbildung 5.5 dargestellte Korrelationsmatrix bestätigt. Es gibt eine starke Korrelation zwischen benachbarten Bins. Grund dafür ist die ordinale Natur des Problems. Jede Klasse repräsentiert bestimmte Energien, die einer definierten Ordnung folgen.

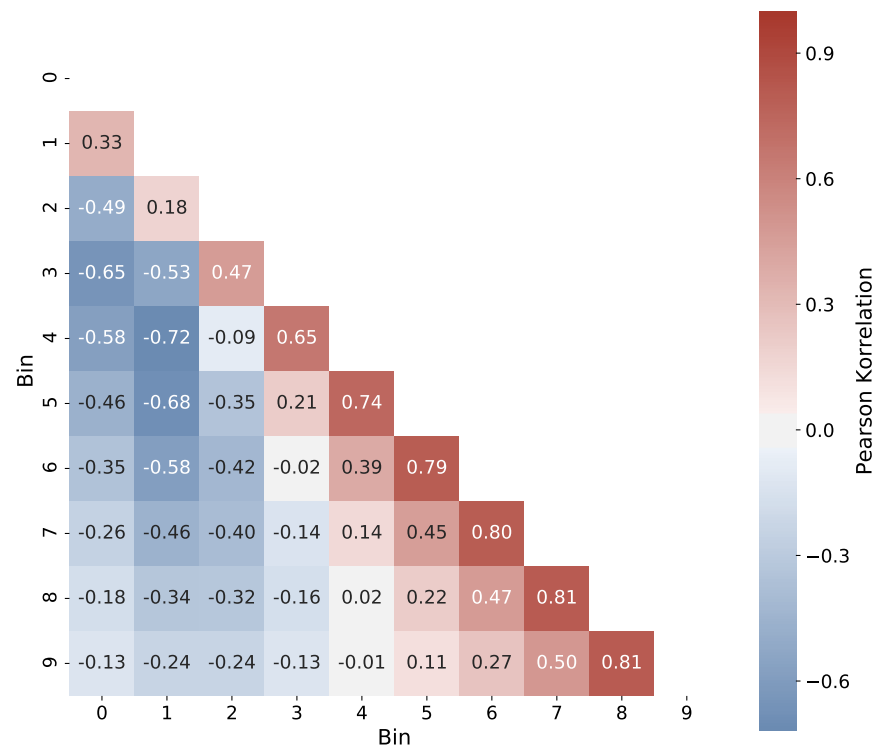


Abbildung 5.5: Darstellung der relevanten Komponenten der symmetrischen Korrelationsmatrix. Sie gibt die Korrelationen zwischen den 10 Bins für das NN an.

5.3 Entfaltung mit DSEA

5.3.1 Initialisierung

DSEA ist für Klassifikationsalgorithmen der Scikit-Learn-Bibliothek[19] ausgerichtet. Diese ist nicht für die Implementierung neuronaler Netze optimiert. In der Arbeit wird dazu das Framework *Tensorflow*[14] und dessen API *Keras* verwendet. Statt den Quellcode von DSEA anzupassen, wird ein Wrapper ("Verpackung") für die Keras-API entwickelt. Der Wrapper kann im Anhang C.2 eingesehen werden. Die Methoden *fit()* und *predict()* sind, wie die Methoden der Klassifizierer der Scikit-Learn-Bibliothek aufgebaut. Die Konsequenz ist, dass bereits bei der Initialisierung die Hyperparameter übergeben werden. Das sind Parameter die üblicherweise in der *fit()* Methode auftreten. Betroffen sind die Anzahl der Epochen, Batchgröße und die Lernrate.

Der neu eingeführte Parameter **one_model** [Bool] gibt an, ob im gesamten Trainingsprozess nur ein Modell verwendet wird:

True Ein Modell wird trainiert, wobei die Gewichte in jeder DSEA-Iteration aktualisiert werden

False In jeder DSEA-Iteration wird ein neues Modell erstellt, welches auf den angepassten Gewichten trainiert wird

Die meisten Klassifizierer basieren nicht auf einer Verlustfunktion und können somit den Trainingsprozess nicht fortsetzen. Das bedeutet, dass die Klassifizierer der Scikit-Learn-Bibliothek in jeder DSEA-Iteration neu initialisiert werden. NN in DSEA bieten also die Freiheit, zwischen den beiden Szenarien zu wählen.

Die Anzahl an Epochen pro DSEA *Iteration* ist durch den Parameter *Epochen* definiert. Die Gesamtanzahl der Epochen ergibt sich durch $Epochen \times Iterationen$.

5.3.2 Optimierung der Hyperparameter

Zur systematischen Untersuchung der drei Parameter *Anzahl Epochen*, *Anzahl Iterationen in DSEA* und *one_model* wird eine Gittersuche durchgeführt. Es wird das bereits bekannte Modell B.1 mit der gleichen Batchsize und Lernrate verwendet. Aufgrund der Rechenzeit wird für die Gittersuche nur ein Teil des Trainingsdatensatzes mit $\sim 1\,000\,000$ Events genutzt.

Zur Evaluation der Modelle werden die Abstände zwischen dem wahren und entfalteten Spektrum betrachtet. Es werden $\sim 500\,000$ Events entfaltet und ausgewertet. In Tabelle 5.2 sind die Parameter der 10 besten Modelle der Gittersuche, nach der X^2 -Distanz sortiert, aufgelistet.

Tabelle 5.2: Die 10 besten Modelle der Gittersuche für die Parameter *Anzahl Epochen*, *Anzahl Iterationen in DSEA* und *one_model* aufgeführt. Die Ergebnisse sind nach der X^2 -Distanz sortiert. Ebenfalls ist der Root Mean Square Error (RMSE) angegeben.

Epochen	Iterationen	one_model	RMSE	X^2 -Distanz
75	12	False	0.003123	0.000269
60	16	True	0.003282	0.000293
50	16	True	0.004092	0.000439
50	8	False	0.004332	0.000449
10	12	True	0.005303	0.000518
100	20	False	0.006635	0.000623
25	12	False	0.006319	0.000653
100	16	True	0.006286	0.000686
25	16	True	0.005821	0.000688
75	8	True	0.005655	0.000694

Ebenfalls wird der Root Mean Square Error („Wurzel des mittleren quadratischen Fehlers“), kurz **RMSE** als Abstandsmaß betrachtet.

Zur Veranschaulichung sind die Ergebnisse in einem Streudiagramm für *one_model=True* in Abbildung 5.6 bzw. für *one_model=False* in Abbildung C.1 dargestellt. Die logarithmisch skalierte Farbskala gibt die X^2 -Distanz an. Auffällig ist, dass bei einer geringen Anzahl an Iterationen in DSEA der Abstand besonders groß ist. Mit steigender Anzahl an Iterationen konvergiert dieser.

Im Bezug auf die Anzahl der Epochen liegen die Modelle mit den kleinsten Abständen zwischen 50 und 75 Epochen.

Die zwei Modelle mit dem geringsten X^2 -Abstand werden genauer untersucht:

Modell 1

Epochenzahl 75

Iterationen 12

one_model False

Grafiken im Anhang C.2 C.3 C.8

Modell 2

Epochenzahl 60

Iterationen 16

one_model True

Grafiken in Abbildung 5.7 5.8 5.9

In den folgenden Abschnitten 5.3.3 bis 5.3.5 wird exemplarisch die Trainingshistorie, das Energiespektrum und die Korrelationsmatrix des 2. Modells gezeigt. Analog dazu befinden sich die Plots des 1. Modells der Übersichtlichkeit halber im Anhang.

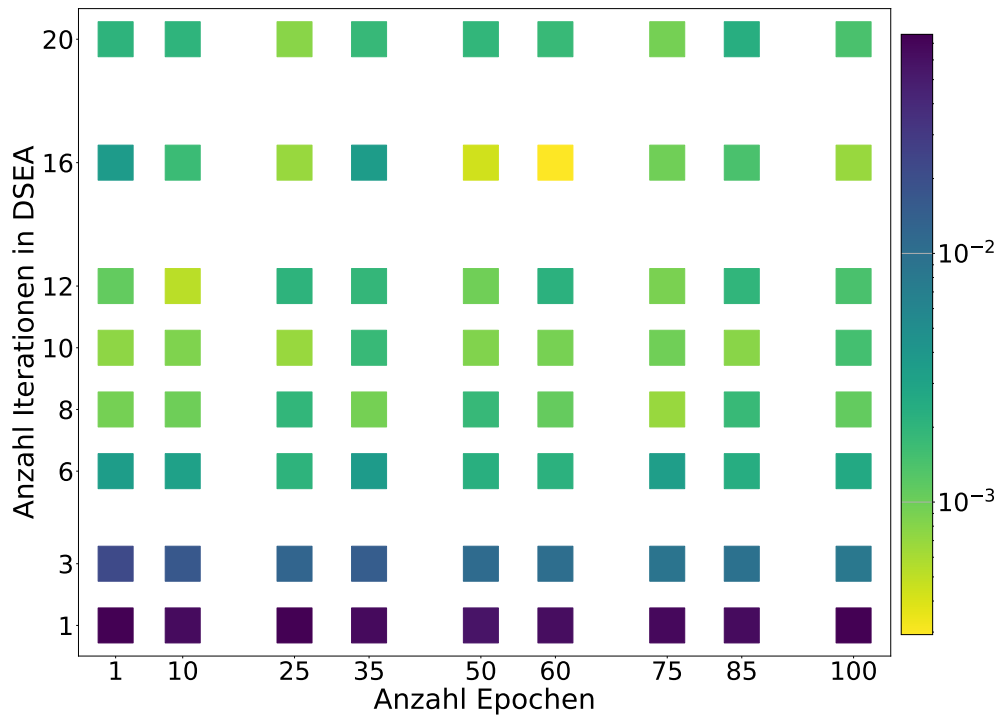
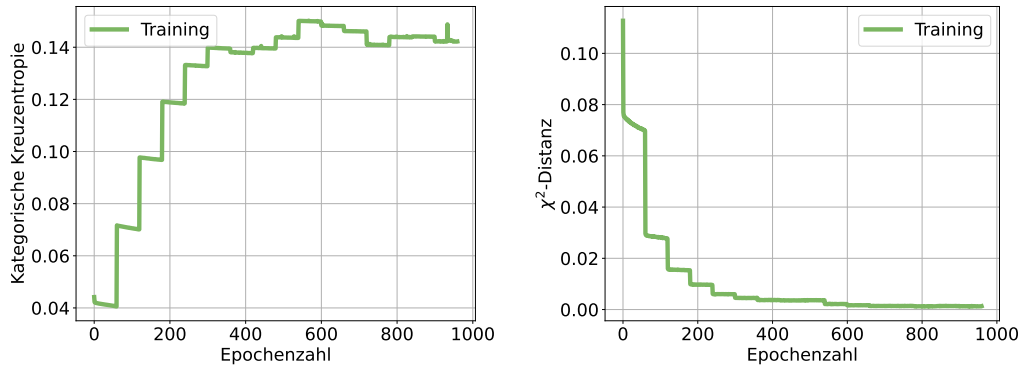


Abbildung 5.6: Streudiagramm zur Darstellung der Ergebnisse der Gittersuche für *one_model=True*. Der Trainingsprozess des Modells wird in jeder DSEA-Iteration mit angepassten Gewichten fortgesetzt. Die logarithmisch-skalierte Farbskala gibt den X^2 -Abstand zwischen dem wahren und vorhergesagten Spektrum an.

5.3.3 Trainingsprozess

Der Verlauf des Trainingsprozess des 2. Modells in Abbildung 5.7 hat einen näherungsweisen stetigen Verlauf. Das liegt daran, dass über den gesamten Prozess nur



(a) Kostenfunktion: Kategorische Kreuzentropie

(b) Metrik: X^2 -Distanz

Abbildung 5.7: Verlauf des Trainingsprozess des 2. Modells mit dem Parameter `one_model=True`. Im gesamten Trainingsverlauf wird ein Modell trainiert, wobei die Gewichte in jeder DSEA-Iteration angepasst werden. Zum einen ist die Kostenfunktion *kategorische Kreuzentropie* und zum anderen die *Chi-Quadrat-Distanz* als Metrik in Abhängigkeit der Epochenzahl aufgeführt.

die Kostenfunktion eines Modells betrachtet wird. Kleine Unstetigkeiten werden durch die angepassten Gewichte am Ende jeder DSEA-Iteration verursacht. Anders sieht es bei dem Trainingsverlauf des 1. Modells in Abbildung C.2 aus. In jeder DSEA-Iteration wird ein neues Modell erstellt und die Optimierung der Kostenfunktion beginnt von vorne. Dies ist der Grund für den sprunghaftigen Verlauf der Metrik und der Verlustfunktion.

Die X^2 -Distanz beider Modelle sinkt mit der Epochenzahl und konvergiert. Hingegen steigt die kategorische Kreuzentropie bei jeder Neugewichtung. Die Gewichte werden in der Kostenfunktion aktualisiert und führen zu einem Anstieg. Die angepasste Kostenfunktion wird in den darauffolgenden Epochen bis zur nächsten Iteration minimiert. Sie konvergiert, wenn auch die Gewichte konvergieren.

5.3.4 Spektrum

Das entfaltete Spektrum ist für das 2. Modell in Abbildung 5.8 bzw. für das 1. Modell in Abbildung C.3 abgebildet. Es treten Oszillationen im gesamten Energiebereich auf. Diese verstärken sich, insbesondere bei Modell 1 zu höheren Energien.

Auffällig sind auch die ungewöhnlich großen absoluten Abstände in den ersten vier Energie-Bins. Aufgrund der hohen Statistik in diesen Bins (viele Events im niedrigen Energiebereich) werden hier kleine Abweichungen erwartet. Eine mögliche Ursache ist, dass die Hochenergie-Bins eine stärkere Gewichtung als die Bins im niederenergetischen Bereich erfahren. Dadurch werden die Events mit hohen Energien im Trainingsprozess mehr berücksichtigt. Die Verteilung der Ergebnisse des Bootstrap-Verfahrens zur Bestimmung der Unsicherheiten kann im Anhang eingesehen werden (Modell 1 im Anhang C.5, Modell 2 im Anhang C.4).

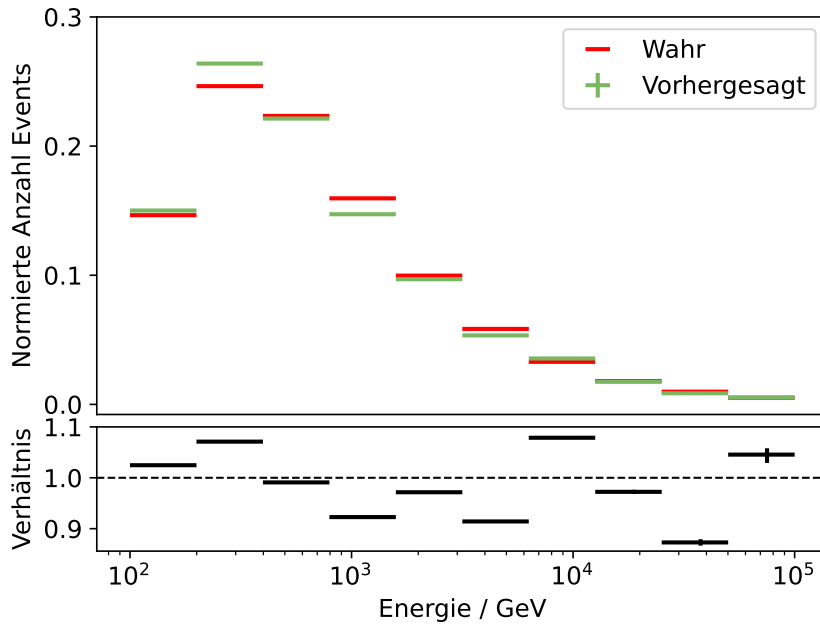


Abbildung 5.8: Das entfaltete Spektrum des 2. Modells mit dem Parameter `one_model=True`. Im gesamten Trainingsverlauf wird ein Modell trainiert, wobei die Gewichte in jeder DSEA-Iteration angepasst werden. Der untere Teil zeigt das Verhältnis zwischen wahren und vorhergesagtem Spektrum und dient als Maß für die relative Abweichung.

5.3.5 Korrelation

Die Vorhersage einzelner Events (Modell 1 im Anhang C.6, Modell 2 im Anhang C.7) weisen für beide Modelle nur geringe Unterschiede auf. Je nach Event variiert die Verteilung der Konfidenzen unterschiedlich stark.

Auffällig sind hier die starken Korrelationen zu den benachbarten Bins. Dies wird auch durch die Korrelationsmatrix in Abbildung 5.9 für Modell 2 (bzw. für Modell 1 im Anhang C.8) bestätigt. Die benachbarten Bins sind stark positiv korreliert. Wird

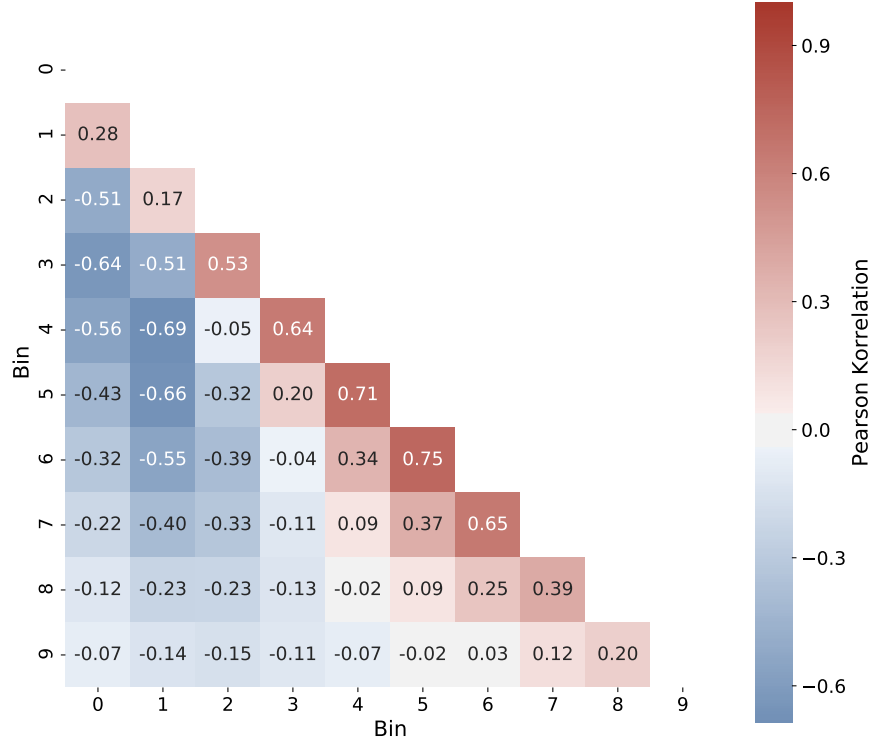


Abbildung 5.9: Die Korrelationsmatrix des 2. Modells (*one_model=True*). Sie gibt die Korrelationen zwischen den entfaltenen Energie-Bins an.

also der n -te Bin vorhergesagt, so werden dem $(n-1)$ -ten und $(n+1)$ -ten Bin ebenfalls hohe Konfidenzen zugeordnet. Die Korrelation nimmt zu entfernten Energien ab. Im niederenergetischen Bereich treten starke negative Korrelationen auf. Insbesondere sind die Bins 3-6 mit Bin 1 korreliert. Dies lässt sich auf die hohe Statistik im Bin 1 zurückführen. Diesem Energiebereich sind die meisten Events zuzuordnen. Wird nun exemplarisch der 4. Bin vorhergesagt, so wird dem 1. Bin eine niedrigere Konfidenz als im Durchschnitt zugeordnet.

5.4 Abhängigkeit der Entfaltung vom Trainingsspektrum

In diesem Kapitel wird die Abhängigkeit des entfalteten Spektrums von den Trainingsdaten (*Bias*) untersucht. Dazu wird aus dem bekannten MC-Datensatz eine Teilprobe mit gleichverteilten Klassen erstellt. Die Methode wird als *Undersampling* bezeichnet.

Die Probe beinhaltet von jeder der 10 Energie-Klassen 50 000 Events und umfasst somit insgesamt 500 000 Events. Diese Teilprobe wird als Trainingsdatensatz verwendet. Entfaltet und evaluiert werden 500 000 Events des bekannten Neutrino-Spektrums. Zum einen wird der Bias des gleichen NN, wie in Abschnitt 5.2 untersucht. Zum anderen wird die Entfaltung eines NN in DSEA auf eine Abhängigkeit geprüft. In Abbildung 5.10 sind die Ergebnisse des auf 50 Epochen trainierten NN dargestellt. Zu sehen ist das Spektrum der gleichverteilten Trainingsdaten in Blau, der entfalte-

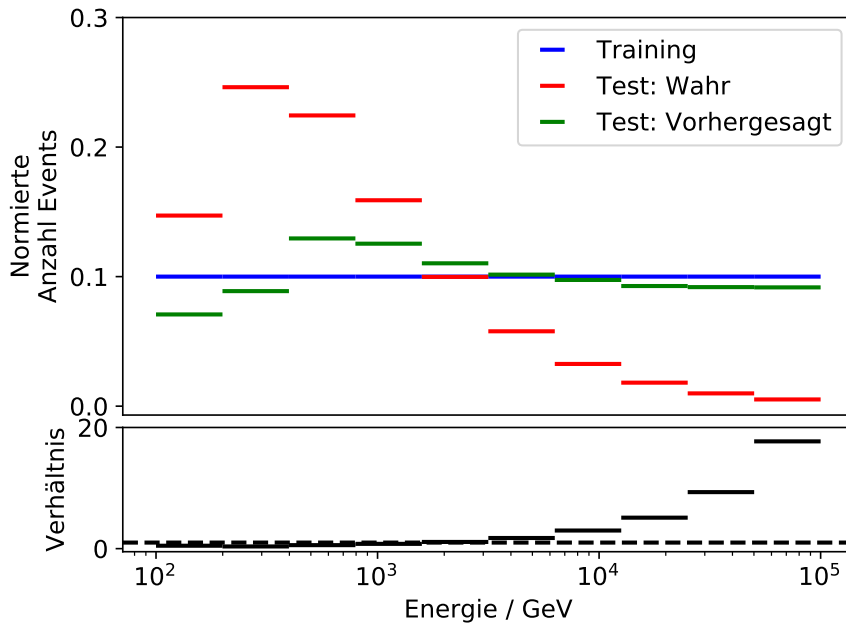


Abbildung 5.10: Das mit einem NN entfaltete Energiespektrum, der Neutrinos bei Verwendung von Trainingsdaten mit gleichverteilten Klassen. Vergleich der Spektren der Trainingsdaten (Blau), der wahren Evaluationsdaten (Rot) und der entfalteten Evaluationsdaten (Grün).

ten Daten in Grün und des wahren Spektrums in Rot. Das vorhergesagte Spektrum liegt nahe dem Trainingspektrum. Nur in Bin 3 und 4 wird das Maximum des zu

entfaltenden Spektrums leicht angedeutet. Insgesamt ist also ein starker Bias zu beobachten.

Im folgenden wird das 1. Modell (12 Iterationen, 75 Epochen, *one_model=False*) zur Entfaltung des Neutrino-Spektrums verwendet. Trainiert wird das Modell auf den gleichverteilten Daten. Die Spektren sind analog zum vorherigen Plot in Abbildung 5.11 graphisch dargestellt. Das entfaltete Spektrum unterscheidet sich erheblich von den Trainingsdaten. Im Hochenergiebereich werden die Bins stark überschätzt. Auch die anderen entfalteten Energiebereiche passen nur bedingt mit dem wahren Spektrum überein. Dies liegt hier nicht an einer starken Abhängigkeit von den Daten, sondern lässt sich auf die geringe Accuracy zurückführen.

Der gleiche Prozess wird für das 2. Modell (16 Iterationen, 60 Epochen, *one_model=True*) durchgeführt. Das entfaltete Spektrum in Abbildung D.1 weist starke Abweichungen von der wahren Verteilung auf. Insbesondere die Hochenergie-Bins werden überschätzt. Insgesamt unterscheiden sich die entfalteten Verteilungen signifikant von dem Trainingsspektrum. Die Modellunabhängigkeit von DSEA wird dadurch wiederum bestätigt.

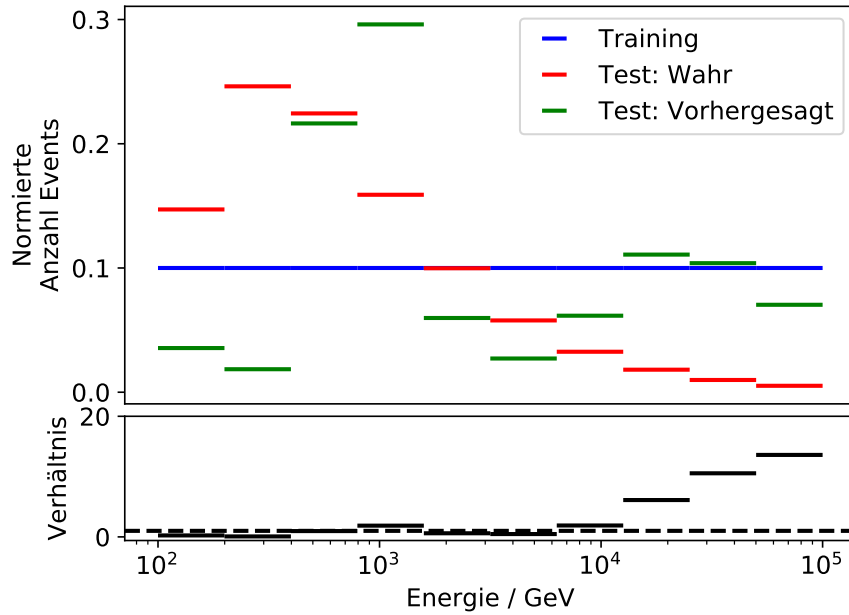


Abbildung 5.11: Das mit einem NN in DSEA entfaltete Energiespektrum der Neutrinos. Das NN (Modell 1) wird in DSEA auf einem Datensatz mit gleichverteilten Klassen trainiert. Dargestellt ist das Spektrum der Trainingsdaten (Blau), der wahren Evaluationsdaten (Rot) und der entfalteten Evaluationsdaten (Grün).

6 Fazit und Ausblick

Ziel der Arbeit ist die Untersuchung von Entfaltungen mit neuronalen Netzen in **DSEA**. Das Verhalten der Parameter *Epochenzahl* und Anzahl *DSEA-Iterationen* wurde systematisch untersucht. Es wurde geprüft, wie sich NN in DSEA entwickeln, wenn in jeder Iteration ein neues Modell erstellt und trainiert wird. Ebenfalls wurde eine neue Methode geprüft, die nur bei auf Verlustfunktion basierenden Klassifizierern möglich ist. Statt in jeder Iteration ein neues Modell zu erstellen, werden die Gewichte der Kostenfunktion **eines** Modells angepasst. Dies führt zu einer schnelleren Konvergenz. Ein signifikanter Unterschied im resultierenden Spektrum konnte jedoch nicht festgestellt werden.

Ebenfalls wurde eine Entfaltung über eine klassische Klassifikation mit einem NN betrachtet. Die Abhängigkeit vom Trainingsspektrum dieses Modells und der von neuronalen Netze in DSEA wurde zuletzt geprüft. Die Entfaltung über eine klassische Klassifikation zeigt einen starken Bias. NN in DSEA weisen auch eine große Abweichung vom wahren Spektrum auf. In diesem Fall liegt das nicht an einem Bias, sondern an der geringen Accuracy $< 39\%$. Die Modellunabhängigkeit von DSEA konnte hier bestätigt werden.

Allgemein weisen die Entfaltungen des vorliegenden Datensatzes große Abweichungen auf. Die Entfaltung mit NN in DSEA kann möglicherweise durch einen Datensatz der Rohdaten beinhaltet verbessert werden. Grund dafür ist, dass die Stärke von NN nicht in der Klassifikation von tabellarischen Daten liegt. Bei der Klassifikation von graphischen Daten sind Convolutional Neural Networks („Faltendes Neuronales Netzwerk“), kurz **CNN** unersetzbar. Optimal ist dafür ein Datensatz, der die drei-dimensionale Struktur des Detektors widerspiegelt. Dies zeigt auch die Arbeit [16] über die Rekonstruktion von Neutrinoereignissen mit graphischen neuronalen Netzen (GNN), die eine Erweiterung der CNNs [16] darstellen.

Größere Modelle mit mehr Parameter haben grundsätzlich ein größeres Potenzial. Um Overfitting zu reduzieren, können Methoden der Regularisierung, wie die L2-Parameter-Regularisierung, Dropout oder die Batch-Norm verwendet werden.

Auffällig waren die in allen Entfaltungen auftretenden Oszillationen. Zur Glättung dieser Störung könnte ebenfalls ein Regularisierungsterm in der Kostenfunktion beitragen. Analog zur Tikhonov-Regularisierung[9], könnte Glattheit über eine kleine zweite Ableitung gefordert werden.

A Monte-Carlo Sample

Tabelle A.1: Namen der verwendeten Features aus dem Monte-Carlo 11374 Datensatz.

Featurename
SplineMPEDirectHitsICE.n_dir_doms
VariousVariables.Cone_Angle
SplineMPECramerRaoParams.variance_theta
Borderness.Q_ratio_in_border
SplineMPETruncatedEnergy_SPICEMie_BINS_MuEres.value
SplineMPETruncatedEnergy_SPICEMie_DOMS_Neutrino.energy
SplineMPEDirectHitsICB.n_late_doms
Dustyness.n_doms_in_dust
LineFitGeoSplit1Params.n_hits
SplineMPEDirectHitsICC.dir_track_hit_distribution_smoothness
SPEFit2GeoSplit1BayesianFitParams.logl
SplineMPECharacteristicsIC.avg_dom_dist_q_tot_dom

B Entfaltung als Klassifikationsproblem

Ebene	Aktivierungsfunktion	Eingangsdimension	Ausgangsdimension
Eingang	-	12	12
Tief	ReLU	12	120
Tief	ReLU	120	240
Tief	ReLU	240	120
Tief	ReLU	120	12
Ausgang	Softmax	12	10

Tabelle B.1: Die Struktur des neuronalen Netzes von oben nach unten gelesen. Die Aktivierungsfunktion und Eingangs- und Ausgangsdimensionen sind für jede Ebene aufgeführt.

Parameter	Wert
Epochenzahl	50
Batchgröße	2048
Lernrate (ADAM)	0.0005

Tabelle B.2: Hyperparameter des neuronalen Netzes.

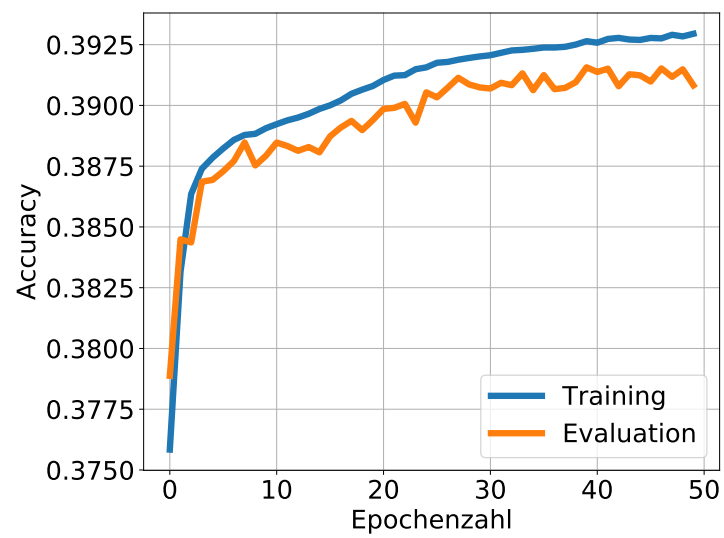


Abbildung B.1: Verlauf der Accuracy in Abhängigkeit der Epochenzahl für den Trainingsprozess des NN.

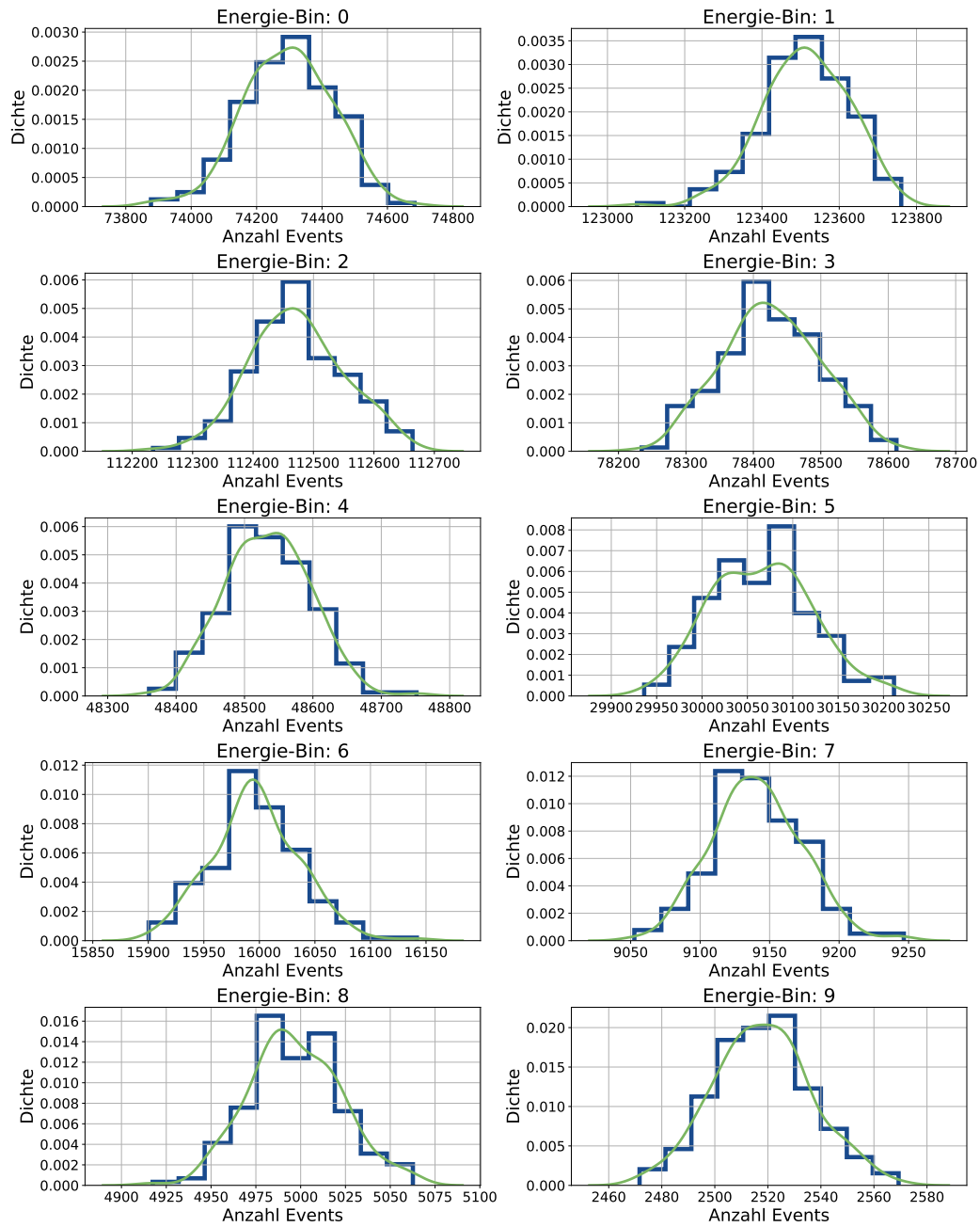


Abbildung B.2: Das blaue Histogramm stellt die Verteilung der Bootstrap-Ergebnisse mit 200 Iterationen dar. Die rote Funktion repräsentiert einen Kerndichtschätzer(KDE) mit Gaußkern. Die Bin-Höhe wird durch den Median angegeben. Über das untere und obere Quantil werden die Unsicherheiten bestimmt.

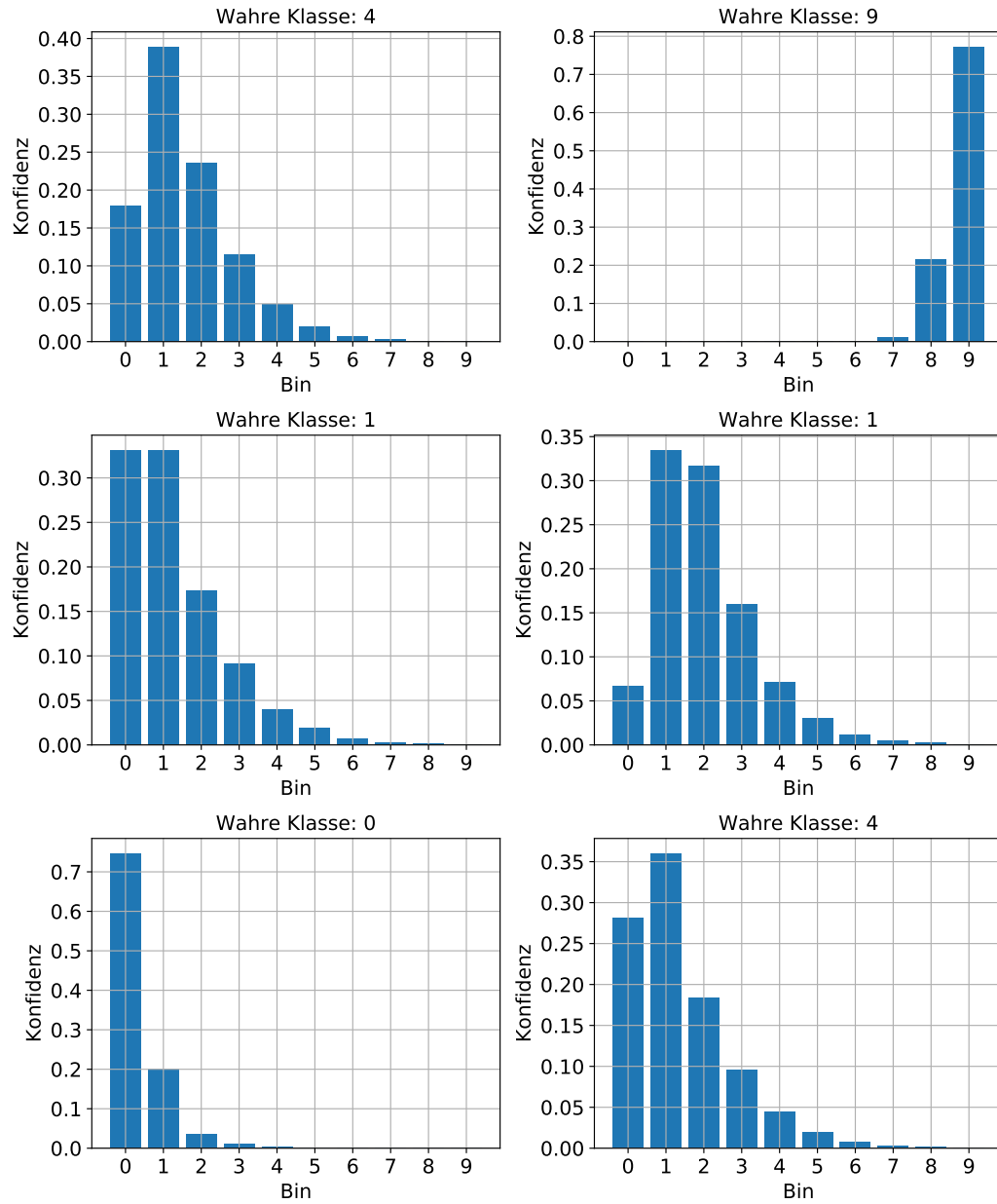


Abbildung B.3: Vorhersage einzelner Events des NN. Jedem Energie-Bin wird eine Konfidenz zugeordnet. Diese gibt die Wahrscheinlichkeit an, dass das betrachtete Event zu dem Energie-Bin gehört.

C Entfaltung mit DSEA

```
def make_model(num_features, num_classes, learning_rate=0.0005):  
    """  
    num_features: Number of features  
    num_classes: Number of energy classes  
    learning_rate: Hyperparameter of the ADAM optimizer  
    """  
    model = tf.keras.Sequential()  
  
    # input layer  
    model.add(tf.keras.layers.Dense(120, input_shape=num_features,  
    activation='relu'))  
  
    # dense layer  
    model.add(tf.keras.layers.Dense(240, activation='relu'))  
    model.add(tf.keras.layers.Dense(120, activation='relu'))  
    model.add(tf.keras.layers.Dense(12, activation='relu'))  
  
    # output layer  
    model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))  
  
    # define optimizer, loss function and metric  
    opt = tf.keras.optimizers.Adam(learning_rate=learning_rate)  
    loss = tf.keras.losses.CategoricalCrossentropy()  
    acc = tf.keras.metrics.CategoricalAccuracy(  
        name="categorical_accuracy", dtype=None)  
  
    # compile the tensorflow model  
    model.compile(optimizer=opt, loss=loss, metrics=[acc, chi2])  
  
    return model
```

Listing C.1: Die Funktion wird zur Erstellung des Modells verwendet. Hier wird das NN mithilfe der Keras-API [14] definiert. Die Ebenen können beliebig angepasst und erweitert werden.

```
class MyClassifier():
    """
    Parameters
    """
    one_model: bool
        If True, dsea will train only one model instead of
        generating a new one in each dsea iteration
    batch_size: int
        Split data into batches with the size of batch_size
    epochs: int
        Number of epochs
    learning_rate: float
        Step size of the optimizer ADAM
    """

    def __init__(self, batch_size=2048, epochs=5, learning_rate=0.0005,
one_model=True):
        self.batch_size = batch_size
        self.epochs = epochs
        self.learning_rate = learning_rate
        self.one_model = one_model
        self.model = None #tensorflow model if created
        self.Niter = 0 #current DSEA iteration
        self.history = [] #model history for each dsea iteration

    def fit(self, X, y, sample_weight=None):
        # train self.model on weighted data X with label y

        # print progress
        self.Niter += 1
        print(f'\nNumber of iteration in DSEA: {self.Niter}')

        # y is NOT one-hot encoded yet
        y_hot = np.zeros((y.size, y.max()+1))
        y_hot[np.arange(y.size),y] = 1

        # create new model if (one_model=False) or
        # (one_model = True and there exist no model yet)
        if self.model is None or self.one_model is False:
            self.model = make_model(num_features=(len(feature_list), ),
            num_classes=y.max()+1, learning_rate=self.learning_rate)

        # save training history
        history = self.model.fit(X, y_hot, sample_weight=sample_weight,
        batch_size=self.batch_size, epochs=self.epochs)
        self.history.append(history)
```

```

    return self

def predict_proba(self, X):
    # predict X, returns confidences c_ij for each row i and class j
    return self.model.predict(X)

def get_model(self):
    # return the tensorflow model
    return self.model

def get_model_history(self):
    # return the history of loss, accuracy and chi2 distance
    # of the training process
    list_loss = []
    list_acc = []
    list_chi = []

    # list of lists → one list
    for history in self.history:
        list_loss += history.history['loss']
        list_acc += history.history['categorical_accuracy']
        list_chi += history.history['chi2']

    # list to numpy array
    list_loss = np.array(list_loss, dtype='float32')
    list_acc = np.array(list_acc, dtype='float32')
    list_chi = np.array(list_chi, dtype='float32')

    return list_loss, list_acc, list_chi

```

Listing C.2: Quellcode des Wrappers zur Verwendung der Keras-API[14] in DSEA. Es wird die Funktion **make_model** im Anhang C.1 zur Erstellung der Tensorflow-Modelle benötigt.

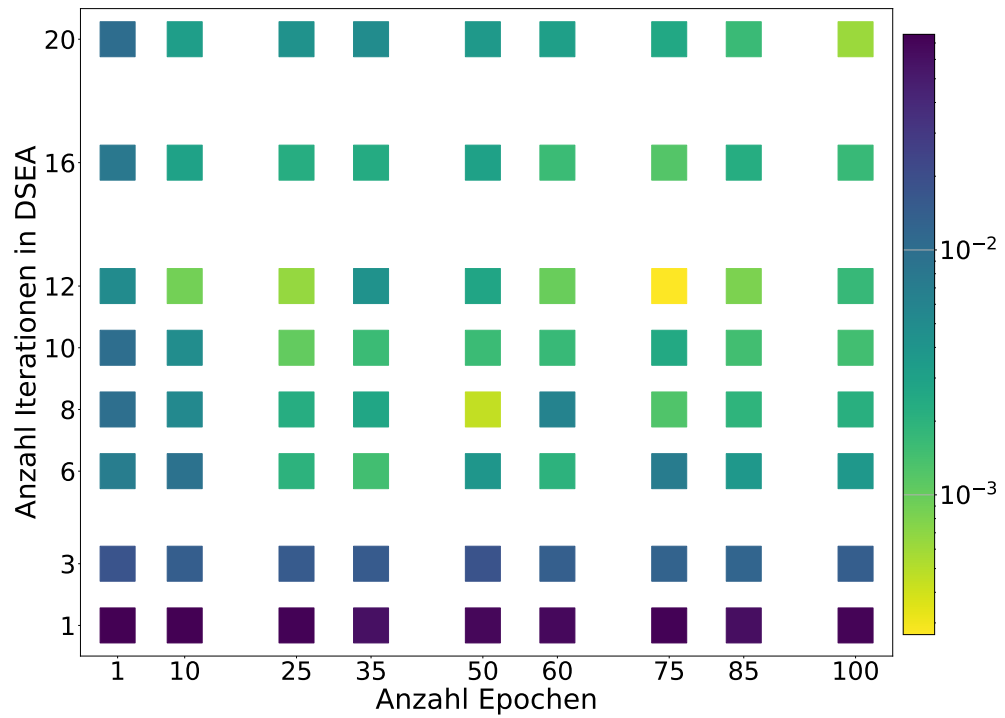
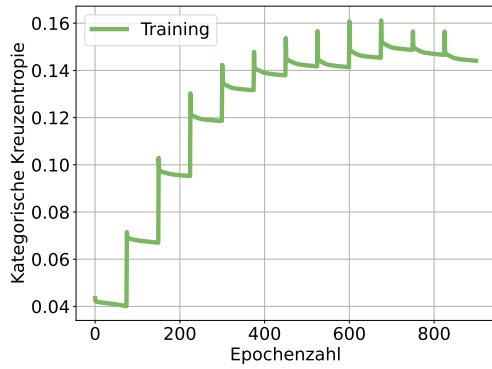
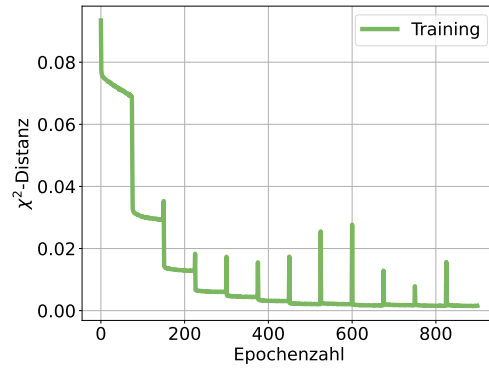


Abbildung C.1: Streudiagramm zur Darstellung der Ergebnisse der Gittersuche für Modell 1 (`one_model=False`). In jeder DSEA-Iteration wird ein neues Modell erstellt und der Trainingsprozess beginnt von vorne. Die logarithmische Farbskala gibt den X^2 -Abstand zwischen dem wahren und vorhergesagten Spektrum an.



(a) Kostenfunktion: Kategorische Kreuzentropie



(b) Metrik: χ^2 -Distanz

Abbildung C.2: Verlauf des Trainingsprozess des 1. Modells mit dem Parameter `one_model=False`. In jeder DSEA-Iteration wird ein neues Modell mit aktualisierten Gewichten trainiert. Zum einen ist die Kostenfunktion *kategorische Kreuzentropie* und zum anderen die *Chi-Quadrat-Distanz* als Metrik in Abhängigkeit der Epochenzahl aufgeführt.

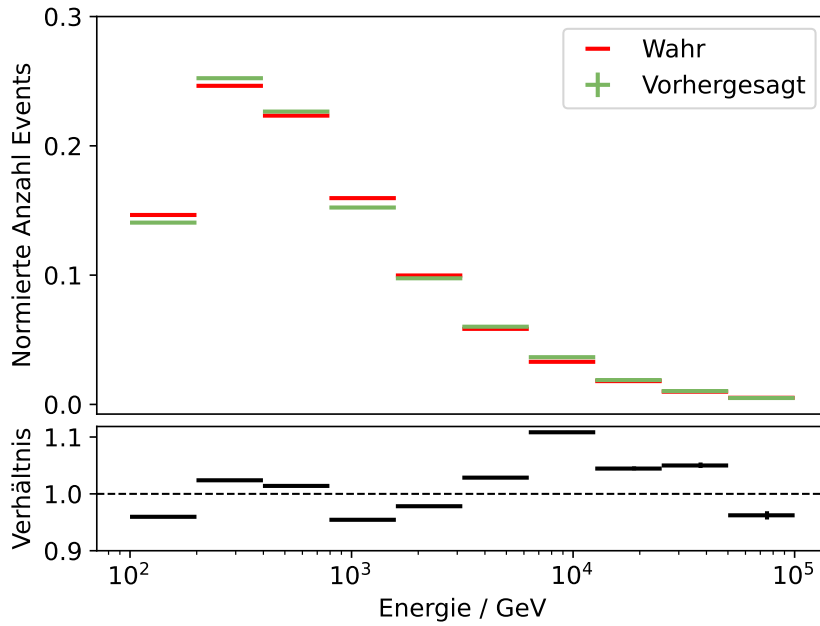


Abbildung C.3: Das entfaltete Spektrum des 1. Modells mit dem Parameter `one_model=False` und der zugehörige Verhältnis-Plot. In jeder DSEA-Iteration wird ein neues Modell mit aktualisierten Gewichten trainiert.

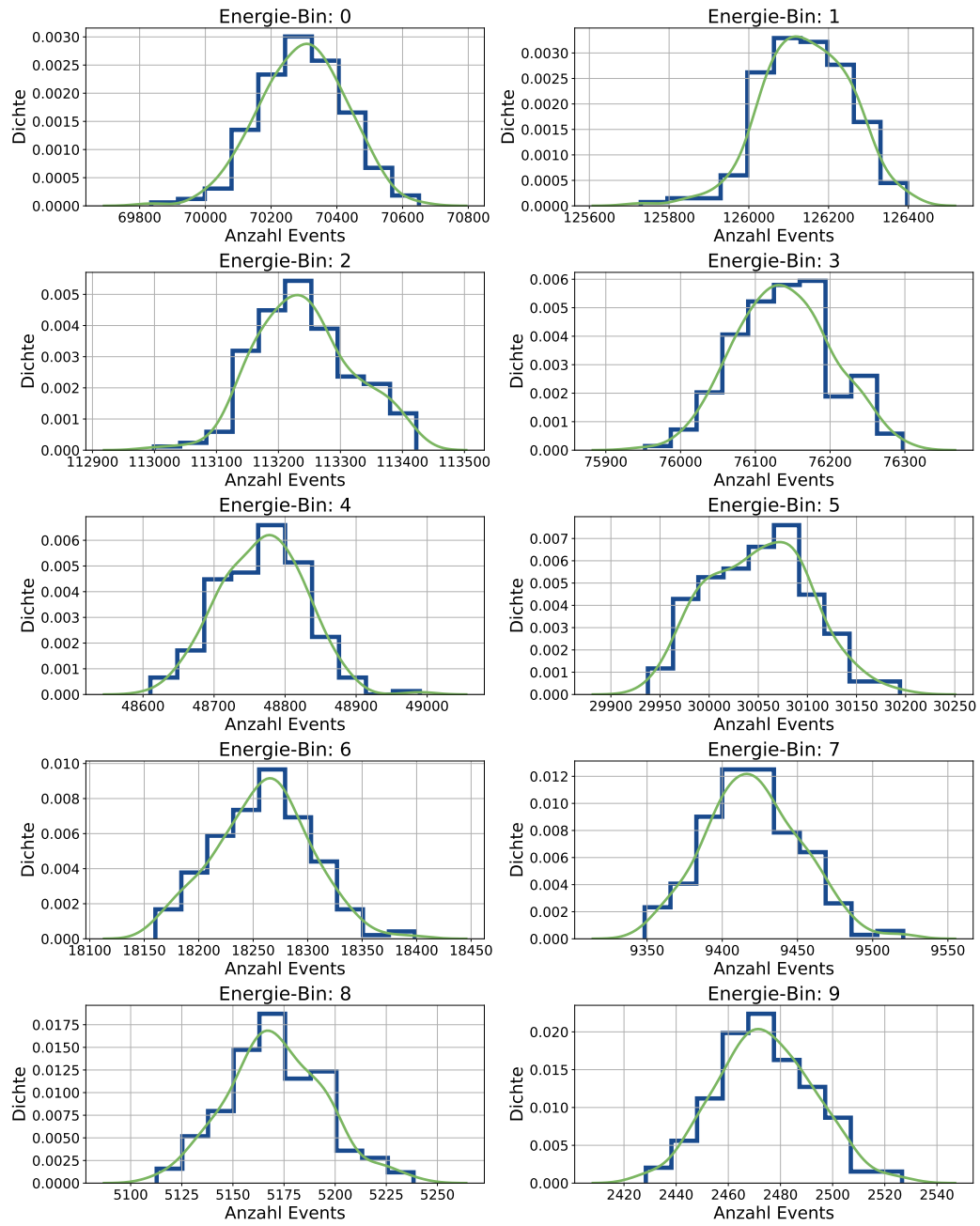


Abbildung C.4: Das blaue Histogramm stellt die Verteilung der Bootstrap-Ergebnisse für Modell 1 ($one_model=False$) mit 200 Iterationen dar. Die rote Funktion repräsentiert einen Kern-Dichteschätzer(KDE) mit Gaußkern. Die Bin-Höhe wird durch den Median angegeben. Über das untere und obere Quantil werden die Unsicherheiten bestimmt.

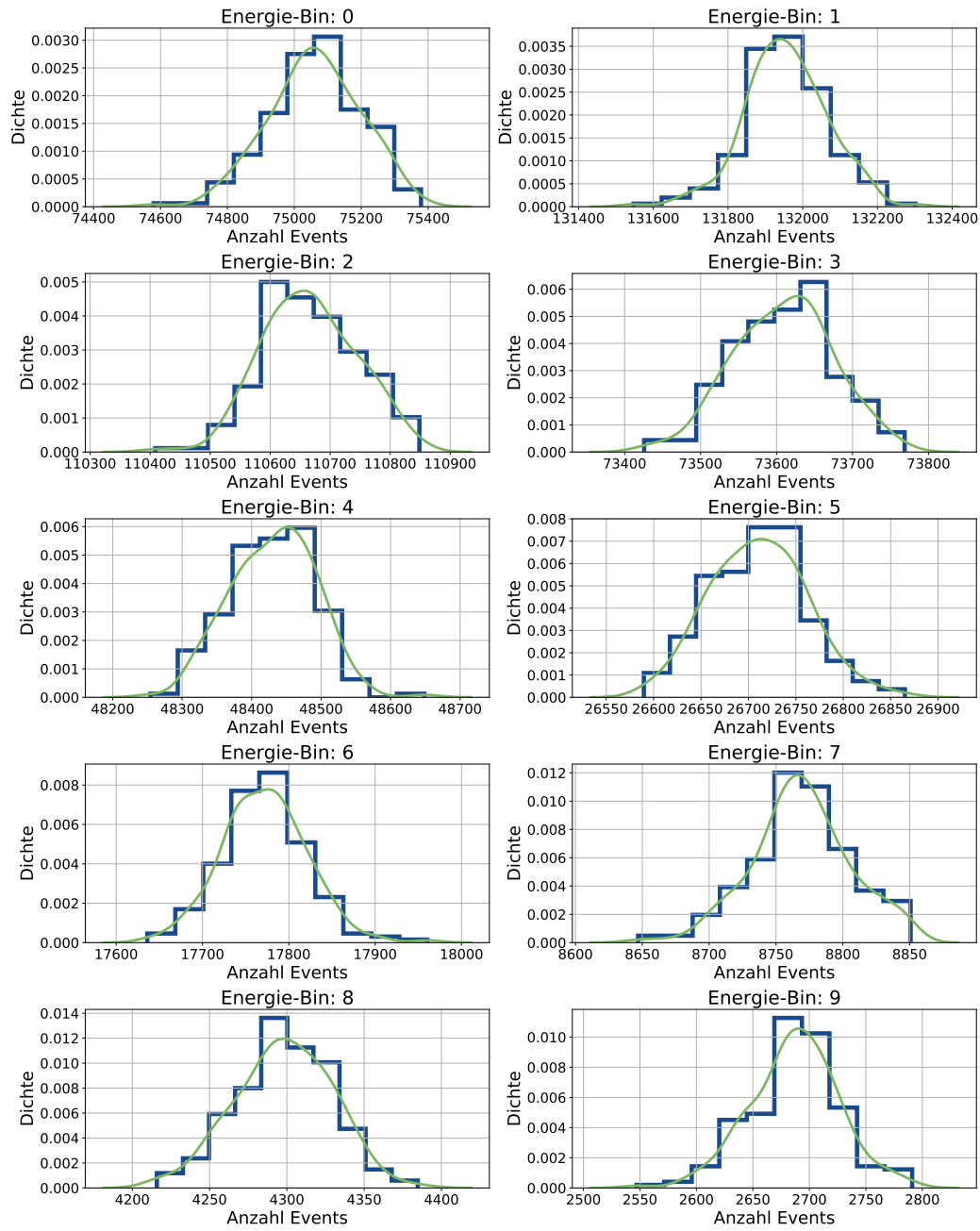


Abbildung C.5: Das blaue Histogramm stellt die Verteilung der Bootstrap-Ergebnisse für Modell 2 (*one_model=True*) mit 200 Iterationen dar. Die rote Funktion repräsentiert einen Kerndichteschätzer(KDE) mit Gaußkern. Die Bin-Höhe wird durch den Median angegeben. Über das untere und obere Quantil werden die Unsicherheiten bestimmt.

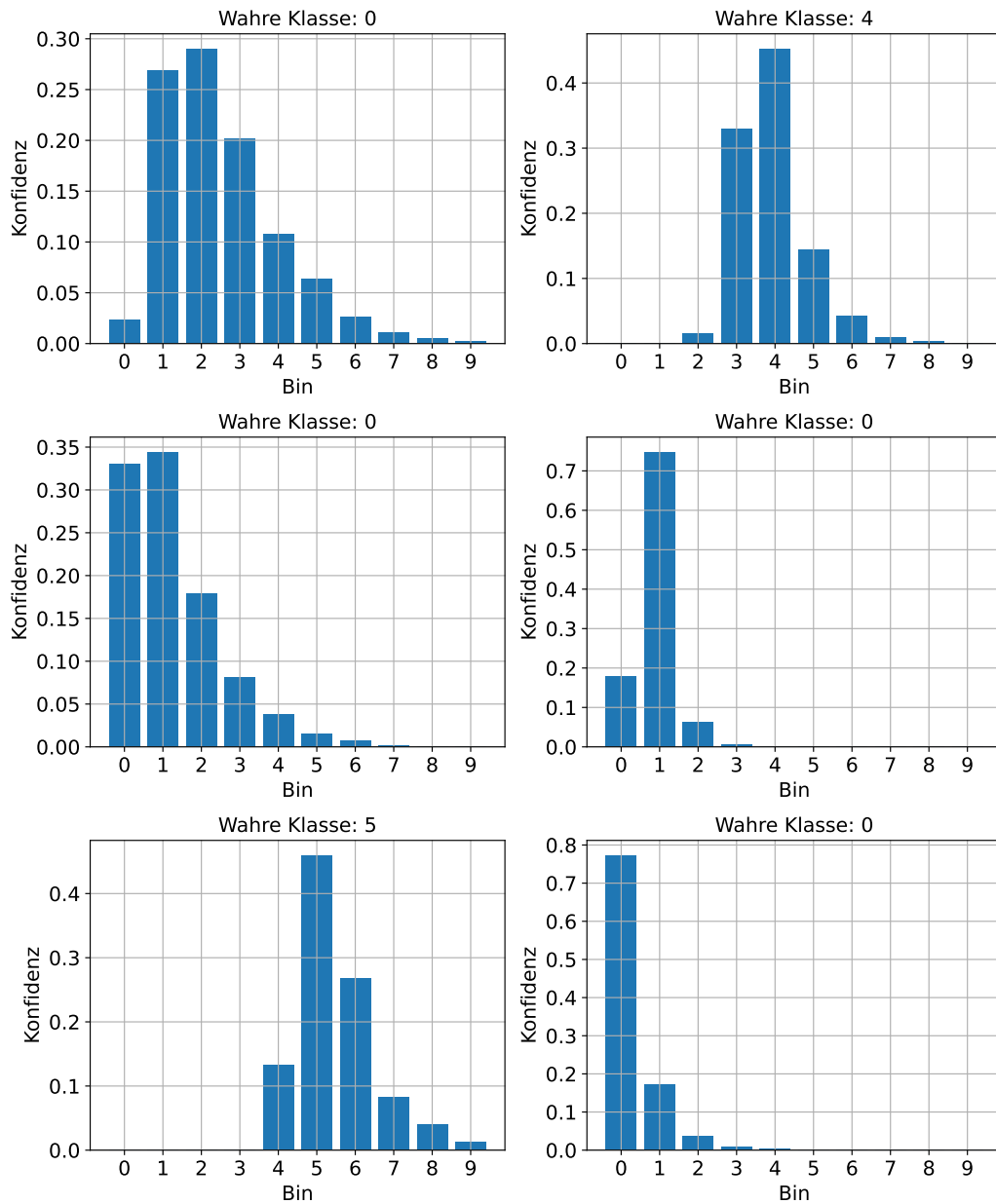


Abbildung C.6: Vorhersage einzelner Events des 1. Modells (*one_model=False*). Jedem Energie-Bin wird eine Konfidenz zugewiesen. Sie gibt die Zugehörigkeit des Events zu diesem Bin an.

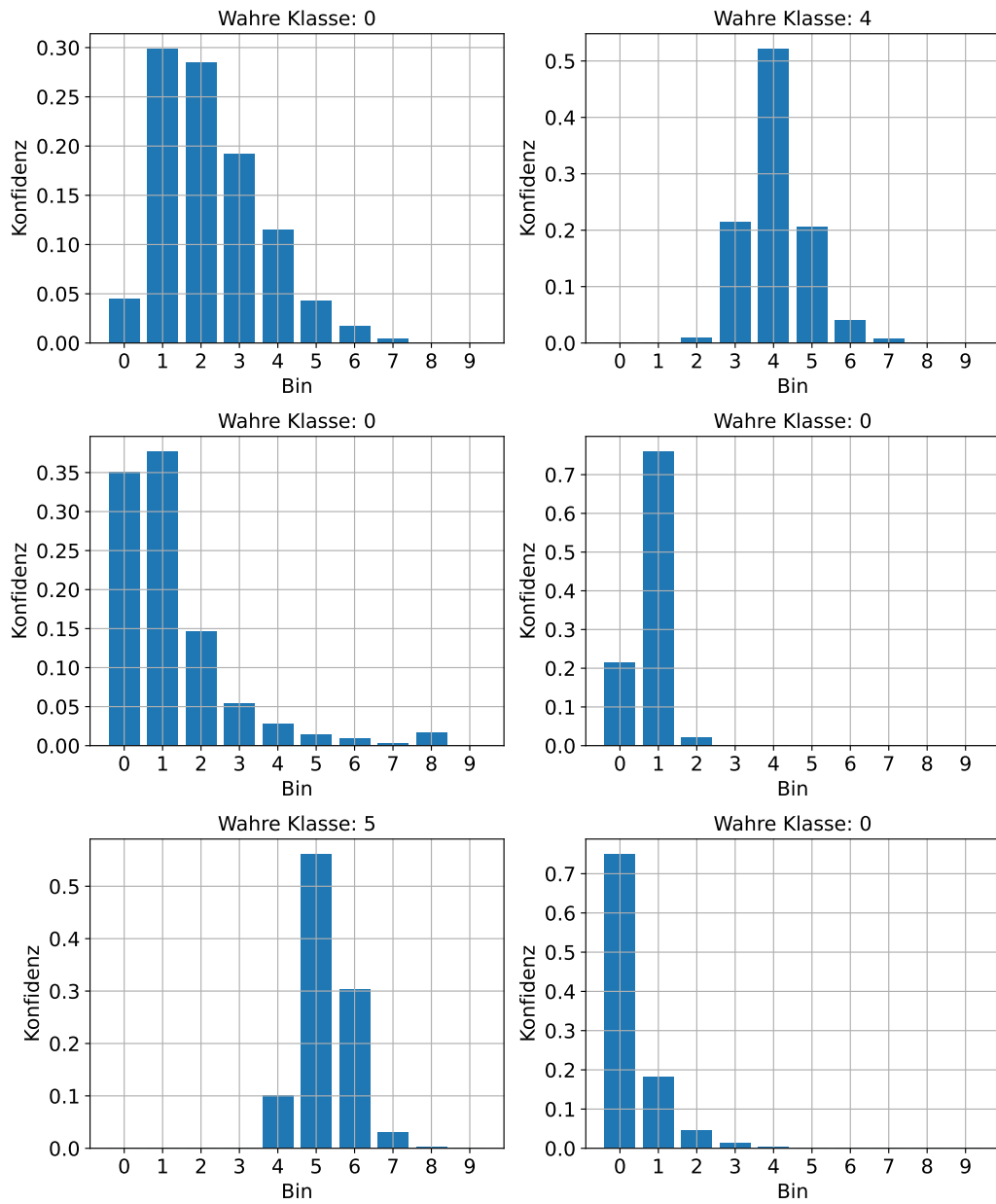


Abbildung C.7: Vorhersage einzelner Events des 2. Modells (*one_model=True*). Jedem Energie-Bin wird eine Konfidenz zugewiesen. Sie gibt die Zugehörigkeit des Events zu diesem Bin an.

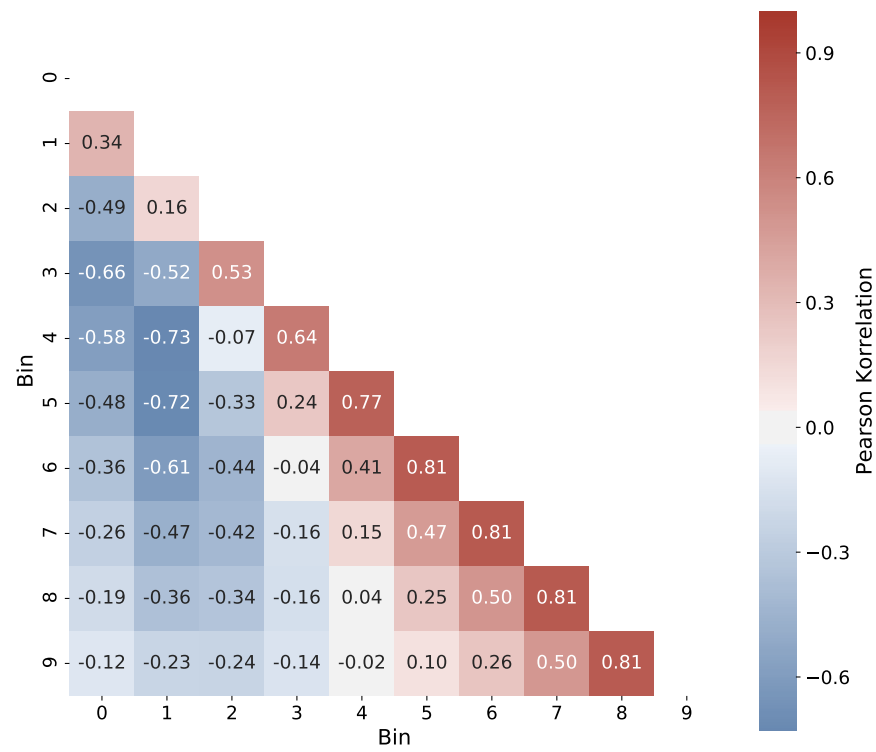


Abbildung C.8: Die Korrelationsmatrix gibt die Korrelationen zwischen den entfalteten Energie-Bins an. Es handelt sich um das 1. Modell (*one_model=False*).

D Abhängigkeit der Entfaltung vom Trainingspektrum

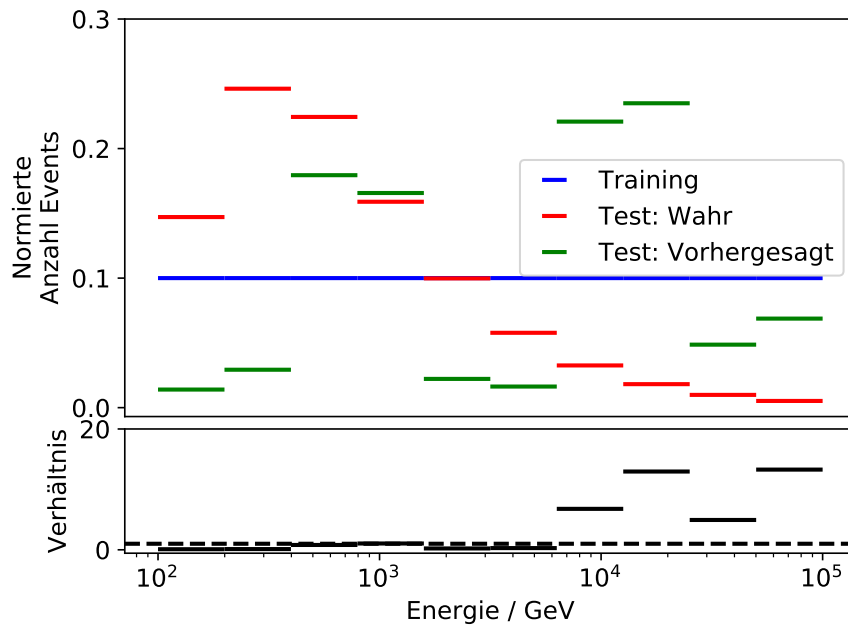


Abbildung D.1: Das mit einem NN in DSEA entfaltete Energiespektrum der Neutrinos. Das NN (Modell 2) wird in DSEA auf einem Datensatz mit gleichverteilten Klassen trainiert. Zum Vergleich der Spektren der Trainingsdaten (Blau), der wahren Evaluationsdaten (Rot) und der entfalteten Evaluationsdaten (Grün).

Abbildungsverzeichnis

2.1	IceCube-Detektor und IceTop Array	2
2.2	Neutrinofluss in Abhängigkeit der Energie	4
3.1	Struktur eines NN mit einer tiefen Ebene	7
3.2	Aufbau des verwendeten NN	8
5.1	Graphische Darstellung der Ausgabe des NN	13
5.2	Verlauf des Trainingsprozesses des NN ohne DSEA	13
5.3	Spektrum des NN ohne DSEA bei einer Klassifikation nach dem „maximum a posteriori“-Prinzip	14
5.4	Spektrum des NN ohne DSEA mit Beachtung der Wahrscheinlich- keitsinterpretation	15
5.5	Korrelationsmatrix des NN ohne DSEA	16
5.6	Ergebnisse der Gittersuche für ein NN in DSEA	19
5.7	Verlauf des Trainingsprozess des 2. Modells in DSEA	20
5.8	Spektrum des 2. Modells in DSEA	21
5.9	Korrelationsmatrix des 2. Modells in DSEA	22
5.10	Überprüfung des Bias: Spektrum des NN ohne DSEA	23
5.11	Überprüfung des Bias: Spektrum des 1. Modells in DSEA	24
B.1	Verlauf der Accuracy für den Trainingsprozess des NN ohne DSEA .	28
B.2	Ergebnisse des Bootstrapping-Vefahrens für das NN ohne DSEA . .	29
B.3	Vorhersage einzelner Events des NN ohne DSEA	30
C.1	Ergebnisse der Gittersuche für mehrere (Anzahl: $n_{\text{Iterationen}}$) NN in DSEA	34
C.2	Verlauf des Trainingsprozesses des 1. Modells in DSEA	35
C.3	Spektrum des 1. Modells in DSEA	35
C.4	Ergebnisse des Bootstrapping-Vefahrens für das 1. Modell in DSEA .	36
C.5	Ergebnisse des Bootstrapping-Vefahrens für das 2. Modell in DSEA .	37
C.6	Vorhersage einzelner Events des 1. Modells in DSEA	38
C.7	Vorhersage einzelner Events des 2. Modells in DSEA	39
C.8	Korrelationsmatrix des 1. Modells in DSEA	40

D.1 Überprüfung des Bias: Spektrum des 2. Modells in DSEA	41
---	----

Literatur

- [1] M. G. Aartsen et al. „Measurement of the Atmospheric ν_e Flux in IceCube“. In: *Phys. Rev. Lett.* (2013). URL: <https://link.aps.org/doi/10.1103/PhysRevLett.110.151105>.
- [2] Markus Ahlers, Klaus Helbing und Carlos Pérez de los Heros. „Probing particle physics with IceCube“. In: *The European Physical Journal C* 78.11 (2018). URL: <https://doi.org/10.1140/epjc/s10052-018-6369-9>.
- [3] S. M. Bilenky und S. T. Petcov. „Massive neutrinos and neutrino oscillations“. In: *Rev. Mod. Phys.* 59 (1987), S. 671–754. URL: <https://link.aps.org/doi/10.1103/RevModPhys.59.671>.
- [4] Jason Brownlee. „What is the Difference Between a Batch and an Epoch in a Neural Network“. In: *Machine Learning Mastery* 20 (2018).
- [5] Mirko Bunse. „DSEA Rock-Solid Regularization and Comparison with other Deconvolution Algorithms“. Masterarbeit. Technische Universität Dortmund, Deutschland, 2018.
- [6] Juan Carlos. *Simulation Production IceCube. Dataset 11374*. 2015. URL: <https://grid.icecube.wisc.edu/simulation/dataset/11374>.
- [7] G. Cybenko, D.P. O’Leary und J. Rissanen. *The Mathematics of Information Coding, Extraction and Distribution*. The IMA Volumes in Mathematics and its Applications. Springer New York, 1998. ISBN: 9780387986654.
- [8] Tyce DeYoung. „NEUTRINO ASTRONOMY WITH ICECUBE“. In: *Modern Physics Letters A* 24 (2009), S. 1543–1557.
- [9] Gene H. Golub, Per Christian Hansen und Dianne P. O’Leary. „Tikhonov Regularization and Total Least Squares“. In: *SIAM Journal on Matrix Analysis and Applications* 21.1 (1999), S. 185–194. URL: <https://doi.org/10.1137/S0895479897326432>.
- [10] Diederik P. Kingma und Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. URL: <https://arxiv.org/abs/1412.6980>.
- [11] Lev Davidovich Landau et al. *Electrodynamics of continuous media*. Bd. 8. elsevier, 2013.

- [12] Jeffrey B. Lewis und Keith T. Poole. „Measuring Bias and Uncertainty in Ideal Point Estimates via the Parametric Bootstrap“. In: *Political Analysis* 12.2 (2004), S. 105–127. DOI: 10.1093/pan/mph015.
- [13] Xiuwen Liu und DeLiang Wang. „A spectral histogram model for texton modeling and texture discrimination“. In: *Vision Research* 42.23 (2002), S. 2617–2634. ISSN: 0042-6989. DOI: [https://doi.org/10.1016/S0042-6989\(02\)00297-3](https://doi.org/10.1016/S0042-6989(02)00297-3).
- [14] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: <https://www.tensorflow.org/>.
- [15] Börner Mathis. „Der Entfaltungsalgorithmus DSEA. Systematische Studien und Anwendung auf astrophysikalische Fragestellungen“. Masterarbeit. Technische Universität Dortmund, Deutschland, 2014.
- [16] Martin Ha Minh. *Reconstruction of Neutrino Events in title= TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: <https://www.tensorflow.org/>.
- [17] Titus Neupert et al. „Introduction to Machine Learning for the Sciences“. In: *ArXiv* (2021). URL: <https://arxiv.org/abs/2102.04883>.
- [18] Dabal Pedamonti. *Comparison of non-linear activation functions for deep neural networks on MNIST classification task*. 2018. URL: <https://arxiv.org/abs/1804.02763>.
- [19] F. Pedregosa et al. „Scikit-learn: Machine Learning in Python“. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830.
- [20] Tim Ruhe. „Data mining on the rocks: A measurement of the atmospheric muon neutrino flux using IceCube in the 59-string configuration and a novel data mining based approach to unfolding“. Diss. Technical University of Dortmund, Germany, Jan. 2013.
- [21] Mohammad S Sorower. „A literature survey on algorithms for multi-label learning“. In: *Oregon State University, Corvallis* 18 (2010), S. 1–25.
- [22] Christian Spiering. „Towards high-energy neutrino astronomy“. In: *The European Physical Journal H* 37.3 (2012), S. 515–565. URL: <https://doi.org/10.1140/2Fepjh%2Fe2012-30014-2>.
- [23] Sean Twomey. „On the numerical solution of Fredholm integral equations of the first kind by the inversion of the linear system produced by quadrature“. In: *Journal of the ACM (JACM)* 10.1 (1963), S. 97–101.
- [24] Xing Wan. „Influence of feature scaling on convergence of gradient iterative algorithm“. In: *Journal of physics: Conference series*. Bd. 1213. 3. IOP Publishing. 2019, S. 032021.

- [25] Lincoln Wolfenstein. „Neutrino oscillations in matter“. In: *Solar Neutrinos*. CRC Press, 2018, S. 294–299.
- [26] R. L. Workman et al. „Review of Particle Physics“. In: *PTEP* (2022). DOI: 10.1093/ptep/ptac097.

Danksagung

Ich möchte mich bei dem gesamten Lehrstuhl bedanken, bei dem ich mich die letzten Monate sehr gut aufgehoben gefühlt habe. Besonderen Dank gilt dabei Prof. Dr. Dr. Wolfgang Rhode und meinen direkten Ansprechpartnern Karolin Hymon, Leonora Kardum und Tim Ruhe, die mir bei jeglichen Fragen weitergeholfen haben.

Die Diskussionen und Anregungen im wöchentlichen Meeting mit der DSEA-Gruppe haben die Arbeit erst ermöglicht. Auch möchte ich Mirko Bunse für die Einbringung der technischen Seite danken.

Mein Dank gilt ebenfalls Nicolai Weitkämper, der seine Bachelorarbeit parallel zu einem ähnlichen Thema geschrieben hat. So konnten wir uns bei organisatorischen Dingen und technischen Problemen unterstützen und über inhaltliche Aspekte diskutieren.

Eidesstattliche Versicherung

(Affidavit)

Haefs, Samuel

209887

Name, Vorname
(surname, first name)

Matrikelnummer
(student ID number)

☒ Bachelorarbeit
(Bachelor's thesis)

☐ Masterarbeit
(Master's thesis)

Titel
(Title)

Lösungen inverser Probleme:


Entfaltung von Energiespektren mit neuronalen Netzen in DSEA

Ich versichere hiermit an Eides statt, dass ich die vorliegende Abschlussarbeit mit dem oben genannten Titel selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

I declare in lieu of oath that I have completed the present thesis with the above-mentioned title independently and without any unauthorized assistance. I have not used any other sources or aids than the ones listed and have documented quotations and paraphrases as such. The thesis in its current or similar version has not been submitted to an auditing institution before.

Dortmund, 25.08.2022

Ort, Datum
(place, date)


Unterschrift
(signature)

Belehrung:

Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -).

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird ggf. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin“) zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Official notification:

Any person who intentionally breaches any regulation of university examination regulations relating to deception in examination performance is acting improperly. This offense can be punished with a fine of up to EUR 50,000.00. The competent administrative authority for the pursuit and prosecution of offenses of this type is the Chancellor of TU Dortmund University. In the case of multiple or other serious attempts at deception, the examinee can also be unenrolled, Section 63 (5) North Rhine-Westphalia Higher Education Act (*Hochschulgesetz, HG*).

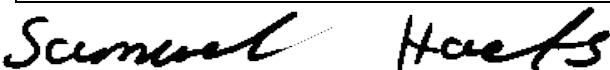
The submission of a false affidavit will be punished with a prison sentence of up to three years or a fine.

As may be necessary, TU Dortmund University will make use of electronic plagiarism-prevention tools (e.g. the "turnitin" service) in order to monitor violations during the examination procedures.

I have taken note of the above official notification:*

Dortmund, 25.08.2022

Ort, Datum
(place, date)


Unterschrift
(signature)

***Please be aware that solely the German version of the affidavit ("Eidesstattliche Versicherung") for the Bachelor's/ Master's thesis is the official and legally binding version.**