

# **Frankfurt University of Applied Sciences**

Fachbereich 2: Informatik & Ingenieurwissenschaften



Bachelorarbeit zur Erlangung des akademischen Grades

Bachelor of Science

(B. Sc.)

## **Stochastischer Gradientenabstieg zum Trainieren neuronaler Netze: Implementierung und vergleichende Untersuchung des RMSprop-Lernverfahrens für den n++-Simulator**

Eingereicht von Artur Brening  
geboren am 06.02.1992 in Krasnoturjinsk, Russland

Erstgutachter: Prof. Dr. Thomas Gabel

Zweitgutachter: Prof. Dr. Christian Baun

## **Vorwort**

Die vorliegende Arbeit wurde im Zeitraum vom 15.07.2020 bis 16.10.2020 unter der Anleitung von Prof. Dr. Thomas Gabel des Fachbereichs für Informatik & Ingenieurwissenschaften der University of Applied Sciences in Frankfurt am Main angefertigt.

Ein besonderer Dank geht an Herr Prof. Dr. Thomas Gabel für die Überlassung des interessanten Themas. Außerdem möchte ich mich für die vielen Hilfen hinsichtlich von Problemen, den anregenden Diskussionen bezüglich der Vorgehensweisen und Rechnungen, die ständige Erreichbarkeit während der SARS-CoV-2-Pandemie und den Zugang zu dem FRA-UNIted-Teamserver bedanken.

Meiner Frau, Tamara, möchte ich herzlichst für die Unterstützung während und vor der Bachelorarbeit danken. Sie hat mir Kraft gegeben, mich stets aufgebaut und mir ein offenes Ohr geliehen.

Abschließend möchte ich meiner Familie, der Familie meiner Frau sowie meinen Freunden danken, die zu jeder Zeit für mich da waren und mich ebenfalls unterstützt haben.

# Inhaltsverzeichnis

1.	Einleitung .....	1
1.1.	Aufgabenstellung .....	2
1.2.	Gliederung .....	2
2.	Grundlagen .....	4
2.1.	Maschinelles Lernen.....	4
2.1.1.	Überwachtes Lernen.....	4
2.1.2.	Unüberwachtes Lernen .....	5
2.1.3.	Bestärkendes Lernen .....	5
2.2.	Einfaches Perzeptron .....	6
2.3.	Vorwärtsgerichtetes neuronales Netzwerk .....	9
2.4.	Vorwärtspropagierung .....	11
2.5.	Gradientenabstiegsverfahren.....	12
2.5.1.	Stapel-Gradientenabstieg.....	15
2.5.2.	Stochastischer Gradientenabstieg.....	16
2.5.3.	Mini-Stapel-Gradientenabstieg .....	16
2.6.	Gradientenabstiegsalgorithmen.....	17
2.6.1.	Backpropagation .....	18
2.6.2.	Resilient Propagation .....	19
2.6.3.	Root Mean Square Propagation .....	21
3.	Implementierung von RMSprop in den n++-Simulatorkern.....	23
3.1.	Erläuterung des n++-Simulatorkerns.....	23
3.2.	Vorstellung der Implementierung von RMSprop .....	23

4.	Experimentelle Untersuchungen .....	27
4.1.	Iris.....	29
4.1.1.	Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate....	30
4.1.2.	Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors .....	32
4.1.3.	Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfefaktors.....	34
4.1.4.	Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel.....	35
4.1.5.	Vergleich der Gradientenabstiegsalgorithmen.....	36
4.2.	Yacht Hydrodynamics .....	38
4.2.1.	Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate....	39
4.2.2.	Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors .....	41
4.2.3.	Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfefaktors.....	42
4.2.4.	Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel.....	44
4.2.5.	Vergleich der Gradientenabstiegsalgorithmen.....	45
4.3.	Real Estate Valuation .....	47
4.3.1.	Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate....	48
4.3.2.	Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors .....	50
4.3.3.	Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfefaktors.....	51

4.3.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel.....	53
4.3.5. Vergleich der Gradientenabstiegsalgorithmen.....	54
4.4. Balance Scale .....	56
4.4.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate....	57
4.4.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors .....	59
4.4.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors.....	60
4.4.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel.....	61
4.4.5. Vergleich der Gradientenabstiegsalgorithmen.....	63
4.5. QSAR Biodegradation.....	65
4.5.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate....	68
4.5.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors .....	69
4.5.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors.....	70
4.5.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel.....	72
4.5.5. Vergleich der Gradientenabstiegsalgorithmen.....	73
4.6. Car Evaluation.....	75
4.6.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate....	77
4.6.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors .....	79

4.6.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors.....	80
4.6.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel.....	81
4.6.5. Vergleich der Gradientenabstiegsalgorithmen.....	83
4.7. Wine Quality Red.....	85
4.7.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate....	86
4.7.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors .....	88
4.7.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors.....	90
4.7.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel.....	91
4.7.5. Vergleich der Gradientenabstiegsalgorithmen.....	93
4.8. Wine Quality White.....	95
4.8.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate....	97
4.8.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors .....	98
4.8.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors.....	100
4.8.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel.....	101
4.8.5. Vergleich der Gradientenabstiegsalgorithmen.....	103
4.9. MAGIC Gamma Telescope .....	105
4.9.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate..	106

4.9.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors .....	108
4.9.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors.....	109
4.9.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel.....	111
4.9.5. Vergleich der Gradientenabstiegsalgorithmen.....	112
4.10. Zusammenfassung der experimentellen Untersuchungen.....	114
5. Diskussion der Ergebnisse .....	117
6. Fazit .....	122
7. Tabellenverzeichnis .....	123
8. Formelverzeichnis.....	124
9. Abbildungsverzeichnis.....	126
10. Literaturverzeichnis .....	133

## 1. Einleitung

Seien es Roboter, die die Gestalt eines Tieres besitzen, Multiagentensysteme, die sich zu vogelähnlichen Schwärmen organisieren, oder evolutionäre Algorithmen, die die Evolution natürlicher Lebewesen nachahmen, die Informatik lässt sich oftmals von der Natur inspirieren. Doch das menschliche Gehirn und dessen Fähigkeit, aus Erfahrung zu lernen, beschäftigt Wissenschaftler bereits eine längere Zeit. Um diese Lernfähigkeit zu simulieren, haben Wissenschaftler die Funktionsweise des Elementarbestandteils des Gehirns, das sogenannte Neuron, betrachtet. Dieses besteht aus einem Zellkörper, mehreren Dendriten und einem Axon. Die Dendriten sind Fortsätze des Neurons und erhalten elektrische Signale vorgesetzter Neuronen, mit denen sie in Verbindung stehen. Dendriten agieren demnach als eine Eingangsebene für Informationen. Die eingehenden elektrischen Signale werden anschließend an den Zellkörper übertragen. Im Zellkörper werden die elektrischen Signale aufsummiert und anschließend, bei Überschreitung eines bestimmten Schwellenwertes, weitergeleitet. Dazu wird das elektrische Signal vom Zellkörper über das Axon, welcher über den sogenannten synaptischen Spalt mit Dendriten nachgeschalteter Neuronen in Verbindung steht, geleitet. Das elektrische Signal des Axons wird am synaptischen Spalt zu chemischen Signalen, sogenannte Botenstoffe, umgewandelt und in den synaptischen Spalt ausgeschüttet. Abhängig von der Menge der ausgeschütteten Botenstoffe werden in den Dendriten nachgeschalteter Neuronen elektrische Signale erzeugt und somit Informationen zwischen Neuronen ausgetauscht. Das Axon agiert demnach als Ausgangsebene für Informationen. Die Lernfähigkeit des Gehirns basiert im Aufbau und der Verstärkung bzw. Schwächung existenter Verbindungen zwischen Neuronen. Mit dem Neuron als Vorbild gelang es Frank Rosenblatt 1958 ein mathematisches Modell für ein künstliches Neuron aufzustellen, den sogenannten einfachen Perzeptron, der selbst zur heutigen Zeit als Grundlage zur Entwicklung von künstlichen neuralen Netzwerken verwendet wird [1]. Diese künstlichen neuralen Netzwerke können mit Hilfe von Lernalgorithmen trainiert werden, um diverse Klassifikations- und Regressionsprobleme lösen zu können. Heutzutage werden künstliche neuronale Netzwerke in zahlreichen Bereichen, wie z.B. der Bioinformatik, Bild- und Spracherkennung und der Finanzprognose, angewandt [2, 3, 4]. Abhängig von der Menge an Daten, die zum Training des künstlichen neuronalen Netzes verwendet werden, muss jedoch die Wahl des verwendeten Lernalgorithmus bedacht werden.

## 1.1. Aufgabenstellung

Der neuartige Gradientenabstiegsalgorithmus RMSprop wird, obwohl es von dessen Erfinder, G. Hinton, keine Publikation dazu gibt, von vielen Quellen verwendet und ist sogar in den gängigsten Applikationen zum überwachten Lernen mittels künstlicher neuronaler Netzwerke implementiert [5]. Da der Gradientenabstiegsalgorithmus Rprop den gesamten Datensatz verwenden muss, um die Aktualisierungswerte der einzelnen Gewichte zu bestimmen und somit einen stabilen Gradientenabstieg zu gewährleisten, eignet sich Rprop nicht in Form eines stochastischen Gradientenabstiegs oder eines Mini-Stapel-Gradientenabstiegs [6]. Somit stößt Rprop bei sehr großen Datensätzen an seine Grenzen. Der Gradientenabstiegsalgorithmus BP kann zwar in Form eines stochastischen Gradientenabstiegs verwendet werden, kann jedoch steile Täler und flache Ebenen einer Kostenfunktion schlecht bewältigen und hat dementsprechend Schwierigkeiten bei hochkomplexen Kostenfunktionen. RMSprop stellt eine vielversprechende Alternative zu Rprop und BP dar und kann, da es sich um einen Mini-Stapel-Gradientenabstiegsalgorithmus handelt, auch für große Datensätze verwendet werden. Das Ziel der vorliegenden Arbeit ist es, unter Verwendung des n++-Simulatorkerns, den Lernerfolg des neuartigen Gradientenabstiegsalgoritmus RMSprop mit den Gradientenabstiegsalgorithmen, BP und Rprop, zu vergleichen. Dazu muss RMSprop vorher im n++-Simulatorkern implementiert werden. Anschließend sollen künstliche neuronale Netzwerke an ausgewählten Datensätzen mit Hilfe der beschriebenen Gradientenabstiegsalgorithmen trainiert und der Lernerfolg miteinander verglichen werden. Ferner sollen die Auswirkungen der einzelnen Parameter von RMSprop, sowie die Größe der verwendeten Mini-Stapel, auf den Lernerfolg der künstlichen neuronalen Netzwerke untersucht werden.

## 1.2. Gliederung

Kapitel 2 umfasst den theoretischen Teil der Bachelorarbeit. Zu Beginn des Kapitels wird der Begriff des maschinellen Lernens erläutert und ein Überblick über die verschiedenen Arten des maschinellen Lernens gegeben. Anschließend wird der Aufbau und die Funktionsweise des einfachen Perzeptrons sowie von vorwärtsgerichteten künstlichen neuronalen Netzwerken vorgestellt. Zuletzt wird auf die Vorwärtspropagierung und das Gradientenabstiegsverfahren eingegangen und die Gradientenabstiegsalgorithmen BP, Rprop und RMSprop erläutert.

In Kapitel 3 wird der n++-Simulatorkern, der im Rahmen der experimentellen Untersuchungen zum Aufbau und Training der vorwärtsgerichteten künstlichen neuronalen Netzwerke verwendet wurde, vorgestellt und dessen Funktionalitäten zusammengefasst. Daraufhin wird detailliert auf die Implementierung von RMSprop in den n++-Simulatorkern eingegangen.

In Kapitel 4 wird zuerst ausführlich auf die allgemeine Vorgehensweise der experimentellen Untersuchungen eingegangen. Anschließend werden die einzelnen Datensätze, an denen die vergleichenden Untersuchungen zwischen den Gradientenabstiegsalgorithmen BP, Rprop und RMSprop durchgeführt und der Effekt der einzelnen Parameter von RMSprop untersucht wurden, vorgestellt und die Ergebnisse der experimentellen Untersuchungen präsentiert. Zum Schluss werden die ermittelten Ergebnisse zusammengefasst.

Kapitel 5 schließt die Bachelorarbeit ab, indem die Ergebnisse der experimentellen Untersuchungen diskutiert werden.

## 2. Grundlagen

Die folgenden Kapitel erläutern den Aufbau und die Funktionsweise des einfachen Perzeptrons sowie von vorwärtsgerichteten künstlichen neuralen Netzwerken und stellen damit die Grundlagen für das maschinelle Lernen mittels künstlicher neuronaler Netzwerke dar. Dazu wird zu Beginn der Begriff des maschinellen Lernens erörtert und näher auf die Vorwärtspropagierung und das Gradientenabstiegsverfahren eingegangen. Zum Schluss wird ein Überblick über die verschiedenen Arten des Gradientenabstiegsverfahrens gegeben und die drei verwendeten Gradientenabstiegsalgorithmen für künstliche neuronale Netzwerke, BP (engl. Backpropagation), Rprop (engl. Resilient Propagation) und RMSprop (engl. Root Mean Square Propagation), vorgestellt.

### 2.1. Maschinelles Lernen

Unter dem maschinellen Lernen versteht man den vollständig automatisierten Prozess eines Systems auf Basis von Daten zu lernen und sich hinsichtlich zukünftiger Aktivitäten zu verbessern [7]. Somit ist das maschinelle Lernen eine Unterkategorie der künstlichen Intelligenz, in dem Algorithmen entwickelt werden, die selbstständig auf Daten zugreifen und aus ihnen lernen können, um die erworbenen Erfahrungen auf noch unbekannte Daten anwenden zu können und bestimmte Aufgaben zu bewerkstelligen [8, 9]. Dies kann unter anderem die Vorhersage von Werten oder Wahrscheinlichkeiten auf Basis bereits analysierter Daten oder die Erkennung von Gruppen mit ähnlichen Eigenschaften in einem Datensatz sein [7]. Dafür wird ein ausgewähltes Modell mit einem vorbereiteten Datensatz optimiert und dessen Lernfortschritt beobachtet. Im Allgemeinen wird das maschinelle Lernen in drei verschiedene Typen kategorisiert, dem überwachten Lernen (engl. Supervised Learning), dem unüberwachten Lernen (engl. Unsupervised Learning) und dem bestärkenden Lernen (engl. Reinforcement Learning) [9].

#### 2.1.1. Überwachtes Lernen

Beim überwachten Lernen (engl. Supervised Learning) wird ein Modell mit einem definierten Bruchteil des gesamten Datensatzes, den Trainingsbeispielen, trainiert. Für jedes Trainingsbeispiel

existiert ein Zielwert, dessen Zusammenhang von dem ausgewählten Modell erlernt werden soll. Der Zielwert kann entweder eine Regression oder eine Klassifikation sein. Nach dem Training des Modells wird der restliche Bruchteil des gesamten Datensatzes, die Testbeispiele, dazu verwendet den Lernfortschritt des jeweiligen Modells mitzuverfolgen [9]. Unter das überwachte Lernen fallen z.B. Modelle wie Support Vector Machines, künstliche neuronale Netzwerke (engl. Artificial Neural Networks) oder Random Forests [7]. Es existiert ebenfalls eine Mischform zwischen dem überwachten Lernen und dem unüberwachten Lernen, bei der Daten mit bekannten Zielwerten dazu genutzt werden ein ausgewähltes Modell zu trainieren und mit Hilfe des Modells anschließend Daten, die keinen Zielwert besitzen, einen Zielwert zuzuordnen. Das überwachte Lernen mit Hilfe von künstlichen neuronalen Netzwerken ist Hauptbestandteil dieser Arbeit.

### 2.1.2. Unüberwachtes Lernen

Beim unüberwachten Lernen (engl. Unsupervised Learning) existieren, im Gegensatz zum überwachten Lernen, zu den einzelnen Daten keine bekannten Zielwerte. Folglich sind die möglichen Ergebnisse der einzelnen Daten unbekannt und lassen sich nicht dazu nutzen ein Modell, wie bei dem überwachten Lernen, zu trainieren. Die verwendeten Modelle werten dementsprechend die bereitgestellten Daten explorativ aus und versuchen Rückschlüsse auf versteckte Gruppen bzw. Strukturen aus diesen Daten zu ziehen [9]. Unter das unüberwachte Lernen fallen z.B. Modelle wie k-Means-Algorithm und Principal Component Analysis [7].

### 2.1.3. Bestärkendes Lernen

Bei dem bestärkenden Lernen (engl. Reinforcement Learning) werden, im Gegenzug zu den vorherigen genannten Kategorien, keine Datensätze und kein menschliches Vorwissen benötigt, um ein ausgewähltes Modell zu optimieren. Allerdings soll ein Modell iterativ in einer definierten Umgebung einen vorgegebenen Endzustand erreichen. Um dies zu bewerkstelligen, soll das Modell selbstständig eine Strategie erlernen [7]. Dabei erhält das Modell bei jeder Interaktion mit der Umgebung über eine Kostenfunktion oder ein Belohnungssystem eine Rückmeldung. Dadurch

## 2. Grundlagen

gewinnt das Modell an Informationen darüber, welche Interaktionen sich positiv bzw. negativ auf die Kostenfunktion bzw. das Belohnungssystem auswirken. Mit diesen Informationen wiederum lernt das Modell bestärkend die Kostenfunktion hinsichtlich des zu erreichenden Endzustands zu minimieren bzw. das Belohnungssystem zu maximieren [7].

### 2.2. Einfaches Perzeptron

Bei dem einfachen Perzeptron (engl. Perceptron), der in Abbildung 1 dargestellt ist, handelt es sich um eine mathematische Nachbildung des biologischen Neurons [1].

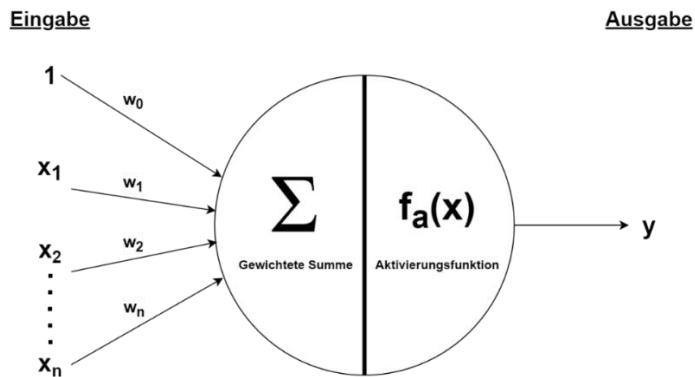


Abbildung 1: Vereinfachte Darstellung eines einfachen Perzeptrons.

Das einfache Perzeptron stellt einen Knoten dar, der mit Eingabewerten  $x_1, x_2, \dots, x_n$  über gewichtete Kanten  $w_1, w_2, \dots, w_n$  verbunden ist. Zusätzlich zu den Eingabewerten ist der Knoten mit einem Schwellenwert  $b$ , auch Bias genannt, verbunden. Dieser hat einen konstanten Eingabewert, der häufig den Wert 1 beträgt, und ein zugehöriges Gewicht  $w_0$ . Die eingehenden Eingabewerte werden als gewichtete Summe gemäß (1) im Knoten aufsummiert.

$$Eingabe = w_0 b + \sum_{i=1}^n w_i x_i \quad (1)$$

Formel (1): Berechnung der gewichteten Summe anhand der Eingabewerte  $x_i$ , den zugehörigen Gewichten  $w_i$  und dem Schwellenwert  $b$  mit dem zugehörigen Gewicht  $w_0$ .

Die Gewichte sind ein Maß für die Signifikanz der jeweiligen Eingabewerte. Je geringer das Gewicht, desto weniger trägt er zur gewichteten Summe bei. Anschließend wird der Ausgabewert des einfachen Perzeptrons über eine sogenannte Aktivierungsfunktion  $f_a(x)$  bestimmt. Die Berechnung des Ausgabewerts, die auch Aktivierung genannt wird, erfolgt gemäß (2).

$$\text{Ausgabe} = f_a(\text{Eingabe}) \quad (2)$$

Formel (2): Berechnung des Ausgabewerts des einfachen Perzeptrons anhand der gewichteten Summe und der Aktivierungsfunktion  $f_a(x)$ .

Als Aktivierungsfunktion  $f_a(x)$  kann unter anderem die Identitätsfunktion, die Sigmoidfunktion, der hyperbolische Tangens, die Sprungfunktion oder ReLU (Rectified Linear Unit) verwendet werden. Mit Hilfe der Aktivierungsfunktion wird der Ausgabewert des Perzeptrons in einen bestimmten Bereich, abhängig von der verwendeten Funktion, eingeschränkt. Die genannten Aktivierungsfunktionen und deren erzielten Wertebereiche sind in Tabelle 1 aufgelistet und in Abbildung 2 übersichtlich dargestellt.

Tabelle 1: Aktivierungsfunktionen mit den dazugehörigen Formeln, Ableitungen und Wertebereichen.

Aktivierungsfunktion	Formel	Erste Ableitung	Wertebereich
1. Identitätsfunktion	$f_a(x) = x$	$f'_a(x) = 1$	$]-\infty, \infty[$
2. Sigmoidfunktion	$f_a(x) = \frac{1}{1 + e^{-x}}$	$f'_a(x) = f_a(x) * (1 - f_a(x))$	$]0, 1[$
3. Hyperbolischer Tangens	$f_a(x) = \tanh(x)$	$f'_a(x) = 1 - \tanh^2(x)$	$]-1, 1[$
4. ReLU	$f_a(x) = \begin{cases} 0 & \text{für } x < 0 \\ x & \text{für } x \geq 0 \end{cases}$	$f'_a(x) = \begin{cases} 0 & \text{für } x < 0 \\ 1 & \text{für } x \geq 0 \end{cases}$	$[0, \infty[$
5. Sprungfunktion	$f_a(x) = \begin{cases} +1 & x \geq 0 \\ -1 & x < 0 \end{cases}$ oder $f_a(x) = \begin{cases} +1 & x \geq 0 \\ 0 & x < 0 \end{cases}$	Keine Differenzierbarkeit	$\{-1, 1\}$ oder $\{0, 1\}$

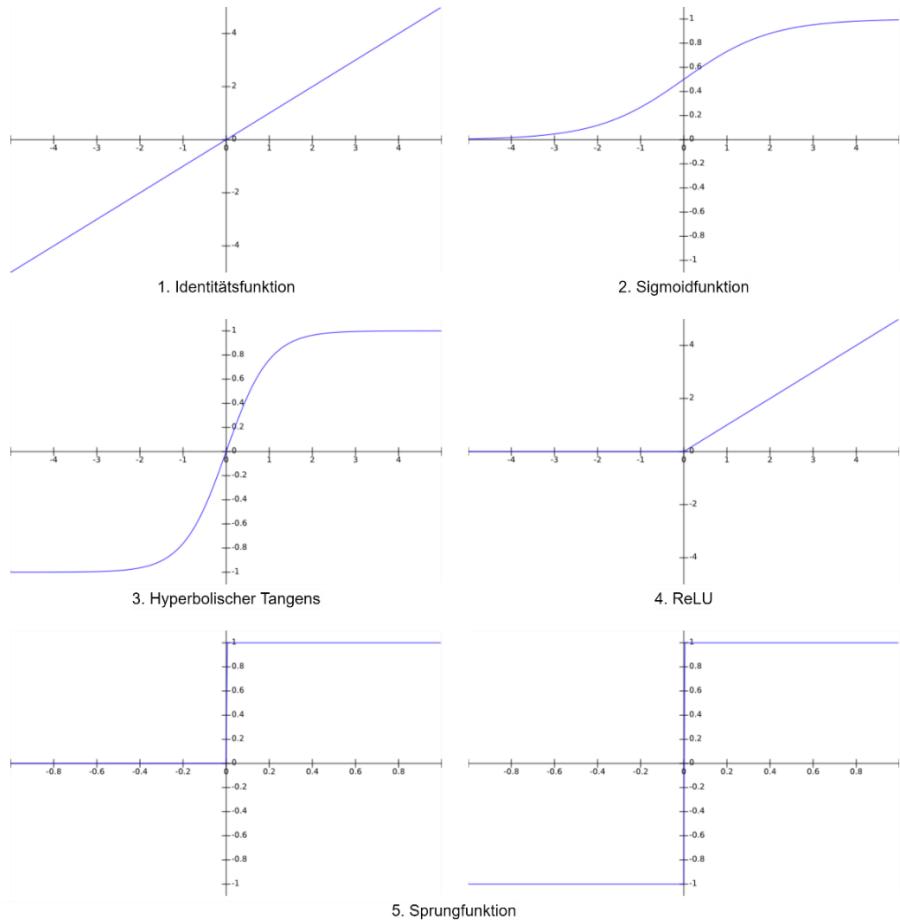


Abbildung 2: Übersicht ausgewählter Aktivierungsfunktionen.

Im Falle des in Abbildung 1 dargestellten einfachen Perzeptrons wird eine Sprungfunktion mit einem Wertebereich von  $\{0, 1\}$  verwendet. Ein einfacher Perzeptron ist dementsprechend in der Lage aus einer beliebigen Anzahl aus Eingabewerten eine einzige, binäre Ausgabe zu erzeugen. Für die Lernalgorithmen, die im weiteren Verlauf noch erläutert werden, müssen die verwendeten Aktivierungsfunktionen jedoch differenzierbar sein [10]. Wird jedoch eine andere Aktivierungsfunktion als die Sprungfunktion verwendet, spricht man nicht mehr von einem einfachen Perzeptron, sondern im Allgemeinen von einem künstlichen Neuron [11]. Mit Hilfe des einfachen Perzeptrons lassen sich linear separierbare Klassifikationen wie beispielsweise die AND- und die OR-Funktion lösen, wohingegen es bei der XOR-Funktion bereits an seine Grenze stößt [12]. Der Grund dafür liegt darin, dass sich die Ergebnismenge der AND- und der OR-Funktion durch eine Hyperebene in zwei Klassen geteilt werden kann. Bei der XOR-Funktion ist dies jedoch nicht nur durch eine einzige Hyperebene möglich [10]. Um komplexere Aufgaben lösen zu können, werden mehrere künstliche Neuronen miteinander verbunden.

### 2.3. Vorwärtsgerichtetes neuronales Netzwerk

Ähnlich wie biologische Neuronen, lassen sich künstliche Neuronen zu Netzwerken verknüpfen. Das daraus resultierende Netzwerk aus künstlichen Neuronen wird auch als ein künstliches neuronales Netzwerk (engl. Artificial Neural Network) bezeichnet [11]. In Abbildung 3 ist ein solches künstliches neuronales Netzwerk vereinfacht dargestellt.

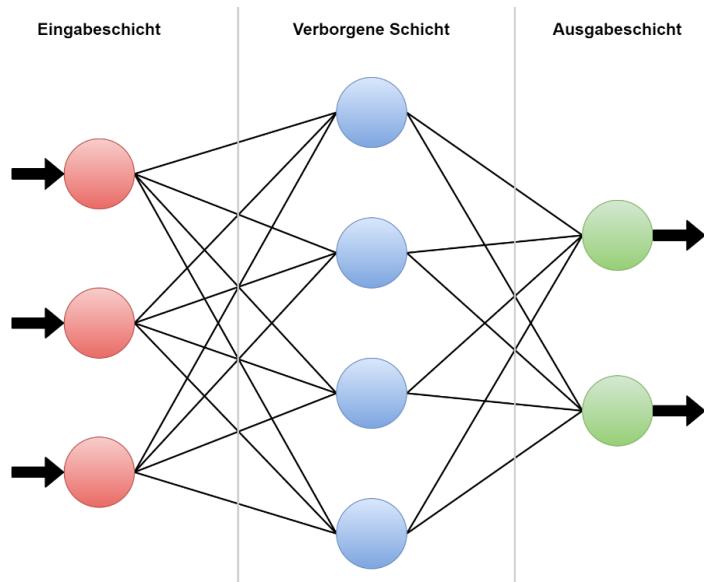


Abbildung 3: Aufbau eines künstlichen neuronalen Netzwerkes.

Die künstlichen Neuronen sind in Abbildung 3 als Knoten dargestellt. Typischerweise lassen sich die künstlichen Neuronen des künstlichen neuronalen Netzwerkes in drei Schichten unterteilen - eine Eingabeschicht, eine oder mehrere verborgene Schicht/en und eine Ausgabeschicht. Jedes künstliche Neuron einer Schicht ist mit jedem künstlichen Neuron der vorherigen und der nachfolgenden Schicht, sofern eine vorhanden ist, über gewichtete Kanten verbunden. Jedes künstliche Neuron in der verborgenen Schicht und in der Ausgabeschicht ist zusätzlich über eine gewichtete Kante mit einem konstanten Schwellenwert verbunden, der in Abbildung 3 nicht sichtbar ist. Besitzt das künstliche neuronale Netzwerk, unter der in Abbildung 3 dargestellten Topologie, mindestens eine verborgene Schicht, wird dies auch als ein vorwärtsgerichtetes neuronales Netzwerk (engl. Feedforward Neural Network) bezeichnet. Falls das künstliche neuronale Netzwerk mehrere verborgene Schichten besitzt, spricht man auch von einem tiefen vorwärtsgerichteten neuronalen Netzwerk [4]. Die Anzahl an Eingabeneuronen in der Eingabeschicht hängt von der Anzahl an Eingabewerten des jeweiligen Problems, das man lösen möchte, ab. Jedem Eingabeneuron der Eingabeschicht wird jeweils ein

## 2. Grundlagen

---

Eingabewert zugeordnet. Eingabeneuronen besitzen keine Aktivierungsfunktionen und ändern dementsprechend nicht den verwendeten Eingabewert [11]. Die künstlichen Neuronen in der verborgenen Schicht hingegen besitzen Aktivierungsfunktionen und verarbeiten die eingehenden, gewichteten Signale von den Eingabeneuronen. Die optimale Anzahl an verborgenen Neuronen und die Anzahl der verborgenen Schichten wird oftmals experimentell ermittelt, indem man die Anzahl der verborgenen Neuronen und der verborgenen Schichten variiert und den Lernerfolg des künstlichen neuronalen Netzwerkes betrachtet. Es existiert jedoch eine Daumenregel zur Ermittlung der Anzahl an verborgenen Neuronen [13]. Diese ist in (3) dargestellt.

$$|Neuron_{verborgen}| = \frac{2}{3} * |Neuron_{Eingabe}| + |Neuron_{Ausgabe}| \quad (3)$$

Formel (3): Daumenregel zur Ermittlung der Anzahl an verborgenen Neuronen,  $|Neuron_{verborgen}|$ , in der verborgenen Schicht aus der Anzahl der Neuronen in der Eingabeschicht,  $|Neuron_{Eingabe}|$ , und der Anzahl der Neuronen in der Ausgabeschicht,  $|Neuron_{Ausgabe}|$ .

Damit hängt die Anzahl der verborgenen Neuronen von der Anzahl der Eingabeneuronen und der Anzahl der Ausgabeneuronen ab. Die Anzahl der verborgenen Schichten hingegen hängt von der Komplexität des zu lösenden Problems ab. Ist ein Problem linear separierbar, wie z.B. die AND- oder die OR-Funktion, ist sogar keine verborgene Schicht notwendig, da die Ergebnismenge der AND- und der OR-Funktion durch eine Hyperebene in zwei Klassen geteilt werden kann [12]. Sobald ein Problem nicht-linear separiert werden muss, werden verborgene Schichten benötigt [12]. Für einfache Probleme sind oftmals zwei oder weniger verborgene Schichten ausreichend. Damit lassen sich die meisten Probleme mit einer hinreichenden Genauigkeit lösen [13]. Unter den verborgenen Schichten kann sich die Anzahl der verborgenen Neuronen jedoch unterscheiden. Die verborgenen Neuronen verarbeiteten die Eingabewerte der Eingabeneuronen und leiten diese an die Ausgabeschicht weiter und die Ausgabeneuronen geben das resultierende Ergebnis des künstlichen neuronalen Netzwerkes, für die jeweiligen Eingabewerte, aus. Hierbei kann es sich um ein Klassifikations- oder um ein Regressionsergebnis handeln, wovon auch die Anzahl der Ausgabeneuronen in der Ausgabeschicht abhängt. Bei Klassifikationsproblemen werden die jeweiligen Eingabewerte, im besten Fall, auf eine bestimmte Klasse von mehreren möglichen Klassen abgebildet. Die Anzahl der Ausgabeneuronen hängt bei Klassifikationsproblemen demnach von der Anzahl an möglichen Klassen, die in der Problemstellung abgebildet werden können, ab. Im Falle einer Klassifikation besitzen die Ausgabeneuronen oft die Sigmoidfunktion als Aktivierungsfunktion, die eine gebrochene Zahl im offenen Intervall  $]0, 1[$  ausgeben und damit eine Wahrscheinlichkeit für die Abbildung auf die jeweilige/n Klasse/n darstellen. Bei Regressionsproblemen hingegen werden die jeweiligen

Eingabewerte dazu verwendet eine bestimmte reelle Zahl zu erzeugen. Aufgrund dessen wird bei Regressionsproblemen meistens nur ein einzelnes Ausgabeneuron in der Ausgabeschicht benötigt. Dieses Ausgabeneuron besitzt oft die Identitätsfunktion als Aktivierungsfunktion, die eine beliebige reelle Zahl im offenen Intervall  $]-\infty, \infty[$  annehmen kann.

## 2.4. Vorwärtspropagierung

Mit Hilfe der Vorwärtspropagierung (engl. Feedforward) lässt sich aus bestimmten Eingabewerten die jeweilige Ausgabe eines vorwärtsgerichteten neuronalen Netzwerkes berechnen. Beginnend mit der Eingabeschicht, berechnet man die Ausgaben bzw. Aktivierungen der Neuronen in der ersten verborgenen Schicht. Die Aktivierungen der künstlichen Neuronen in der ersten verborgenen Schicht dienen wiederum als Eingaben der nächsten verborgenen Schicht, falls eine vorhanden ist. Dieser Vorgang wiederholt sich soweit, bis mit Hilfe der Aktivierungen der künstlichen Neuronen der letzten verborgenen Schicht die Aktivierungen der Ausgabeneuronen, die die Ausgabe des neuronalen Netzwerkes darstellen, berechnet werden können. Dieser Vorgang ist in (4) und (5) dargestellt [14].

$$z_k^{(l)} = w_{k,0}^{(l-1)} * b^{(l-1)} + w_{k,1}^{(l-1)} * a_1^{(l-1)} + \dots + w_{k,m}^{(l-1)} * a_m^{(l-1)} \quad (4)$$

Formel (4): Berechnung der Eingabe  $z_k^{(l)}$  des  $k$ -ten Neurons der  $l$ -ten Schicht mit  $b^{(l-1)}$  = Schwellenwert der  $(l-1)$ -ten Schicht,  $w_{k,m}^{(l-1)}$  = Gewicht der Kante des  $m$ -ten Neurons der  $(l-1)$ -ten Schicht zum  $k$ -ten Neuron der  $l$ -ten Schicht und  $a_m^{(l-1)}$  = Aktivierung des  $m$ -ten Neurons der  $(l-1)$ -ten Schicht.

$$a_k^{(l)} = f_a(z_k^{(l)}) \quad (5)$$

Formel (5): Berechnung der Aktivierung  $a_k^{(l)}$  des  $k$ -ten Neurons der  $l$ -ten Schicht mit  $f_a(x)$  = Aktivierungsfunktion des  $k$ -ten Neurons und  $z_k^{(l)}$  = Eingabe des  $k$ -ten Neurons der  $l$ -ten Schicht.

Dazu wird gemäß (4) die Eingabe des  $k$ -ten künstlichen Neurons in der  $l$ -ten Schicht aus der gewichteten Summe aller eingehenden Aktivierungen der künstlichen Neuronen in der  $(l-1)$ -ten Schicht berechnet. Das Resultat der gewichteten Summe wird schließlich gemäß (5) dazu verwendet die Aktivierung des  $k$ -ten künstlichen Neurons in der  $l$ -ten Schicht zu berechnen. Dafür wird die Aktivierungsfunktion des jeweiligen Neurons verwendet. Wenn alle Aktivierungen der

## 2. Grundlagen

---

künstlichen Neuronen in der  $l$ -ten Schicht berechnet wurden, wird die Aktivierungen der  $l$ -ten Schicht dazu verwendet die Aktivierungen der  $(l+1)$ -ten Schicht zu berechnen bis schließlich die Aktivierungen der Ausgabeneuronen, und folglich das Ergebnis des künstlichen neuronalen Netzwerkes hinsichtlich der jeweiligen Eingabewerte, feststehen.

### 2.5. Gradientenabstiegsverfahren

Nachdem man die Aktivierungen der Ausgabeneuronen eines vorwärtsgerichteten neuronalen Netzwerkes mittels der Vorwärtspropagierung ermittelt hat, wird auffallen, dass aus den verwendeten Eingabewerten nicht das erwünschte Resultat erzeugt wird. Der Grund dafür liegt darin, dass die Gewichte der einzelnen Kanten im künstlichen neuronalen Netzwerk auf die Problemstellung angepasst werden müssen. Hier kommt der Aspekt der Lernfähigkeit von künstlichen neuronalen Netzwerken ins Spiel. Mit Hilfe von Eingabewerten und den dazugehörigen erwarteten Ausgabewerten lässt sich der Fehler der tatsächlichen Ausgabe von der erwarteten Ausgabe berechnen. Dieser Fehler wird als eine sogenannte Kostenfunktion  $C$  zusammengefasst und kann z.B. die mittlere quadratische Abweichung (engl. Mean Squared Error, MSE) der tatsächlichen Ausgaben zu den jeweiligen erwarteten Ausgaben aller Ausgabeneuronen sein [14, 15]. Diese ist in (6) dargestellt.

$$C_{MSE} = \frac{1}{2} \sum_k (a_k^{(l)} - y_k)^2 \quad (6)$$

Formel (6): Berechnung der mittleren quadratischen Abweichung  $C_{MSE}$  aller Ausgabeneuronen mit  $a_k^{(l)}$  = Aktivierung des  $k$ -ten Neurons in der Ausgabeschicht ( $l$ ) und dem erwarteten Ausgabewert  $y_k$  des  $k$ -ten Ausgabeneurons.

Substituiert man nun  $a_k^{(l)}$  durch die in (4) und (5) dargestellten Ausdrücke, erhält man den in (7) dargestellten Ausdruck in der die Abhängigkeit der Kostenfunktion von den Aktivierungen der letzten verborgenen Schicht erkennbar ist.

$$C_{MSE} = \frac{1}{2} \sum_k (f_a(w_{k,0}^{(l-1)} * b^{(l-1)} + w_{k,1}^{(l-1)} * a_{k,1}^{(l-1)} + \dots + w_{k,m}^{(l-1)} * a_m^{(l-1)}) - y_k)^2 \quad (7)$$

Formel (7): Darstellung der mittleren quadratischen Abweichung aller Ausgabeneuronen in Abhängigkeit zu den Aktivierungen der letzten verborgenen Schicht ( $l-1$ ).

Das Ziel, zum Erlernen einer Problemstellung, ist nun die in (6) bzw. (7) dargestellte Kostenfunktion, durch die Anpassung der Gewichte des gesamten künstlichen neuronalen Netzwerkes, weitestgehend zu minimieren. Dazu verwendet man die Änderung der Kostenfunktion in Abhängigkeit zu den einzelnen Gewichten des künstlichen neuronalen Netzwerkes. Dieser wird auch als der Gradient von der Kostenfunktion  $C$  in  $w_{k,j}$  bezeichnet und ist mit Hilfe der Kettenregel in (8) dargestellt [11].

$$\frac{\partial C}{\partial w_{k,j}^{(l-1)}} = \frac{\partial C}{\partial a_k^{(l)}} * \frac{\partial a_k^{(l)}}{\partial w_{k,j}^{(l-1)}} = \frac{\partial C}{\partial a_k^{(l)}} * \frac{\partial a_k^{(l)}}{\partial z_k^{(l)}} * \frac{\partial z_k^{(l)}}{\partial w_{k,j}^{(l-1)}} \quad (8)$$

Formel (8): Berechnung des Gradienten von  $C$  in  $w_{k,j}^{(l-1)}$  aus dem Produkt des Gradienten von  $C$  in  $a_k^{(l)}$  und dem Gradienten von  $a_k^{(l)}$  in  $w_{k,j}^{(l-1)}$ . Mit  $a_k^{(l)}$  = Aktivität des  $k$ -ten Neurons in der  $l$ -ten Schicht,  $w_{k,j}^{(l-1)}$  = Gewicht der Kante des  $j$ -ten Neurons der  $(l-1)$ -ten Schicht zum  $k$ -ten Neuron der  $l$ -ten Schicht und  $z_k^{(l)}$  = Eingabe des  $k$ -ten Neurons in der  $l$ -ten Schicht.

Durch die Berechnung der einzelnen partiellen Ableitungen, die in (8) zu sehen sind, lässt sich der Wert des Gradienten der Kostenfunktion jeder gewichteten Kante, beginnend mit der Ausgabeschicht und rückpropagierend zur Eingabeschicht, über die Kettenregel ermitteln. Hierzu wurden die genannten partiellen Ableitungen in (9) bis (11) übersichtlich aufgelistet.

$$\frac{\partial C}{\partial a_k^{(l)}} = a_k^{(l)} - y_k \quad (9)$$

Formel (9): Berechnung des Gradienten von  $C$  in  $a_k^{(l)}$  mit  $a_k^{(l)}$  = Aktivität des  $k$ -ten Neurons in der Ausgabeschicht ( $l$ ) und  $y_k$  = erwartete Ausgabe des  $k$ -ten Ausgabeneurons.

$$\frac{\partial a_k^{(l)}}{\partial z_k^{(l)}} = f'_a(z_k^{(l)}) \quad (10)$$

Formel (10): Berechnung des Gradienten von  $a_k^{(l)}$  in  $z_k^{(l)}$  mit  $f'_a(z_k^{(l)})$  = erste Ableitung der Aktivierungsfunktion nach  $z_k^{(l)}$ .

$$\frac{\partial z_k^{(l)}}{\partial w_{k,j}^{(l-1)}} = a_j^{(l-1)} \quad (11)$$

Formel (11): Berechnung des Gradienten von  $z_k^{(l)}$  in  $w_{k,j}^{(l-1)}$  mit  $a_j^{(l-1)}$  = Aktivierung des  $j$ -ten Neurons der  $(l-1)$ -ten Schicht.

## 2. Grundlagen

---

Zu bemerken ist hier, dass die in (9) dargestellte partielle Ableitung nur die partielle Ableitung der in (6) dargestellten mittleren quadratischen Abweichung repräsentiert. Falls eine andere Kostenfunktion verwendet wurde, ergibt sich ein dementsprechend anderes Ergebnis aus der Berechnung des Gradienten von  $C$  in  $a_k^{(l)}$ . Außerdem ist das Ergebnis von (10) abhängig von der verwendeten Aktivierungsfunktion des jeweiligen Neurons. Eine Auswahl der möglichen ersten Ableitungen der Aktivierungsfunktionen ist in Tabelle 1 übersichtlich dargestellt. Aus (11) ist, unter Berücksichtigung von (4), zu entnehmen, dass der Gradient von  $z_k^{(l)}$  in  $w_{k,j}^{(l-1)}$  lediglich die Aktivität des ausgehenden Neurons  $j$ , das über die Kante  $w_{k,j}^{(l-1)}$  in den Neuron  $k$  eingeht, beinhaltet. Unter der Annahme, dass man im gesamten künstlichen neuronalen Netzwerk nur zwei gewichtete Kanten besitzt, lässt sich z.B. die Kostenfunktion in Abbildung 4 folgendermaßen darstellen.

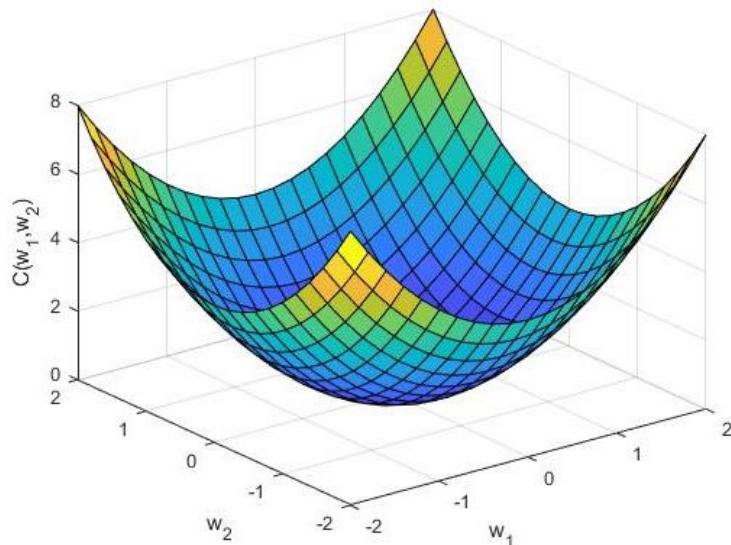


Abbildung 4: Räumliche Darstellung einer von zwei Gewichten abhängigen Kostenfunktion.

Durch Berechnung des Gradienten relativ zu einem bestimmten Gewicht berechnet man die Steigung der Kostenfunktion in Abhängigkeit zum betrachteten Gewicht und damit die Richtung der Kostenfunktion in die sich die Kostenfunktion hinsichtlich des Gewichts entwickelt [11]. Besitzt der Gradient des Gewichts einen positiven Wert, deutet es auf eine steigende Kostenfunktion hin. Ein negativer Gradient des Gewichts hingegen deutet darauf hin, dass die Kostenfunktion fällt. Entspricht der Gradient des Gewichts gleich Null, dann handelt es sich entweder um ein Maximum oder Minimum der Kostenfunktion hinsichtlich des jeweiligen Gewichts. Mit Hilfe dieser Informationen ist es möglich über das sogenannte Gradientenabstiegsverfahren (engl. Gradient Descent) zu einem Minimum der Kostenfunktion zu konvergieren. Dabei bestimmt man den aktuellen Gradienten des jeweiligen

Gewichts und ändert den Wert des betrachteten Gewichts um einen inkrementellen Wert in Richtung eines Minimums der Kostenfunktion. Dieser Vorgang wird iterativ für alle Gewichte des künstlichen neuronalen Netzwerkes wiederholt, bis ein Minimum der Kostenfunktion erreicht wird. Vorstellen kann man sich dieses Verfahren durch einen imaginären Ball, den man auf die in Abbildung 4 dargestellte Ebene an einem beliebigen Punkt auflegt und die Ebene herunter rollen lässt, bis dieser zum Stillstand kommt und somit ein Minimum erreicht hat. Dies ist im Falle der in Abbildung 4 dargestellten Kostenfunktion recht simpel zu bewerkstelligen, jedoch sind in der Praxis die Kostenfunktionen der künstlichen neuronalen Netzwerke von weitaus mehr als zwei Gewichten abhängig und damit im n-dimensionalen Raum schwer darstellbar. Die Berechnung des Gradienten jedes Gewichts im künstlichen neuronalen Netzwerk kann, abhängig von der Größe des verwendeten Datensatzes, viel Zeit in Anspruch nehmen. Dazu wird das Gradientenabstiegsverfahren in drei Kategorien unterteilt, die sich von der Recheneffizienz unterscheiden, dem Stapel-Gradientenabstieg, dem stochastischen Gradientenabstieg und dem Mini-Stapel-Gradientenabstieg [5].

### 2.5.1. Stapel-Gradientenabstieg

Beim Stapel-Gradientenabstieg (engl. Batch Gradient Descent) wird der Gradient der Kostenfunktion jedes Gewichts des künstlichen neuronalen Netzwerks für den gesamten Trainingsdatensatz berechnet und aufsummiert. Das Durchlaufen aller Trainingsbeispiele wird auch als eine Epoche bezeichnet. Nach dem Beenden einer Epoche werden alle Gewichte des künstlichen neuronalen Netzwerkes entsprechend des resultierenden Gradienten der Kostenfunktion aktualisiert. Mit Hilfe des Stapel-Gradientenabstiegs erreicht man, durch die Einbeziehung des gesamten Trainingsdatensatzes zur Berechnung des Gradienten, einen stabilen Verlauf der Kostenfunktion in Folge des Gradientenabstiegs, ist jedoch anfällig für lokale Minima der Kostenfunktion und erreicht dementsprechend eventuell einen nicht optimalen Konvergenzzustand, da der Ausgangspunkt des Gradientenabstiegs von den initialen Gewichtswerten des künstlichen neuronale Netzwerks abhängt [5]. Außerdem berechnet man bei großen Trainingsdatensätzen Gradienten, die man bereits in ähnlichen Trainingsbeispielen berechnet hat, erneut, wodurch man zum einen redundant Berechnungen wiederholt und zum anderen die Redundanzen in den Gradienten einfließen lässt [5].

## 2. Grundlagen

---

### 2.5.2. Stochastischer Gradientenabstieg

Im Gegensatz zum Stapel-Gradientenabstieg wird beim stochastischen Gradientenabstieg (engl. Stochastic Gradient Descent) zur Berechnung des Gradienten der Kostenfunktion für jedes Gewicht des künstlichen neuronalen Netzwerkes nur ein einzelnes, zufällig ausgewähltes, Trainingsbeispiel des gesamten Trainingsdatensatzes verwendet und entsprechend dessen Resultat die Gewichte des künstlichen neuronalen Netzwerkes aktualisiert [5]. Folglich richtet sich die Anzahl der Aktualisierungen der Gewichte in einer Epoche nach der Anzahl der Trainingsbeispiele im gesamten Trainingsdatensatz. Der stochastische Gradientenabstieg erfolgt, abhängig von der jeweiligen Problemstellung, schneller als der Stapel-Gradientenabstieg aufgrund der hohen Anzahl an Gewichtsaktualisierungen in einer Epoche. Wegen der häufigen Gewichtsaktualisierungen ist der stochastische Gradientenabstieg, im Gegensatz zum Stapel-Gradientenabstieg, rechenintensiver, was bei künstlichen neuronalen Netzwerken mit vielen Knoten ein Problem darstellen kann. Zudem führt es, wegen den häufigen Gewichtsaktualisierungen, zu großen Schwankungen im Verlauf der Kostenfunktion in Folge des Gradientenabstiegs. Dadurch, dass ein zufällig bestimmtes Trainingsbeispiel zur Bestimmung des Gradienten verwendet wird, sind Sprünge zu neuen und eventuell besseren lokalen Minima der Kostenfunktion möglich [5].

### 2.5.3. Mini-Stapel-Gradientenabstieg

Der Mini-Stapel-Gradientenabstieg (Mini Batch Gradient Descent) stellt eine Mischform des Stapel-Gradientenabstiegs und des stochastischen Gradientenabstiegs dar. Dazu wird der gesamte Trainingsdatensatz zufällig in Mini-Stapel eingeteilt. Anschließend wird der Gradient der Kostenfunktion jedes Trainingsbeispiels im Mini-Stapel berechnet und aufsummiert. Daraufhin wird für einen Mini-Stapel jeweils eine Gewichtsaktualisierung durchgeführt, wodurch die Anzahl der Gewichtsaktualisierungen in einer Epoche der Anzahl der Mini-Stapel entspricht. Durch die Verwendung von Mini-Stapeln wird die Anzahl der Gewichtsaktualisierungen im Vergleich zum stochastischen Gradientenabstieg verringert, was wiederum zu einem stabileren Verlauf der Kostenfunktion in Folge des Gradientenabstiegs führt. Neben der Verringerung der Wahrscheinlichkeit auf die Berechnung von bereits berechneten Gradienten werden gleichzeitig, durch die zufällige Wahl der Trainingsbeispiele in den Mini-Stapel, Sprünge zu eventuell besseren lokalen Minima der Kostenfunktion ermöglicht [5]. Die Sprunggröße hängt dabei von der Anzahl der

Trainingsbeispiele im Mini-Stapel ab. Handelt es sich bei den Mini-Stapeln um jeweils ein einzelnes Trainingsbeispiel, findet in einer Epoche für jeweils ein Trainingsbeispiel eine Gewichtsaktualisierung statt, was wiederum dem stochastischen Gradientenabstieg gleicht. Handelt es sich hingegen bei den Mini-Stapeln sozusagen nur um einen einzelnen Mini-Stapel mit dem gesamten Trainingsdatensatz, findet in einer Epoche nur eine Gewichtsaktualisierung statt, was wiederum dem Stapel-Gradientenabstieg gleicht.

## 2.6. Gradientenabstiegsalgorithmen

Nach Berechnung aller Gradienten der Kostenfunktion für jedes Gewicht im vorwärtsgerichteten künstlichen neuronalen Netzwerk gemäß (8), sind die Einflüsse jedes Gewichts auf die Kostenfunktion hinsichtlich einer bestimmten Problemstellung bekannt. Mit Hilfe dieser Informationen kann der eigentliche Prozess des Gradientenabstiegs mit Hilfe von Gradientenabstiegsalgorithmen begonnen werden. Gradientenabstiegsalgorithmen beschreiben die Art und Weise, wie das Optimierungsproblem der Kostenfunktion, durch Anpassung der Gewichte des künstlichen neuronalen Netzwerkes, gelöst werden kann. Dazu wird eine Lernrate  $\varepsilon$  eingeführt, die im Allgemeinen die Rate der Gewichtsanpassung und damit die Schrittweite des Gradientenabstiegsverfahrens entlang des multidimensionalen Raums der Kostenfunktion beschreibt [5]. Besitzt die Lernrate einen geringen Wert, kann der Prozess des Gradientenabstiegs aufgrund der geringen Schrittweite des Gradientenabstiegs deutlich verlangsamt werden und dadurch viel Zeit in Anspruch nehmen. Hohe Lernraten hingegen tendieren dazu Minima der Kostenfunktion aufgrund der hohen Schrittweite des Gradientenabstiegs zu überspringen. Dadurch wird die Konvergenz der Kostenfunktion zu einem Minimum erschwert und es kann zu einer Divergenz führen [5]. Gradientenabstiegsalgorithmen unterscheiden sich in der Art und Weise, wie und ob eine Lernrate verwendet wird, um durch Trainingsbeispiele sich an eine Problemstellung anzupassen. Wird ein künstliches neuronales Netzwerk übermäßig an die Trainingsbeispiele spezialisiert, kann es zu einer sogenannten Überanpassung des künstlichen neuronalen Netzwerkes kommen [16]. In diesem Zustand verliert das künstliche neuronale Netzwerk die Fähigkeit fremde Daten korrekt auszuwerten, und damit zu Generalisieren, da die Gewichte vollständig auf die Trainingsbeispiele angepasst sind. Ist das künstliche neuronale Netzwerk hingegen zu wenig den Trainingsbeispielen ausgesetzt, kommt es ebenfalls zu fehlerhaften Auswertungen fremder Daten. In diesem Fall spricht man von einer Unteranpassung.

### 2.6.1. Backpropagation

Der Algorithmus zur Backpropagation (BP) stellt die einfachste und gängigste Form des Gradientenabstiegsverfahrens dar. Dazu wird der Lernrate  $\varepsilon$  ein statischer Wert zugeordnet, der dazu verwendet wird, die Gewichte in jeder Iteration  $t$  des Gradientenabstiegsverfahrens gemäß (12) anzupassen [17].

$$w_{k,j}^{(t+1)} = w_{k,j}^{(t)} - \varepsilon * \frac{\partial C}{\partial w_{k,j}}^{(t)} \quad (12)$$

Formel (12): BP. Berechnung des neuen Gewichtswertes für die gewichtete Kante des  $j$ -ten Neurons zum  $k$ -ten Neuron anhand der Lernrate  $\varepsilon$  und dem Gradienten der Kostenfunktion  $C$  in  $w_{k,j}$ .  $(t)$  bezeichnet die aktuelle Iteration und  $(t+1)$  die darauffolgende Iteration.

Dabei wird das Gewicht  $w_{k,j}$  um einen Bruchteil, der durch die Lernrate  $\varepsilon$  bestimmt ist, des Gradienten der Kostenfunktion  $C$  in  $w_{k,j}$  geändert. Ist der Gradient des jeweiligen Gewichts negativ, also ist die Kostenfunktion fallend hinsichtlich des Gewichts, wird das Gewicht um einen Bruchteil erhöht. Bei einem positiven Gradienten, also einer steigenden Kostenfunktion hinsichtlich des jeweiligen Gewichts, wird das Gewicht um einen Bruchteil verringert. Dadurch steigt man den Gradienten der Kostenfunktion, hin zu einem lokalen oder globalen Minimum, hinab. BP ist stark abhängig von der Wahl der Lernrate  $\varepsilon$ . Wählt man eine zu kleine Lernrate, kann der Gradientenabstieg aufgrund der kleinen Schrittweite im multidimensionalen Raum der Kostenfunktion viel Zeit in Anspruch nehmen, konvergiert jedoch nach einer gewissen Zeit in einem Minimum. Wählt man eine zu große Lernrate, wird eine Konvergenz wegen der hohen Schrittweite verhindert und es kann zu einer Divergenz kommen. Außerdem stellen flache Hügel und steile Täler eine große Herausforderung für die einfache Form von BP dar [18]. Im Falle von einem flachen Hügel bzw. flachen Ebenen haben die Gradienten der Gewichte einen Wert von etwa 0, wodurch man in jeder Iteration sehr geringe Anpassungen der Gewichte und damit sehr kleine Schrittweiten tätigt. Im Falle von steilen Tälern ist die Kostenfunktion sehr steil in einer Dimension als in einer anderen, wodurch es zu einem hin- und herspringen zwischen den Wänden des steilen Tals kommen kann [5]. In diesem Fall kann es eine gewisse Zeit in Anspruch nehmen, den niedrigsten Punkt des steilen Tals zu erreichen, und dann den Gradientenabstieg fortzuführen oder es kann sogar zu einer Divergenz aus dem Tal führen. Um diesen Fällen entgegenzuwirken führt man BP einen Parameter  $\gamma$  hinzu, der den Einfluss vorheriger Schritte auf den aktuellen Schritt darstellt [17]. Dies führt zum Algorithmus zur schwungbasierten BP, die in (13) dargestellt ist [5].

$$w_{k,j}^{(t+1)} = w_{k,j}^{(t)} - v^{(t)} \quad \text{mit} \quad v^{(t)} = \gamma * v^{(t-1)} + \varepsilon * \frac{\partial C}{\partial w_{k,j}}^{(t)} \quad (13)$$

Formel (13): Schwungbasierte BP. Berechnung des neuen Gewichtswertes für die gewichtete Kante des  $j$ -ten Neurons zum  $k$ -ten Neuron anhand der Lernrate  $\varepsilon$ , dem Schwungparameter  $\gamma$  und dem Gradienten der Kostenfunktion  $C$  in  $w_{k,j}$ . ( $t$ ) beschreibt die aktuelle Iteration, ( $t-1$ ) die vorherige Iteration und ( $t+1$ ) die darauffolgende Iteration.

Parameter  $\gamma$  wird auch als Schwungparameter bezeichnet und besitzt oftmals einen statischen Wert von etwa 0.9. Der Wert von  $\gamma$  bestimmt den Bruchteil des Einflusses der vorherigen Schritte, die dem Bruchteil des aktuellen Gradienten aufaddiert werden. Im Allgemeinen erreicht man durch  $v^{(t)}$ , dass bei aufeinanderfolgenden Gradienten mit gleichem Vorzeichen ein Schwung aufgebaut wird und bei aufeinanderfolgenden Gradienten mit wechselndem Vorzeichen Schwung abgebaut wird [5]. Dadurch wird die durch  $\gamma$  bestimmte initiale Beschleunigung des Gradientenabstiegs bei Erreichen eines Minimums abgebaut und ermöglicht eine Konvergenz [5]. Mittels schwungbasierter BP lassen sich flache Ebenen besser bewältigen, jedoch können Minima aufgrund des eventuell zu hohem Schwung sogar übersprungen werden. Zusätzlich kann der Schwung zu noch stärkerem hin- und herspringen in Minima führen und dadurch das Risiko einer Divergenz erhöhen. Dadurch muss man bei BP für die jeweilige Problemstellung, neben der Lernrate  $\varepsilon$ , zusätzlich einen Schwungparameter  $\gamma$  finden, der zu einem optimalen Minimum führt.

### 2.6.2. Resilient Propagation

Der von M. Riedmiller und H. Braun 1993 eingeführte Gradientenabstiegsalgorithmus, die Resilient Propagation (Rprop), führt für jedes Gewicht im vorwärtsgerichteten künstlichen neuronalen Netzwerk individuelle Aktualisierungswerte  $\Delta_{k,j}$  ein [17]. Diese können mit Hilfe der Informationen des Gradienten der Kostenfunktion  $C$  in  $w_{k,j}$  gemäß (14) angepasst werden.

$$\Delta_{k,j}^{(t)} = \begin{cases} \eta^+ * \Delta_{k,j}^{(t-1)} & \text{für } \frac{\partial C}{\partial w_{k,j}}^{(t-1)} * \frac{\partial C}{\partial w_{k,j}}^{(t)} > 0 \\ \eta^- * \Delta_{k,j}^{(t-1)} & \text{für } \frac{\partial C}{\partial w_{k,j}}^{(t-1)} * \frac{\partial C}{\partial w_{k,j}}^{(t)} < 0 \\ \Delta_{k,j}^{(t-1)} & \text{für } \frac{\partial C}{\partial w_{k,j}}^{(t-1)} * \frac{\partial C}{\partial w_{k,j}}^{(t)} = 0 \end{cases} \quad (14)$$

mit  $\eta^+ \geq 1$  und  $0 < \eta^- \leq 1$

Formel (14): Anpassungen des individuellen Aktualisierungswertes  $\Delta_{k,j}^{(t)}$ , der gewichteten Kante des  $j$ -ten Neurons zum  $k$ -ten Neuron, in der Iteration  $t$  mittels Gradienten von  $C$  in  $w_{k,j}$  der Iteration  $t$  und der vorherigen Iteration  $(t-1)$ .  $\eta^+$  bezeichnet den Erhöhungsfaktor und  $\eta^-$  den Verringerungsfaktor.

Aus dem Produkt des Gradienten von  $C$  in  $w_{k,j}$  der aktuellen Iteration  $t$  und dem Gradienten von  $C$  in  $w_{k,j}$  der vorherigen Iteration  $(t-1)$  kann ein Vorzeichenwechsel festgestellt werden. Wenn das Produkt der Gradienten negativ ist, dann bedeutet dies, dass die letzte Aktualisierung des Gewichts zu groß gewesen ist und man ein lokales Minimum verfehlt hat [17]. In diesem Fall wird der Aktualisierungswert  $\Delta_{k,j}$  um den Verringerungsfaktor  $\eta^-$  verringert. Wenn das Produkt der Gradienten positiv ist, dann bedeutet dies, dass man noch kein lokales Minimum erreicht hat. Dementsprechend wird der Aktualisierungswert  $\Delta_{k,j}$  um den Erhöhungsfaktor  $\eta^+$  erhöht. Wurde der Aktualisierungswert bestimmt, erfolgt die Aktualisierung des jeweiligen Gewichts gemäß (15).

$$w_{k,j}^{(t+1)} = \begin{cases} w_{k,j}^{(t)} - \Delta_{k,j}^{(t)} & \text{für } \frac{\partial C}{\partial w_{k,j}}^{(t)} > 0 \\ w_{k,j}^{(t)} + \Delta_{k,j}^{(t)} & \text{für } \frac{\partial C}{\partial w_{k,j}}^{(t)} < 0 \\ 0 & \text{für } \frac{\partial C}{\partial w_{k,j}}^{(t)} = 0 \end{cases} \quad (15)$$

Formel (15): Rprop. Berechnung des neuen Gewichtswertes für die gewichtete Kante des  $j$ -ten Neurons zum  $k$ -ten Neuron anhand des Aktualisierungswertes  $\Delta_{k,j}^{(t)}$  und dem Gradienten der Kostenfunktion  $C$  in  $w_{k,j}$ .  $(t)$  beschreibt die aktuelle Iteration und  $(t+1)$  die darauffolgende Iteration.

Aus (15) ist zu entnehmen, dass die Gewichtsaktualisierung mittels Rprop, im Gegensatz zu BP, unabhängig von dem Gradienten des Gewichts erfolgt. Der Gradient wird lediglich dazu verwendet, um Informationen darüber zu bekommen, ob die Kostenfunktion hinsichtlich des Gewichts steigt oder

fällt. Ist der Gradient des Gewichts positiv, deutet es auf eine steigende Kostenfunktion hin und man senkt das Gewicht um den Aktualisierungswert  $\Delta_{k,j}$ . Ein negativer Gradient hingegen deutet auf eine fallende Kostenfunktion hin. Dementsprechend erhöht man das Gewicht um den Aktualisierungswert  $\Delta_{k,j}$ . Mit Hilfe dieser Vorgehensweise erreicht man, dass sich die Schrittweite bei flachen Ebenen stetig erhöht und bei Erreichen eines Minimums verringert, sodass es effizient zu einer Konvergenz kommt. Gleichzeitig wird der Effekt des Hin- und Herspringens in steilen Tälern geschwächt und dadurch das Risiko einer Divergenz minimiert. Ein weiterer Vorteil von Rprop ist, dass die Suche nach optimalen Parametern, wie z.B. der Lernrate oder dem Schwungparameter der schwungbasierten BP, nicht notwendig ist, weil jedes Gewicht durch seinen eigenen Aktualisierungswert an die Problemstellung im Laufe des Gradientenabstiegsverfahrens angepasst wird. Lediglich der minimale und maximale Aktualisierungswert,  $\Delta_{min}$  und  $\Delta_{max}$ , müssen festgelegt werden [17]. Diese beschreiben den minimalen und maximalen Wert, den ein Aktualisierungswert  $\Delta_{k,j}$  annehmen kann. Ein Nachteil von Rprop ist jedoch, dass dieser auf einen Stapel-Gradientenabstieg limitiert ist. Durch die vielen Sprünge des Gradienten eines stochastischen Gradientenabstiegs können die Aktualisierungswerte von Rprop schlecht an die Problemstellung angepasst werden, sodass eine Konvergenz in diesem Falle erheblich erschwert wird. Dadurch kann man den Vorteil des stochastischen Gradientenabstiegs, Sprünge zu neuen und eventuell besseren lokalen Minima der Kostenfunktion zu ermöglichen, nicht nutzen.

### 2.6.3. Root Mean Square Propagation

Der von G. Hinton 2012 vorgestellte, jedoch unveröffentlichte, Gradientenabstiegsalgorithmus, die Root Mean Square Propagation (RMSprop), ist ein Mini-Stapel Gradientenabstieg, der den Nachteil von Rprop, auf einen Stapel-Gradientenabstieg beschränkt zu sein, ausgleichen soll. Die zentrale Idee hinter RMSprop ist, dass man einen gleitenden Durchschnitt der quadrierten Gradienten für jedes Gewicht gemäß (16) mitführt, um eine effektive Mittelung der Gradienten über die Mini-Stapel zu erreichen [19].

$$v^{(t)} = \beta * v^{(t-1)} + (1 - \beta) * \left( \frac{\partial C}{\partial w_{k,j}} \right)^2 \quad \text{mit } 0 \leq \beta \leq 1 \quad (16)$$

Formel (16): Berechnung des gleitenden Durchschnitts des quadrierten Gradienten von  $C$  in  $w_{k,j}$  mit dem Vergessensfaktor  $\beta$ . ( $t$ ) beschreibt die aktuelle Iteration und ( $t-1$ ) die vorherige Iteration.

## 2. Grundlagen

---

Durch Verwendung des gleitenden Durchschnitts des quadrierten Gradienten werden Gradienten der Gewichte vorheriger Mini-Stapel in Erinnerung gehalten. Dabei bestimmt ein Vergessensfaktor  $\beta$  in welchem Ausmaß dies geschieht. Bei einem kleinen Vergessensfaktor haben Gradienten vorheriger Mini-Stapel weniger Einfluss auf den gleitenden Durchschnitt. Im Gegenteil dazu, ist bei einem großen Vergessensfaktor der Einfluss der Gradienten vorheriger Mini-Stapel auf den gleitenden Durchschnitt größer. G. Hinton empfiehlt hierbei einen Vergessensfaktor von 0.9 [19]. Mit Hilfe dessen lassen sich die Gewichte gemäß (17) aktualisieren.

$$w_{k,j}^{(t+1)} = w_{k,j}^{(t)} - \frac{\varepsilon}{\sqrt{v^{(t)} + \varphi}} * \frac{\partial C}{\partial w_{k,j}}^{(t)} \quad (17)$$

Formel (17): RMSprop. Berechnung des neuen Gewichtswertes für die gewichtete Kante des  $j$ -ten Neurons zum  $k$ -ten Neuron anhand der Lernrate  $\varepsilon$ , dem gleitenden Durchschnitt  $v^{(t)}$ , dem Unschärfeefaktor  $\varphi$  und dem Gradienten der Kostenfunktion  $C$  in  $w_{k,j}$ .  
 $(t)$  beschreibt die aktuelle Iteration und  $(t+1)$  die darauffolgende Iteration.

Durch die Division der Lernrate  $\varepsilon$  mit der Quadratwurzel der Summe aus dem gleitenden Durchschnitt  $v^{(t)}$  und dem Unschärfeefaktor  $\varphi$ , findet eine Anpassung der Lernrate statt, sodass jedes Gewicht individuell aktualisiert werden kann. Dabei wird die Lernrate selbst nicht verändert. Der Unschärfeefaktor  $\varphi$  ist eine möglichst kleine Dezimalzahl, z.B.  $\varphi = 10^{-6}$ , die lediglich verhindern soll, dass der Nenner des Bruches 0 ergibt. Abhängig von dem gewählten Vergessensfaktor  $\beta$ , fließen hierbei Gradienten vorheriger Mini-Stapel zu einem bestimmten Grad in die Anpassung der Lernrate  $\varepsilon$  ein. Der von G. Hinton empfohlene Wert der Lernrate ist 0.001 [5]. Da der gleitende Durchschnitt aufgrund des Quadrates des Gradienten keinen Einfluss auf das Vorzeichen hat, wird die Richtung des Gradientenabstiegs lediglich durch das Produkt aus der angepassten Lernrate und dem Gradienten bestimmt. Entsprechend des Gradientenabstiegsverfahrens wird bei einem negativen Gradienten, also einer fallenden Kostenfunktion, das jeweilige Gewicht um einen angepassten Bruchteil der Lernrate in Richtung des Minimums erhöht. Im Gegenteil dazu wird bei einem positiven Gradienten, also einer steigenden Kostenfunktion, das jeweilige Gewicht um einen angepassten Bruchteil der Lernrate in Richtung des Minimums verringert.

### 3. Implementierung von RMSprop in den n++-Simulatorkern

In diesem Kapitel wird der n++-Simulatorkern und dessen Funktionalitäten erläutert. Ferner wird die Implementierung des Gradientenabstiegsalgorithmus RMSprop innerhalb des n++-Simulatorkerns vorgestellt.

#### 3.1. Erläuterung des n++-Simulatorkerns

Bei dem n++-Simulatorkern handelt es sich um eine C++-Hilfsbibliothek, die von M. Riedmiller 1994 entwickelt wurde. Es beinhaltet alle Kernfunktionalitäten, die im Umgang mit künstlichen neuronalen Netzwerken notwendig sind, und ermöglicht es gleichzeitig weitere Funktionalitäten hinzuzufügen. Nach Einbindung des n++-Simulatorkerns, lassen sich mit Hilfe dessen vorwärtsgerichtete künstliche neuronale Netzwerke mit spezifischen Topologien erstellen. Diese können anschließend dazu genutzt werden, um mittels der implementierten Vorwärtspropagierung Eingabewerte auswerten zu lassen und damit eine Ausgabe des künstlichen neuronalen Netzwerkes zu erzeugen. Nach Ermittlung des Fehlers der Ausgabeneuronen, lässt sich der ermittelte Fehler über das gesamte künstliche neuronale Netzwerk mit Hilfe des n++-Simulatorkerns rückpropagieren, um die Gradienten der einzelnen Gewichte berechnen zu lassen. Abhängig von dem gewählten Gradientenabstiegsalgorithmus lassen sich die Gewichte des Netzwerks, durch die berechneten Gradienten der Gewichte hinsichtlich des ermittelten Fehlers, an die Problemstellung anpassen. Der n++-Simulatorkern ermöglicht es außerdem die verwendeten künstlichen neuronalen Netzwerke zu jedem beliebigen Zeitpunkt in einem Textformat abzuspeichern oder bereits abgespeicherte Netzwerke zu laden und anschließend zu nutzen. In dem Textformat wird die Topologie des Netzwerkes, der Gradientenabstiegsalgorithmus und dessen verwendete Parameter, jeder Knoten mitsamt seiner Aktivierungsfunktion und die Gewichte der Kanten der jeweiligen Knoten übersichtlich aufgelistet.

#### 3.2. Vorstellung der Implementierung von RMSprop

Zu Beginn der vorliegenden Arbeit waren nur BP und Rprop im n++-Simulatorkern als mögliche Gradientenabstiegsalgorithmen implementiert. Dieser sollte nun um RMSprop erweitert werden, um

### 3. Implementierung von RMSprop in den n++-Simulatorkern

---

vergleichende Untersuchungen zwischen den Gradientenabstiegsalgorithmen zu ermöglichen. Dazu wurde die Funktion *set\_update\_f* des n++-Simulatorkerns, in der die Fallunterscheidung zwischen BP und Rprop stattfindet, um den Fall RMSprop erweitert. Diese ist in Abbildung 5 zu sehen.

```
int Net::set_update_f(int typ, float* params)
/* set current update function, init update_function, set update_params */
{
    int i;

    switch(typ){
        case BP:
            bp_init(params,unit,&topo_data); /* prepare bp, set update_params */
            update_f = bp_update;
            update_id = BP;
            break;
        case RPROP:
            rprop_init(params,unit,&topo_data); /* prepare rprop */
            update_f = rprop_update;
            update_id = RPROP;
            break;
        case RMSPROP:
            rmsprop_init(params,unit,&topo_data); /* prepare rmsprop */
            update_f = rmsprop_update;
            update_id = RMSPROP;
            break;
        default:
            fprintf(stderr,"set update_f: Unknown update function %d\n",typ);
            return (ERROR);
    }
    /* copy to internal update_params */
    for(i=0;i<MAX_PARAMS;i++)
        update_params[i] = params[i];
    return(0);
}
```

Abbildung 5: Quellcode-Ausschnitt der *set\_update\_f* Funktion des n++-Simulatorkerns.

Mit Hilfe der Funktion lässt sich der Gradientenabstiegsalgorithmus des jeweiligen künstlichen neuronalen Netzwerkes bestimmen, der für die Aktualisierung der Gewichte verwendet wird. Als Argumente werden der Funktion *set\_update\_f* die Wahl des Gradientenabstiegsalgorithmus als eine Ganzzahl und die jeweiligen Parameter des Gradientenabstiegsalgorithmus als Array übergeben. Da für BP der Wert 0 und für Rprop der Wert 1 als erstes Argument erwartet wird, wurde für RMSprop der Wert 2 in der Header-Datei des n++-Simulatorkerns eingeführt. Innerhalb des jeweiligen Falles der *set\_update\_f* Funktion wird der Gradientenabstiegsalgorithmus über die *bp\_init* bzw. *rprop\_init* Funktion mit den an *set\_update\_f* übergebenen Parametern initialisiert und die Gradienten der Gewichte, sowie die Aktualisierungswerte der Gewichte des künstlichen neuronalen Netzwerkes auf 0 gesetzt. Werden den Funktionen keine Parameter übergeben, findet eine Initialisierung der Parameter des jeweiligen Gradientenabstiegsalgorithmus mit Standardparametern statt. Hier wurde dementsprechend im n++-Simulatorkern für RMSprop die Funktion *rmsprop\_init* hinzugefügt, die in Abbildung 6 zu sehen ist.

```

void rmsprop_init(float *params, unit_typ unit[], topo_typ *topo)
{
    weight_typ *wptr;

    if(! params[0])
        params[0] = EPSILON;
    if(! params[1])
        params[1] = BETA;
    if(! params[2])
        params[2] = PHI;
    if(! params[3])
        params[3] = BATCH_SIZE;
    FORALL_WEIGHTS(wptr){
        wptr->delta = wptr->dEdw = (FTYPE) 0;
    }
}
    
```

Abbildung 6: Quellcode-Ausschnitt der rmsprop\_init Funktion des n++-Simulatorkerns.

Diese überprüft ebenfalls, ob Parameter vorhanden sind, die verwendet werden können, um RMSprop verwenden zu können. Falls ein Parameter nicht vorhanden ist, findet eine Initialisierung des jeweiligen Parameters mittels eines Standardparameters statt. Dazu wurden gemäß (16) und (17) eine Variable für die Lernrate  $\epsilon$  mit dem Wert 0.001, eine Variable für den Vergessensfaktor  $\beta$  mit dem Wert 0.9, eine Variable für den Unschärfeefaktor  $\varphi$  mit dem Wert  $10^{-6}$  und eine Variable für die Größe des Mini-Stapels in den n++-Simulatorkern eingeführt. Der Parameter für die Größe der Mini-Stapel wird nicht in der Aktualisierung der Gewichte verwendet und dient lediglich dazu, dass man die künstlichen neuronalen Netzwerke in abgespeicherter Form identifizieren kann, da der verwendete Gradientenabstiegsalgorithmus und dessen Parameter im Textformat aufgelistet werden. Wird der *set\_update\_f* kein Wert für die Größe des Mini-Stapels übergeben, findet eine Initialisierung der Größe des Mini-Stapels mit dem Wert 16 statt. Hier lässt sich darüber diskutieren, ob ein Parameter für die Größe des Mini-Stapels innerhalb des n++-Simulatorkerns notwendig ist. Ähnlich zu den anderen Initialisierungsfunktionen, werden die Gradienten der Gewichte sowie die Aktualisierungswerte der Gewichte des künstlichen neuronalen Netzwerkes zum Schluss auf den Wert 0 gesetzt. Im Anschluss des jeweiligen Falles der *set\_update\_f* Funktion wird die Aktualisierungsfunktion des Gradientenabstiegsalgoritmus, die für die Anpassung der Gewichte verwendet wird, über die *bp\_update* bzw. *rprop\_update* Funktion zugewiesen. Hier wurde die Aktualisierungsfunktion *rmsprop\_update* für RMSprop eingeführt, die in Abbildung 7 zu sehen ist. Die entsprechende Aktualisierungsfunktion wird durch die *update\_weights* Funktion des n++-Simulatorkerns aufgerufen.

### 3. Implementierung von RMSprop in den n++-Simulatorkern

---

```
void rmsprop_update(unit_typ unit[], topo_typ *topo, float *params)
{
    weight_typ *wptr;
    float learnrate = params[0];
    float forget_factor = params[1];
    float fuzz_factor = params[2];

    FORALL_WEIGHTS(wptr){
        wptr->variable[0] = forget_factor*wptr->variable[0] + (1 - forget_factor)*wptr->dEdw*wptr->dEdw;
        wptr->delta = (learnrate/sqrt(wptr->variable[0] + fuzz_factor))*wptr->dEdw;
        wptr->value -= wptr->delta;
        wptr->dEdw = (FTYPE) 0; /* important: clear dEdw !*/
    }
}
```

Abbildung 7: Quellcode-Ausschnitt der *rmsprop\_update* Funktion des n++-Simulatorkerns.

Zu Beginn der *rmsprop\_update* Funktion werden die Parameter an entsprechende Variablen zugewiesen. Schließlich wird für alle Gewichte des künstlichen neuronalen Netzwerks gemäß (16) der gleitende Durchschnitt des quadrierten Gradienten des jeweiligen Gewichtes bestimmt und in einer Variable *variable[0]* gespeichert, die dem Gewicht zugewiesen ist. Die Gradienten der Gewichte des künstlichen neuronalen Netzwerkes müssen vorher mittels der im n++-Simulatorkern implementierten *backward\_pass* Funktion ermittelt werden. Daraufhin wird der Aktualisierungswert des Gewichts bestimmt und das Gewicht um den ermittelten Aktualisierungswert gemäß (17) aktualisiert. Zum Schluss wird der Gradient des Gewichts zurückgesetzt auf den Wert 0, sodass das künstliche neuronale Netzwerk für die nächste Epoche verwendet werden kann. Damit wurde der Gradientenabstiegsalgorithmus RMSprop vollständig in den n++-Simulatorkern integriert, der nun dazu verwendet werden kann, um künstliche neuronale Netzwerke mittels RMSprop zu trainieren. Die Funktionalität des Ladens eines Datensatzes in den Arbeitsspeicher und der Aufteilung des Datensatzes in Mini-Stapel findet nicht innerhalb des n++-Simulatorkerns statt.

## 4. Experimentelle Untersuchungen

Für eine verlässliche Beurteilung des Lernerfolgs der künstlichen neuronalen Netze wurde das 10-fache Kreuzvalidierungsverfahren angewandt. Dazu wurden für alle folgenden experimentellen Untersuchungen jeweils 10 vorwärtsgerichtete künstliche neuronale Netzwerke gemäß (3) mit zwei verborgenen Schichten mittels des n++-Simulatorkerns erstellt. Hierbei wurden die einzelnen Gewichte der Kanten im Netzwerk zufallsgeneratorbasiert mit einem Wert zwischen -0.5 und 0.5 initialisiert, um unterschiedliche Ausgangspunkte des Gradientenabstiegsverfahrens für jedes Netzwerk zu gewährleisten. Alle ausgewählten Datensätze, die zum Training der künstlichen neuronalen Netzwerke genutzt wurden, sind dem UCI Machine Learning Repository entnommen worden [20]. Die Eingabewerte der verwendeten Datensätze wurden im Vorfeld auf das Intervall [0,1] gemäß (18) normalisiert, um den Suchraum der Kostenfunktion auf einen einheitlichen Hyperwürfel einzuschränken und damit das Gradientenabstiegsverfahren möglichst zu beschleunigen [21].

$$x_n = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (18)$$

Formel (18): Normalisierung des Eingabewertes  $x$  mittels des kleinstmöglichen Eingabewertes  $x_{min}$  von  $x$  und des größtmöglichen Eingabewertes  $x_{max}$  von  $x$ .

Zu Beginn des Trainings wurde der normalisierte Datensatz zufallsgeneratorbasiert gemischt und auf 80% Trainingsbeispiele sowie 20% Testbeispiele aufgeteilt. Dadurch soll gewährleistet werden, dass möglichst jedes Beispiel im Datensatz zur Ermittlung eines optimalen Minimums in Folge des Gradientenabstiegsverfahrens verwendet werden kann. Bei der Aufteilung des Datensatzes wurde bei Klassifikationsproblemen oftmals darauf geachtet, dass in den Trainingsbeispielen und in den Testbeispielen dasselbe Verhältnis der möglichen Klassen repräsentiert wird. Dadurch sollte erreicht werden, dass möglichst jede Klasse beim Training des künstlichen neuronalen Netzwerkes verwendet wird und es nicht zu einem Ungleichgewicht der möglichen Klassen zwischen den Trainingsbeispielen und den Testbeispielen kommt. Hierbei muss angemerkt werden, dass diese Vorgehensweise für einen Gradientenabstiegsalgorithmus nicht notwendig ist. Die Trainingsbeispiele wurden anschließend dazu verwendet eines der 10 erstellten künstlichen neuronalen Netzwerke zu trainieren. Hierbei wurde innerhalb jeder Epoche die mittlere quadratische Abweichung (engl. Mean Squared Error, MSE) der Ausgabeneuronen gemäß (19) für alle Trainingsbeispiele sowie für alle Testbeispiele bestimmt, sodass der Trainingsprozess des künstlichen neuronalen Netzwerkes mitverfolgt werden kann.

$$MSE = \frac{1}{n} \sum_{i=1}^n ((a_1 - y_1)^2 + \dots + (a_k - y_k)^2) \quad (19)$$

Formel (19): Berechnung der mittleren quadratischen Abweichung innerhalb einer Epoche.  $n$  = Anzahl der Trainingsbeispiele bzw. Testbeispiele,  $a_k$  = Aktivierung des  $k$ -ten Ausgabeneurons,  $y_k$  = erwarteter Wert des  $k$ -ten Ausgabeneurons.

Zusätzlich dazu wurde bei Klassifikationsproblemen für jedes Testbeispiel eine korrekte Zuordnung des künstlichen neuronalen Netzwerkes bewertet. Dies erfolgte gemäß (20) anhand der totalen Quadratsumme der Ausgabeneuronen (engl. Total Sum of Squares, TSS).

$$TSS = (a_1 - y_1)^2 + \dots + (a_k - y_k)^2 \quad (20)$$

Formel (20): Berechnung der totalen Quadratsumme der Ausgabeneuronen.  $a_k$  = Aktivierung des  $k$ -ten Ausgabeneurons,  $y_k$  = erwarteter Wert des  $k$ -ten Ausgabeneurons.

Sofern sich der ermittelte Wert des TSS unterhalb des Wertes 0.05 befand, wurde von einer korrekten Klassifikation der Eingabewerte ausgegangen. Dementsprechend wurde die Genauigkeit des künstlichen neuronalen Netzwerkes aus dem Verhältnis der Anzahl aller korrekten Zuweisungen und der Anzahl der Testbeispiele bestimmt. Bei Regressionsproblemen wurde anstatt der Genauigkeit des künstlichen neuronalen Netzwerkes der mittlere absolute prozentuale Fehler (engl. Mean Absolute Percentage Error, MAPE) aller Testbeispiele gemäß (21) berechnet.

$$MAPE = \frac{1}{n} \sum_{i=1}^n \frac{|y_i - a_i|}{y_i} \quad (21)$$

Formel (21): Berechnung des mittleren absoluten prozentualen Fehlers des Ausgabeneurons.  
 $n$  = Anzahl der Testbeispiele,  $a$  = Aktivierung des Ausgabeneurons,  $y$  = erwarteter Wert des Ausgabeneurons

Dieser soll ein prozentuelles Maß der mittleren Abweichung der Ausgabe des künstlichen neuronalen Netzwerkes von dem erwarteten Ausgabewert darstellen. Jedes künstliche neuronale Netzwerk wurde 10000 Epochen mit den Trainingsbeispielen trainiert. Bei BP sowie Rprop wurde hierbei der Stapel-Gradientenabstieg verwendet. Das Gradientenabstiegsverfahren mittels BP wurde stets mit einer Lernrate  $\varepsilon = 0.001$  und einem Schwungparameter  $\gamma = 0.95$  durchgeführt. Das Gradientenabstiegsverfahren mit Rprop erfolgte stets mit einem minimalen Aktualisierungswert  $\Delta_{min} = 0.001$  und einem maximalen Aktualisierungswert  $\Delta_{max} = 50$ . Die Gradientenabstiegsverfahren von

BP und Rprop galten lediglich als Referenz für den Lernerfolg von RMSprop, dementsprechend stellen die gewählten Parameter womöglich nicht die optimale Wahl für den Lernerfolg des künstlichen neuronalen Netzwerkes dar. Nichtsdestotrotz führen die gewählten Parameter von BP und Rprop oftmals zu einem guten Ergebnis. Für RMSprop wurde der Mini-Stapel-Gradientenabstieg verwendet. Die Mini-Stapel wurden zufällig aus den Trainingsbeispielen herausgezogen, sodass auch die Möglichkeit bestand, dass einzelne Trainingsbeispiele mehrmals innerhalb einer Epoche zur Berechnung der Gradienten verwendet werden könnten. Dabei gilt eine Epoche als abgeschlossen, wenn alle Mini-Stapel zum Training des künstlichen neuronalen Netzwerkes verwendet wurden. Zur Untersuchung der Auswirkungen der einzelnen Parameter von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes wurde ein Parameter variiert, während den anderen Parametern ihr von G. Hinton empfohlener Standardwert zugeordnet wurde [19, 5]. Dementsprechend wurde die Lernrate  $\varepsilon = 0.001$ , der Vergessensfaktor  $\beta = 0.9$  und der Unschärfeefaktor  $\varphi = 10^{-6}$  als Standardwerte angenommen. Zusätzlich zu der Variation der einzelnen Parameter von RMSprop, wurde ebenfalls die Größe der Mini-Stapel variiert und deren Auswirkung auf den Lernerfolg untersucht. Nach Abschluss des Trainings jedes künstlichen neuronalen Netzwerkes mit ihren individuell gemischten Datensätzen, wurde ein Mittelwert der mittleren quadratischen Abweichung der Trainings- und Testbeispiele sowie der Genauigkeit bzw. des mittleren absoluten prozentualen Fehlers zwischen den Ergebnissen aller künstlichen neuronalen Netzwerkes gebildet und die Standardabweichung der genannten Werte ermittelt. In den folgenden Unterkapiteln wird zuerst ein Überblick über den verwendeten Datensatz verschafft und dessen Eingabewerte sowie erwarteten Ausgabewerte vorgestellt. Anschließend werden die experimentellen Ergebnisse des Trainings des künstlichen neuronalen Netzwerkes mittels RMSprop unter Variation der einzelnen Parameter des jeweiligen Datensatzes vorgestellt und der Lernerfolg der Gradientenabstiegsalgorithmen BP, Rprop und RMSprop verglichen.

## 4.1. Iris

Bei dem Iris-Datensatz des UCI Machine Learning Repository handelt es sich um ein Klassifikationsproblem [22]. Der Datensatz besitzt insgesamt 150 Beispiele, die auf 120 Trainingsbeispiele und 30 Testbeispiele aufgeteilt wurden. In dem Datensatz existieren 3 mögliche Klassen mit jeweils 50 Beispielen, die ausgewertet werden können. Bei den möglichen Klassen handelt es sich um Arten der Iris-Pflanze – Iris Setosa, Iris Versicolour und Iris Virginica. Eine der Arten ist laut des Erstellers des Datensatzes linear-separierbar von den anderen beiden Arten, wohingegen

## 4. Experimentelle Untersuchungen

---

die anderen beiden Arten nicht voneinander linear-separierbar sind. Jedes Beispiel besitzt insgesamt 4 Attribute, die als Eingabewerte für das künstliche neuronale Netzwerk zur Ermittlung der Art der Iris-Pflanze dienen. Diese sind in Tabelle 2 aufgeführt.

Tabelle 2: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Iris-Datensatzes.

Attribut	Information	Wertebereich
A <sub>1</sub>	Kelchblattlänge in cm	[4.3, 7.9]
A <sub>2</sub>	Kelchblattbreite in cm	[2, 4.4]
A <sub>3</sub>	Blütenblattlänge in cm	[1, 6.9]
A <sub>4</sub>	Blütenblattbreite in cm	[0.1, 2.5]

Es wurde darauf geachtet, dass der Datensatz zufallsgeneratorbasiert derart gemischt wurde, dass jeweils 40 Beispiele jeder Art von Iris-Pflanze in den Trainingsbeispielen vorhanden sind. Folglich befanden sich jeweils 10 Beispiele jeder Art von Iris-Pflanze in den Testbeispielen. Für den Iris-Datensatz wurden künstliche neuronale Netzwerke gemäß (3) mit zwei verborgenen Schichten mittels des n++-Simulatorkerns erstellt. Dementsprechend befanden sich in der Eingabeschicht 4 Eingabeneuronen, in den verborgenen Schichten jeweils 6 verborgene Neuronen und in der Ausgabeschicht 3 Ausgabeneuronen. Als Aktivierungsfunktionen der verborgenen Neuronen und der Ausgabeneuronen wurde die Sigmoidfunktion verwendet. Jedes Ausgabeneuron repräsentiert die Wahrscheinlichkeit der Eingabewerte zu einer jeweiligen Art der Iris-Pflanze zu gehören.

### 4.1.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate

Dieses Experiment soll den Effekt der Lernrate von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Iris-Datensatzes analysieren. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 8 dargestellt.

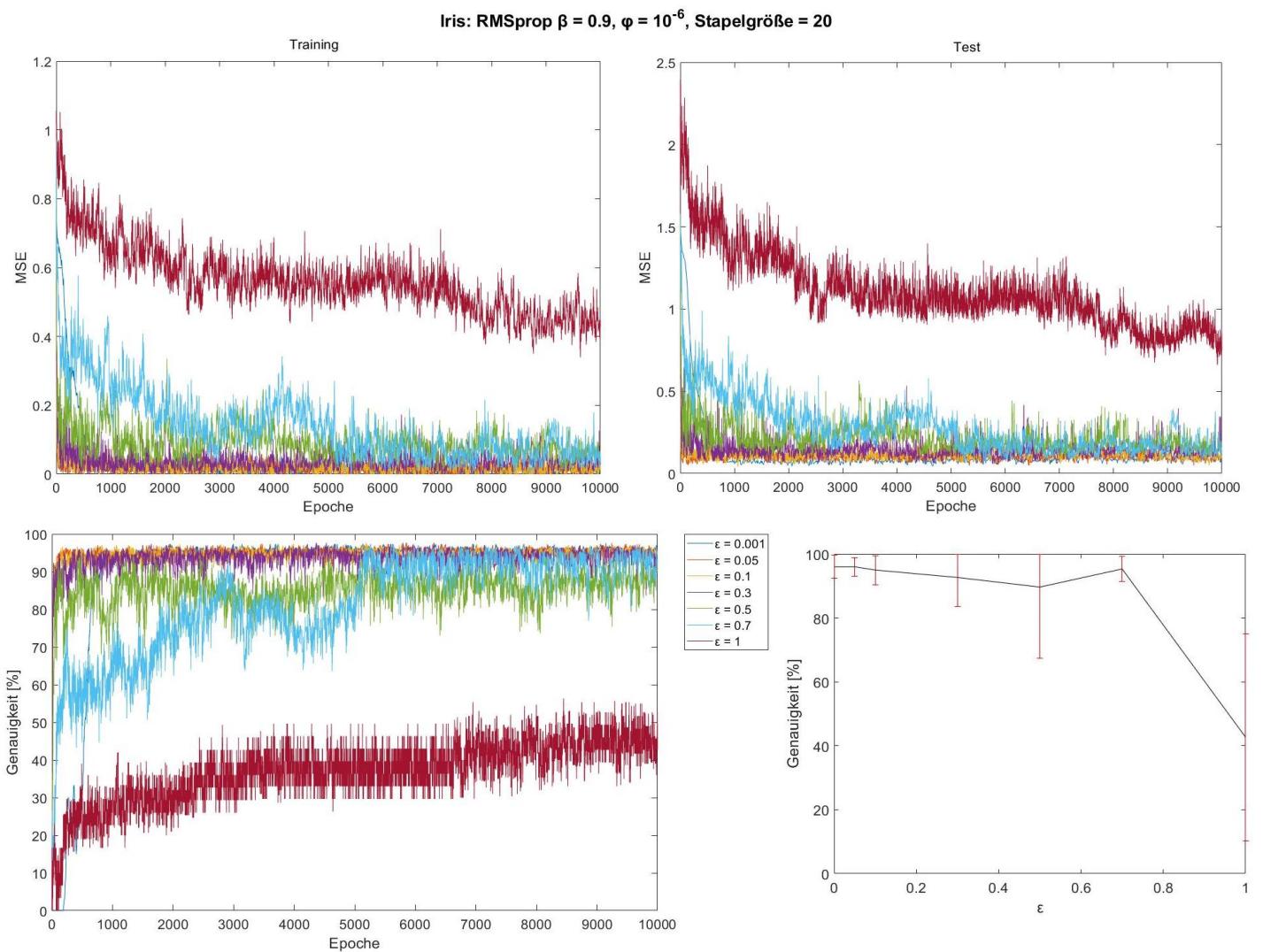


Abbildung 8: Verlauf des MSE der Trainings- (oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Iris-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Lernrate dargestellt.

In Abbildung 8 ist zu beobachten, dass die Oszillation der Verläufe des MSE der Trainings- und Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerks mit steigender Lernrate zunehmend stärker werden. Der Grund dafür liegt in der Schrittweite des Gradientenabstiegs, die durch die Lernrate bestimmt wird. Durch eine hohe Lernrate werden innerhalb einer Epoche mehrere große Schritte hin zu einem möglichen Minimum getätigt. Zusätzlich dazu wird durch eine hohe Lernrate die Konvergenz in einem Minimum erschwert. Bei einer Lernrate von 0.001, 0.05 und 0.1 oszillieren die Verläufe ebenfalls, jedoch ist dies vermutlich größtenteils auf die mehreren Gewichtsaktualisierungen durch den Mini-Stapel-Gradientenabstieg zurückzuführen. Der Verlauf der Lernrate 0.001 besitzt die wenigsten Oszillationen. Zusätzlich zu der steigenden Stärke der Oszillationen, sinkt die Genauigkeit des künstlichen neuronalen Netzwerkes mit steigender Lernrate im Mittel zu niedrigen Genauigkeitswerten. Gleichzeitig steigt der MSE der Trainings- und Testbeispiele im Mittel hin zu

#### 4. Experimentelle Untersuchungen

höheren Werten. Dieser Effekt scheint jedoch erst ab einer Lernrate von 0.3 aufzutreten. In den genannten Abbildungen kann man ebenfalls erkennen, dass eine Lernrate von 0.001 mehr Epochen benötigt, um einen Konvergenzzustand bei den höchsten Genauigkeitswerten bzw. niedrigsten MSE-Werten zu erreichen, als eine Lernrate von 0.05 oder 0.1.

##### **4.1.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors**

Dieses Experiment soll den Effekt des Vergessensfaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Iris-Datensatzes ermitteln. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 9 dargestellt.

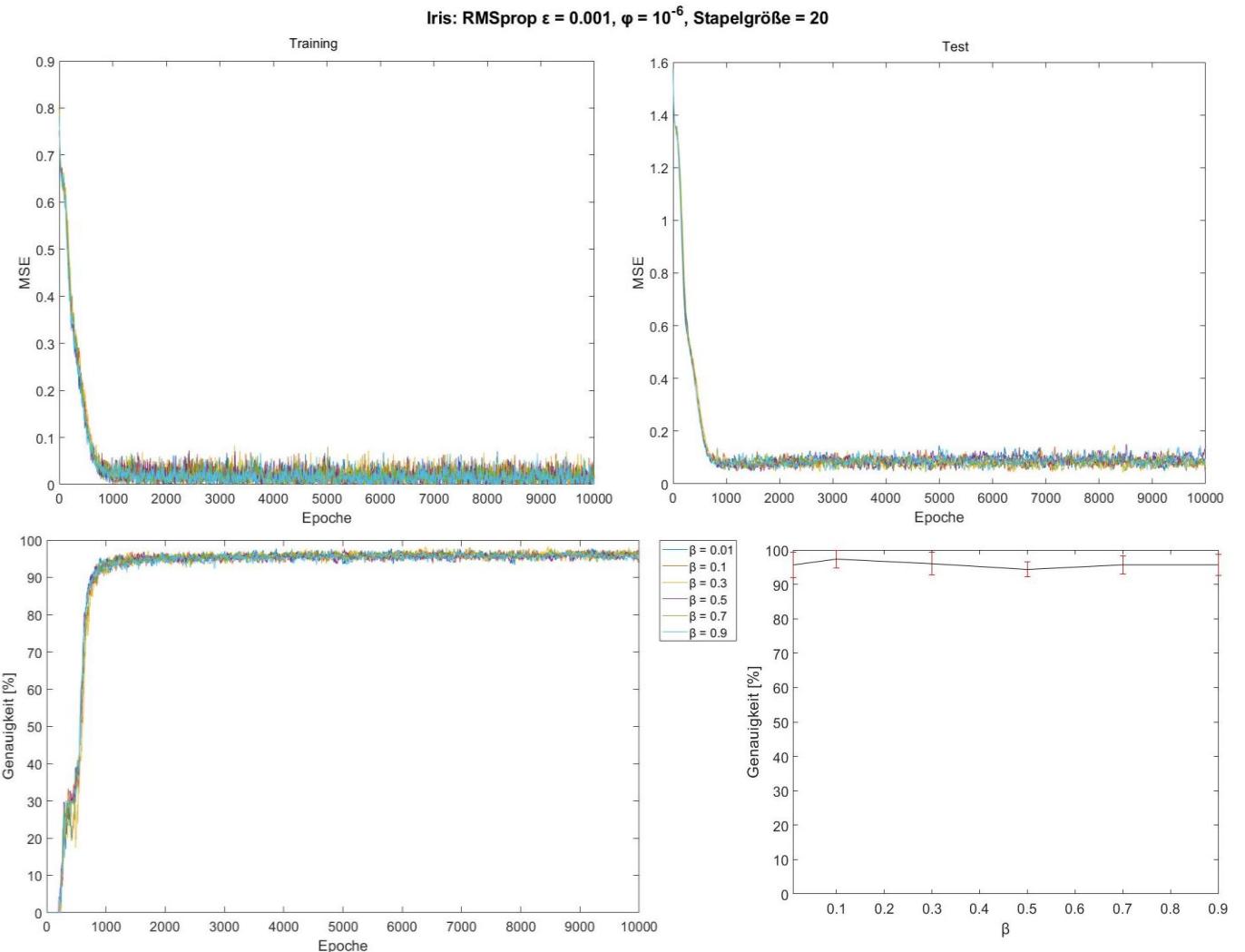


Abbildung 9: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Iris-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des Vergessensfaktors dargestellt.

Aus Abbildung 9 ist zu entnehmen, dass der Vergessensfaktor keinen deutlichen Effekt auf den Lernerfolg des künstlichen neuronalen Netzwerks unter Verwendung des Iris-Datensatzes aufzuweisen hat. Nennenswert ist jedoch der Verlauf der Genauigkeit des künstlichen neuronalen Netzwerkes. Bei einer Epoche von etwa 500 scheint der RMSprop Gradientenabstiegsalgorithmus bei einem Vergessensfaktor von 0.3 eine leicht andere Route entlang des Hyperraums der Kostenfunktion zu nehmen als bei den anderen Vergessensfaktoren. Allerdings scheint diese Route zu dem gleichen oder einem ähnlichen Minimum zu führen, da jeweils die gleiche Genauigkeit am Ende erreicht wird.

## 4. Experimentelle Untersuchungen

### 4.1.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors

Dieses Experiment soll den Effekt des Unschärfeefaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Iris-Datensatzes überprüfen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 10 dargestellt.

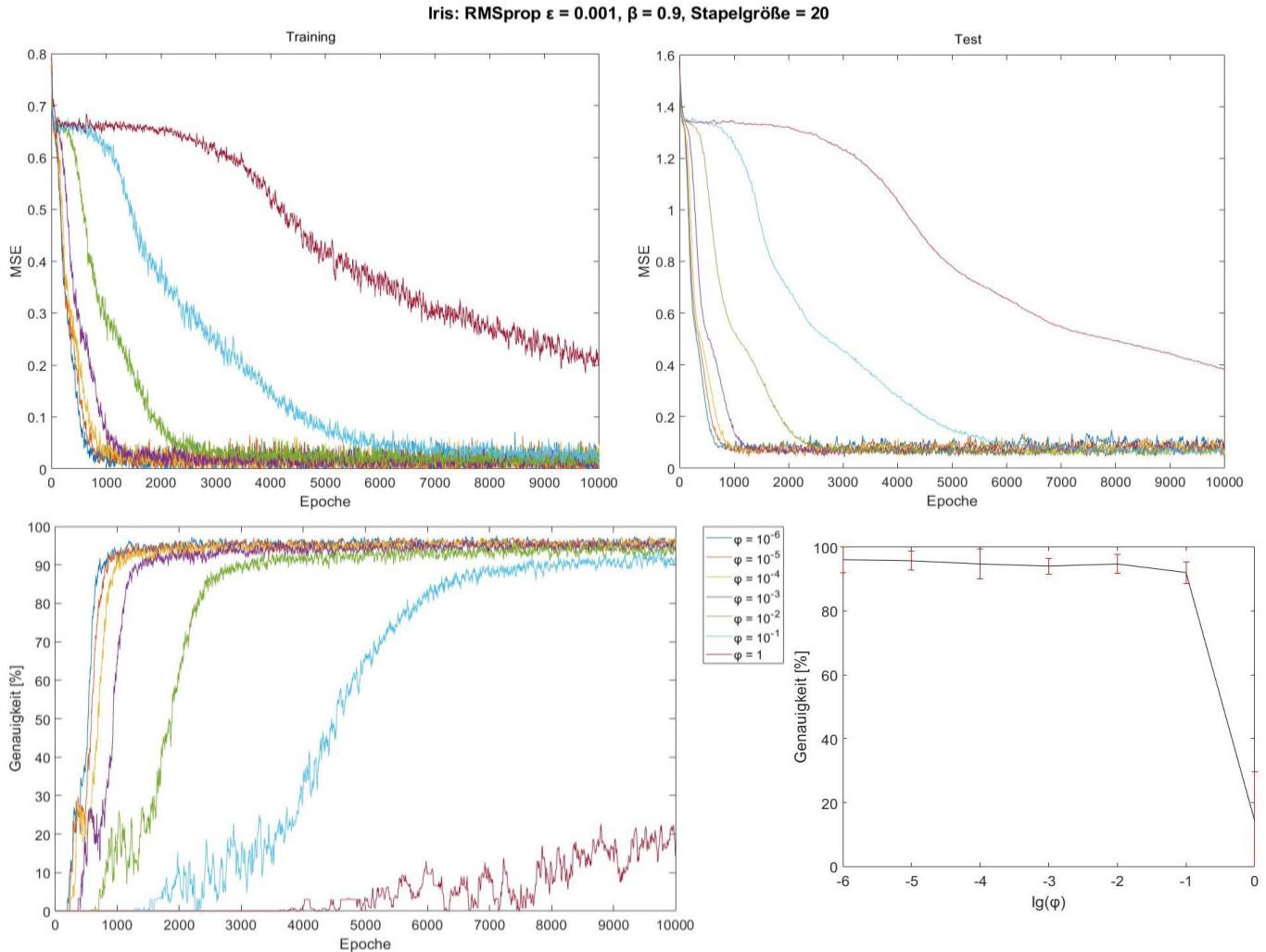


Abbildung 10: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Iris-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des logarithmierten Unschärfeefaktors dargestellt.

In Abbildung 10 ist zu sehen, dass das Gradientenabstiegsverfahren mit einem Unschärfeefaktor von  $10^{-6}$  weniger Epochen benötigt, um den minimalen MSE bzw. die höchste Genauigkeit des künstlichen neuronalen Netzwerkes zu erreichen als die anderen Unschärfeefaktoren. Mit steigendem Unschärfeefaktor scheint der Prozess des Gradientenabstiegsverfahrens verlangsamt zu werden.

Gleichzeitig scheint die maximal erreichbare Genauigkeit des künstlichen neuronalen Netzwerkes mit steigendem Unschärfe faktor zu niedrigeren Werten zu sinken.

#### 4.1.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel

Dieses Experiment soll den Effekt der Größe des Mini-Stapels von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Iris-Datensatzes untersuchen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 11 dargestellt.

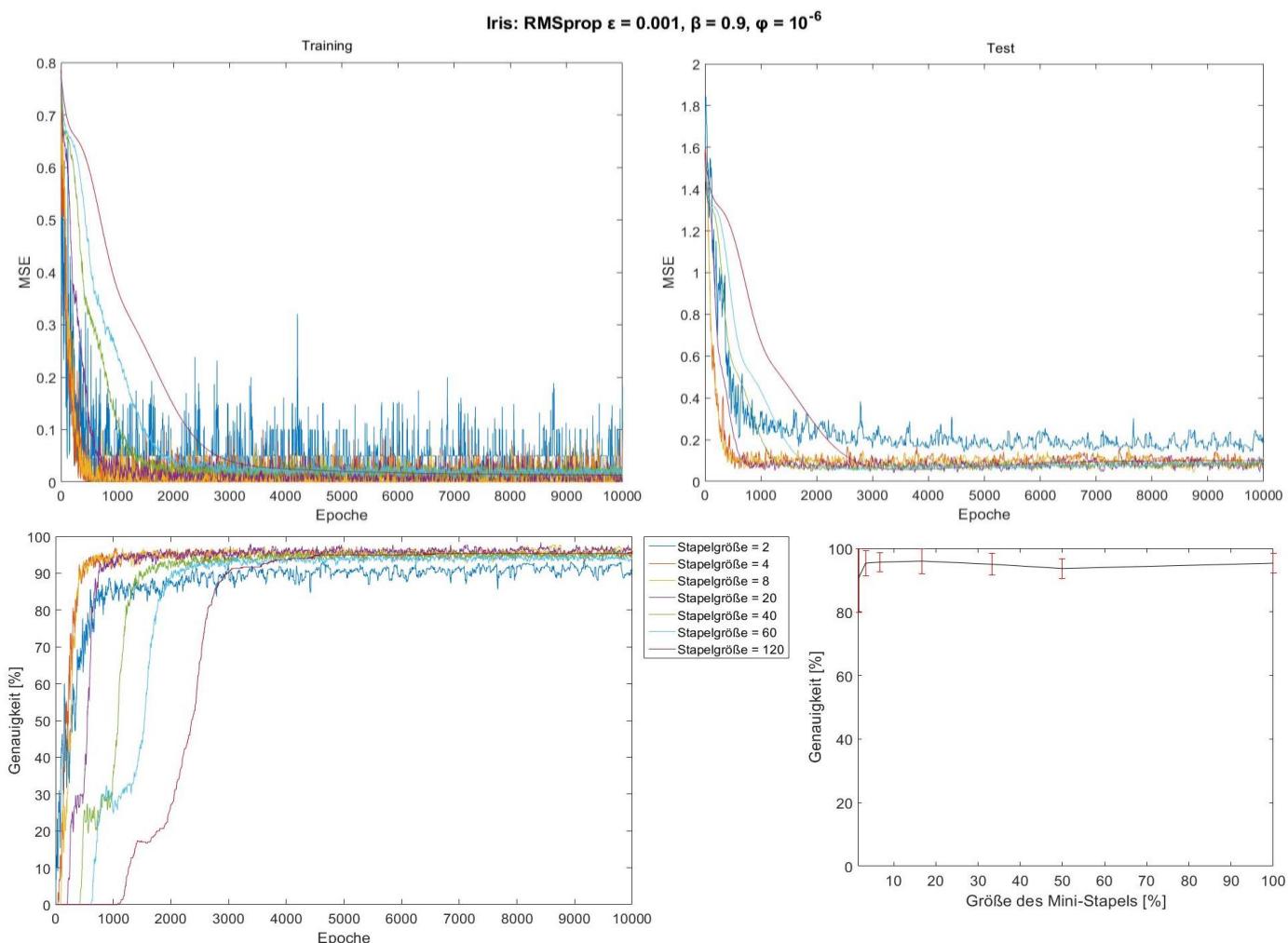


Abbildung 11: Verlauf des MSE der Trainings- (oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Iris-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Größe des Mini-Stapels im Verhältnis zu der gesamten Anzahl der Trainingsbeispiele dargestellt.

## 4. Experimentelle Untersuchungen

---

Aus Abbildung 11 ist festzustellen, dass das Gradientenabstiegsverfahren mit einer Mini-Stapelgröße von 4, 8 und 20 weniger Epochen benötigen, um den minimalen MSE bzw. die höchste Genauigkeit des künstlichen neuronalen Netzwerkes zu erreichen. Erwähnenswert ist hierbei, dass eine Mini-Stapelgröße von 1 einem stochastischen Gradientenabstieg von RMSprop gleichen würde, wohingegen eine Mini-Stapelgröße von 120 einem Stapel-Gradientenabstieg von RMSprop gleichen würde. Aus den genannten Abbildungen lässt sich demnach erkennen, dass bei den Mini-Stapelgrößen von 4, 8 und 20 höhere Genauigkeitswerte und niedrigere MSE-Werte erreicht werden im Vergleich zum Stapel-Gradientenabstieg von RMSprop. Die Mini-Stapelgröße von 2 oszilliert aufgrund der hohen Anzahl an Gewichtsaktualisierungen innerhalb einer Epoche am meisten und erreicht die niedrigsten Genauigkeitswerte bzw. die höchsten MSE-Werte zwischen allen Mini-Stapelgrößen. Der Gradientenabstieg mit der Mini-Stapelgröße von 120 ist im Vergleich, aufgrund nur einer Gewichtsaktualisierung innerhalb einer Epoche, am langsamsten, erreicht jedoch höhere Genauigkeitswerte als ein Gradientenabstieg mit einer Mini-Stapelgröße von 2, 40 oder 60.

### 4.1.5. Vergleich der Gradientenabstiegsalgorithmen

Hier soll der Lernerfolg des künstlichen neuronalen Netzwerkes mittels des Gradientenabstiegsalgorithmus RMSprop mit dem Lernerfolg des künstlichen neuronalen Netzwerkes mittels der Gradientenabstiegsalgorithmen BP und Rprop verglichen werden. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 12 dargestellt.

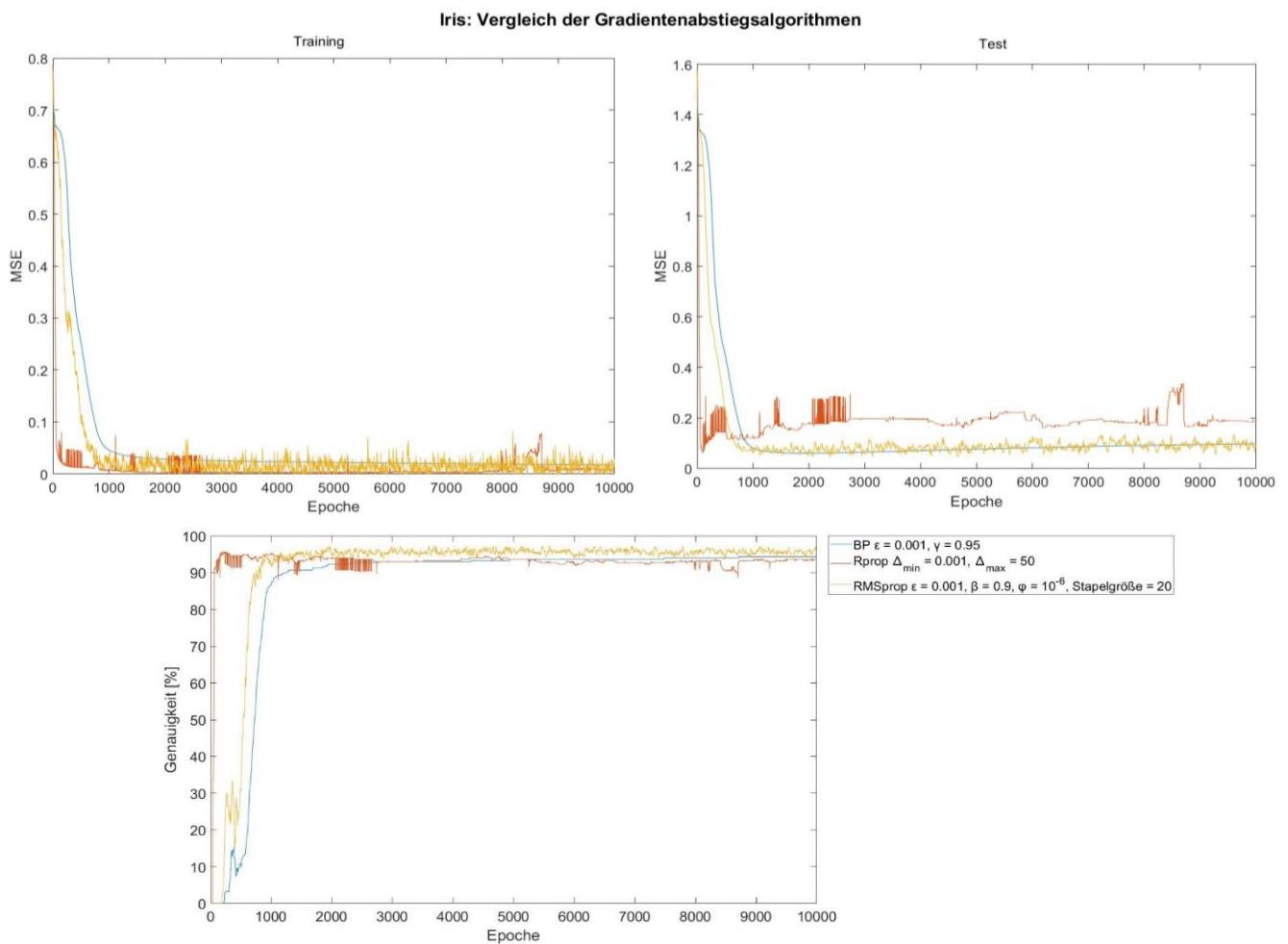


Abbildung 12: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten) des künstlichen neuronalen Netzwerkes des Iris-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop.

Abbildung 12 zeigt, dass Rprop effizient innerhalb weniger Epochen einen hohen Genauigkeitswert erreicht, jedoch nimmt der Genauigkeitswert aufgrund von Überanpassung des künstlichen neuronalen Netzwerkes an die Trainingsbeispiele ab. Dies lässt sich z.B. durch den Anstieg des MSE der Testbeispiele erkennen. Mittels BP erreicht man zwar keinen hohen Genauigkeitswert wie bei RMSprop oder Rprop, jedoch scheint dieser innerhalb der 10000 Epochen nicht einen Plateauwert erreicht zu haben. Nichtsdestotrotz erreicht man innerhalb 10000 Epochen mit BP den niedrigsten Genauigkeitswert aller Gradientenabstiegsalgorithmen. RMSprop hingegen erreicht den höchsten Genauigkeitswert und scheint, ähnlich zu BP, keine bzw. nur schwach ausgeprägte Anzeichen einer Überanpassung des künstlichen neuronalen Netzwerks an die Trainingsbeispiele zu zeigen. In Abbildung 13 wird die erreichte Genauigkeit nach 10000 Epochen, die durchschnittliche Genauigkeit der letzten 1000 Epochen und die maximal erreichte Genauigkeit innerhalb der 10000 Epochen des jeweiligen Gradientenabstiegsalgorithmus übersichtlich dargestellt.

## 4. Experimentelle Untersuchungen

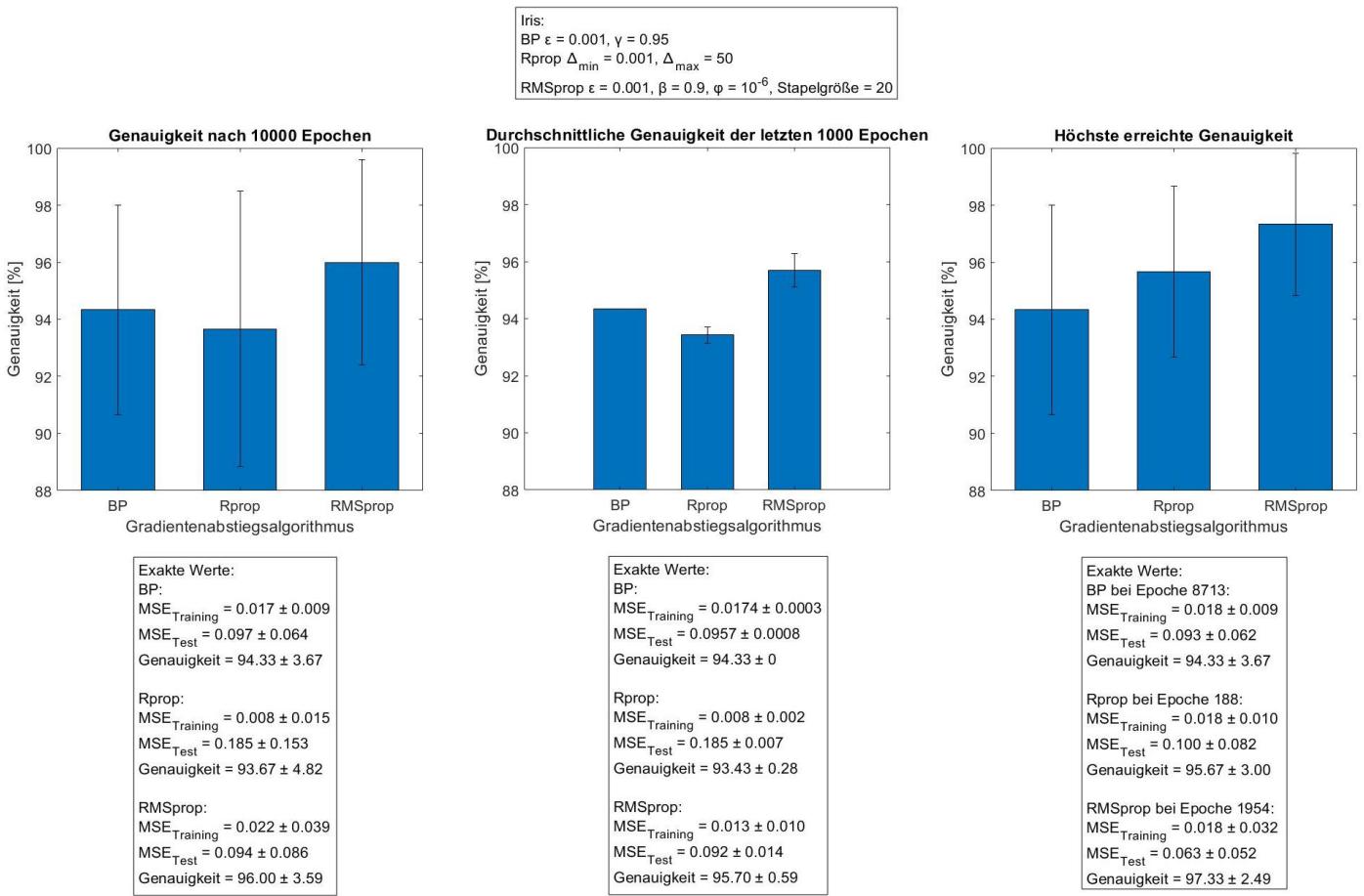


Abbildung 13: Übersicht der nach 10000 Epochen erreichten Genauigkeit (links), der durchschnittlichen Genauigkeit der letzten 1000 Epochen (mittig) und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen (rechts) des künstlichen neuronalen Netzwerkes des Iris-Datensatzes von BP, Rprop und RMSprop.

## 4.2. Yacht Hydrodynamics

Bei dem Yacht Hydrodynamics-Datensatz des UCI Machine Learning Repository handelt es sich um ein Regressionsproblem [23]. Der Datensatz besitzt insgesamt 308 Beispiele, dem jedoch 3 Beispiele entnommen wurden, um eine Aufteilung in 80 % Trainingsbeispiele und 20 % Testbeispiele zu ermöglichen. Dementsprechend wurden 305 Beispiele des Datensatzes verwendet, die auf 244 Trainingsbeispiele und 61 Testbeispiele aufgeteilt wurden. Der Datensatz soll dazu verwendet werden den Rückstandswiderstand von Segelyachten vorherzusagen. Dieser wiederum dient zur Bewertung der Schiffsleistung und für die Abschätzung der erforderlichen Antriebskraft. Jedes Beispiel besitzt insgesamt 6 Attribute, die als Eingabewerte für das künstliche neuronale Netzwerk zur Ermittlung des Rückstandswiderstandes der Segelyacht dienen. Diese sind in Tabelle 3 übersichtlich dargestellt.

Tabelle 3: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Yacht Hydrodynamics-Datensatzes.

Attribut	Information	Wertebereich
A <sub>1</sub>	Längsposition des Auftriebszentrums, dimensionslos	[-5, 0]
A <sub>2</sub>	Prismatischer Koeffizient, dimensionslos	[0.53, 0.6]
A <sub>3</sub>	Längen-Verschiebungsverhältnis, dimensionslos	[4.34, 5.14]
A <sub>4</sub>	Strahl-Zug-Verhältnis, dimensionslos	[2.81, 5.35]
A <sub>5</sub>	Längen-Strahl-Verhältnis, dimensionslos	[2.73, 3.64]
A <sub>6</sub>	Froude Nummer, dimensionslos	[0.125, 0.45]

Für den Yacht Hydrodynamics-Datensatz wurden künstliche neuronale Netzwerke gemäß (3) mit zwei verborgenen Schichten mittels des n++-Simulatorkerns erstellt. Dementsprechend befanden sich in der Eingabeschicht 6 Eingabeneuronen, in den verborgenen Schichten jeweils 5 verborgene Neuronen und in der Ausgabeschicht 1 Ausgabeneuron. Als Aktivierungsfunktionen der verborgenen Neuronen wurde die Sigmoidfunktion verwendet. Für die Aktivierungsfunktion des Ausgabeneurons wurde die Identitätsfunktion verwendet.

#### 4.2.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate

Dieses Experiment soll den Effekt der Lernrate von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Yacht Hydrodynamics-Datensatzes erforschen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie der mittlere absolute prozentuale Fehler der Testbeispiele innerhalb jeder Epoche in Abbildung 14 dargestellt.

## 4. Experimentelle Untersuchungen

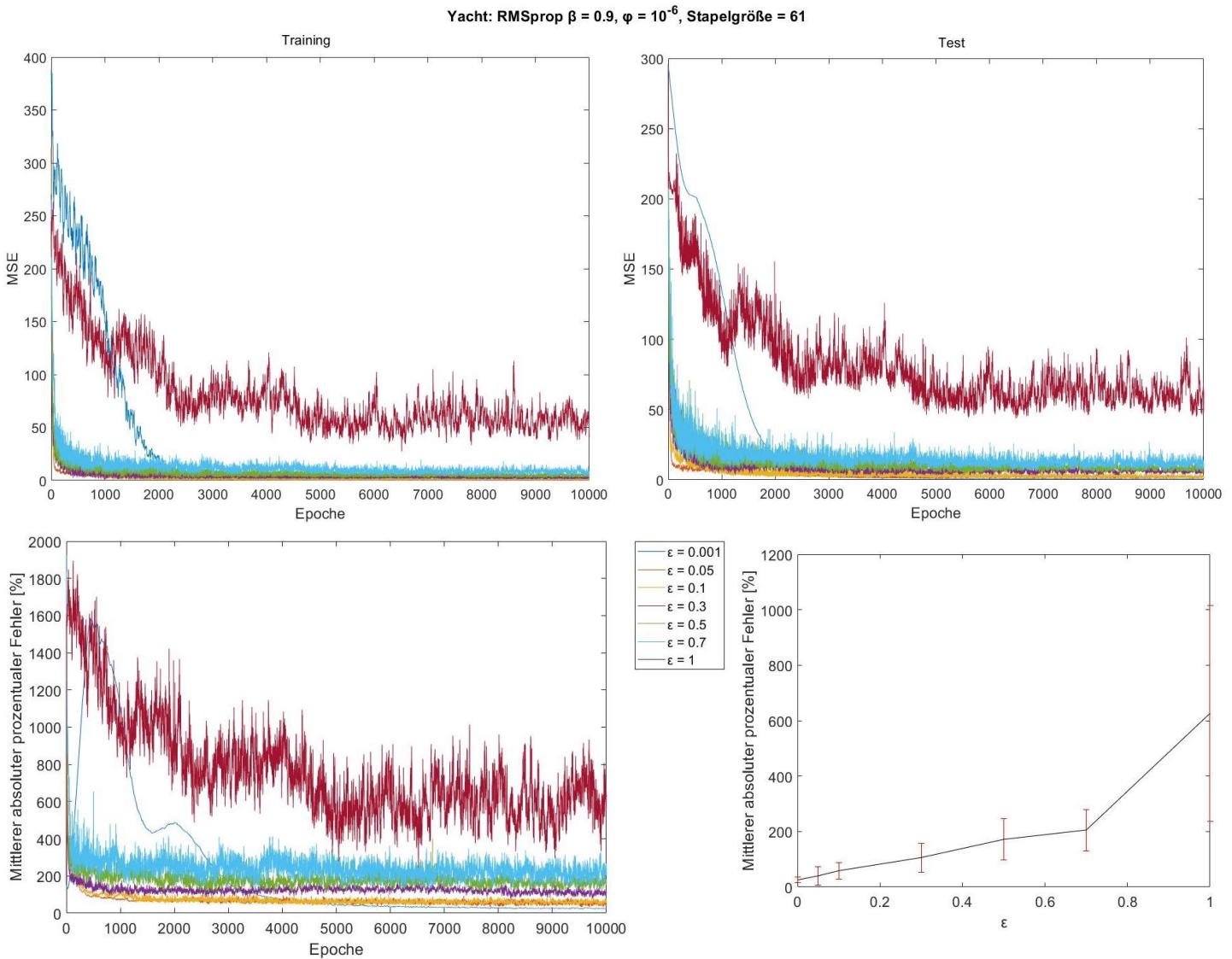


Abbildung 14: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele (unten links) des Yacht Hydrodynamics-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate. Unten rechts ist der erreichte mittlere absolute prozentuale Fehler der Testbeispiele nach 10000 Epochen Training in Abhängigkeit der Lernrate dargestellt.

In Abbildung 14 ist zu erkennen, dass die Stärke der Oszillationen der Verläufe des MSE sowie des mittleren absoluten prozentualen Fehlers mit steigender Lernrate zunimmt. Demzufolge repräsentiert eine Lernrate von 0.001 einen stabilen Gradientenabstieg unter Verwendung des Yacht Hydrodynamics-Datensatzes und erreicht auch den niedrigsten mittleren absoluten prozentualen Fehler aller verwendeten Lernraten. Der Verlauf des mittleren absoluten prozentualen Fehlers der Lernrate 0.001 erreicht jedoch erst nach etwa 6000 Epochen einen Plateauwert wohingegen die Lernraten 0.05, 0.1 und 0.3 bereits nach etwa 100 bis 500 Epochen einen Plateauwert erreichen. Diese Plateauwerte sind jedoch höher im Vergleich zu dem Plateauwert der Lernrate 0.001. Ab einer Lernrate von 0.3 beginnt die Schrittweite des Gradientenabstiegs das Gradientenabstiegsverfahren zu

erschweren, sodass es im Mittel zu einem höheren MSE-Werten und einem höheren mittleren absoluten prozentualen Fehler konvergiert.

#### 4.2.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors

Dieses Experiment soll den Effekt des Vergessensfaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Yacht Hydrodynamics-Datensatzes überprüfen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie der mittlere absolute prozentuale Fehler der Testbeispiele innerhalb jeder Epoche in Abbildung 15 dargestellt.

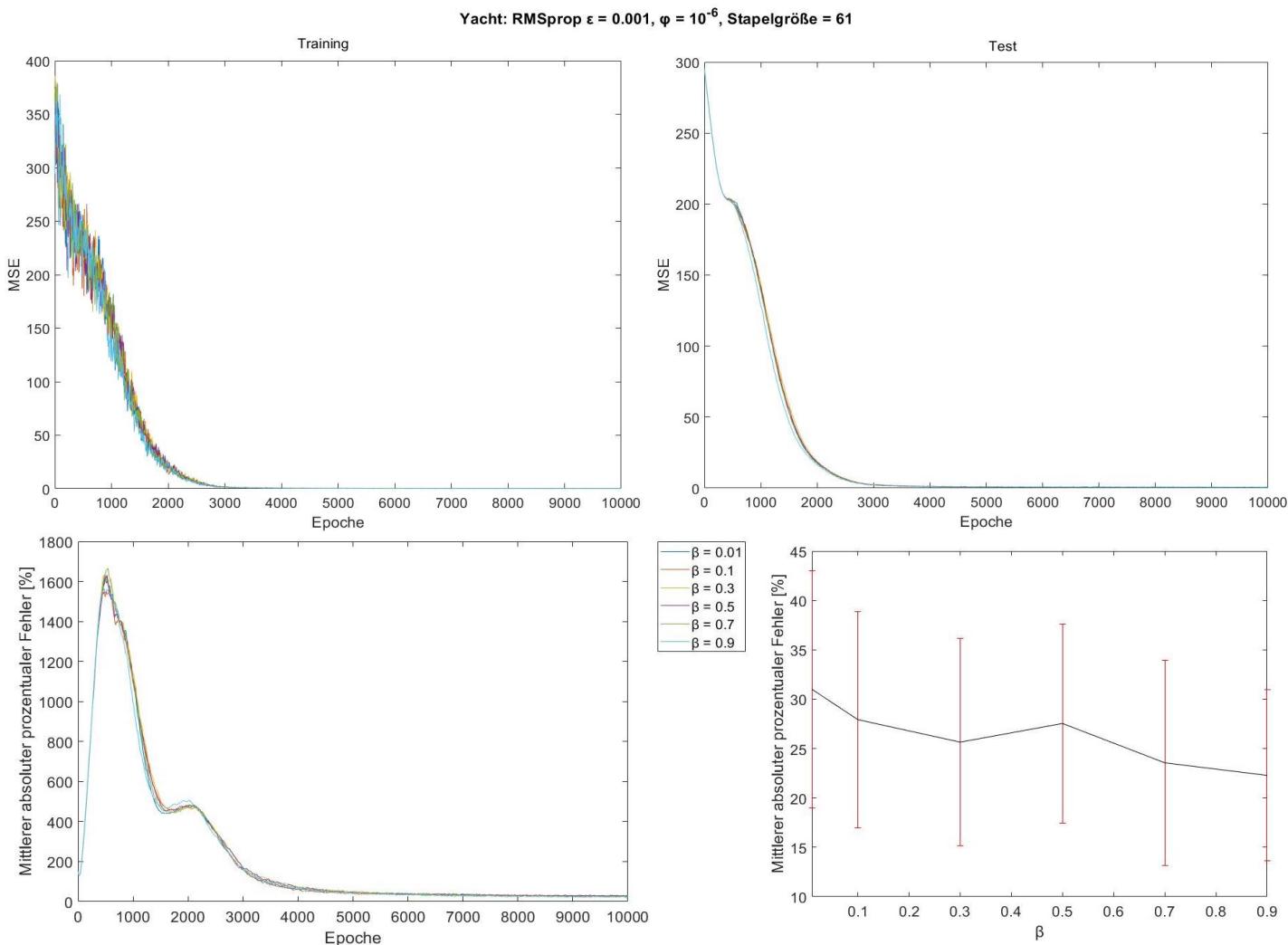


Abbildung 15: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele (unten links) des Yacht Hydrodynamics-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors. Unten rechts ist der erreichte mittlere absolute prozentuale Fehler der Testbeispiele nach 10000 Epochen Training in Abhängigkeit des Vergessensfaktors dargestellt.

#### 4. Experimentelle Untersuchungen

In Abbildung 15 ist zu erkennen, dass der Vergessensfaktor nur einen geringen Effekt auf den Lernerfolg des künstlichen neuronalen Netzes hat. Der Verlauf des MSE der Trainingsbeispiele darauf hin, dass der Gradientenabstiegsalgorithmus abhängig von dem verwendeten Vergessensfaktor eine leicht unterschiedliche Route im Hyperraum der Kostenfunktion einschlägt. Dies ist auch im Verlauf des mittleren absoluten prozentualen Fehlers durch die leichte Verschiebung des Verlaufs erkennbar. Nichtsdestotrotz konvergieren alle Verläufe der Vergessensfaktoren zu nahezu den gleichen MSE-Werten bzw. dem gleichen mittleren absoluten prozentualen Fehler, wobei der niedrigste erreichte mittlere absolute prozentuale Fehler mit steigendem Vergessensfaktor sinkt.

##### **4.2.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors**

Dieses Experiment soll den Effekt des Unschärfeefaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Yacht Hydrodynamics-Datensatzes ermitteln. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie der mittlere absolute prozentuale Fehler der Testbeispiele innerhalb jeder Epoche in Abbildung 16 dargestellt.

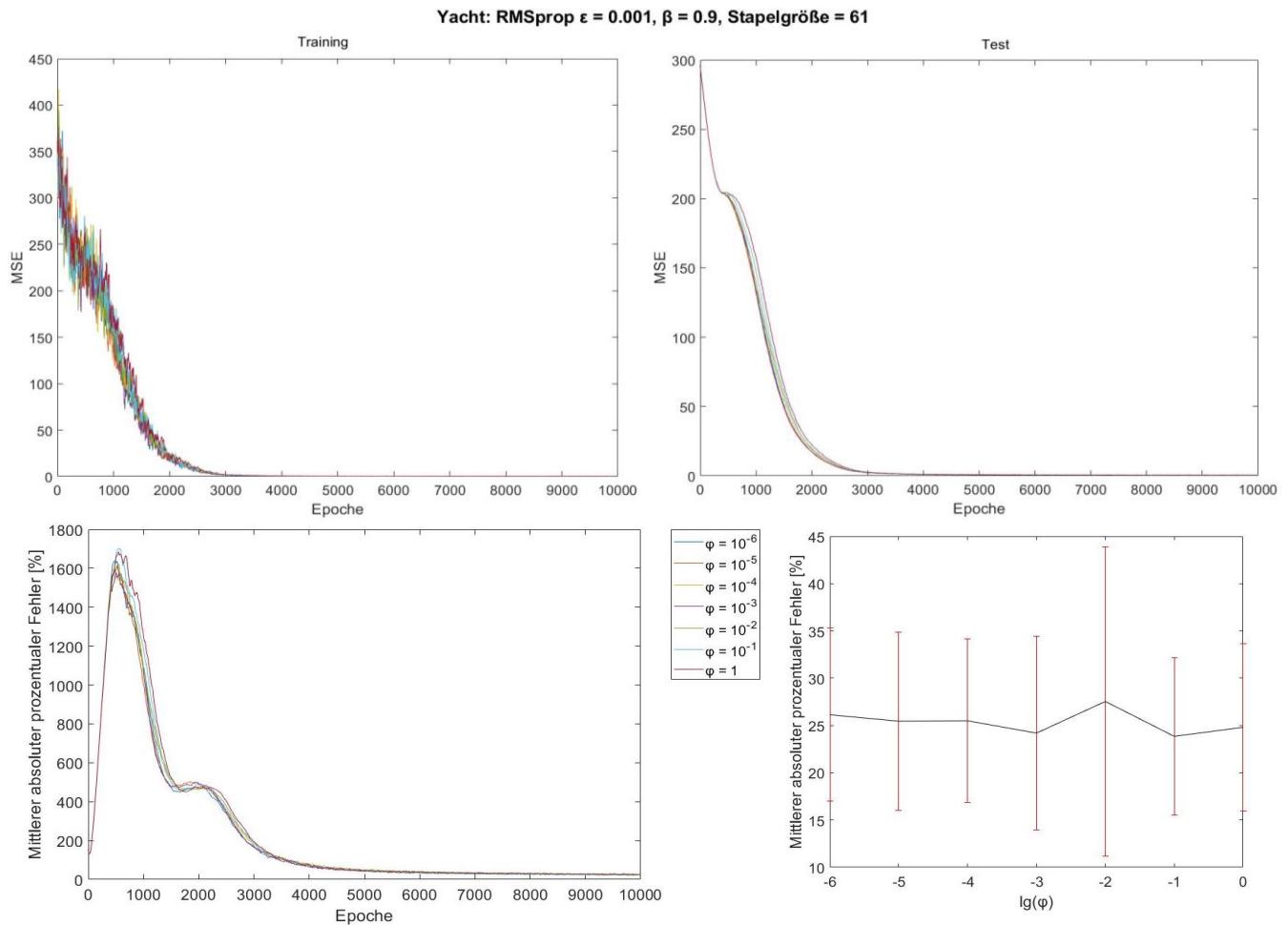


Abbildung 16: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele (unten links) des Yacht Hydrodynamics-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors. Unten rechts ist der erreichte mittlere absolute prozentuale Fehler der Testbeispiele nach 10000 Epochen Training in Abhängigkeit des logarithmierten Unschärfeefaktors dargestellt.

In Abbildung 16 ist zu erkennen, dass der Unschärfeefaktor nur einen geringen Effekt auf den Lernerfolg des künstlichen neuronalen Netzes hat. Aus dem Verlauf der MSE-Werte der Testbeispiele kann man erkennen, dass ab einem Unschärfeefaktor von  $10^{-2}$  der Verlauf des MSE-Wertes bzw. des mittleren absoluten prozentualen Fehlers hin zu höheren Werten verschoben wird. Dadurch scheint ein hoher Unschärfeefaktor lediglich das Gradientenabstiegsverfahren mittels RMSprop zu verlangsamen, da alle Verläufe der Unschärfeefaktoren auf nahezu die gleichen Plateauwerte konvergieren.

## 4. Experimentelle Untersuchungen

### 4.2.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel

Dieses Experiment soll den Effekt der Größe der Mini-Stapel von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Yacht Hydrodynamics-Datensatzes untersuchen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie der mittlere absolute prozentuale Fehler der Testbeispiele innerhalb jeder Epoche in Abbildung 17 dargestellt.

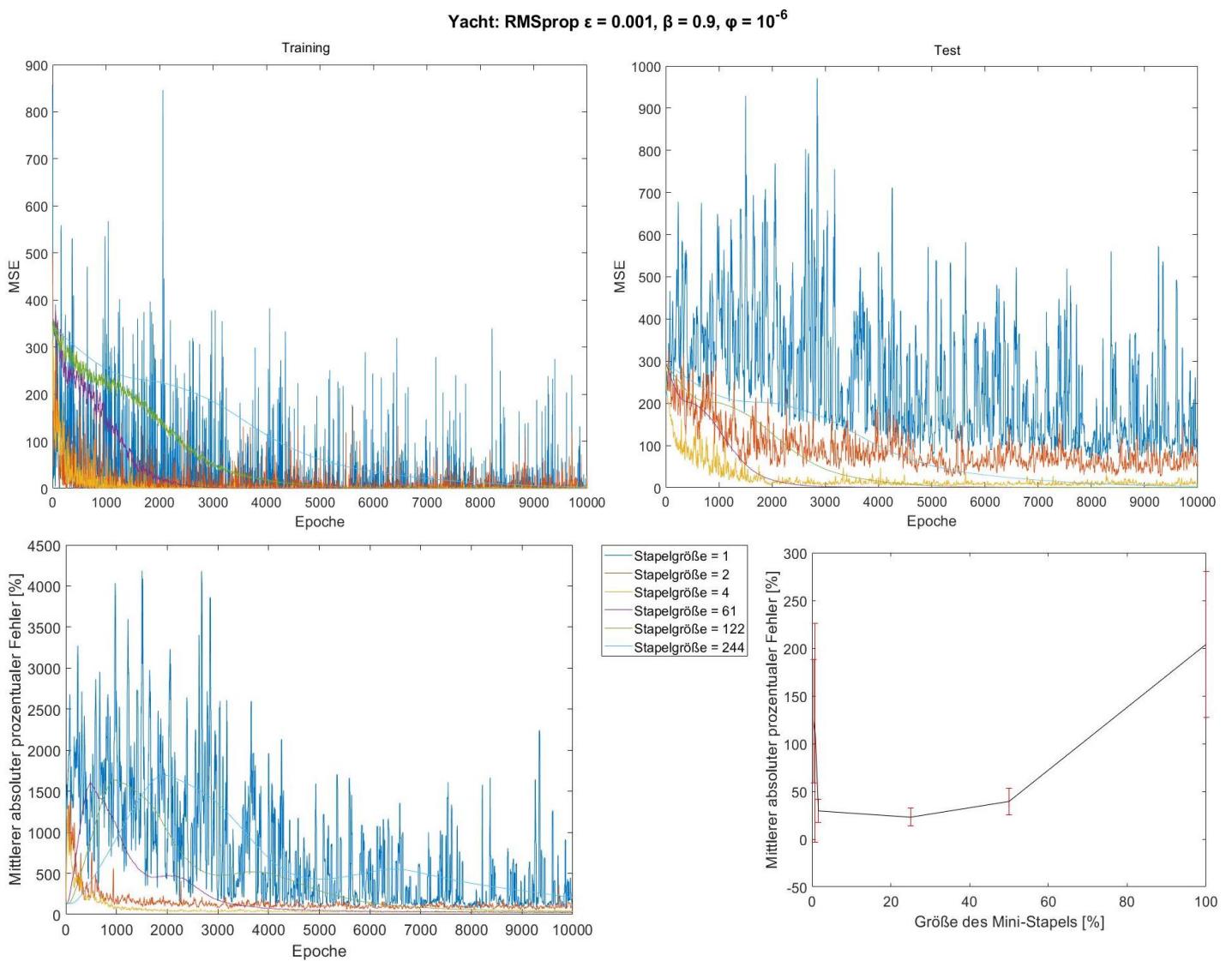


Abbildung 17: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele (unten links) des Yacht Hydrodynamics-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels. Unten rechts ist der erreichte mittlere absolute prozentuale Fehler der Testbeispiele nach 10000 Epochen Training in Abhängigkeit der Größe des Mini-Stapels im Verhältnis zu der gesamten Anzahl der Trainingsbeispiele dargestellt.

Aus Abbildung 17 ist zu entnehmen, dass eine Mini-Stapelgröße von 4 und 61 innerhalb weniger Epochen den niedrigsten MSE-Wert bzw. den niedrigsten Wert des mittleren absoluten prozentualen Fehlers erreicht als eine Mini-Stapelgröße von 1, 2 und 244. Die Mini-Stapelgröße von 122 erreicht einen ähnlich niedrigen MSE-Wert und mittleren absoluten prozentualen Fehler, wie die Mini-Stapelgröße 4 und 61, jedoch erst nach etwa 9000 Epochen. Erwähnenswert ist hierbei, dass eine Mini-Stapelgröße von 1 einem stochastischen Gradientenabstieg von RMSprop gleicht, wohingegen eine Mini-Stapelgröße von 244 einem Stapel-Gradientenabstieg von RMSprop gleicht. Bei den Mini-Stapelgrößen 61, 122 und 244 lässt sich außerdem erkennen, dass der Gradientenabstieg mit steigender Mini-Stapelgröße, und damit einem steigenden Charakter des Stapel-Gradientenabstiegs, verlangsamt wird. Trotz des niedrigen MSE-Wertes der Testbeispiele und der Trainingsbeispiele am Ende der 10000 Epochen, wird bei einer Mini-Stapelgröße von 244 neben der Mini-Stapelgröße von 1 der schlechteste mittlere absolute prozentuale Fehler ermittelt.

#### 4.2.5. Vergleich der Gradientenabstiegsalgorithmen

Hier soll der Lernerfolg des künstlichen neuronalen Netzwerkes mittels des Gradientenabstiegsalgorithmus RMSprop mit dem Lernerfolg des künstlichen neuronalen Netzwerkes mittels der Gradientenabstiegsalgorithmen BP und Rprop verglichen werden. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie der mittlere absolute prozentuale Fehler der Testbeispiele innerhalb jeder Epoche in Abbildung 18 dargestellt.

## 4. Experimentelle Untersuchungen

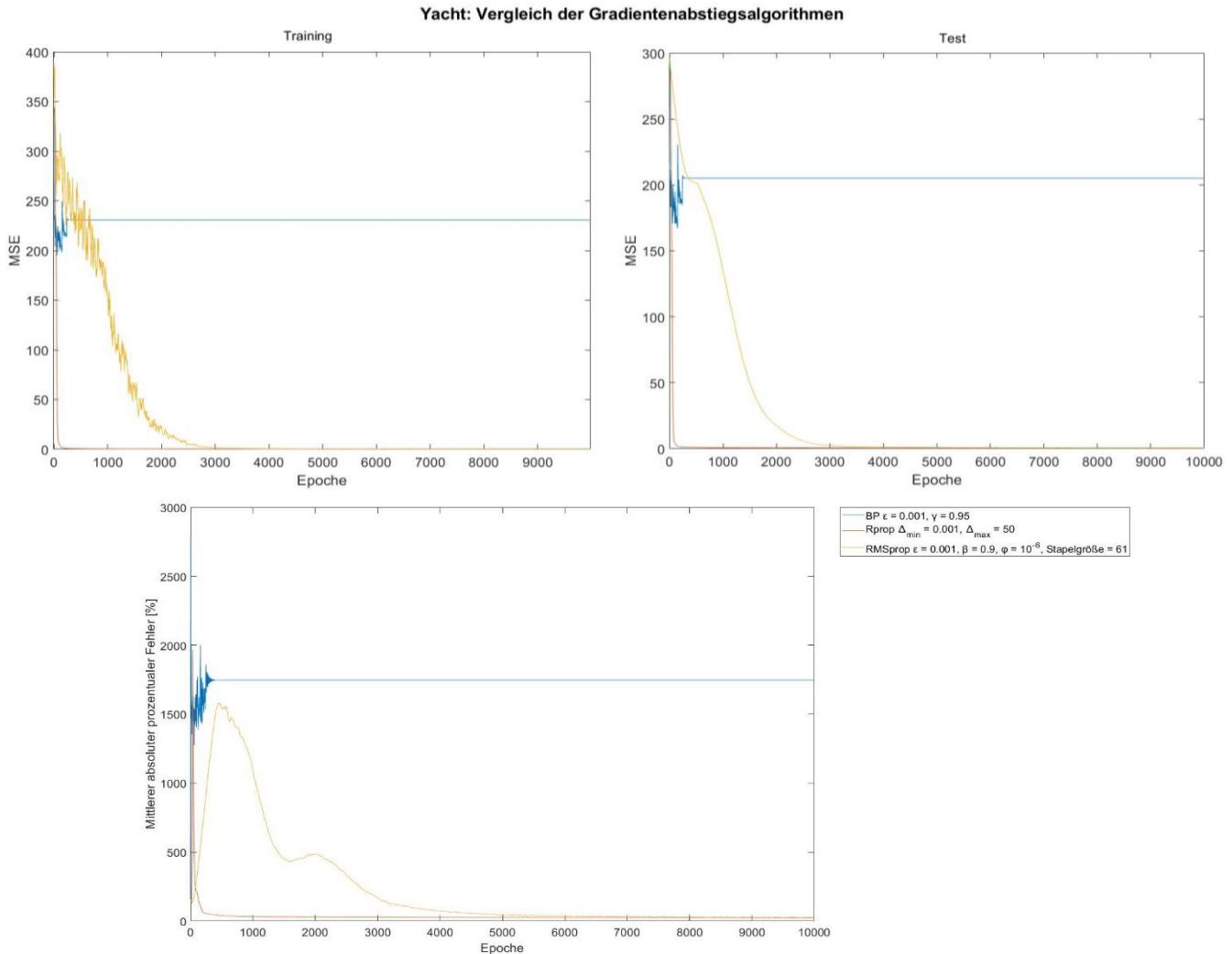


Abbildung 18: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie des mittleren absoluten prozentualen Fehlers (unten) der Testbeispiele des Yacht Hydrodynamics-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop.

In Abbildung 18 ist zu erkennen, dass das Gradientenabstiegsverfahren mittels BP beim Yacht Hydrodynamics-Datensatz innerhalb von 500 Epochen ein Plateauwert erreicht. Dementsprechend ist davon auszugehen, dass mittels BP unter den gegebenen Parametern stets ein lokales Minimum der Kostenfunktion erreicht wird, das im Vergleich zu Rprop und RMSprop schlechter ist. Diese erreichen beide nämlich sehr niedrige MSE-Werte und Werte des mittleren absoluten prozentualen Fehlers. Mittels Rprop geschieht dies innerhalb von etwa 250 Epochen, wohingegen RMSprop deutlich mehr Epochen benötigt. Anhand der Verläufe von RMSprop kann zusätzlich davon ausgegangen werden, dass RMSprop eine andere Route entlang des Hyperraums der Kostenfunktion nimmt als Rprop. Diese scheint jedoch zu ähnlichen Ergebnissen zu führen. In Abbildung 19 wird der erreichte mittlere absolute prozentuale Fehler nach 10000 Epochen, der durchschnittliche mittlere absolute prozentuale Fehler der letzten 1000 Epochen sowie der niedrigste erreichte mittlere absolute prozentuale Fehler innerhalb der 10000 Epochen des jeweiligen Gradientenabstiegsalgoritmus übersichtlich dargestellt.

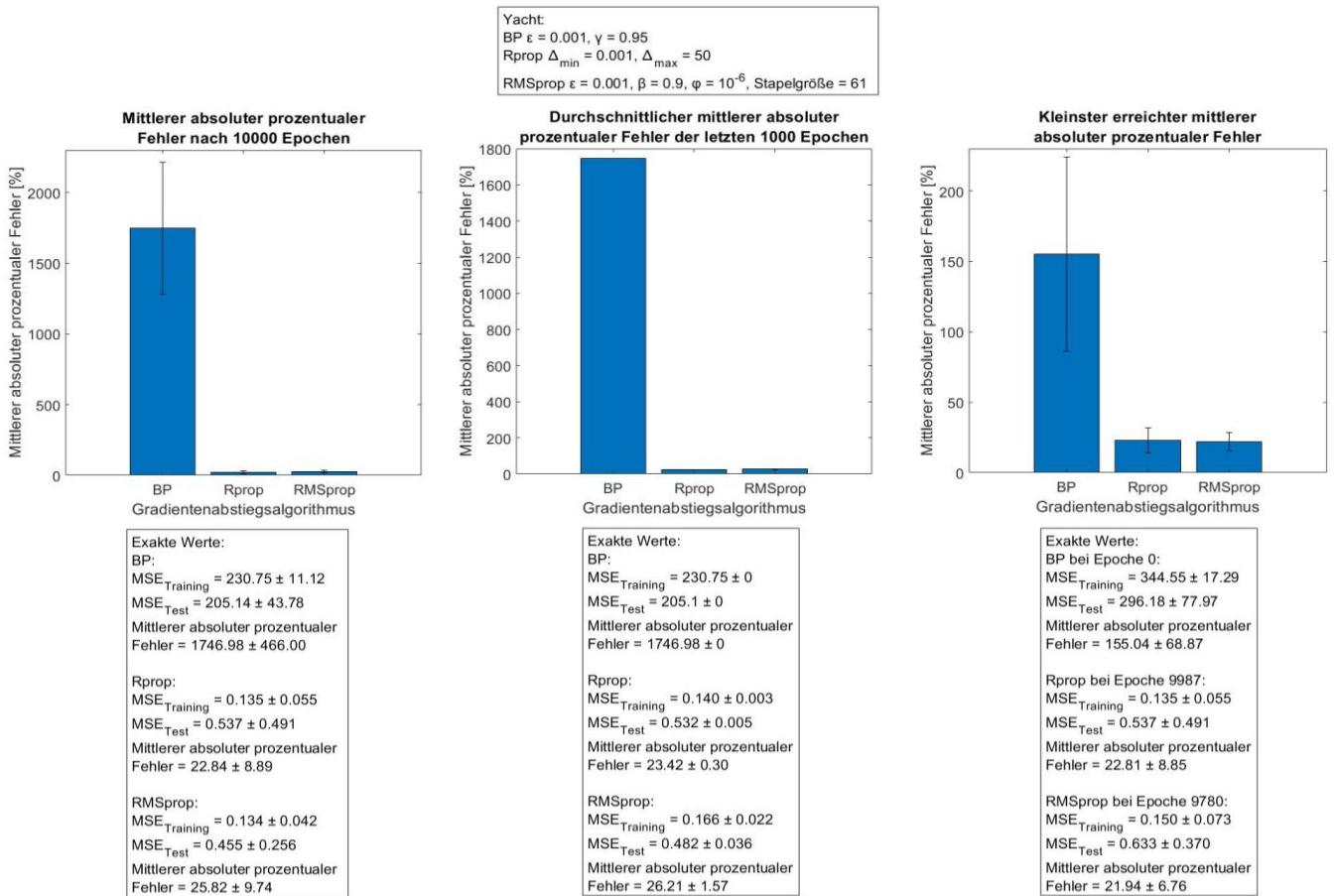


Abbildung 19 Übersicht des nach 10000 Epochen erreichten mittleren absoluten prozentualen Fehlers (links), des durchschnittlichen mittleren absoluten prozentualen Fehlers der letzten 1000 Epochen (mittig) und des kleinsten erreichten mittleren absoluten prozentualen Fehlers innerhalb der 10000 Epochen (rechts) des Yacht Hydrodynamics-Datensatzes von BP, Rprop und RMSprop.

### 4.3. Real Estate Valuation

Bei dem Real Estate Valuation-Datensatz des UCI Machine Learning Repository handelt es sich um ein Regressionsproblem [24]. Der Datensatz besitzt insgesamt 414 Beispiele, dem jedoch 4 Beispiele entnommen wurden, um eine Aufteilung in 80 % Trainingsbeispiele und 20 % Testbeispiele zu ermöglichen. Dementsprechend wurden 410 Beispiele des Datensatzes verwendet, die auf 328 Trainingsbeispiele und 82 Testbeispiele aufgeteilt wurden. Der Datensatz soll dazu verwendet werden den Preis von Häusern pro Flächeneinheit zu berechnen. Als Einheit des Hauspreises wurde 10000 Neuer Taiwan-Dollar pro Ping verwendet, wobei 1 Ping 3.3 Quadratmeter entspricht. Jedes Beispiel besitzt insgesamt 6 Attribute, die als Eingabewerte für das künstliche neuronale Netzwerk zur Ermittlung des Hauspreises pro Flächeneinheit dienen. Diese sind in Tabelle 4 zu sehen.

## 4. Experimentelle Untersuchungen

---

Tabelle 4: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Real Estate Valuation-Datensatzes.

Attribut	Information	Wertebereich
A <sub>1</sub>	Transaktionsdatum	[2012.667, 2013.583]
A <sub>2</sub>	Alter des Hauses in Jahren	[0, 43.8]
A <sub>3</sub>	Distanz zur nächsten Bahn-Station in m	[23.38284, 6488.021]
A <sub>4</sub>	Anzahl der Supermärkte im zu Fuß erreichbaren Umkreis	[0, 10]
A <sub>5</sub>	Geografische Koordinate Breite in °	[24.93207, 25.01459]
A <sub>6</sub>	Geografische Koordinate Länge in °	[121.47353, 121.56627]

Für den Real Estate Valuation-Datensatz wurden künstliche neuronale Netzwerke gemäß (3) mit zwei verborgenen Schichten mittels des n++-Simulatorkerns erstellt. Dementsprechend befanden sich in der Eingabeschicht 6 Eingabeneuronen, in den verborgenen Schichten jeweils 5 verborgene Neuronen und in der Ausgabeschicht 1 Ausgabeneuron. Als Aktivierungsfunktionen der verborgenen Neuronen wurde die Sigmoidfunktion verwendet. Für die Aktivierungsfunktion des Ausgabeneurons wurde die Identitätsfunktion verwendet.

### 4.3.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate

Dieses Experiment soll den Effekt der Lernrate von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Real Estate Valuation-Datensatzes überprüfen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie der mittlere absolute prozentuale Fehler der Testbeispiele innerhalb jeder Epoche in Abbildung 20 dargestellt.

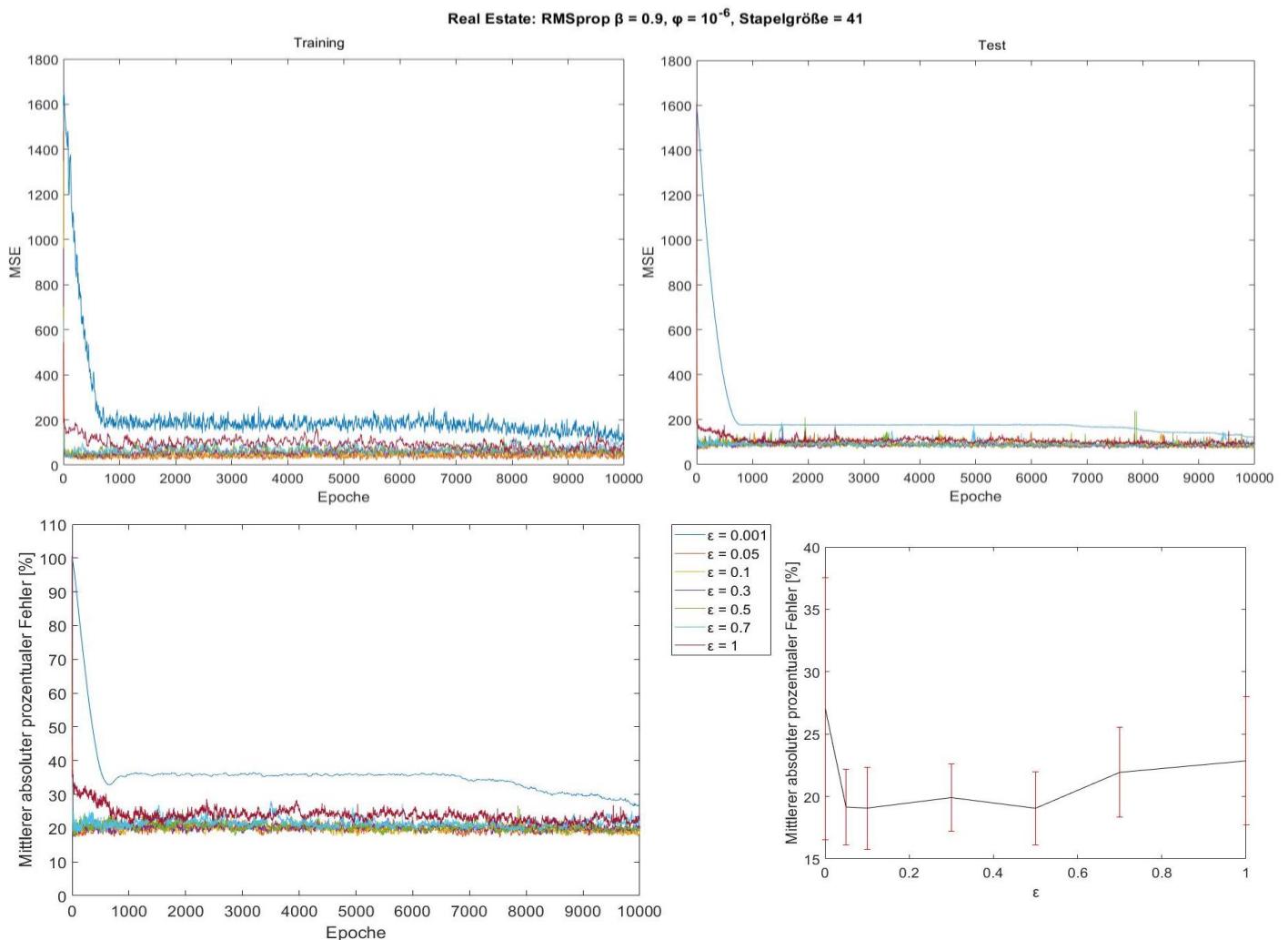


Abbildung 20: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele (unten links) des Real Estate Valuation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate. Unten rechts ist der erreichte mittlere absolute prozentuale Fehler der Testbeispiele nach 10000 Epochen Training in Abhängigkeit der Lernrate dargestellt.

In Abbildung 20 ist zu erkennen, dass eine Lernrate von 0.001 die schlechtesten Ergebnisse bezüglich des Lernerfolg des künstlichen neuronalen Netzwerkes im Vergleich zu den anderen Lernraten erzielt. Jedoch ist ab einer Epoche von etwa 7500 zu erkennen, dass der Verlauf der Lernrate 0.001 den Plateauwert hin zu einem besseren MSE-Wert bzw. einem besseren mittleren absoluten prozentualen Fehler verlässt. Dies deutet darauf hin, dass eine Lernrate von 0.001 eine zu geringe Schrittweite für den Gradientenabstieg anhand des Real Estate Valuation-Datensatzes darstellt. Die Lernraten 0.05, 0.1, 0.3 und 0.5 erzielen dagegen die besten Ergebnisse hinsichtlich des Lernerfolgs des künstlichen neuronalen Netzwerkes. Schließlich scheint ab einer Lernrate von 0.7 die Schrittgröße in Folge des Gradientenabstiegs zu groß zu werden, wodurch im Mittel schlechtere MSE-Werte und ein schlechterer mittlerer absoluter prozentualer Fehler im Vergleich zu den Lernraten 0.05, 0.1, 0.3 und 0.5 erreicht werden.

## 4. Experimentelle Untersuchungen

### 4.3.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors

Dieses Experiment soll den Effekt des Vergessensfaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Real Estate Valuation-Datensatzes analysieren. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie der mittlere absolute prozentuale Fehler der Testbeispiele innerhalb jeder Epoche in Abbildung 21 dargestellt.

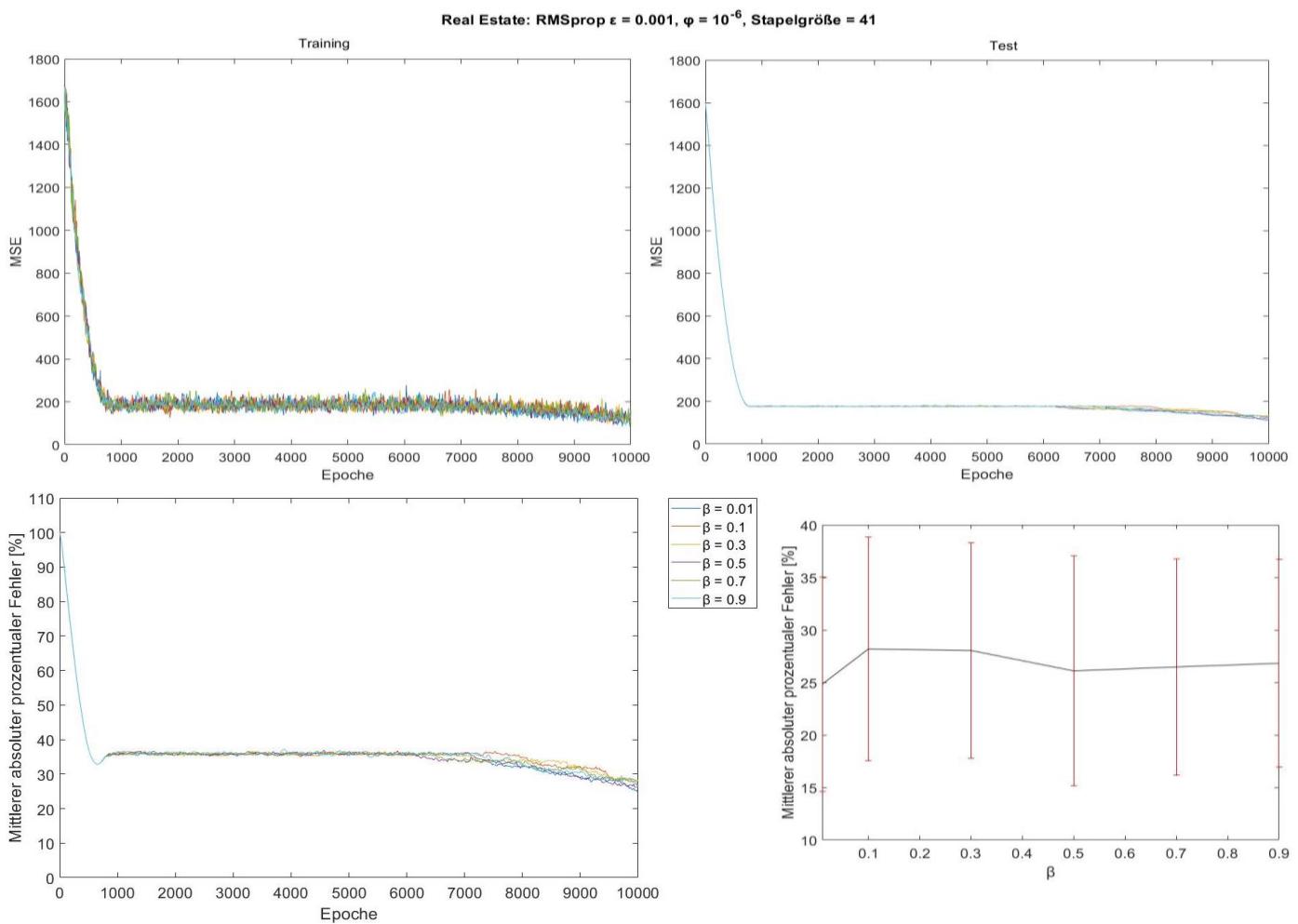


Abbildung 21: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele (unten links) des Real Estate Valuation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors. Unten rechts ist der erreichte mittlere absolute prozentuale Fehler der Testbeispiele nach 10000 Epochen Training in Abhängigkeit des Vergessensfaktors dargestellt.

In Abbildung 21 ist zu beobachten, dass der Vergessensfaktor nur einen geringen Effekt auf den Lernerfolg des künstlichen neuronalen Netzes hat. Im Verlauf des MSE der Testbeispiele und des mittleren absoluten prozentualen Fehlers ist zu sehen, dass der Gradientenabstiegsalgorithmus ab einer Epoche von etwa 6200, abhängig von dem verwendeten Vergessensfaktor, eine unterschiedliche Route im Hyperraum der Kostenfunktion einschlägt. Interessanterweise hat der Vergessensfaktor 0.01 und 0.5 in Epoche 10000 den niedrigsten mittleren absoluten prozentualen Fehler, jedoch kann man vermuten, dass die einzelnen Verläufe, falls man das künstliche neuronale Netzwerk über 10000 Epochen trainieren würde, zu einem gemeinsamen Plateauwert konvergieren würden. Dies ist jedoch nur eine Spekulation und müsste untersucht werden.

### 4.3.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors

Dieses Experiment soll den Effekt des Unschärfeefaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Real Estate Valuation-Datensatzes überprüfen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie der mittlere absolute prozentuale Fehler der Testbeispiele innerhalb jeder Epoche in Abbildung 22 dargestellt.

## 4. Experimentelle Untersuchungen

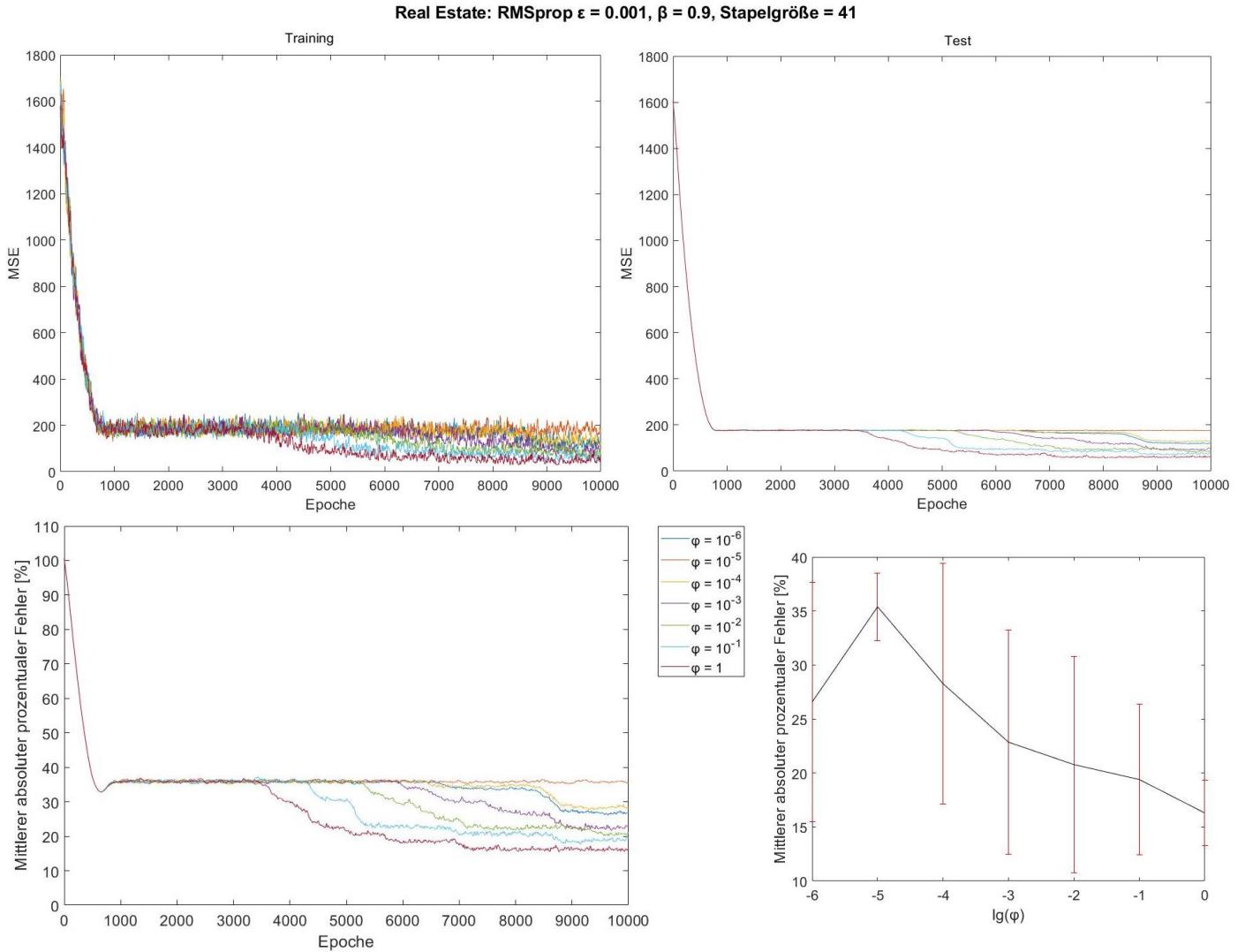


Abbildung 22: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele (unten links) des Real Estate Valuation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors. Unten rechts ist der erreichte mittlere absolute prozentuale Fehler der Testbeispiele nach 10000 Epochen Training in Abhängigkeit des logarithmierten Unschärfeefaktors dargestellt.

Abbildung 22 zeigt, dass der Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Real Estate Valuation-Datensatzes mit steigendem Unschärfeefaktor steigt, wodurch niedrigere Plateauwerte des MSE und des mittleren absoluten prozentualen Fehlers erreicht werden können. Dies scheint bei dem Verlauf für den Unschärfeefaktor von  $10^{-5}$  und  $10^{-4}$  jedoch eine Ausnahme darzustellen, da der Verlauf des Unschärfeefaktors  $10^{-5}$  keine Änderung des Plateauwerts aufweist, wie es bei den anderen Unschärfeefaktoren der Fall ist. Außerdem kann man darauf deuten, dass bei dem Real Estate Valuation-Datensatz das Gradientenabstiegsverfahren mit steigendem Unschärfeefaktor beschleunigt wird, da die Abzweigung des Verlaufs hin zu niedrigeren Plateauwerten jeweils in einer früheren Epoche stattfindet.

#### 4.3.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel

Dieses Experiment soll den Effekt der Größe der Mini-Stapel von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Real Estate Valuation-Datensatzes analysieren. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie der mittlere absolute prozentuale Fehler der Testbeispiele innerhalb jeder Epoche in Abbildung 23 dargestellt.

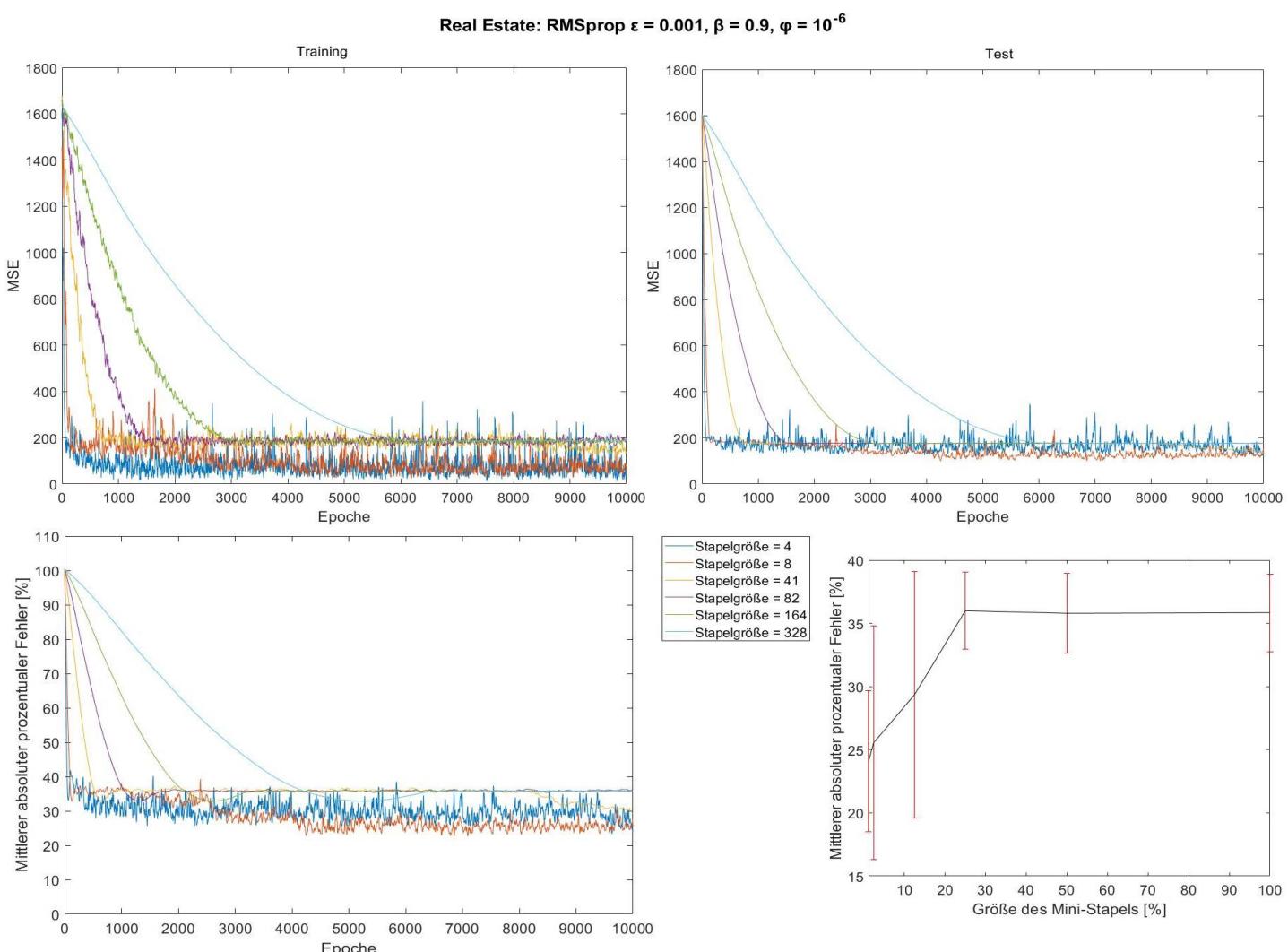


Abbildung 23: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele (unten links) des Real Estate Valuation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels. Unten rechts ist der erreichte mittlere absolute prozentuale Fehler der Testbeispiele nach 10000 Epochen Training in Abhängigkeit der Größe des Mini-Stapels im Verhältnis zu der gesamten Anzahl der Trainingsbeispiele dargestellt.

In Abbildung 23 ist zu erkennen, dass eine Mini-Stapelgröße von 4 und 8 innerhalb weniger Epochen den niedrigsten MSE-Wert bzw. den niedrigsten Wert des mittleren absoluten prozentualen Fehlers

#### 4. Experimentelle Untersuchungen

erreicht als eine Mini-Stapelgröße von 41, 82, 164 und 328. Die Mini-Stapelgröße von 41 erreicht einen recht ähnlich niedrigen MSE-Wert und mittleren absoluten prozentualen Fehler, wie die Mini-Stapelgröße 4 und 8, jedoch erst nach einem Plateauwert, der sich über mehrere Epochen hinwegzieht. Erwähnenswert ist hierbei, dass eine Mini-Stapelgröße von 1 einem stochastischen Gradientenabstieg von RMSprop gleicht, wohingegen eine Mini-Stapelgröße von 328 einem Stapel-Gradientenabstieg von RMSprop gleicht. Bei den Mini-Stapelgrößen 82, 164 und 328 lässt sich außerdem erkennen, dass der Gradientenabstieg mit steigender Mini-Stapelgröße, und damit einem steigenden Charakter des Stapel-Gradientenabstiegs, verlangsamt wird.

##### **4.3.5. Vergleich der Gradientenabstiegsalgorithmen**

Hier soll der Lernerfolg des künstlichen neuronalen Netzwerkes mittels des Gradientenabstiegsalgorithmus RMSprop mit dem Lernerfolg des künstlichen neuronalen Netzwerkes mittels der Gradientenabstiegsalgorithmen BP und Rprop verglichen werden. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie der mittlere absolute prozentuale Fehler der Testbeispiele innerhalb jeder Epoche in Abbildung 24 dargestellt.

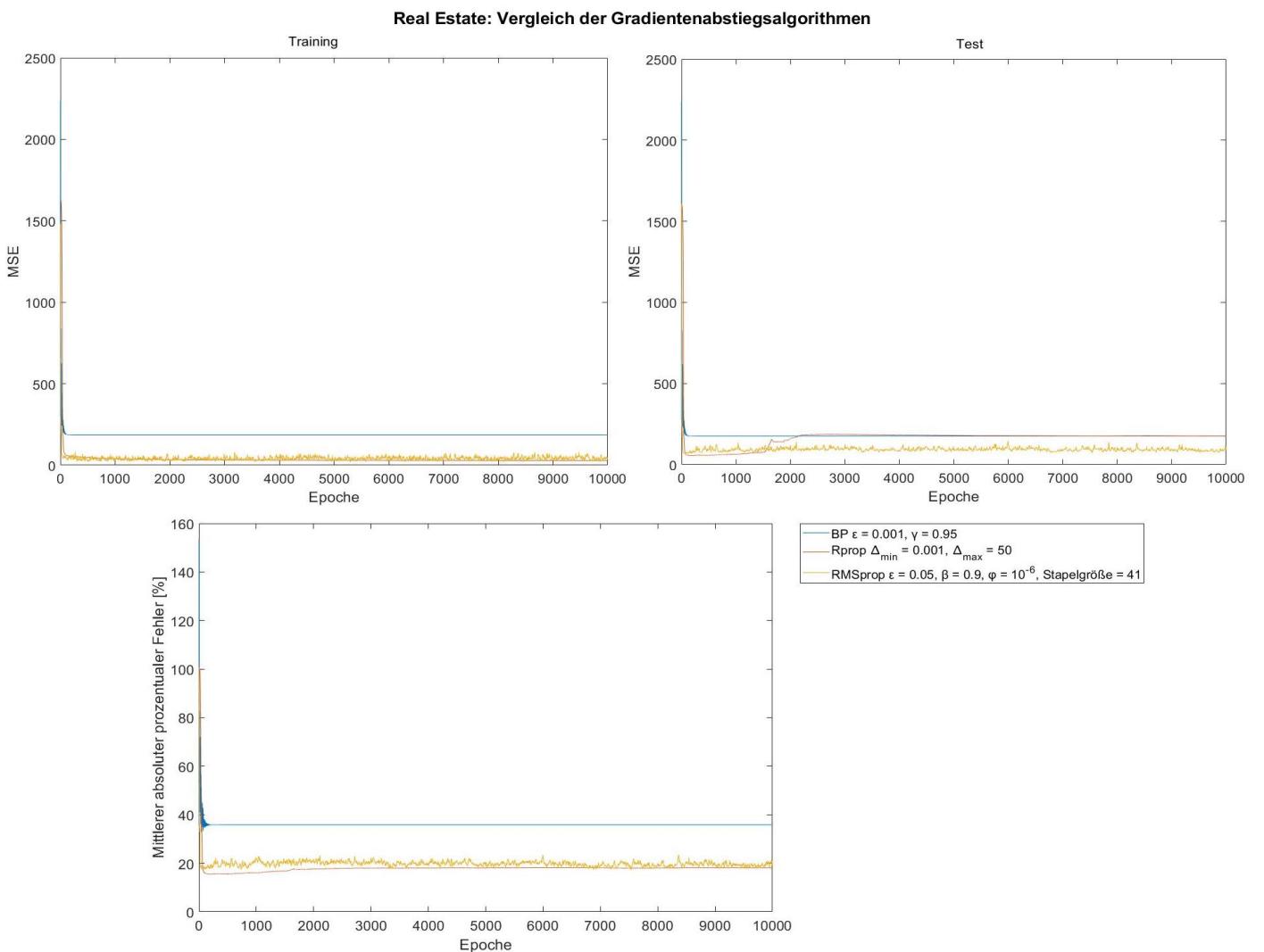


Abbildung 24: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie des mittleren absoluten prozentualen Fehlers (unten) der Testbeispiele des Real Estate Valuation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop.

Aus Abbildung 24 ist festzustellen, dass das Gradientenabstiegsverfahren mittels Rprop, im Vergleich zu den anderen Gradientenabstiegsalgorithmen, zu den niedrigsten MSE-Werten bzw. dem niedrigsten mittleren absoluten prozentualen Fehler führt. Jedoch kann man aus den MSE-Werten der Testbeispiele und dem mittleren absoluten prozentualen Fehler erkennen, dass es zu einer Überanpassung an die Trainingsbeispiele kommt, wodurch die Verläufe zu höheren Werten steigen. Trotz der Überanpassung erreicht Rprop bessere Werte als RMSprop. Bessere Ergebnisse von RMSprop sind durchaus durch Anpassung der einzelnen Parameter denkbar. Betrachtet man z.B. Abbildung 22, so scheint der Unschärfe faktor mitunter eine erhebliche Rolle beim Lernerfolg von RMSprop zu spielen. Das Gradientenabstiegsverfahren mittels BP schneidet im Vergleich am schlechtesten ab. In Abbildung 25 wird der erreichte mittlere absolute prozentuale Fehler nach 10000 Epochen, der durchschnittliche mittlere absolute prozentuale Fehler der letzten 1000 Epochen

## 4. Experimentelle Untersuchungen

sowie der niedrigste erreichte mittlere absolute prozentuale Fehler innerhalb der 10000 Epochen des jeweiligen Gradientenabstiegsalgorithmus übersichtlich dargestellt.

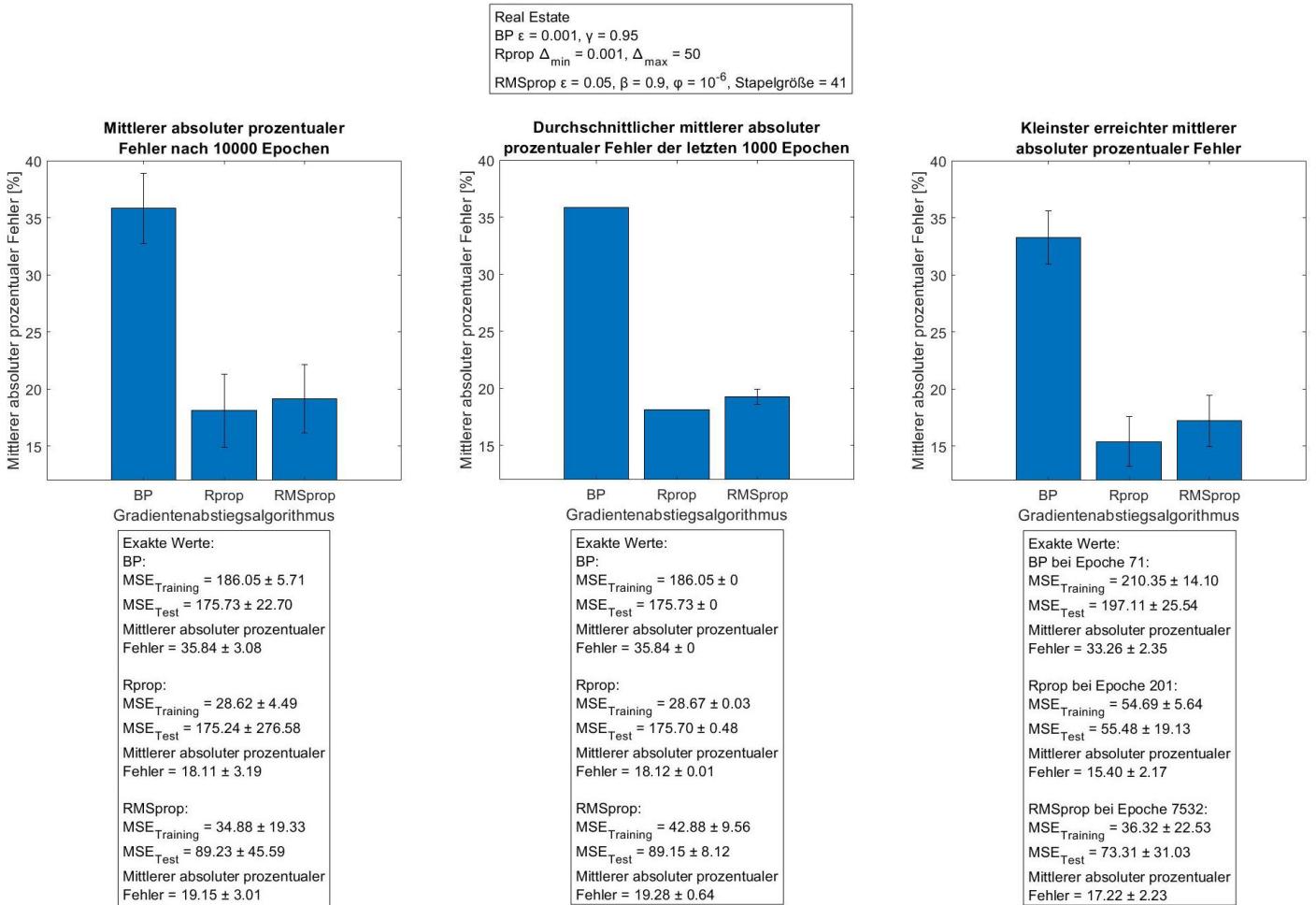


Abbildung 25: Übersicht des nach 10000 Epochen erreichten mittleren absoluten prozentualen Fehlers (links), des durchschnittlichen mittleren absoluten prozentualen Fehlers der letzten 1000 Epochen (mittig) und des kleinsten erreichten mittleren absoluten prozentualen Fehlers innerhalb der 10000 Epochen (rechts) des Real Estate Valuation-Datensatzes von BP, Rprop und RMSprop.

## 4.4. Balance Scale

Bei dem Balance Scale-Datensatz des UCI Machine Learning Repository handelt es sich um ein Klassifikationsproblem [25]. Der Datensatz besitzt insgesamt 625 Beispiele, die auf 500 Trainingsbeispiele und 125 Testbeispiele aufgeteilt wurden. In dem Datensatz existieren 3 mögliche Klassen, die ausgewertet werden können. Bei den möglichen Klassen handelt es sich um mögliche Neigungen einer Waage – L (nach links geneigt), B (balanciert), R (nach rechts geneigt). In den 625 Beispielen des Datensatzes existierten insgesamt 288 Beispiele der Klasse R, 288 Beispiele der Klasse L und 49 Beispiele der Klasse B. Jedes Beispiel besitzt insgesamt 4 Attribute, die als Eingabewerte für

das künstliche neuronale Netzwerk zur Ermittlung der Neigung der Waage dienen. Diese sind in Tabelle 5 beschrieben.

Tabelle 5: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Balance Scale-Datensatzes.

Attribut	Information	Wertebereich
A <sub>1</sub>	Linkes Gewicht	{1, 2, 3, 4, 5}
A <sub>2</sub>	Linke Distanz	{1, 2, 3, 4, 5}
A <sub>3</sub>	Rechtes Gewicht	{1, 2, 3, 4, 5}
A <sub>4</sub>	Rechte Distanz	{1, 2, 3, 4, 5}

Es wurde darauf geachtet, dass der Datensatz zufallsgeneratorbasiert derart gemischt wurde, dass 230 Beispiele für R, 230 Beispiele für L und 40 Beispiele für B in den Trainingsbeispielen vorhanden sind. Folglich befanden sich 58 Beispiele für R, 58 Beispiele für L und 9 Beispiele für B in den Testbeispielen. Für den Balance Scale-Datensatz wurden künstliche neuronale Netzwerke gemäß (3) mit zwei verborgenen Schichten mittels des n++-Simulatorkerns erstellt. Dementsprechend befanden sich in der Eingabeschicht 4 Eingabeneuronen, in den verborgenen Schichten jeweils 6 verborgene Neuronen und in der Ausgabeschicht 3 Ausgabeneuronen. Als Aktivierungsfunktionen der verborgenen Neuronen und der Ausgabeneuronen wurde die Sigmoidfunktion verwendet. Jedes Ausgabeneuron repräsentiert die Wahrscheinlichkeit der Eingabewerte zu einer jeweiligen Neigung der Waage zu gehören.

#### 4.4.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate

Dieses Experiment soll den Effekt der Lernrate von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Balance Scale-Datensatzes untersuchen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 26 dargestellt.

## 4. Experimentelle Untersuchungen

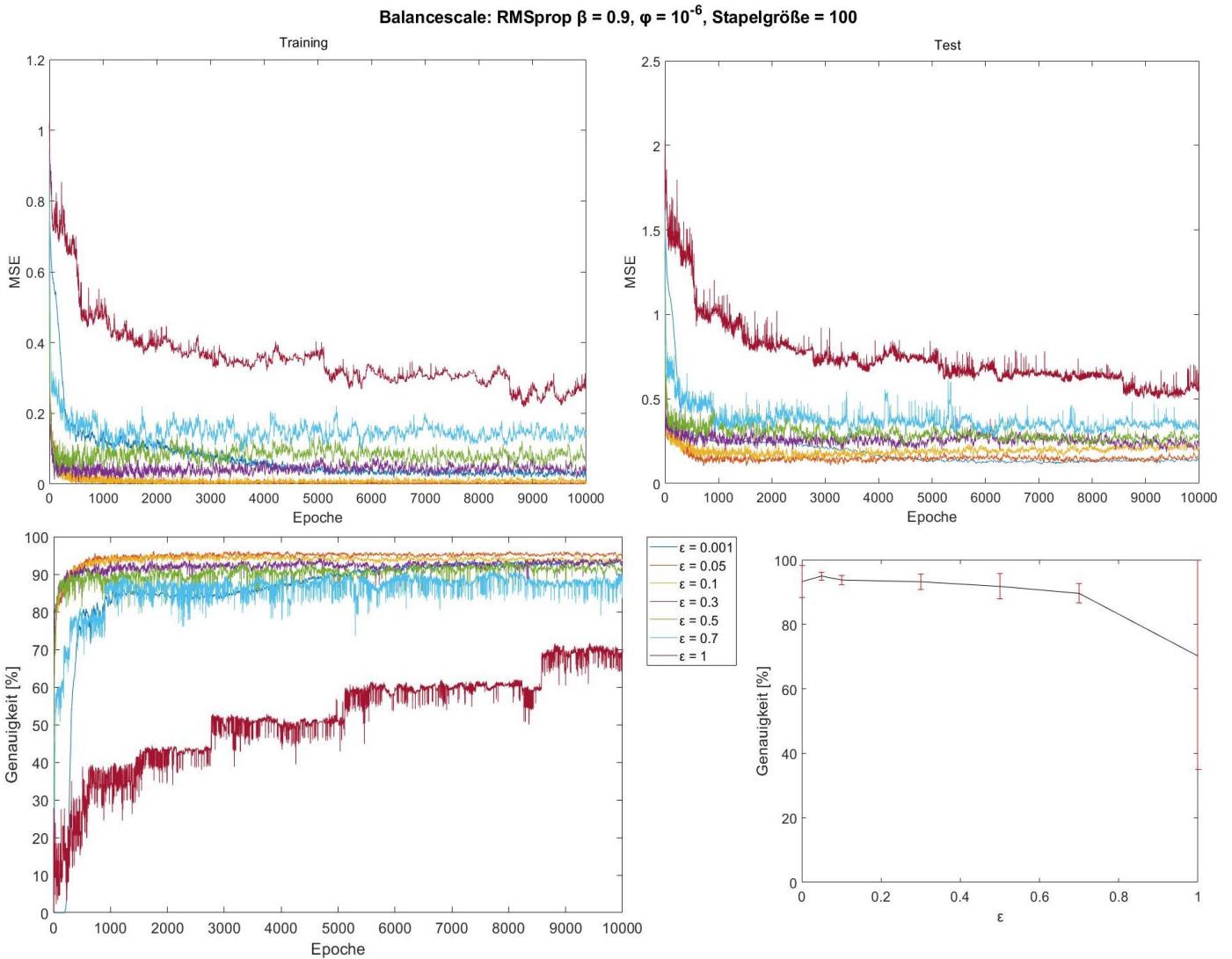


Abbildung 26: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Lernrate dargestellt.

In Abbildung 26 ist zu beobachten, dass RMSprop mit einer Lernrate von 0.05 den besten Lernerfolg anhand des Balance Scale-Datensatzes erzielt. Außerdem erkennt man, dass eine Lernrate von 0.001, 0.1 und 0.3 gegen Ende des Gradientenabstiegs zu demselben Plateauwert der Genauigkeit konvergiert. Ab einer Lernrate von 0.5 beginnt im Mittel der Lernerfolg des künstlichen neuronalen Netzwerkes zu sinken. Dies deutet auf eine zu große Schrittweite des Gradientenabstiegs hin, wodurch die Konvergenz in einem lokalen Minimum erschwert wird. Bei keiner der Lernraten sind deutliche Anzeichen einer Überanpassung an die Trainingsbeispiele zu erkennen.

#### 4.4.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors

Dieses Experiment soll den Effekt des Vergessensfaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Balance Scale-Datensatzes erforschen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 27 dargestellt.

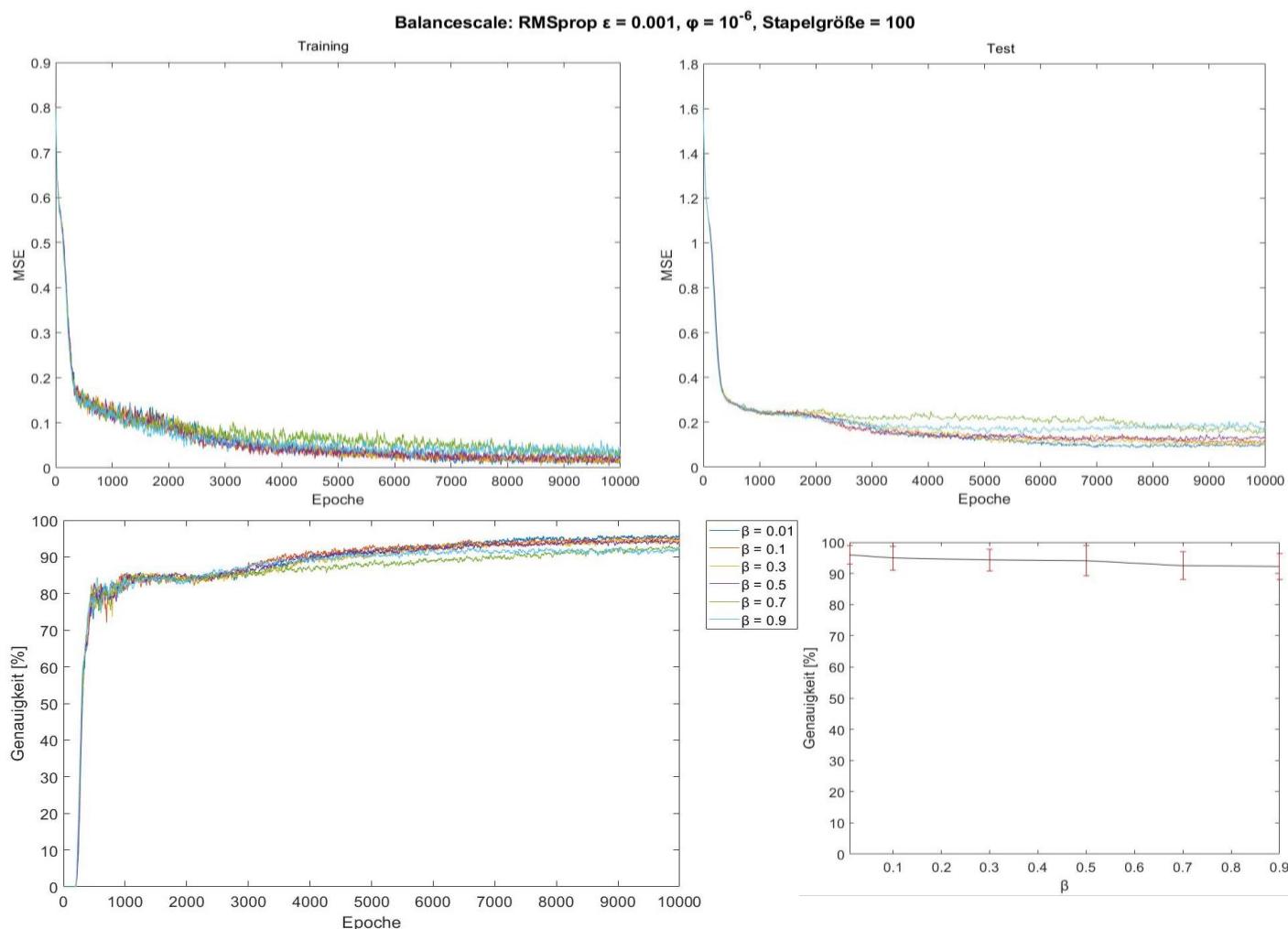


Abbildung 27: Verlauf des MSE der Trainings- (oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des Vergessensfaktors dargestellt.

In Abbildung 27 ist zu sehen, dass mit steigendem Vergessensfaktor der Lernerfolg des künstlichen neuronalen Netzwerks unter Verwendung des Balance Scale-Datensatzes sinkt. Da bei einem kleinen Vergessensfaktor Gradienten vorheriger Mini-Stapel weniger Einfluss auf den gleitenden Durchschnitt haben, könnte bei der Epoche von etwa 2000 der Gradientenabstiegsalgorismus leicht andere Routen

#### 4. Experimentelle Untersuchungen

entlang des Hyperraums der Kostenfunktion, hin zu besseren lokalen Minima, nehmen. Im Gegenteil dazu, ist bei einem großen Vergessensfaktor der Einfluss der Gradienten vorheriger Mini-Stapel auf den gleitenden Durchschnitt größer, wodurch im Vergleich zu den kleineren Vergessensfaktoren weniger günstige Minima erreicht werden. Allerdings könnte man auch untersuchen, wie sich die verschiedenen Vergessensfaktoren bei mehr als 10000 Epochen verhalten, denn es könnte auch durchaus sein, dass alle Verläufe zu einem Plateauwert konvergieren und damit zu dem gleichen oder ähnlichen lokalen Minimum führen.

##### **4.4.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors**

Dieses Experiment soll den Effekt des Unschärfeefaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Balance Scale-Datensatzes ermitteln. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 28 dargestellt.

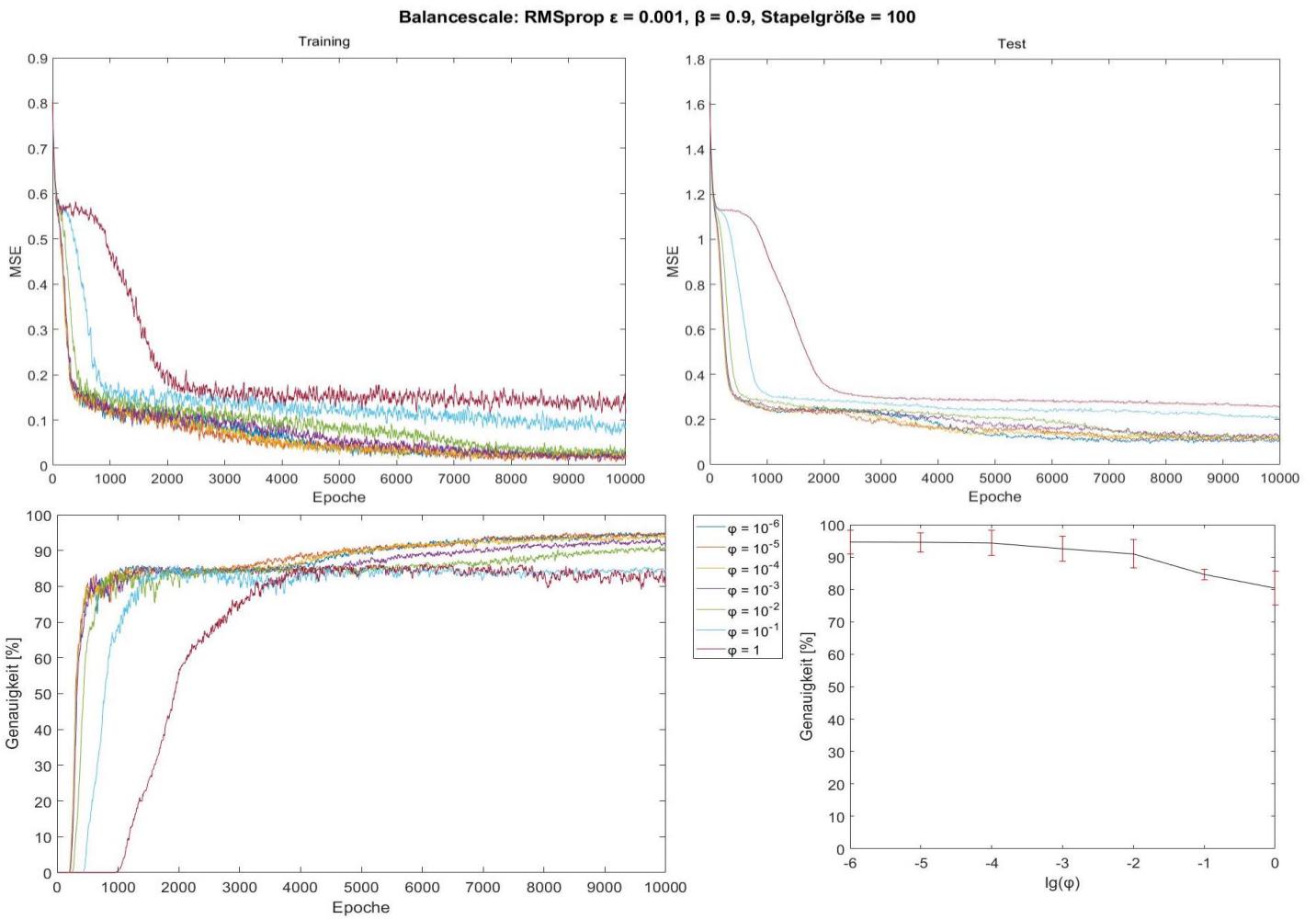


Abbildung 28: Verlauf des MSE der Trainings- (oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des logarithmierten Unschärfeefaktors dargestellt.

In Abbildung 28 kann man eindeutig beobachten, dass der Lernerfolg des künstlichen neuronalen Netzwerkes mit steigendem Unschärfeefaktor abnimmt. Es werden dadurch im Mittel schlechtere MSE-Werte für Trainings- sowie Testbeispiele und dementsprechend schlechtere Genauigkeitswerte mit steigenden Unschärfeefaktoren erreicht. Dieser Effekt tritt deutlich ab einem Unschärfeefaktor von  $10^{-3}$  auf.

#### 4.4.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel

Dieses Experiment soll den Effekt der Größe der Mini-Stapel von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Balance Scale-Datensatzes überprüfen.

#### 4. Experimentelle Untersuchungen

Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 29 dargestellt.

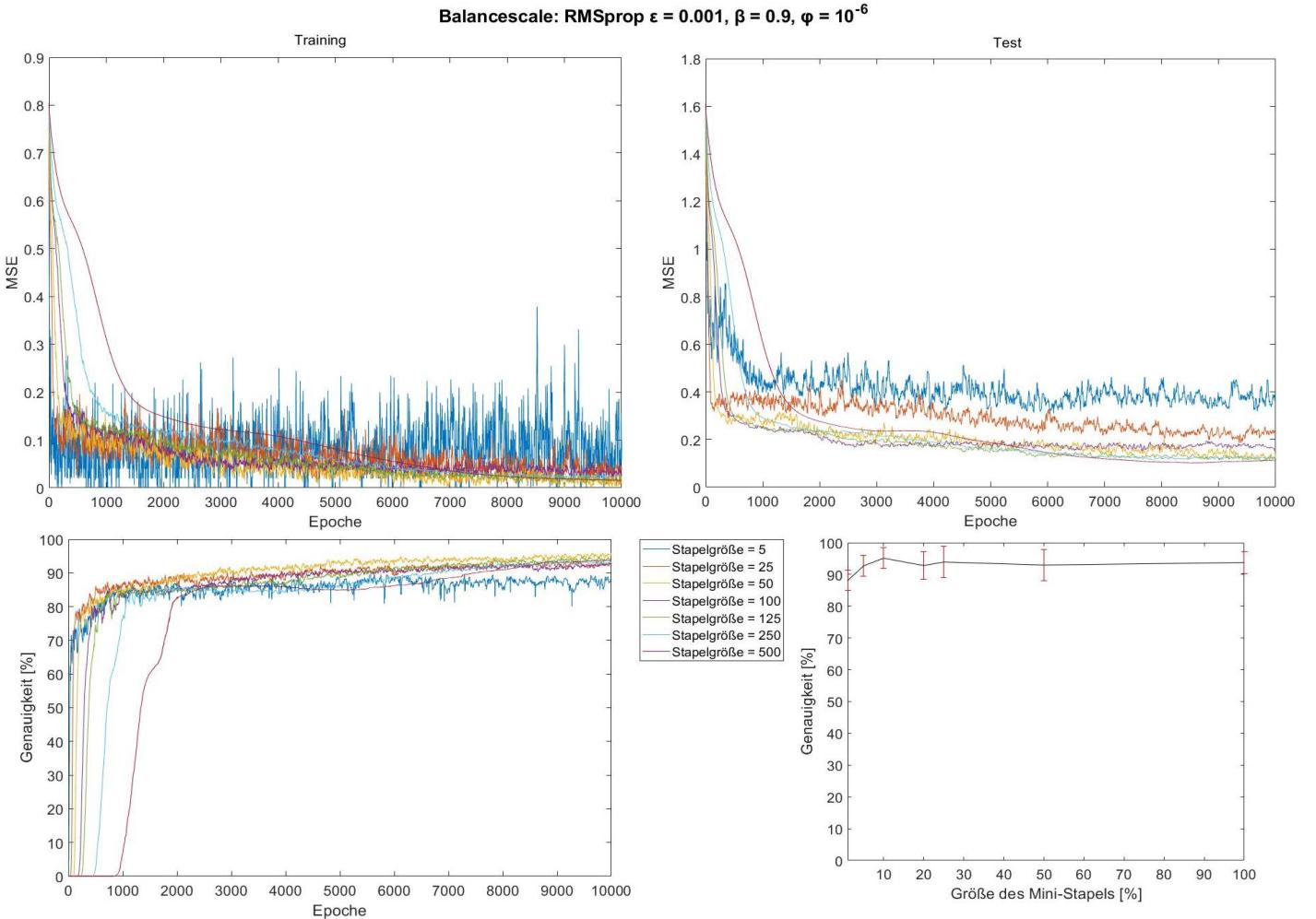


Abbildung 29: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Größe des Mini-Stapels im Verhältnis zu der gesamten Anzahl der Trainingsbeispiele dargestellt.

Aus Abbildung 29 ist zu entnehmen, dass das Gradientenabstiegsverfahren mit einer Mini-Stapelgröße von 5, 25 und 100 den schlechtesten Lernerfolg des künstlichen neuronalen Netzwerkes aufzeigt. Erwähnenswert ist hierbei, dass eine Mini-Stapelgröße von 1 einem stochastischen Gradientenabstieg von RMSprop gleichen würde, wohingegen eine Mini-Stapelgröße von 500 einem Stapel-Gradientenabstieg von RMSprop gleichen würde. Aus der genannten Abbildung lässt sich demnach erkennen, dass bei der Mini-Stapelgröße von 50 die höchsten Genauigkeitswerte erzielt werden. Die Mini-Stapelgröße von 5 oszilliert aufgrund der hohen Anzahl an Gewichtsaktualisierungen innerhalb einer Epoche am meisten und erreicht die niedrigsten Genauigkeitswerte bzw. die höchsten MSE-Werte zwischen allen Mini-Stapelgrößen. Der Gradientenabstieg mit der Mini-Stapelgröße von 500 ist

im Vergleich, aufgrund nur einer Gewichtsaktualisierung innerhalb einer Epoche, am langsamsten, erreicht jedoch Genauigkeitswerte die denen der Mini-Stapelgröße von 125 und 250 ähneln.

#### **4.4.5. Vergleich der Gradientenabstiegsalgorithmen**

Hier soll der Lernerfolg des künstlichen neuronalen Netzwerkes mittels des Gradientenabstiegsalgorithmus RMSprop mit dem Lernerfolg des künstlichen neuronalen Netzwerkes mittels der Gradientenabstiegsalgorithmen BP und Rprop verglichen werden. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 30 dargestellt.

## 4. Experimentelle Untersuchungen

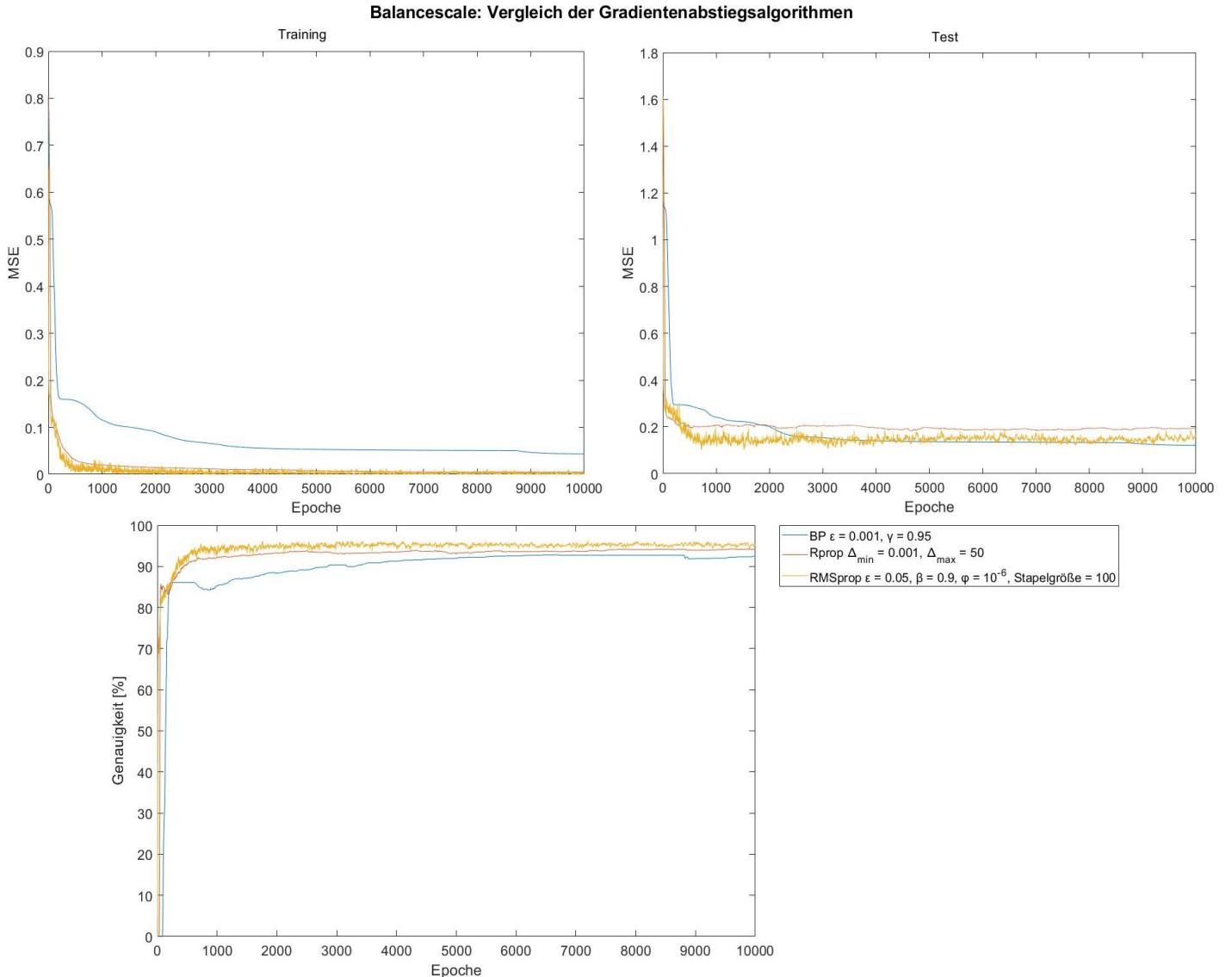


Abbildung 30: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten) des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop.

In Abbildung 30 ist zu erkennen, dass Rprop und RMSprop effizient innerhalb von etwa 1000 Epochen einen hohen Genauigkeitswert erreichen, jedoch erreicht RMSprop im Vergleich zu Rprop einen höheren Plateauwert der Genauigkeit. Aus dem Verlauf des MSE-Wertes der Testbeispiele und der Genauigkeit lässt sich außerdem erkennen, dass BP zwar den niedrigsten Genauigkeitswert aller Gradientenabstiegsalgorithmen erreicht, jedoch nimmt BP gleichzeitig auch die niedrigsten MSE-Werte der Testbeispiele an. In Abbildung 31 wird die erreichte Genauigkeit nach 10000 Epochen, die durchschnittliche Genauigkeit der letzten 1000 Epochen und die maximal erreichte Genauigkeit innerhalb der 10000 Epochen des jeweiligen Gradientenabstiegsalgorithmus übersichtlich dargestellt.

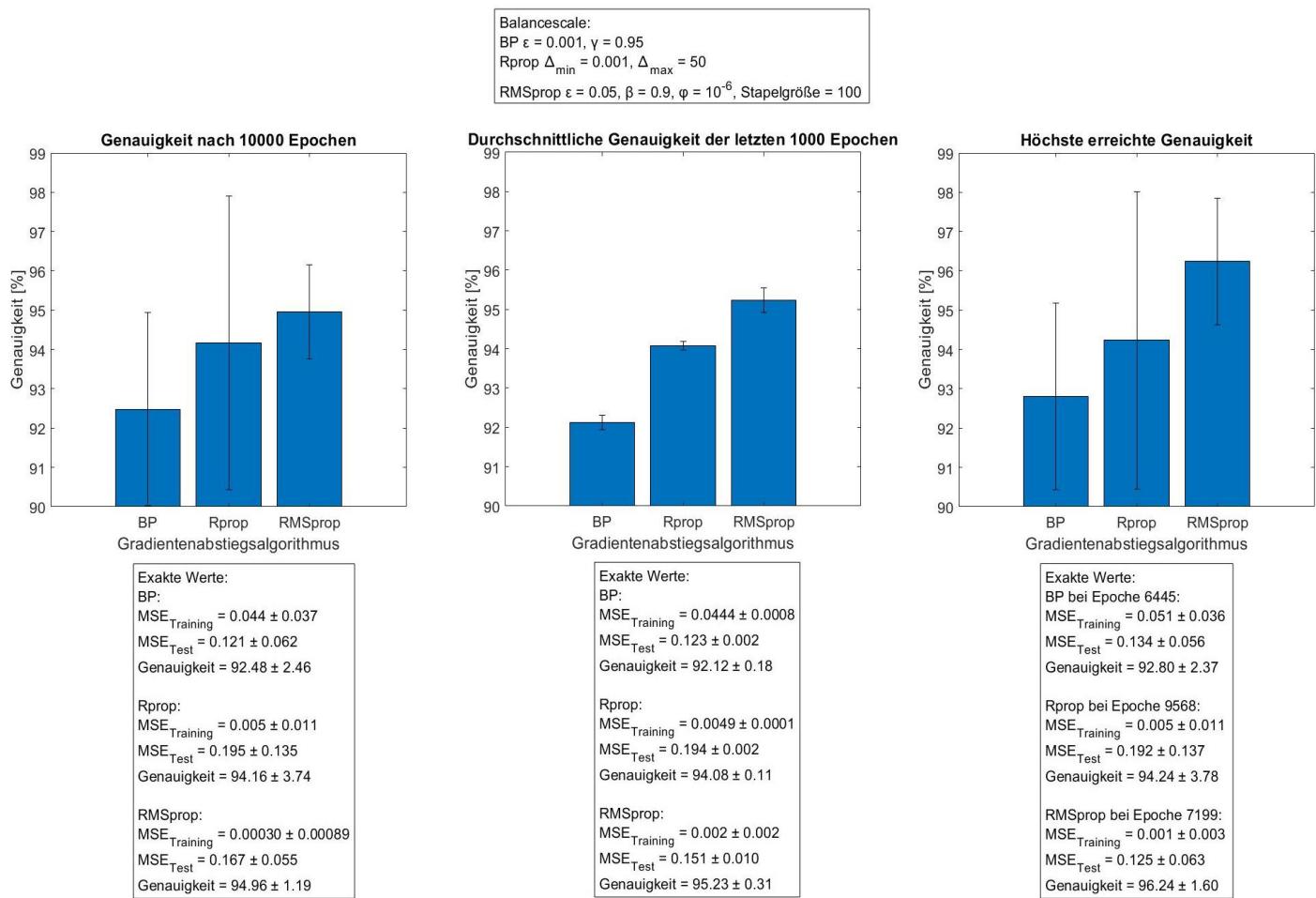


Abbildung 31: Übersicht der nach 10000 Epochen erreichten Genauigkeit (links), der durchschnittlichen Genauigkeit der letzten 1000 Epochen (mittig) und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen (rechts) des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes von BP, Rprop und RMSprop.

## 4.5. QSAR Biodegradation

Bei dem QSAR Biodegradation-Datensatz des UCI Machine Learning Repository handelt es sich um ein Klassifikationsproblem [26]. Der Datensatz besitzt insgesamt 1055 Beispiele, dem jedoch 5 Beispiele entnommen wurden, um eine Aufteilung in 80 % Trainingsbeispiele und 20 % Testbeispiele zu ermöglichen. Dementsprechend wurden 1050 Beispiele des Datensatzes verwendet, die auf 840 Trainingsbeispiele und 210 Testbeispiele aufgeteilt wurden. In dem Datensatz existieren 2 mögliche Klassen, die ausgewertet werden können. Bei den möglichen Klassen handelt es sich um die biologische Abbaubarkeit von Molekülen – RB (ready biodegradable) und NRB (not ready biodegradable). In den 1050 Beispielen des Datensatzes existieren insgesamt 356 Beispiele der Klasse RB und 694 Beispiele der Klasse NRB. Jedes Beispiel besitzt insgesamt 41 Attribute, die als Eingabewerte für das künstliche neuronale Netzwerk zur Ermittlung der biologischen Abbaubarkeit des Moleküls dienen. Diese sind in Tabelle 6 ausführlich dargestellt.

#### 4. Experimentelle Untersuchungen

Tabelle 6: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des QSAR Biodegradation-Datensatzes.

Attribut	Information	Wertebereich
A <sub>1</sub>	SpMax_L: Führender Eigenwert der Laplace-Matrix	[2, 6.496]
A <sub>2</sub>	J_Dz(e): Balaban-ähnlicher Index der Barysz-Matrix, gewichtet mit der Sanderson-Elektronegativität	[0.8039, 9.1775]
A <sub>3</sub>	nHM: Anzahl der Schweratome	[0, 12]
A <sub>4</sub>	F01[N-N]: Frequenz der N-N-Bindung im topologischen Abstand 1	[0, 3]
A <sub>5</sub>	F04[C-N]: Frequenz der C-N-Bindung im topologischen Abstand 4	[0, 16]
A <sub>6</sub>	NssssC: Anzahl der Atome des Typs ssssC	[0, 13]
A <sub>7</sub>	nCb-: Anzahl der substituierten Benzol C(sp <sub>2</sub> )	[0, 18]
A <sub>8</sub>	C%: Prozentsatz der C-Atome	[0, 60.7]
A <sub>9</sub>	nCp: Anzahl von endständigen primären C(sp <sub>3</sub> )	[0, 24]
A <sub>10</sub>	nO: Anzahl der Sauerstoffatome	[0, 12]
A <sub>11</sub>	F03[C-N]: Frequenz der C-N-Bindung im topologischen Abstand 3	[0, 27]
A <sub>12</sub>	SdssC: Summe der dssC E-Zustände	[-5.135, 4.722]
A <sub>13</sub>	HyWi_B(m): Hyper-Wiener-ähnlicher Index (Log-Funktion) der Burden-Matrix gewichtet mit der Masse	[1.544, 5.701]
A <sub>14</sub>	LOC: Abgeschnittener zentrischer Index	[0, 4.491]
A <sub>15</sub>	SM6_L: Spektrales Moment der Ordnung 6 der Laplace-Matrix	[4.174, 12.609]
A <sub>16</sub>	F03[C-O]: Frequenz der C-O-Bindung im topologischen Abstand 3	[0, 40]
A <sub>17</sub>	Me: Mittlere atomare Sanderson-Elektronegativität (skaliert am Kohlenstoffatom)	[0.957, 1.311]
A <sub>18</sub>	Mi: Mittleres erstes Ionisationspotenzial (skaliert am Kohlenstoffatom)	[1.022, 1.377]
A <sub>19</sub>	nN-N: Anzahl der N-Hydrazine	[0, 2]
A <sub>20</sub>	nArNO <sub>2</sub> : Anzahl der aromatischen Nitro-Gruppen	[0, 3]
A <sub>21</sub>	nCRX3: Anzahl von CRX3	[0, 3]
A <sub>22</sub>	SpPosA_B(p): Normalisierte spektrale positive Summe der Burden-Matrix gewichtet mit der Polarisierbarkeit	[0.863, 1.641]
A <sub>23</sub>	nCIR: Anzahl der Ringe	[0, 15]
A <sub>24</sub>	B01[C-Br]: Vorhandensein/Fehlen von C-Br im topologischen Abstand 1	{0, 1}
A <sub>25</sub>	B03[C-Cl]: Vorhandensein/Fehlen von C-Cl im topologischen Abstand 3	{0, 1}
A <sub>26</sub>	N-073: Ar <sub>2</sub> NH / Ar <sub>3</sub> N / Ar <sub>2</sub> N-Al / R..N..R	[0, 3]
A <sub>27</sub>	SpMax_A: Führender Eigenwert der Adjazenzmatrix (Lovasz-Pelkan-Index)	[1, 2.859]

A <sub>28</sub>	Psi_i_1d: Pseudokonnektivitätsindex des inneren Zustands – Typ 1d	[-1.099, 1.073]
A <sub>29</sub>	B04[C-Br]: Vorhandensein/Fehlen von C-Br im topologischen Abstand 4	{0,1}
A <sub>30</sub>	SdO: Summe der dO E-Zustände	[0, 67.469]
A <sub>31</sub>	T12_L: Zweiter Mohar-Index der Laplace-Matrix	[0.444, 17.537]
A <sub>32</sub>	nCrt: Anzahl der Ring-Tertiär-C(sp <sup>3</sup> )	[0, 8]
A <sub>33</sub>	C-026: R-CX-R	[0, 12]
A <sub>34</sub>	F02[C-N]: Frequenz der C-N-Bindung im topologischen Abstand 2	[0, 18]
A <sub>35</sub>	nHDon: Anzahl der Donoratome für H-Bindungen (N und O)	[0, 7]
A <sub>36</sub>	SpMax_B(m): Führender Eigenwert der Burden-Matrix gewichtet mit der Masse	[2.267, 10.695]
A <sub>37</sub>	Psi_i_A: Pseudokonnektivitätsindex des inneren Zustands – Durchschnitt des Typs S	[1.467, 5.825]
A <sub>38</sub>	nN: Anzahl der Stickstoffatome	[0, 6]
A <sub>39</sub>	SM6_B(m): Spektralmoment der Ordnung 6 der Burden-Matrix gewichtet mit der Masse	[4.917, 14.7]
A <sub>40</sub>	nArCOOR: Anzahl der aromatischen Ester	[0, 4]
A <sub>41</sub>	nX: Anzahl der Halogenatome	[0, 27]

Es wurde darauf geachtet, dass der Datensatz zufallsgeneratorbasiert derart gemischt wurde, dass 286 Beispiele für RB und 554 Beispiele für NRB in den Trainingsbeispielen vorhanden sind. Folglich befanden sich 71 Beispiele für RB und 139 Beispiele für NRB in den Testbeispielen. Für den QSAR Biodegradation-Datensatz wurden künstliche neuronale Netzwerke gemäß (3) mit zwei verborgenen Schichten mittels des n++-Simulatorkerns erstellt. Dementsprechend befanden sich in der Eingabeschicht 41 Eingabeneuronen, in den verborgenen Schichten jeweils 29 verborgene Neuronen und in der Ausgabeschicht 2 Ausgabeneuronen. Als Aktivierungsfunktionen der verborgenen Neuronen und der Ausgabeneuronen wurde die Sigmoidfunktion verwendet. Jedes Ausgabeneuron repräsentiert die Wahrscheinlichkeit der Eingabewerte biologisch abbaubar zu sein oder nicht.

## 4. Experimentelle Untersuchungen

### 4.5.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate

Dieses Experiment soll den Effekt der Lernrate von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des QSAR Biodegradation-Datensatzes analysieren. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 32 dargestellt.

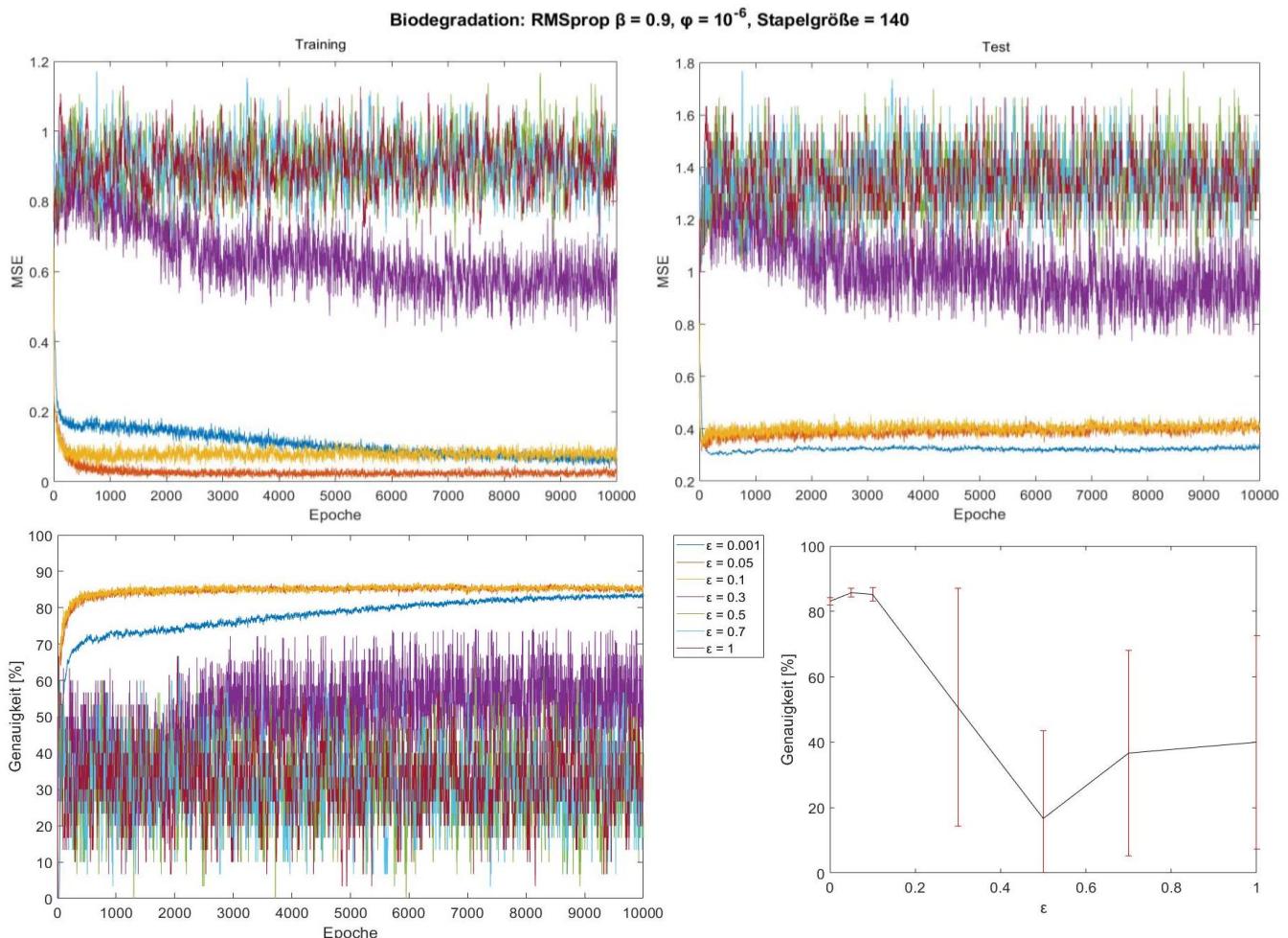


Abbildung 32: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Lernrate dargestellt.

In Abbildung 32 kann man sehen, dass mit Hilfe der Lernraten von 0.05 und 0.1 die höchste und nahezu gleiche Genauigkeit des künstlichen neuronalen Netzwerkes erreicht wird. Eine Lernrate von 0.001 erreicht zwar nicht dieselben Genauigkeitswerte innerhalb der 10000 Epochen, jedoch ist das Ergebnis der Lernrate 0.001 ebenfalls gut und könnte potenziell bei höheren Epochen bessere Genauigkeitswerte erreichen. Alle Lernraten ab dem Wert 0.3 scheinen eine zu große Schrittweite für den

Gradientenabstieg unter Verwendung des QSAR Biodegradation-Datensatzes darzustellen, sodass es zu starken Oszillationen kommt, die man in den Abbildungen erkennen kann.

#### 4.5.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors

Dieses Experiment soll den Effekt des Vergessensfaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des QSAR Biodegradation-Datensatzes ermitteln. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 33 dargestellt.

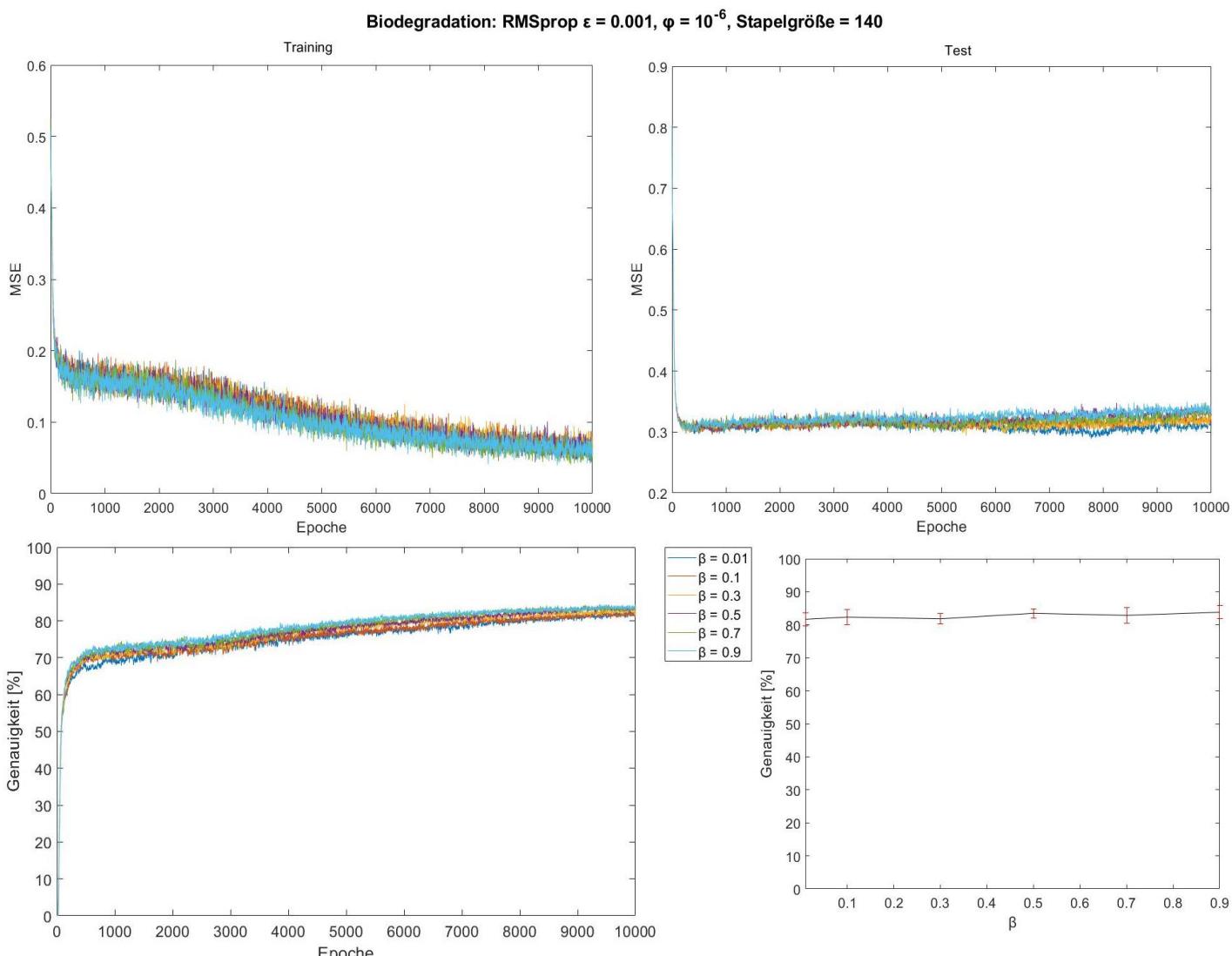


Abbildung 33: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genaugigkeit (unten links) des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors. Unten rechts ist die erreichte Genaugigkeit nach 10000 Epochen Training in Abhängigkeit des Vergessensfaktors dargestellt.

#### 4. Experimentelle Untersuchungen

In Abbildung 33 ist zu erkennen, dass mit steigendem Vergessensfaktor der Lernerfolg des künstlichen neuronalen Netzwerks, unter Verwendung des QSAR Biodegradation-Datensatzes, etwas steigt. Da bei einem kleinen Vergessensfaktor Gradienten vorheriger Mini-Stapel weniger Einfluss auf den gleitenden Durchschnitt haben, könnte bei der Epoche von etwa 300 der Gradientenabstiegsalgorithmus leicht andere Routen entlang des Hyperraums der Kostenfunktion in Richtung schlechterer lokalen Minima nehmen. Im Gegenteil dazu ist bei einem großen Vergessensfaktor der Einfluss der Gradienten vorheriger Mini-Stapel auf den gleitenden Durchschnitt größer, wodurch im Vergleich zu den kleineren Vergessensfaktoren günstigere Minima erreicht werden könnten. Allerdings könnte man auch untersuchen, wie sich die verschiedenen Vergessensfaktoren bei mehr als 10000 Epochen verhalten, denn es könnte auch durchaus sein, dass alle Verläufe zu einem Plateauwert konvergieren und damit zu dem gleichen oder einem ähnlichen lokalen Minimum führen.

##### **4.5.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors**

Dieses Experiment soll den Effekt des Unschärfeefaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des QSAR Biodegradation-Datensatzes untersuchen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 34 dargestellt.

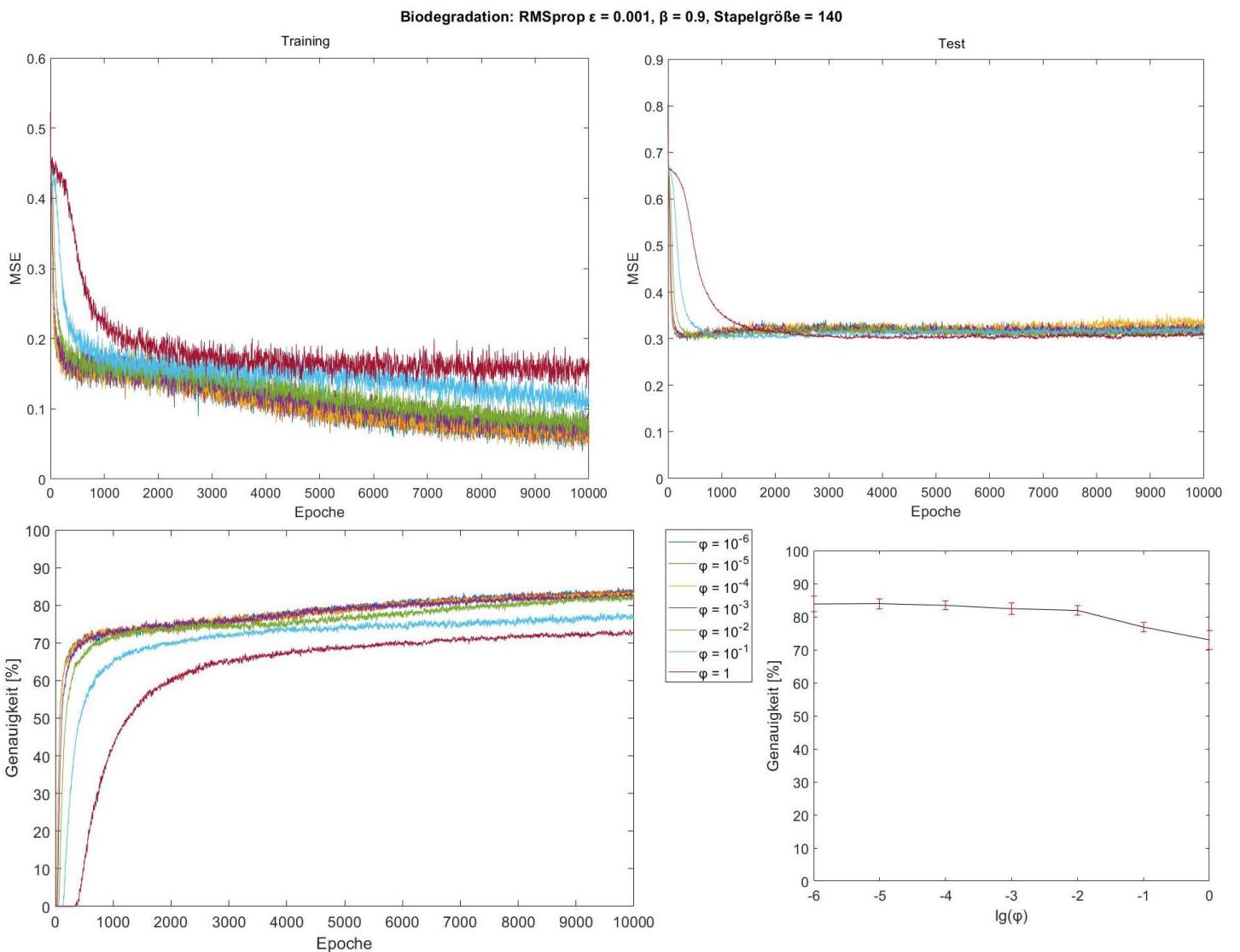


Abbildung 34: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des logarithmierten Unschärfeefaktors dargestellt.

Aus Abbildung 34 kann man anhand der Genauigkeit eindeutig erkennen, dass der Lernerfolg des künstlichen neuronalen Netzwerkes durch einen steigenden Unschärfeefaktor sinkt. Somit erzielt der Unschärfeefaktor von  $10^{-6}$  den besten Lernerfolg. Dies könnte daran liegen, dass durch den Unschärfeefaktor der gleitende Durchschnitt vorheriger Gradienten, durch die Summe der genannten Terme, derart verzerrt wird, dass der Gradientenabstieg mit steigendem Unschärfeefaktor erschwert oder verlangsamt wird.

## 4. Experimentelle Untersuchungen

### 4.5.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel

Dieses Experiment soll den Effekt der Größe der Mini-Stapel von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des QSAR Biodegradation-Datensatzes überprüfen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 35 dargestellt.

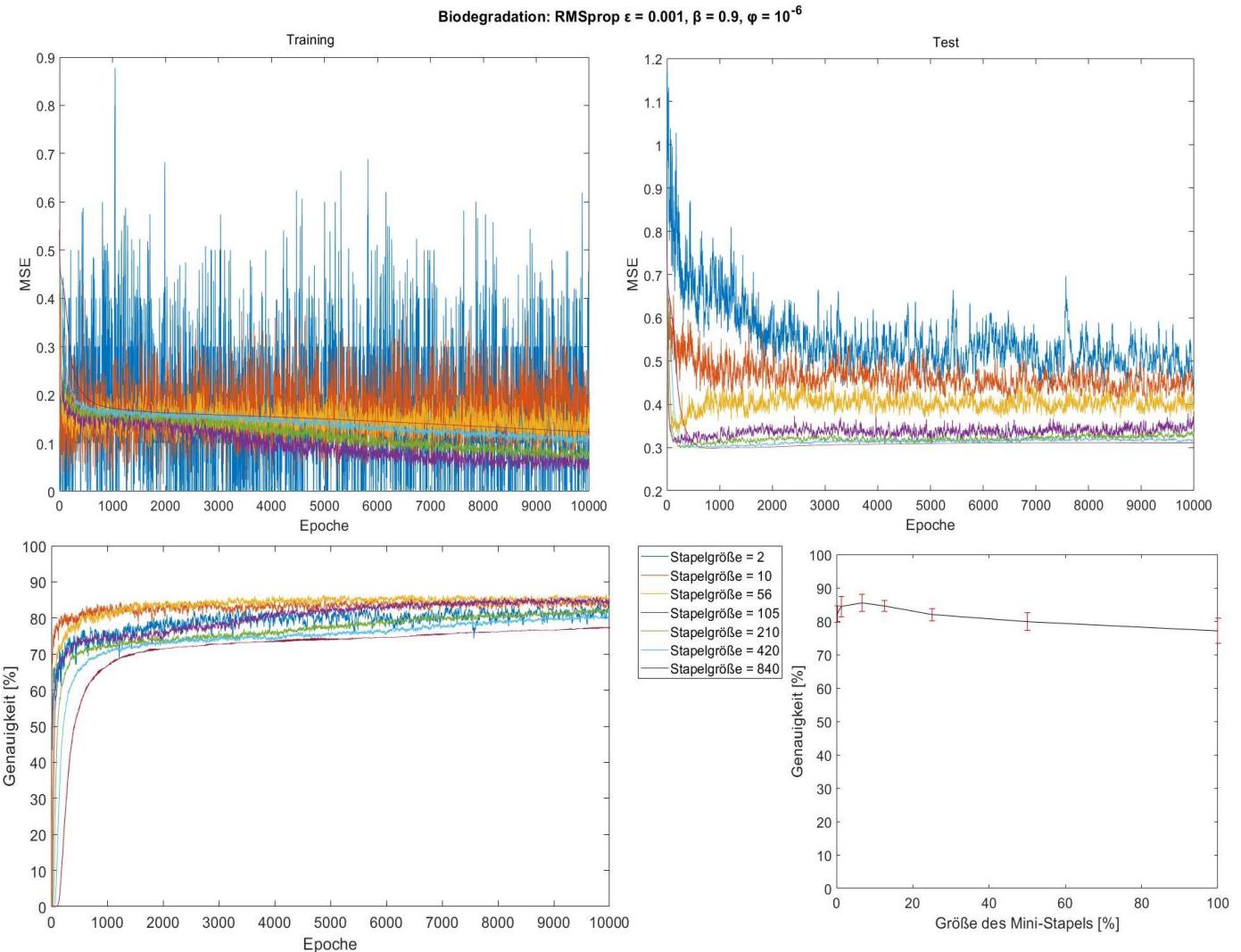


Abbildung 35: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Größe des Mini-Stapels im Verhältnis zu der gesamten Anzahl der Trainingsbeispiele dargestellt.

In Abbildung 35 ist zu erkennen, dass mittels einer Mini-Stapelgröße von 56 die größte Genauigkeit des künstlichen neuronalen Netzwerkes erreicht wird. Mit Hilfe einer Mini-Stapelgröße von 10 oder 105 lassen sich jedoch ähnlich gute Ergebnisse erzielen. Erwähnenswert ist hierbei, dass eine Mini-Stapelgröße von 1 einem stochastischen Gradientenabstieg von RMSprop gleichen würde, wohingegen

eine Mini-Stapelgröße von 840 einem Stapel-Gradientenabstieg von RMSprop gleichen würde. Aus der genannten Abbildung lässt sich demnach erkennen, dass bei der Mini-Stapelgröße von 840, aufgrund nur einer Gewichtsaktualisierung innerhalb einer Epoche, die niedrigsten Genauigkeitswerte erzielt werden. Die Mini-Stapelgröße von 2 oszilliert aufgrund der hohen Anzahl an Gewichtsaktualisierungen innerhalb einer Epoche am meisten, erreicht jedoch durchaus akzeptable Genauigkeitswerte.

#### 4.5.5. Vergleich der Gradientenabstiegsalgorithmen

Hier soll der Lernerfolg des künstlichen neuronalen Netzwerkes mittels des Gradientenabstiegsalgorithmus RMSprop mit dem Lernerfolg des künstlichen neuronalen Netzwerkes mittels der Gradientenabstiegsalgorithmen BP und Rprop verglichen werden. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 36 dargestellt.

## 4. Experimentelle Untersuchungen

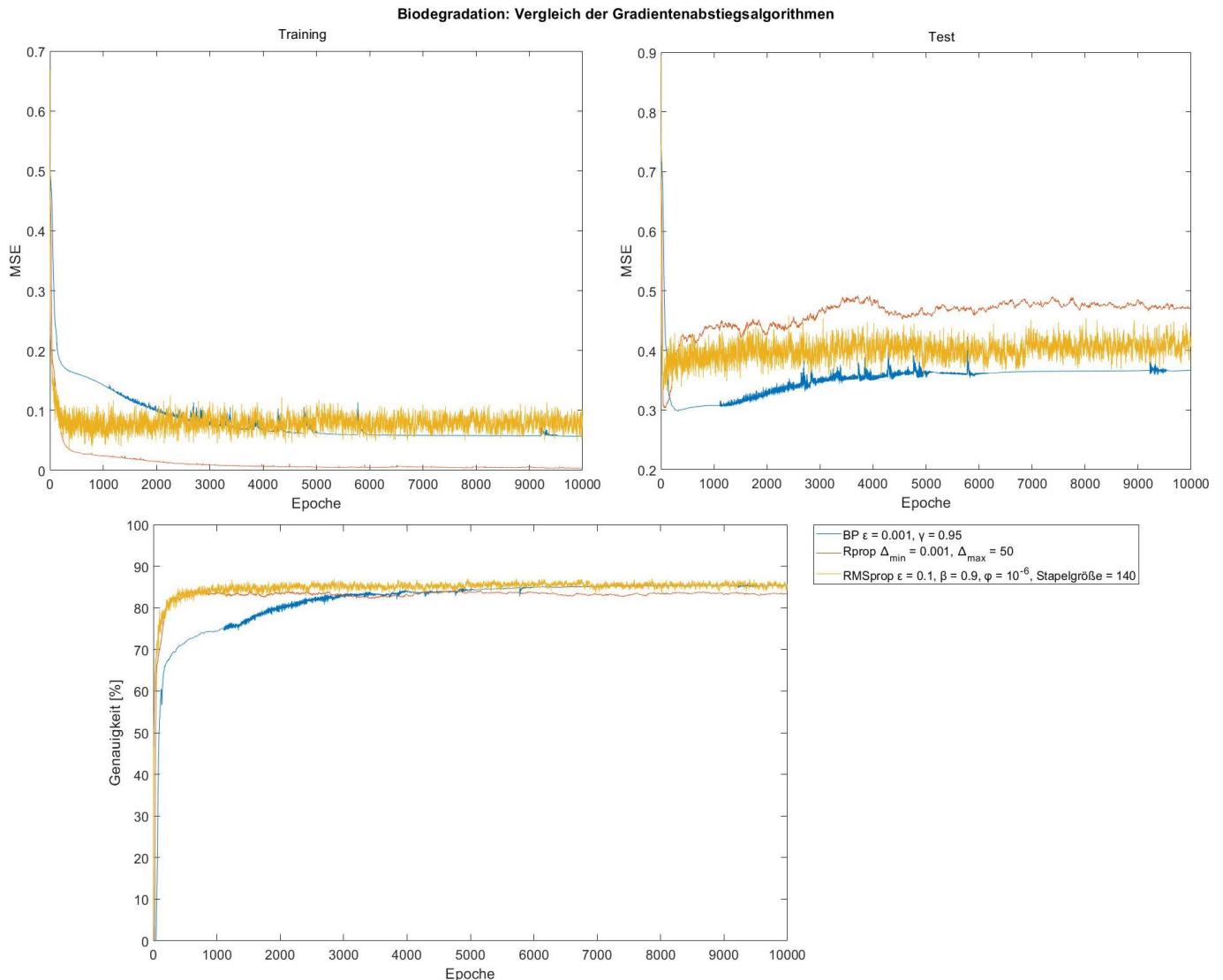


Abbildung 36: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten) des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop.

In Abbildung 36 ist zu beobachten, dass alle drei Gradientenabstiegsalgorithmen hohe Genauigkeitswerte des künstlichen neuronalen Netzwerkes erreichen. Bei Rprop jedoch kommt es, wie man anhand der sinkenden MSE-Werte der Trainingsbeispiele und der steigenden MSE-Werte der Testbeispiele sehen kann, zu einer Überanpassung an die Trainingsbeispiele. Dementsprechend sinkt ab Epoche 800 der Genauigkeitswert von Rprop. Mittels BP und RMSprop scheint es zu keiner oder nur einer schwachen Überanpassung zu kommen, da die Genauigkeitswerte bei beiden Gradientenabstiegsalgorithmen nicht zu sinken scheinen. In Abbildung 37 wird die erreichte Genauigkeit nach 10000 Epochen, die durchschnittliche Genauigkeit der letzten 1000 Epochen und die maximal erreichte Genauigkeit innerhalb der 10000 Epochen des jeweiligen Gradientenabstiegsalgorithmus übersichtlich dargestellt.

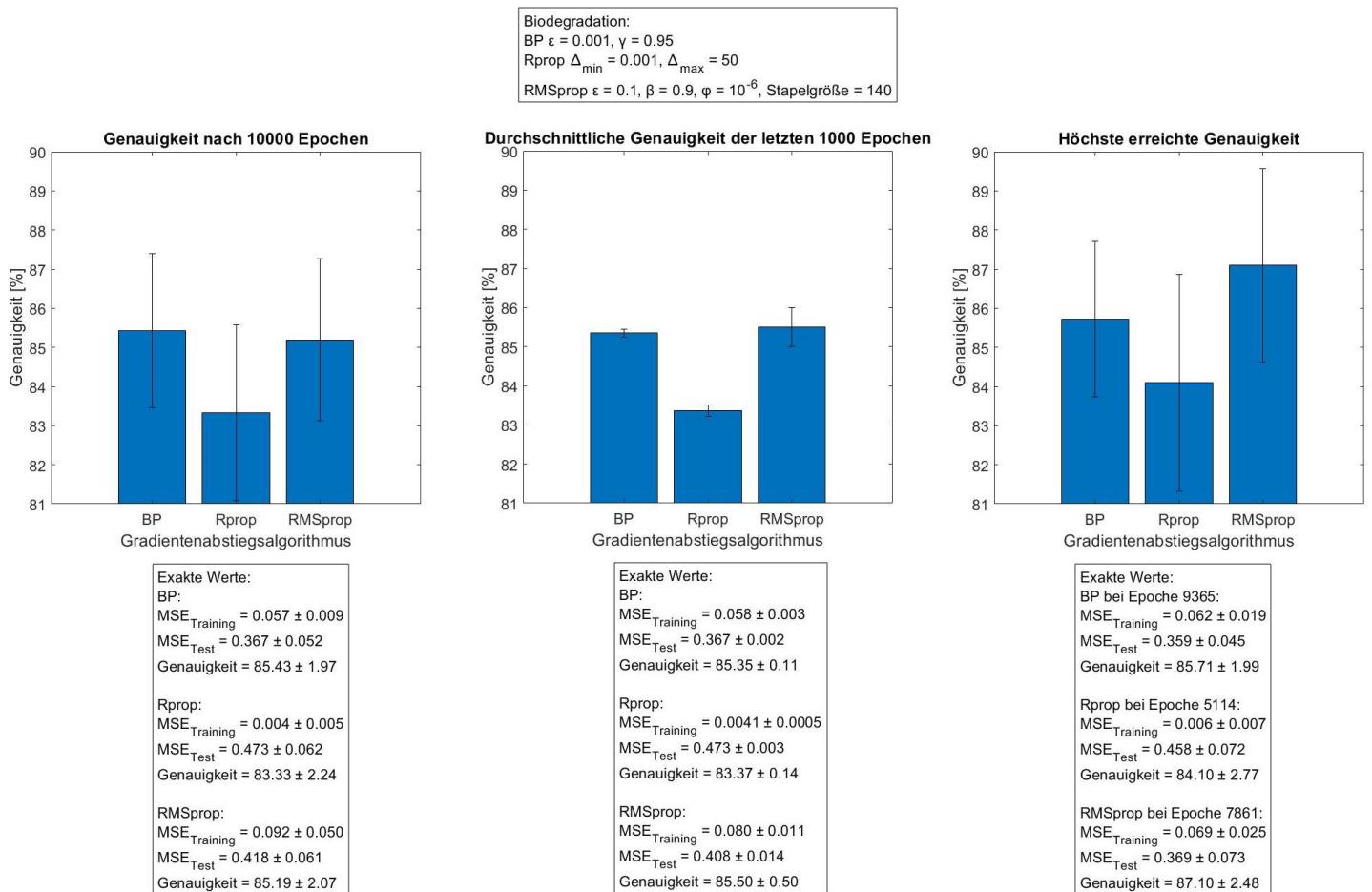


Abbildung 37: Übersicht der nach 10000 Epochen erreichten Genauigkeit (links), der durchschnittlichen Genauigkeit der letzten 1000 Epochen (mittig) und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen (rechts) des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes von BP, Rprop und RMSprop.

## 4.6. Car Evaluation

Bei dem Car Evaluation-Datensatz des UCI Machine Learning Repository handelt es sich um ein Klassifikationsproblem [27]. Der Datensatz besitzt insgesamt 1728 Beispiele, dem jedoch 3 Beispiele entnommen wurden, um eine Aufteilung in 80 % Trainingsbeispiele und 20 % Testbeispiele zu ermöglichen. Dementsprechend wurden 1725 Beispiele des Datensatzes verwendet, die auf 1380 Trainingsbeispiele und 345 Testbeispiele aufgeteilt wurden. In dem Datensatz existieren 4 mögliche Klassen, die ausgewertet werden können. Bei den möglichen Klassen handelt es sich um Akzeptanzwerte von Autos – unacc (unacceptable), acc (acceptable), good und vgood (very good). In den 1725 Beispielen des Datensatzes existierten insgesamt 1207 Beispiele der Klasse unacc, 384 Beispiele der Klasse acc, 69 Beispiele der Klasse good und 65 Beispiele der Klasse vgood. Jedes Beispiel besitzt insgesamt 6 Attribute, die als Eingabewerte für das künstliche neuronale Netzwerk zur Ermittlung des Akzeptanzwertes des Autos dienen. Diese sind Tabelle 7 zu entnehmen.

## 4. Experimentelle Untersuchungen

---

Tabelle 7: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Car Evaluation-Datensatzes.

Attribut	Information	Wertebereich
A <sub>1</sub>	Kaufpreis des Autos	{vhigh, high, med, low}
A <sub>2</sub>	Wartungspreis des Autos	{vhigh, high, med, low}
A <sub>3</sub>	Anzahl der Türen des Autos	{2, 3, 4, 5more}
A <sub>4</sub>	Personenkapazität des Autos	{2, 4, more}
A <sub>5</sub>	Größe des Gepäckraums des Autos	{small, medium, big}
A <sub>6</sub>	Geschätzte Sicherheit des Autos	{low, medium, high}

Wie man aus Tabelle 7 erkennen kann, handelt es sich bei den Attributen zum größten Teil um Merkmale, die als Eingabewerte verwendet werden sollen. Deshalb wurden die Merkmale durch Ganzzahlen ersetzt. Dazu wurde bei A<sub>1</sub> und A<sub>2</sub> low durch den Wert 1, med durch den Wert 2, high durch den Wert 3 und vhigh durch den Wert 4 ersetzt. Für A<sub>3</sub> und A<sub>4</sub> wurde 5more bzw. more durch den Wert 5 ersetzt und für A<sub>5</sub> sowie A<sub>6</sub> wurde small bzw. low durch den Wert 1, medium durch den Wert 2 und big bzw. high durch den Wert 3 ersetzt. Damit ergeben sich folgende Wertebereiche der Attribute, die in Tabelle 8 dargestellt sind.

Tabelle 8: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute nach Anpassung des Car Evaluation-Datensatzes.

Attribut	Information	Wertebereich
A <sub>1</sub>	Kaufpreis des Autos	{4, 3, 2, 1}
A <sub>2</sub>	Wartungspreis des Autos	{4, 3, 2, 1}
A <sub>3</sub>	Anzahl der Türen des Autos	{2, 3, 4, 5}
A <sub>4</sub>	Personenkapazität des Autos	{2, 4, 5}
A <sub>5</sub>	Größe des Gepäckraums des Autos	{1, 2, 3}
A <sub>6</sub>	Geschätzte Sicherheit des Autos	{1, 2, 3}

Es wurde darauf geachtet, dass der Datensatz zufallsgeneratorbasiert derart gemischt wurde, dass 966 Beispiele für unacc, 307 Beispiele für acc, 55 Beispiele für good und 52 Beispiele für vgood in den Trainingsbeispielen vorhanden sind. Folglich befanden sich 241 Beispiele für unacc, 77 Beispiele für acc, 14 Beispiele für good und 13 Beispiele für vgood in den Testbeispielen. Für den Car-Datensatz wurden künstliche neuronale Netzwerke gemäß (3) mit zwei verborgenen Schichten mittels des n++-

Simulatorkerns erstellt. Dementsprechend befanden sich in der Eingabeschicht 6 Eingabeneuronen, in den verborgenen Schichten jeweils 8 verborgene Neuronen und in der Ausgabeschicht 4 Ausgabeneuronen. Als Aktivierungsfunktionen der verborgenen Neuronen und der Ausgabeneuronen wurde die Sigmoidfunktion verwendet. Jedes Ausgabeneuron repräsentiert die Wahrscheinlichkeit der Eingabewerte zu einem jeweiligen Akzeptanzwert der Autos zu gehören.

#### **4.6.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate**

Dieses Experiment soll den Effekt der Lernrate von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Car Evaluation-Datensatzes untersuchen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 38 dargestellt.

## 4. Experimentelle Untersuchungen

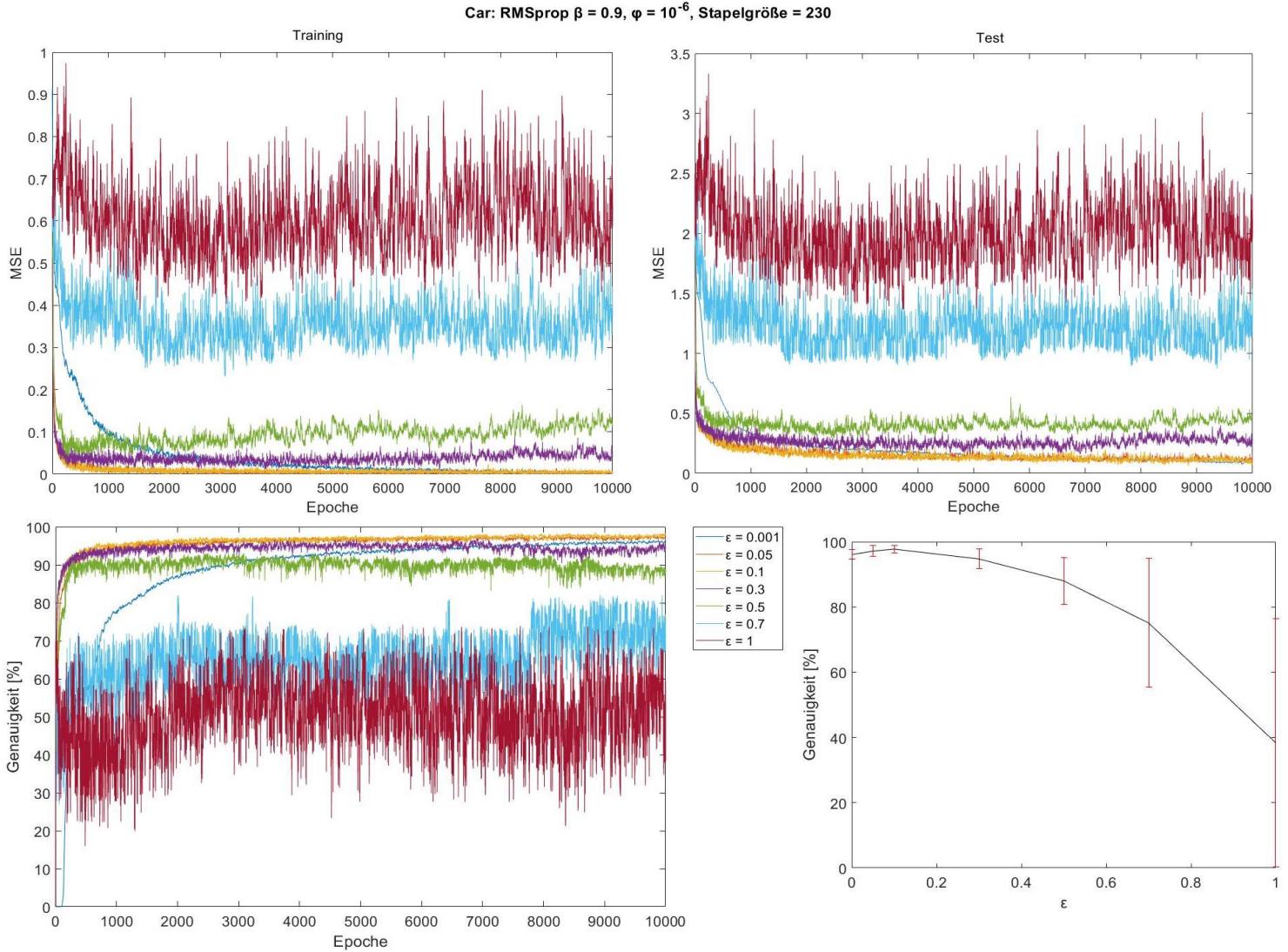


Abbildung 38: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Lernrate dargestellt.

Aus Abbildung 38 folgt, dass mittels einer Lernrate von 0.05 und 0.1 der größte Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Car Evaluation-Datensatzes erzielt werden kann. Mit Hilfe einer Lernrate von 0.001 wäre dies auch möglich, da der Verlauf der Genauigkeit von 0.001 noch kein Plateauwert erreicht hat innerhalb der 10000 Epochen. Ab einer Lernrate von 0.3 scheint die Lernrate des Gradientenabstiegs im Mittel zu schlechteren Genauigkeitswerten und höheren MSE-Werten zu führen. Außerdem verstärkt sich mit steigender Lernrate die Oszillationen der Verläufe.

## 4.6.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors

Dieses Experiment soll den Effekt des Vergessensfaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Car Evaluation-Datensatzes ermitteln. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 39 dargestellt.

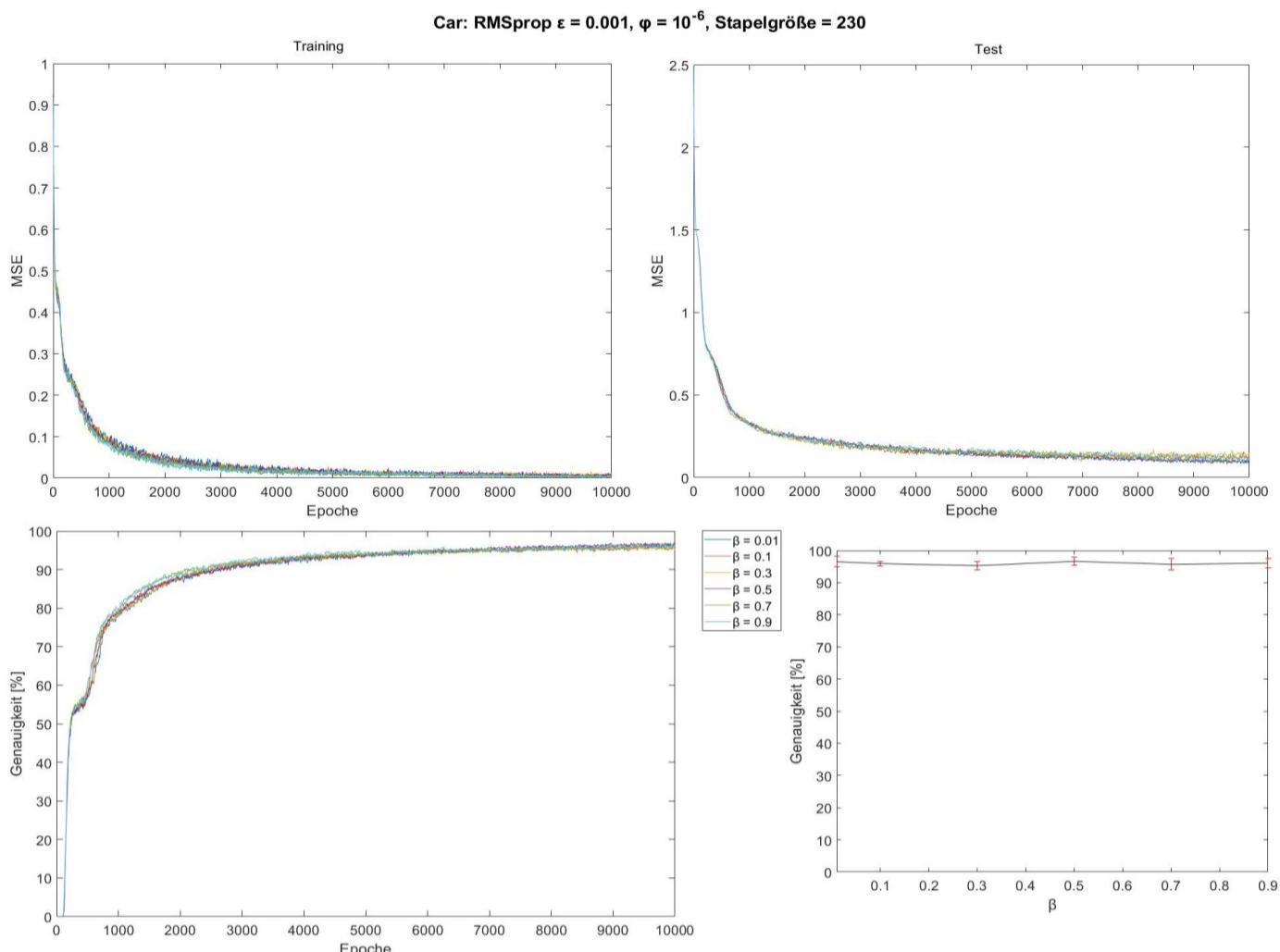


Abbildung 39: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des Vergessensfaktors dargestellt.

Abbildung 39 zeigt, dass der Vergessensfaktor nur einen geringen Effekt auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Car Evaluation-Datensatzes aufweist. Alle MSE-Werte und Genauigkeitswerte konvergieren nahezu auf dieselben Plateauwerte. Interessant ist

## 4. Experimentelle Untersuchungen

hierbei der Verlauf des Vergessensfaktor von 0.7 und 0.9, da diese bei etwa einer Epoche von 800 im Vergleich zu den anderen Vergessensfaktoren höhere Genauigkeitswerte erreichen.

### 4.6.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors

Dieses Experiment soll den Effekt des Unschärfeefaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Car Evaluation-Datensatzes erforschen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 40 dargestellt.

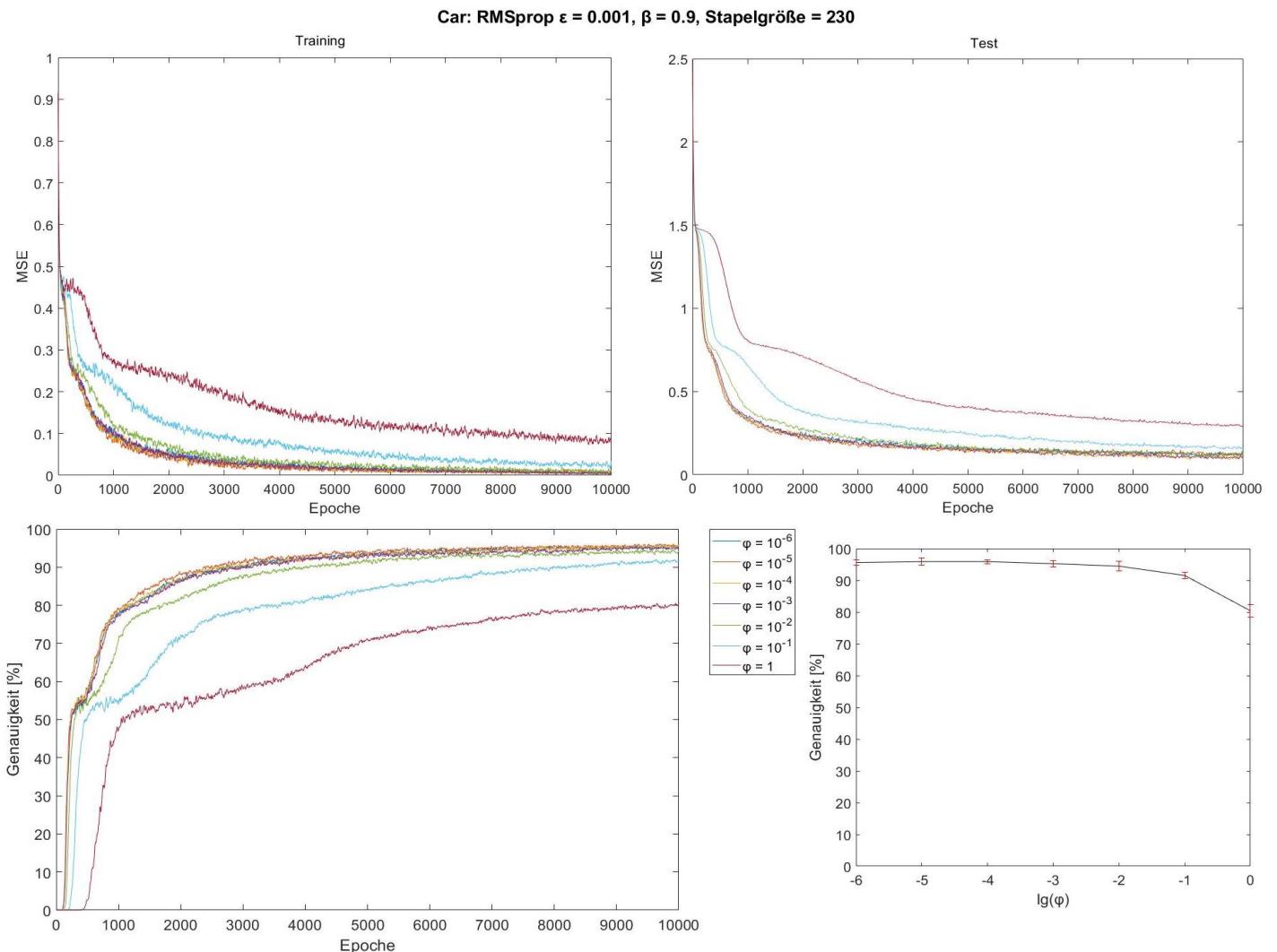


Abbildung 40: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des logarithmierten Unschärfeefaktors dargestellt.

Aus Abbildung 40 ist festzustellen, dass der Gradientenabstieg mit steigendem Unschärfe faktor zunehmend erschwert wird. Dadurch werden im Mittel höhere MSE-Werte und kleinere Genauigkeitswerte in Folge des Gradientenabstiegs erreicht. Dementsprechend erreicht man mit den kleinsten Unschärfe faktoren  $10^{-6}$  bis  $10^{-3}$  den größten Lernerfolg des künstlichen neuronalen Netzwerkes. Ab einem Unschärfe faktor von  $10^{-2}$  beginnt der Unschärfe faktor den Gradientenabstieg negativ zu beeinflussen.

#### **4.6.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel**

Dieses Experiment soll den Effekt der Größe der Mini-Stapel von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Car Evaluation-Datensatzes überprüfen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 41 dargestellt.

## 4. Experimentelle Untersuchungen

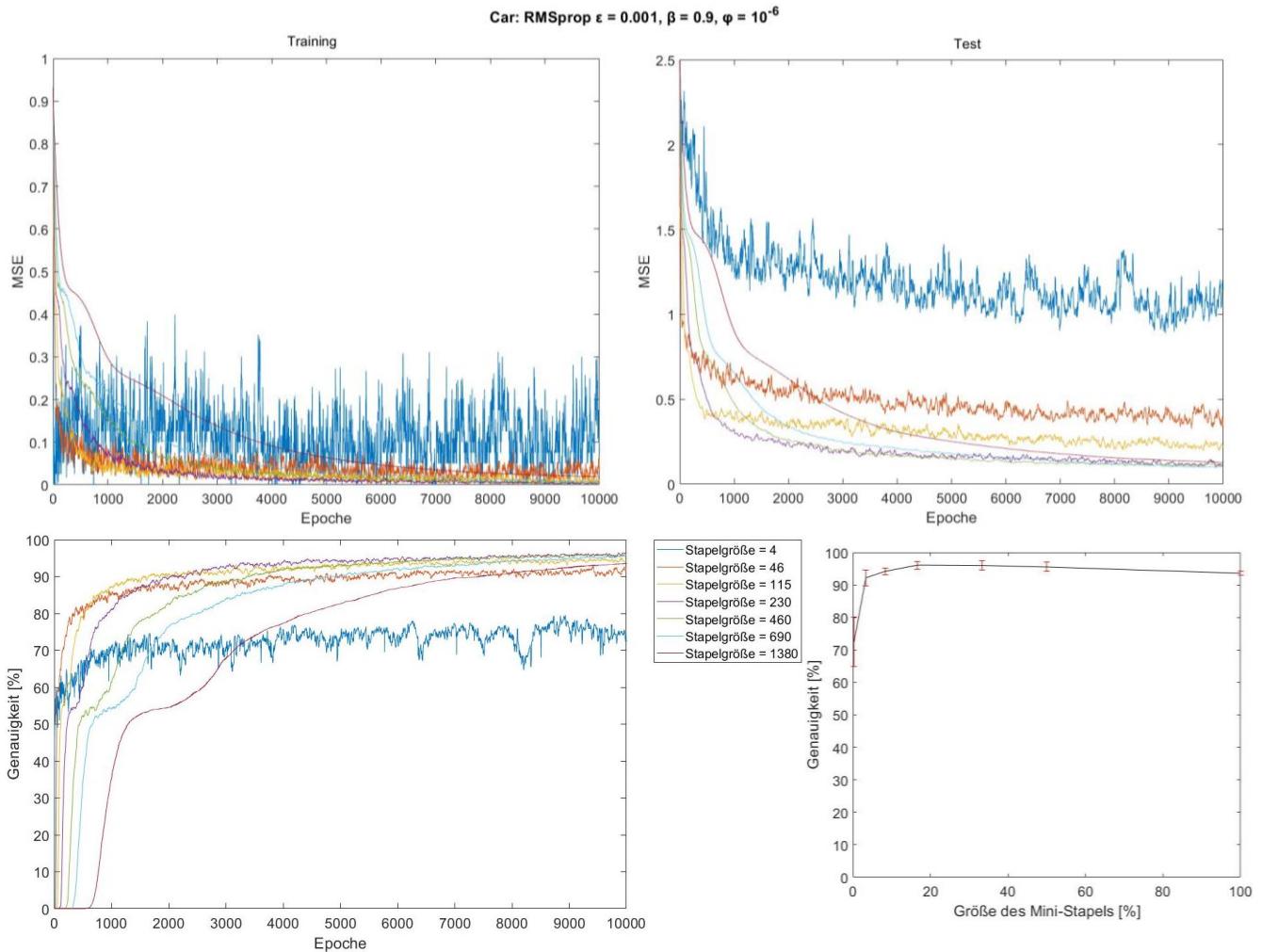


Abbildung 41: Verlauf des MSE der Trainings- (oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Größe des Mini-Stapels im Verhältnis zu der gesamten Anzahl der Trainingsbeispiele dargestellt.

In Abbildung 41 kann man sehen, dass mittels einer Mini-Stapelgröße von 230, 460 und 690 der größte Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Car Evaluation-Datensatzes erreicht werden kann. Erwähnenswert ist hierbei, dass eine Mini-Stapelgröße von 1 einem stochastischen Gradientenabstieg von RMSprop gleichen würde, wohingegen eine Mini-Stapelgröße von 1380 einem Stapel-Gradientenabstieg von RMSprop gleichen würde. Aus der genannten Abbildung lässt sich demnach erkennen, dass bei der Mini-Stapelgröße von 1380, aufgrund nur einer Gewichtsaktualisierung innerhalb einer Epoche, die niedrigsten Genauigkeitswerte erzielt werden. Der Verlauf der Mini-Stapelgröße von 1380 erreicht jedoch innerhalb der 10000 Epochen noch keinen Plateauwert also sind durchaus bessere Ergebnisse denkbar, wenn man das künstliche neuronale Netzwerk über die 10000 Epochen hinaus trainiert. Die Mini-Stapelgröße von 4 oszilliert aufgrund der hohen Anzahl an Gewichtsaktualisierungen innerhalb einer Epoche am meisten und erreicht deshalb die schlechtesten Genauigkeitswerte aller Mini-Stapelgrößen.

#### 4.6.5. Vergleich der Gradientenabstiegsalgorithmen

Hier soll der Lernerfolg des künstlichen neuronalen Netzwerkes mittels des Gradientenabstiegsalgorithmus RMSprop mit dem Lernerfolg des künstlichen neuronalen Netzwerkes mittels der Gradientenabstiegsalgorithmen BP und Rprop verglichen werden. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 42 dargestellt.

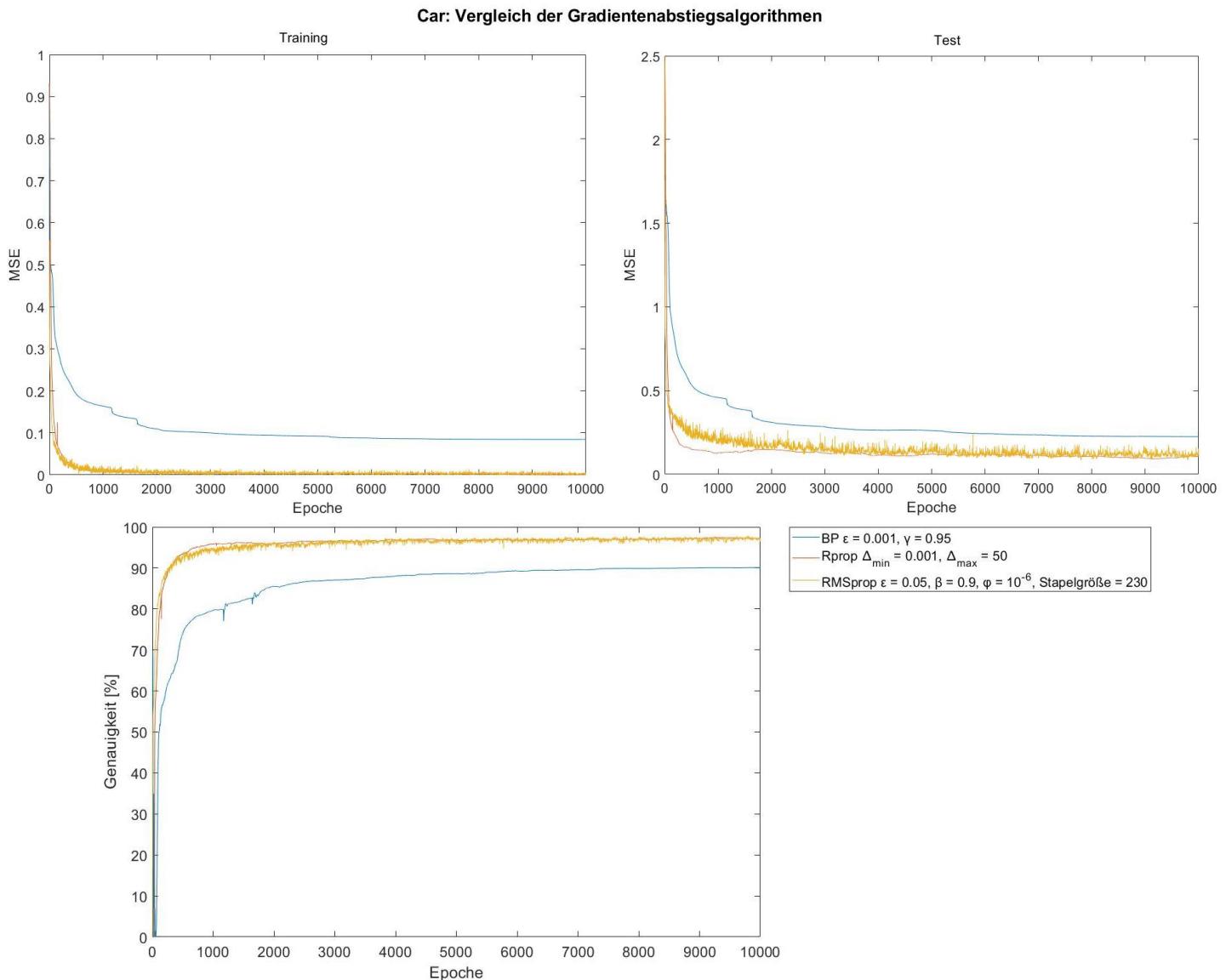


Abbildung 42: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten) des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop.

#### 4. Experimentelle Untersuchungen

In Abbildung 42 kann man erkennen, dass das Gradientenabstiegsverfahren mittels BP die schlechtesten Ergebnisse erzielt. BP erreicht einen Plateauwert bei den höchsten MSE-Werten und den niedrigsten Genauigkeitswerten, was darauf hindeutet, dass BP stets ein lokales Minimum erreicht, das im Vergleich zu Rprop und RMSprop schlechter ist. Rprop und RMSprop erreichen beide einen nahezu selben Plateauwert hinsichtlich der Genauigkeit des künstlichen neuronalen Netzwerkes. In Abbildung 43 wird die erreichte Genauigkeit nach 10000 Epochen, die durchschnittliche Genauigkeit der letzten 1000 Epochen und die maximal erreichte Genauigkeit innerhalb der 10000 Epochen des jeweiligen Gradientenabstiegsalgorithmus übersichtlich dargestellt.

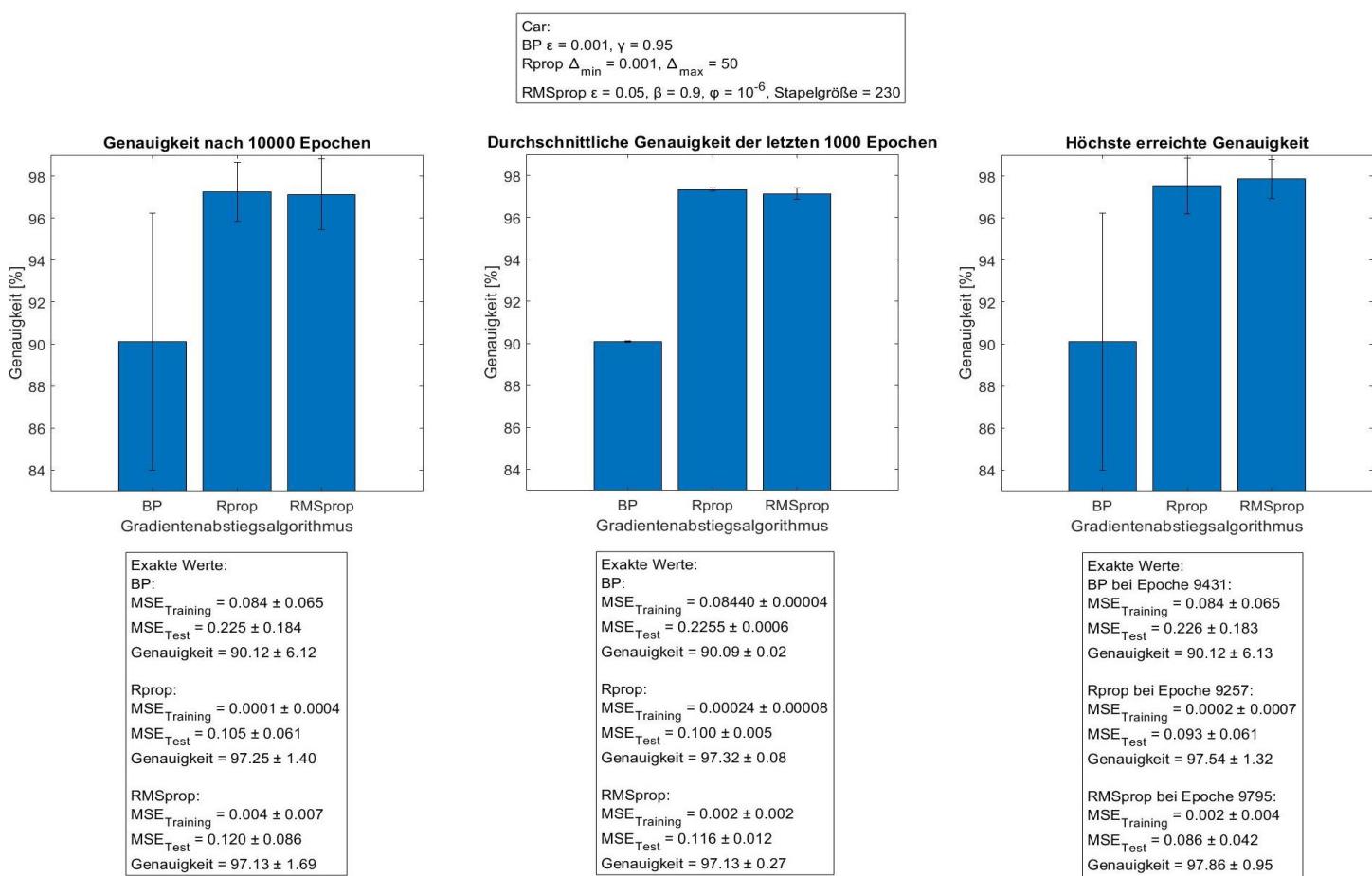


Abbildung 43: Übersicht der nach 10000 Epochen erreichten Genauigkeit (links), der durchschnittlichen Genauigkeit der letzten 1000 Epochen (mittig) und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen (rechts) des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes von BP, Rprop und RMSprop.

## 4.7. Wine Quality Red

Bei dem Wine Quality Red-Datensatz des UCI Machine Learning Repository handelt es sich um ein Klassifikationsproblem [28]. Der Datensatz besitzt insgesamt 1599 Beispiele, dem jedoch 4 Beispiele entnommen wurden, um eine Aufteilung in 80 % Trainingsbeispiele und 20 % Testbeispiele zu ermöglichen. Dementsprechend wurden 1595 Beispiele des Datensatzes verwendet, die auf 1276 Trainingsbeispiele und 319 Testbeispiele aufgeteilt wurden. In dem Datensatz existieren theoretisch 6 mögliche Klassen, die ausgewertet werden können, jedoch wurden 11 mögliche Klassen verwendet. Dementsprechend handelt es sich bei den möglichen Klassen um Qualitätsbewertungen von Rotwein – 0 (schlechteste Qualität) bis 10 (beste Qualität). In den 1595 Beispielen des Datensatzes existieren insgesamt 10 Beispiele für die Qualität 3, 53 Beispiele für die Qualität 4, 680 Beispiele für die Qualität 5, 635 Beispiele für die Qualität 6, 199 Beispiele für die Qualität 7 und 18 Beispiele für die Qualität 8. Für die restlichen Qualitäten existieren keine Beispiele in diesem Datensatz. Jedes Beispiel besitzt insgesamt 11 Attribute, die als Eingabewerte für das künstliche neuronale Netzwerk zur Ermittlung der Qualität des Rotweins dienen. Diese sind in Tabelle 9 zu erkennen.

Tabelle 9: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Wine Quality Red-Datensatzes.

Attribut	Information	Wertebereich
A <sub>1</sub>	Fester Säuregehalt in g(Weinsäure)/dm <sup>3</sup>	[4.6, 15.9]
A <sub>2</sub>	Flüchtiger Säuregehalt in g(Essigsäure)/dm <sup>3</sup>	[0.12, 1.58]
A <sub>3</sub>	Zitronensäure in g/dm <sup>3</sup>	[0, 1]
A <sub>4</sub>	Restzucker in g/dm <sup>3</sup>	[0.9, 15.5]
A <sub>5</sub>	Chloride in g(Natriumchlorid)/dm <sup>3</sup>	[0.012, 0.611]
A <sub>6</sub>	Freies Schwefeldioxid in mg/dm <sup>3</sup>	[1, 72]
A <sub>7</sub>	Gesamtes Schwefeldioxid in mg/dm <sup>3</sup>	[6, 289]
A <sub>8</sub>	Dichte in g/cm <sup>3</sup>	[0.99007, 1.00369]
A <sub>9</sub>	pH-Wert	[2.74, 4.01]
A <sub>10</sub>	Sulfate in g(Kaliumsulfat)/dm <sup>3</sup>	[0.33, 2]
A <sub>11</sub>	Alkoholgehalt in vol.%	[8.4, 14.9]

#### 4. Experimentelle Untersuchungen

Bei diesem Datensatz wurde nicht darauf geachtet, dass die Verhältnisse der einzelnen Klassen in den Trainingsbeispielen sowie in den Testbeispielen aufrechterhalten werden. Demnach könnte es vorkommen, dass beispielsweise in einer Mischung des Datensatzes in den Trainingsbeispielen oder Testbeispielen keine Beispiele für die Qualität 8 vertreten waren. Für den Wine Quality Red-Datensatz wurden künstliche neuronale Netzwerke gemäß (3) mit zwei verborgenen Schichten mittels des n++-Simulatorkerns erstellt. Dementsprechend befanden sich in der Eingabeschicht 11 Eingabeneuronen, in den verborgenen Schichten jeweils 18 verborgene Neuronen und in der Ausgabeschicht 11 Ausgabeneuronen. Als Aktivierungsfunktionen der verborgenen Neuronen und der Ausgabeneuronen wurde die Sigmoidfunktion verwendet. Jedes Ausgabeneuron repräsentiert die Wahrscheinlichkeit der Eingabewerte zu einer jeweiligen Qualität des Rotweins zu gehören.

##### **4.7.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate**

Dieses Experiment soll den Effekt der Lernrate von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Wine Quality Red-Datensatzes untersuchen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 44 dargestellt.

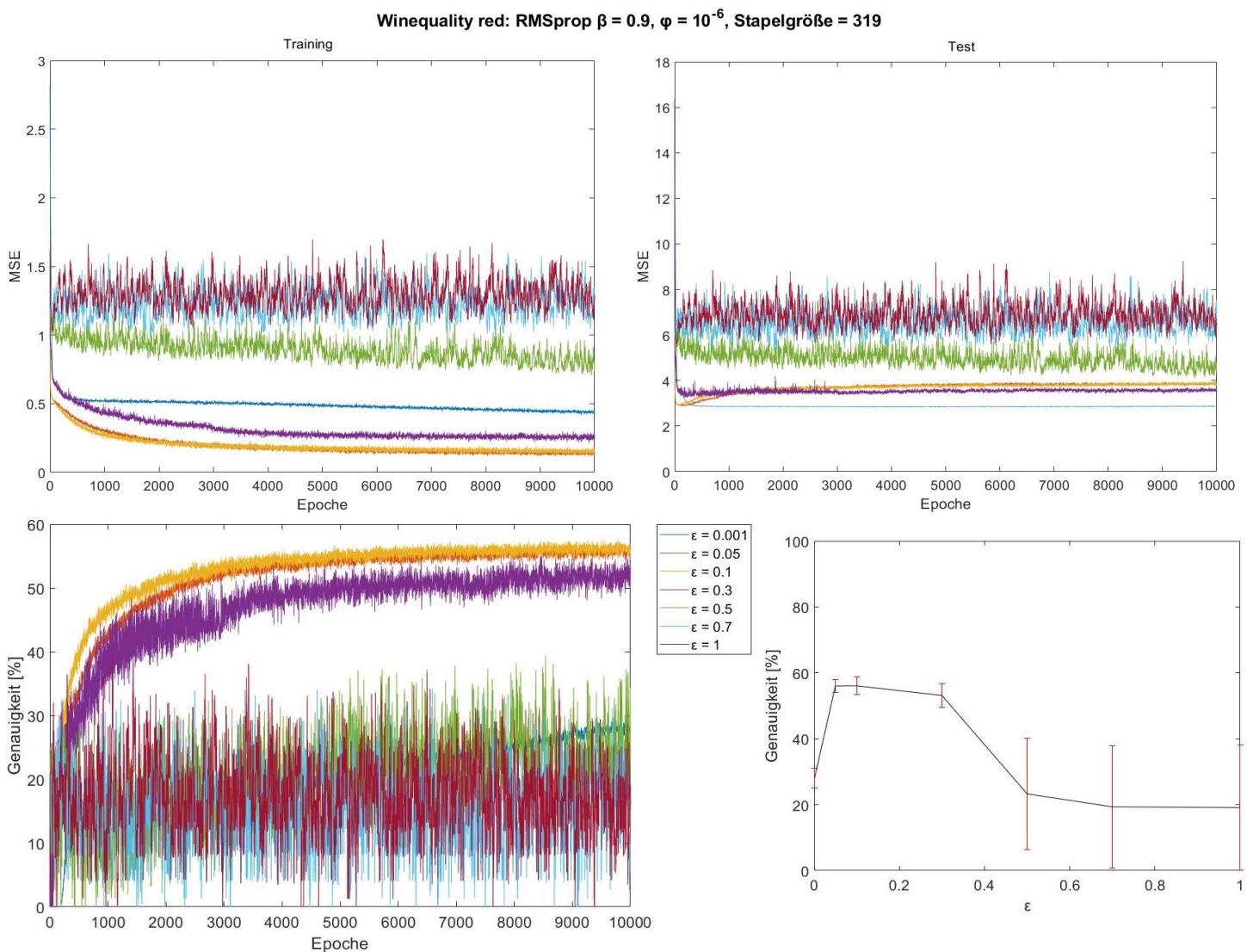


Abbildung 44: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Lernrate dargestellt.

In Abbildung 44 ist zu erkennen, dass mit Hilfe der Lernraten von 0.05 und 0.1 der höchste Lernerfolg des künstlichen neuronalen Netzwerkes erzielt werden kann. Dieser beträgt jedoch etwa nur 55 % unter der Verwendung des Wine Quality Red-Datensatzes. Unter Rücksichtnahme der MSE-Werte der Testbeispiele kann man erkennen, dass das künstliche neuronale Netzwerk ungefähr ab Epoche 150 sich bei den Lernraten 0.05 und 0.1 gut an die Trainingsbeispiele anpasst, wodurch es zu einem Anstieg der MSE-Werte der Testbeispiele kommt. Gleichzeitig ist bei den Verläufen der Lernrate 0.05 und 0.1 kein Verlust der Genauigkeitswerte festzustellen. Dies könnte darauf deuten, dass sich das künstliche neuronale Netzwerk sehr gut an die Beispiele anpasst, die es eindeutig einer Qualität von Rotwein zuordnen kann. Dementsprechend müsste dem Trainingsdatensatz Beispiele der Qualitäten, die das Netzwerk nicht korrekt zuordnen kann, beigefügt werden, um die maximal erreichten Genauigkeitswerte zu erhöhen. Der Gradientenabstieg mittels der Lernrate 0.001 erreicht zwar die

#### 4. Experimentelle Untersuchungen

niedrigsten MSE-Werte der Testbeispiele, jedoch mitunter die niedrigsten Genauigkeitswerte. Aus dem Verlauf des MSE-Wertes der Trainingsbeispiele folgt, dass dies auf eine unzureichende Anpassung an die Trainingsbeispiele zurückzuführen ist. Ab einer Lernrate von 0.3 ist die Schrittweite des Gradientenabstiegs zu groß, sodass es zu zunehmenden Oszillationen der Verläufe kommt.

##### **4.7.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors**

Dieses Experiment soll den Effekt des Vergessensfaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Wine Quality Red-Datensatzes überprüfen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 45 dargestellt.

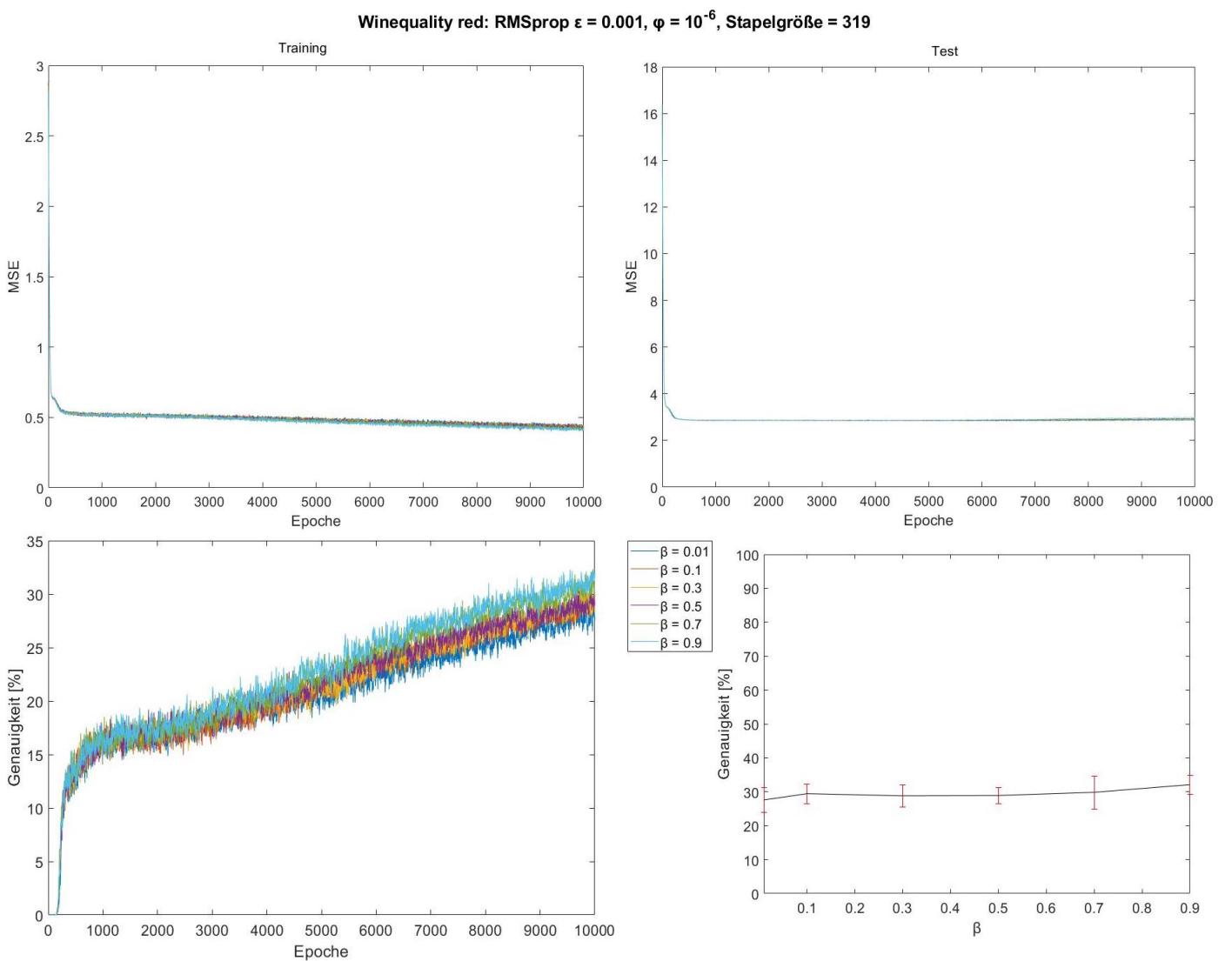


Abbildung 45: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des Vergessensfaktors dargestellt.

In Abbildung 45 kann man eindeutig erkennen, dass der Lernerfolg des künstlichen neuronalen Netzwerkes mit steigendem Vergessensfaktor, unter Verwendung des Wine Quality Red-Datensatzes, steigt. Da bei einem kleinen Vergessensfaktor Gradienten vorheriger Mini-Stapel weniger Einfluss auf den gleitenden Durchschnitt haben, könnte bei der Epoche von etwa 1000 der Gradientenabstiegsalgorithmus leicht andere Routen entlang des Hyperraums der Kostenfunktion in Richtung schlechterer lokaler Minima nehmen. Im Gegenteil dazu, ist bei einem großen Vergessensfaktor der Einfluss der Gradienten vorheriger Mini-Stapel auf den gleitenden Durchschnitt größer, wodurch im Vergleich zu den kleineren Vergessensfaktoren günstigere Minima erreicht werden könnten. Allerdings könnte man auch untersuchen, wie sich die verschiedenen Vergessensfaktoren bei mehr als 10000 Epochen verhalten, denn es könnte auch durchaus sein, dass

## 4. Experimentelle Untersuchungen

alle Verläufe zu dem gleichen Plateauwert konvergieren und damit zu dem gleichen oder ähnlichen lokalen Minimum führen.

### 4.7.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors

Dieses Experiment soll den Effekt des Unschärfeefaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Wine Quality Red-Datensatzes ermitteln. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 46 dargestellt.

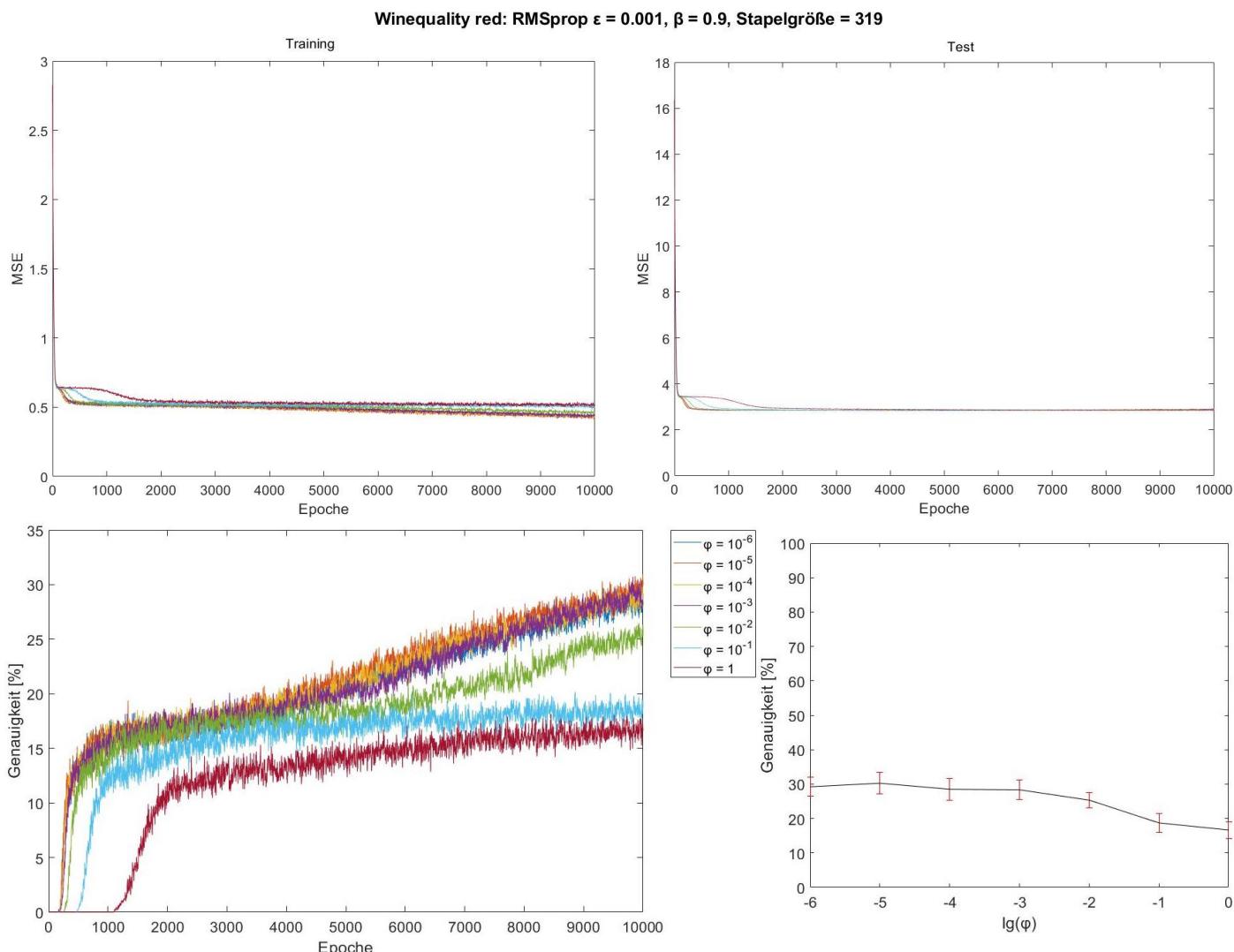


Abbildung 46: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des logarithmierten Unschärfeefaktors dargestellt.

In Abbildung 46 kann man erkennen, dass der Gradientenabstieg mittels RMSprop ab einem Unschärfeefaktor von  $10^{-2}$  durch den Unschärfeefaktor negativ beeinflusst wird, wodurch im Mittel niedrigere Genauigkeitswerte erreicht werden. Da die MSE-Werte der Testbeispiele kaum unterschiedlich voneinander sind und Unterschiede in den MSE-Werten der Trainingsbeispiele auftreten, kann davon ausgegangen werden, dass sich das künstliche neuronale Netzwerk durch einen steigenden Unschärfeefaktor schlechter auf die Trainingsbeispiele anpassen kann, das wiederum zu den schlechten Genauigkeitswerten führt.

#### **4.7.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel**

Dieses Experiment soll den Effekt der Größe der Mini-Stapel von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Wine Quality Red-Datensatzes analysieren. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 47 dargestellt.

## 4. Experimentelle Untersuchungen

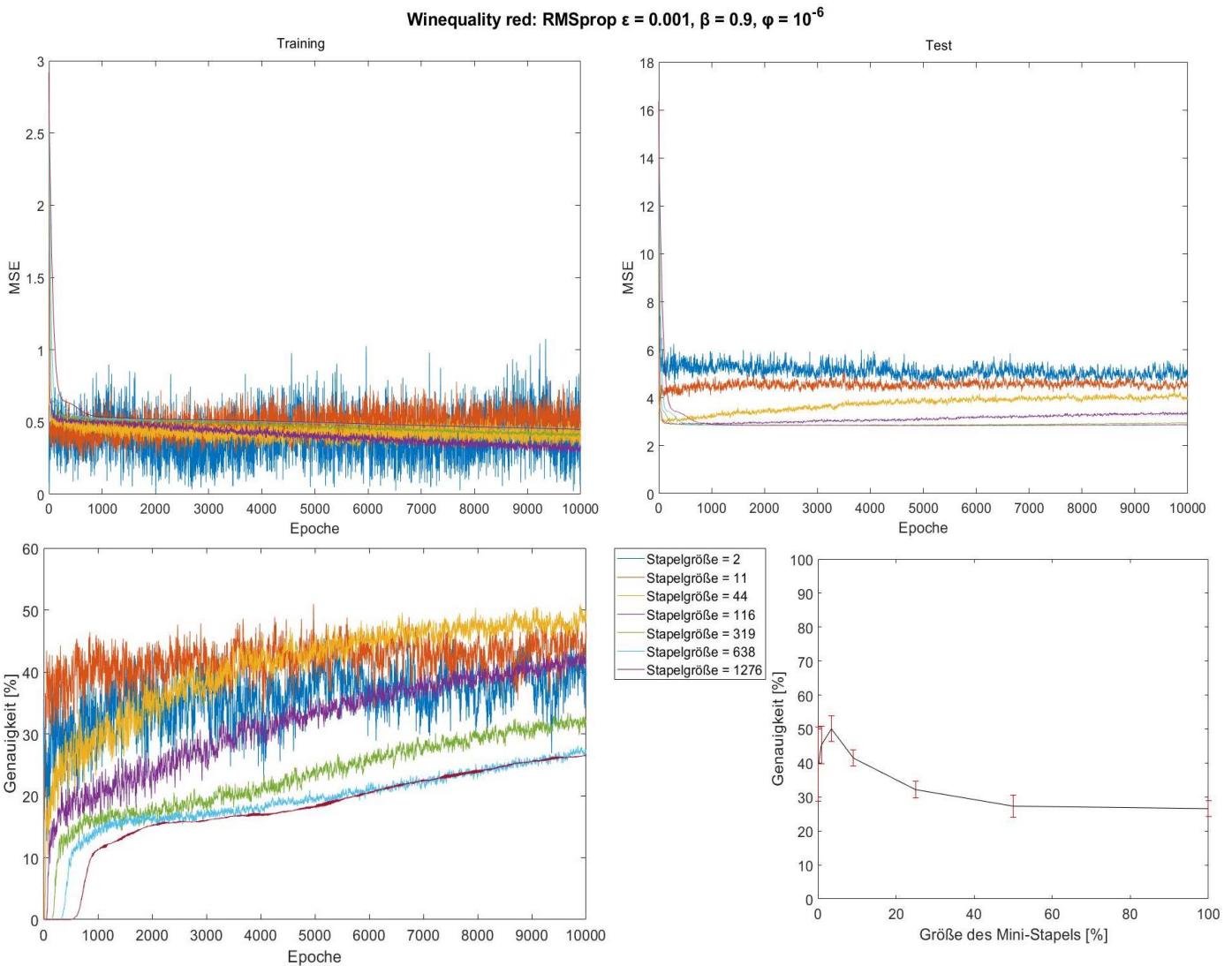


Abbildung 47: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Größe des Mini-Stapels im Verhältnis zu der gesamten Anzahl der Trainingsbeispiele dargestellt.

Aus Abbildung 47 kann man entnehmen, dass mittels eines Gradientenabstiegs mit einer Mini-Stapelgröße von 44 der größte Lernerfolg des künstlichen neuronalen Netzwerkes erzielt werden kann. Bei dem Verlauf der Mini-Stapelgröße von 11 tritt sogar bei einer Epoche von etwa 4900 eine Spitze der Genauigkeitswerte auf, die ein Maximum der erreichten Genauigkeitswerte darstellt. Erwähnenswert ist hierbei, dass eine Mini-Stapelgröße von 1 einem stochastischen Gradientenabstieg von RMSprop gleichen würde, wohingegen eine Mini-Stapelgröße von 1276 einem Stapel-Gradientenabstieg von RMSprop gleichen würde. Aus den genannten Abbildungen lässt sich demnach erkennen, dass ab der Mini-Stapelgröße von 319, aufgrund nur weniger Gewichtsaktualisierungen innerhalb einer Epoche, die niedrigsten Genauigkeitswerte erzielt werden. Die Verläufe der Mini-Stapelgrößen ab 319 erreichen jedoch innerhalb der 10000 Epochen noch keinen Plateauwert also sind

somit bessere Ergebnisse denkbar, wenn man das künstliche neuronale Netzwerk über die 10000 Epochen hinaus trainiert. Die Mini-Stapelgröße von 2 oszilliert aufgrund der hohen Anzahl an Gewichtsaktualisierungen innerhalb einer Epoche am meisten, erreicht jedoch im Vergleich akzeptable Genauigkeitswerte.

#### **4.7.5. Vergleich der Gradientenabstiegsalgorithmen**

Hier soll der Lernerfolg des künstlichen neuronalen Netzwerkes mittels des Gradientenabstiegsalgorithmus RMSprop mit dem Lernerfolg des künstlichen neuronalen Netzwerkes mittels der Gradientenabstiegsalgorithmen BP und Rprop verglichen werden. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 48 dargestellt.

## 4. Experimentelle Untersuchungen

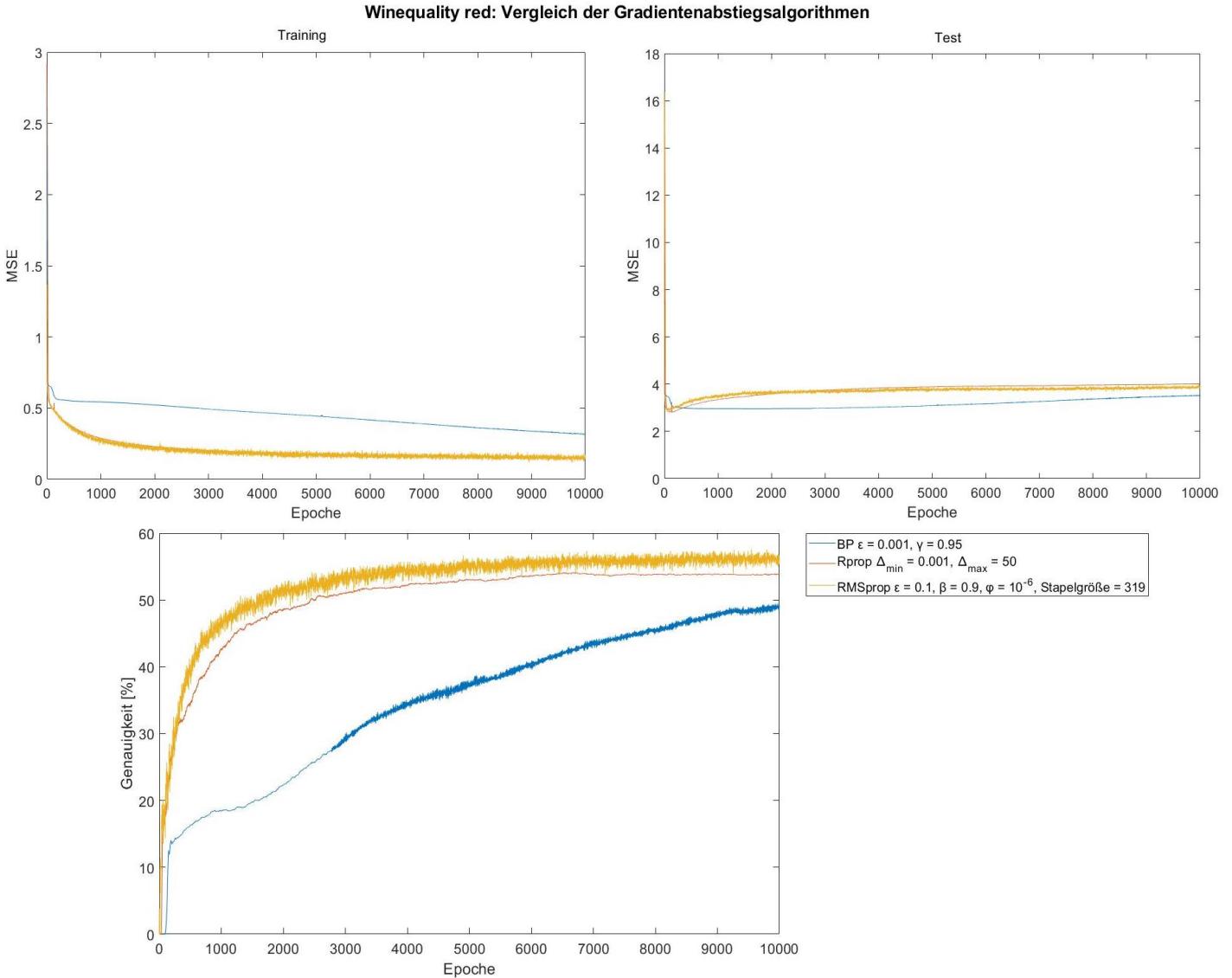


Abbildung 48: Verlauf des MSE der Trainings- (oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten) des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop.

Aus Abbildung 48 ist festzustellen, dass der Gradientenabstieg mittels RMSprop den größten Erfolg unter Verwendung des Wine Quality Red-Datensatzes verspricht. Das künstliche neuronale Netzwerk passt sich mittels RMSprop und Rprop nahezu gleichmäßig effizient an die Trainingsbeispiele an, RMSprop erreicht jedoch im Mittel niedrigere MSE-Werte der Testbeispiele, das wiederum darauf deutet, dass RMSprop ein günstigeres lokales Minimum im Hyperraum der Kostenfunktion erreicht. Der Gradientenabstieg mittels BP fällt unter Verwendung des Wine Quality Red-Datensatzes am schlechtesten aus, erreicht jedoch innerhalb der 10000 Epochen noch keinen Plateauwert der Genauigkeitswerte. Dementsprechend sind bessere Genauigkeitswerte von BP möglich. In Abbildung 49 wird die erreichte Genauigkeit nach 10000 Epochen, die durchschnittliche Genauigkeit der letzten

1000 Epochen und die maximal erreichte Genauigkeit innerhalb der 10000 Epochen des jeweiligen Gradientenabstiegsalgorithmus übersichtlich dargestellt.

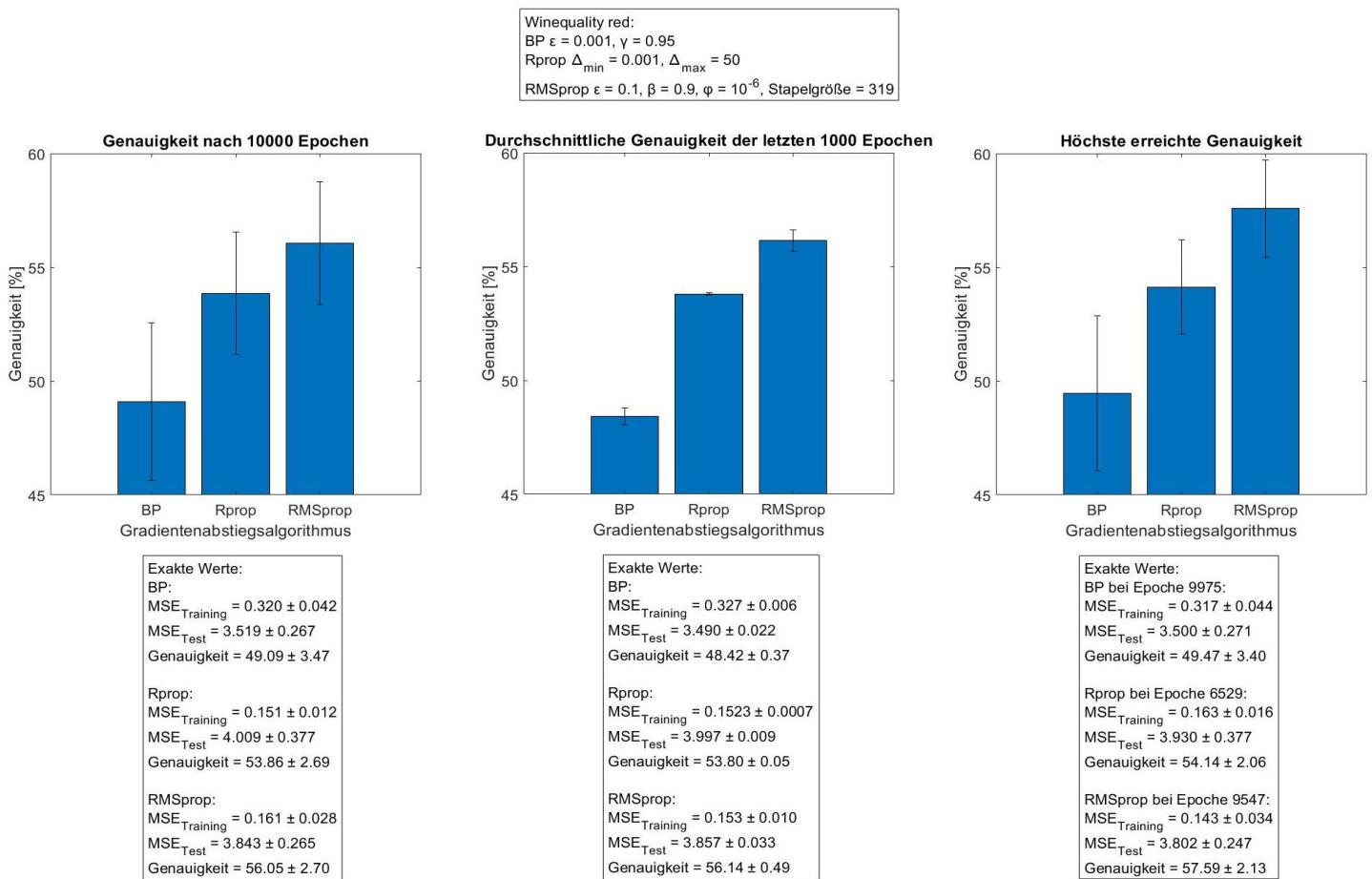


Abbildung 49: Übersicht der nach 10000 Epochen erreichten Genauigkeit (links), der durchschnittlichen Genauigkeit der letzten 1000 Epochen (mittig) und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen (rechts) des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes von BP, Rprop und RMSprop.

## 4.8. Wine Quality White

Bei dem Wine Quality White-Datensatz des UCI Machine Learning Repository handelt es sich um ein Klassifikationsproblem [28]. Der Datensatz besitzt insgesamt 4896 Beispiele, dem jedoch 1 Beispiel entnommen wurden, um eine Aufteilung in 80 % Trainingsbeispiele und 20 % Testbeispiele zu ermöglichen. Dementsprechend wurden 4895 Beispiele des Datensatzes verwendet, die auf 3916 Trainingsbeispiele und 979 Testbeispiele aufgeteilt wurden. In dem Datensatz existieren theoretisch 7 mögliche Klassen, die ausgewertet werden können, jedoch wurden 11 mögliche Klassen verwendet. Dementsprechend handelt es sich bei den möglichen Klassen um Qualitätsbewertungen von Weißwein – 0 (schlechteste Qualität) bis 10 (beste Qualität). In den 4895 Beispielen des Datensatzes existieren

#### 4. Experimentelle Untersuchungen

---

insgesamt 20 Beispiele für die Qualität 3, 163 Beispiele für die Qualität 4, 1457 Beispiele für die Qualität 5, 2196 Beispiele für die Qualität 6, 879 Beispiele für die Qualität 7, 175 Beispiele für die Qualität 8 und 5 Beispiele für die Qualität 9. Für die restlichen Qualitäten existieren keine Beispiele in dem Datensatz. Jedes Beispiel besitzt insgesamt 11 Attribute, die als Eingabewerte für das künstliche neuronale Netzwerk zur Ermittlung der Qualität des Weißweins dienen. Diese sind in Tabelle 10 übersichtlich dargestellt.

Tabelle 10: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Wine Quality White-Datensatzes.

Attribut	Information	Wertebereich
A <sub>1</sub>	Fester Säuregehalt in g(Weinsäure)/dm <sup>3</sup>	[3.8, 14.2]
A <sub>2</sub>	Flüchtiger Säuregehalt in g(Essigsäure)/dm <sup>3</sup>	[0.08, 1.1]
A <sub>3</sub>	Zitronensäure in g/dm <sup>3</sup>	[0, 1.66]
A <sub>4</sub>	Restzucker in g/dm <sup>3</sup>	[0.6, 65.8]
A <sub>5</sub>	Chloride in g(Natriumchlorid)/dm <sup>3</sup>	[0.009, 0.346]
A <sub>6</sub>	Freies Schwefeldioxid in mg/dm <sup>3</sup>	[2, 289]
A <sub>7</sub>	Gesamtes Schwefeldioxid in mg/dm <sup>3</sup>	[9, 440]
A <sub>8</sub>	Dichte in g/cm <sup>3</sup>	[0.98711, 1.03898]
A <sub>9</sub>	pH-Wert	[2.72, 3.82]
A <sub>10</sub>	Sulfate in g(Kaliumsulfat)/dm <sup>3</sup>	[0.22, 1.08]
A <sub>11</sub>	Alkoholgehalt in vol.%	[8, 14.2]

Bei diesem Datensatz wurde nicht darauf geachtet, dass die Verhältnisse der einzelnen Klassen in den Trainingsbeispielen sowie in den Testbeispielen aufrechterhalten werden. Demnach könnte es vorkommen, dass beispielsweise in einer Mischung des Datensatzes in den Trainingsbeispielen oder Testbeispielen keine Beispiele für die Qualität 9 vertreten waren. Für den Wine Quality White-Datensatz wurden künstliche neuronale Netzwerke gemäß (3) mit zwei verborgenen Schichten mittels des n++-Simulatorkerns erstellt. Dementsprechend befanden sich in der Eingabeschicht 11 Eingabeneuronen, in den verborgenen Schichten jeweils 18 verborgene Neuronen und in der Ausgabeschicht 11 Ausgabeneuronen. Als Aktivierungsfunktionen der verborgenen Neuronen und der Ausgabeneuronen wurde die Sigmoidfunktion verwendet. Jedes Ausgabeneuron repräsentiert die Wahrscheinlichkeit der Eingabewerte zu einer jeweiligen Qualität des Weißweins zu gehören.

### 4.8.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate

Dieses Experiment soll den Effekt der Lernrate von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Wine Quality White-Datensatzes ermitteln. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 50 dargestellt.

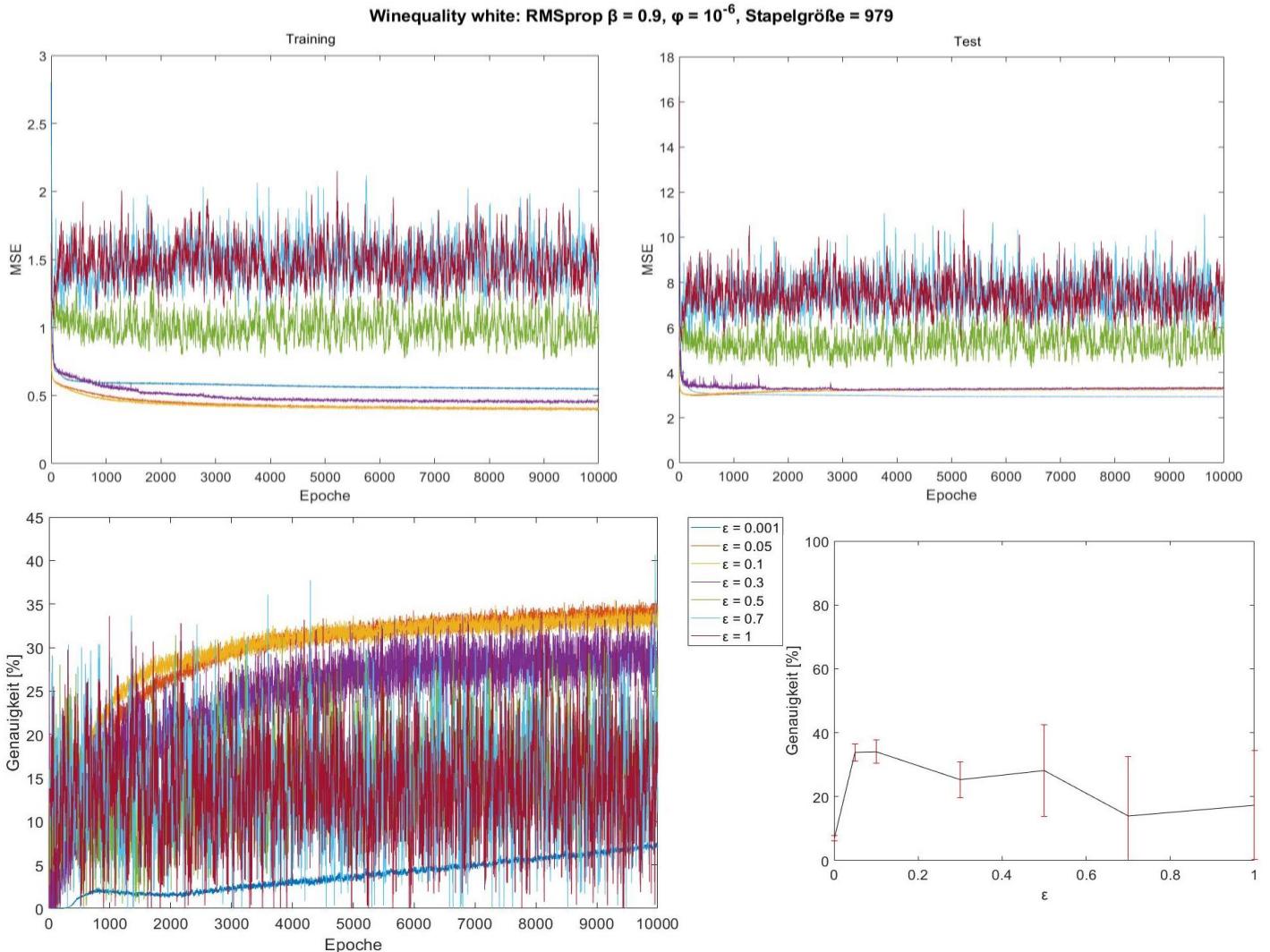


Abbildung 50: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Lernrate dargestellt.

In Abbildung 50 ist zu beobachten, dass mittels der Lernrate 0.05 und 0.1 der größte Lernerfolg des künstlichen neuronalen Netzwerkes erreicht werden kann. Ab einer Lernrate von 0.3 beginnt die Schrittweite des Gradientenabstiegs zu groß zu werden, sodass es zu starken Oszillationen der Verläufe der MSE-Werte und der Genauigkeitswerte kommt. Bei der Lernrate von 0.7 tritt sogar eine Spitze

#### 4. Experimentelle Untersuchungen

auf, die oberhalb einer Genauigkeit von 40 % liegt und die maximal erreichte Genauigkeit darstellt. Den schlechtesten Lernerfolg erreicht der Gradientenabstieg mittels einer Lernrate von 0.001, jedoch wird innerhalb der 10000 Epochen kein Plateauwert der Genauigkeitswerte erreicht. Dementsprechend sind bei einer Lernrate von 0.001 durchaus bessere Ergebnisse möglich.

##### **4.8.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors**

Dieses Experiment soll den Effekt des Vergessensfaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Wine Quality White-Datensatzes untersuchen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 51 dargestellt.

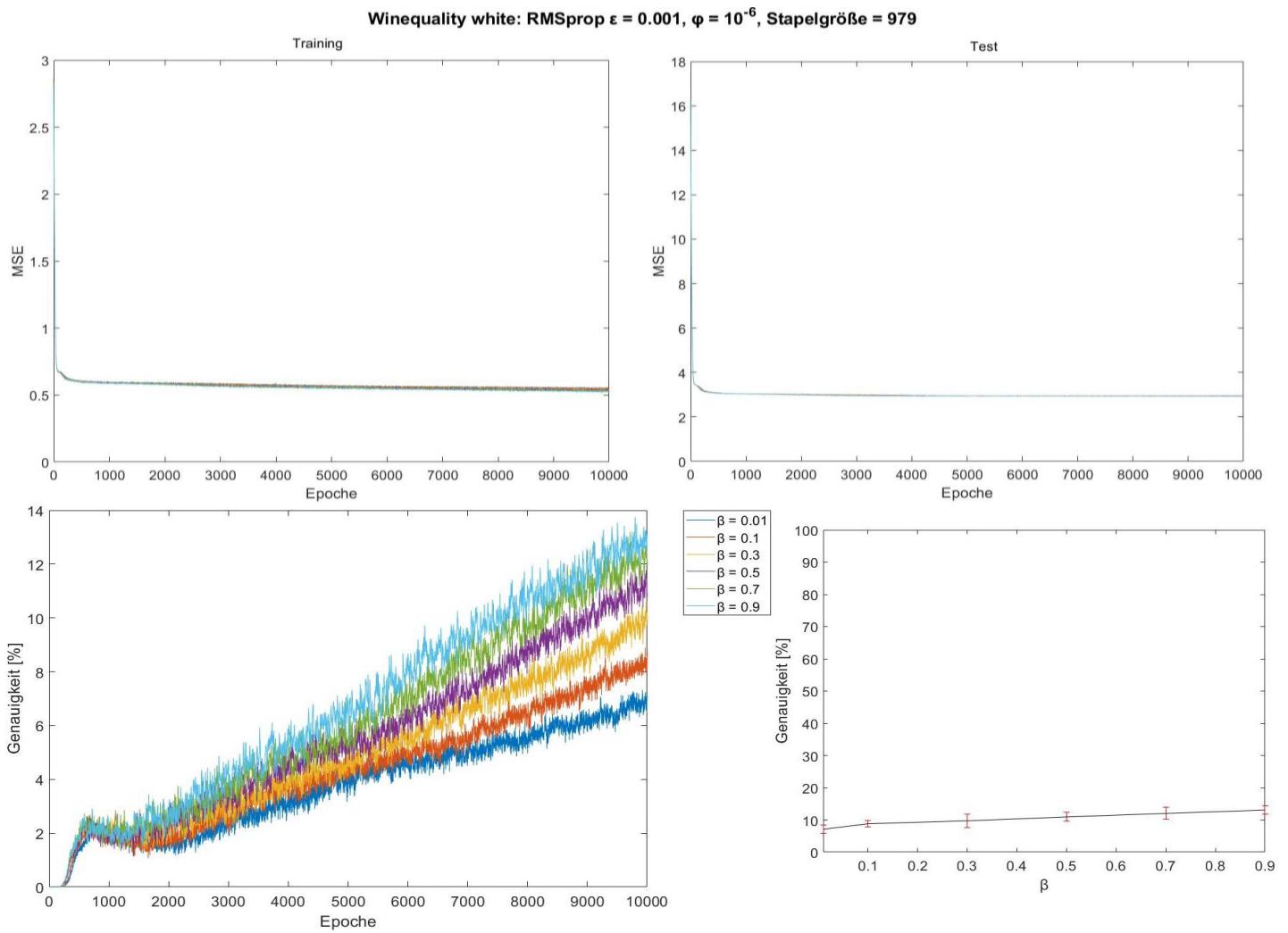


Abbildung 51: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des Vergessensfaktors dargestellt.

In Abbildung 51 kann man eindeutig erkennen, dass der Lernerfolg des künstlichen neuronalen Netzwerkes mit steigendem Vergessensfaktor, unter Verwendung des Wine Quality White-Datensatzes, steigt. Da bei einem kleinen Vergessensfaktor Gradienten vorheriger Mini-Stapel weniger Einfluss auf den gleitenden Durchschnitt haben, könnte bei der Epoche von etwa 500 der Gradientenabstiegsalgorithmus leicht andere Routen entlang des Hyperraums der Kostenfunktion in Richtung schlechterer lokaler Minima nehmen. Im Gegenteil dazu, ist bei einem großen Vergessensfaktor der Einfluss der Gradienten vorheriger Mini-Stapel auf den gleitenden Durchschnitt größer, wodurch im Vergleich zu den kleineren Vergessensfaktoren günstigere Minima erreicht werden könnten. Allerdings könnte man auch untersuchen, wie sich die verschiedenen Vergessensfaktoren bei mehr als 10000 Epochen verhalten, denn es könnte auch durchaus sein, dass

## 4. Experimentelle Untersuchungen

alle Verläufe zu dem gleichen Plateauwert konvergieren und damit zu dem gleichen oder ähnlichen lokalen Minimum führen.

### 4.8.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors

Dieses Experiment soll den Effekt des Unschärfeefaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Wine Quality White-Datensatzes überprüfen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 52 dargestellt.

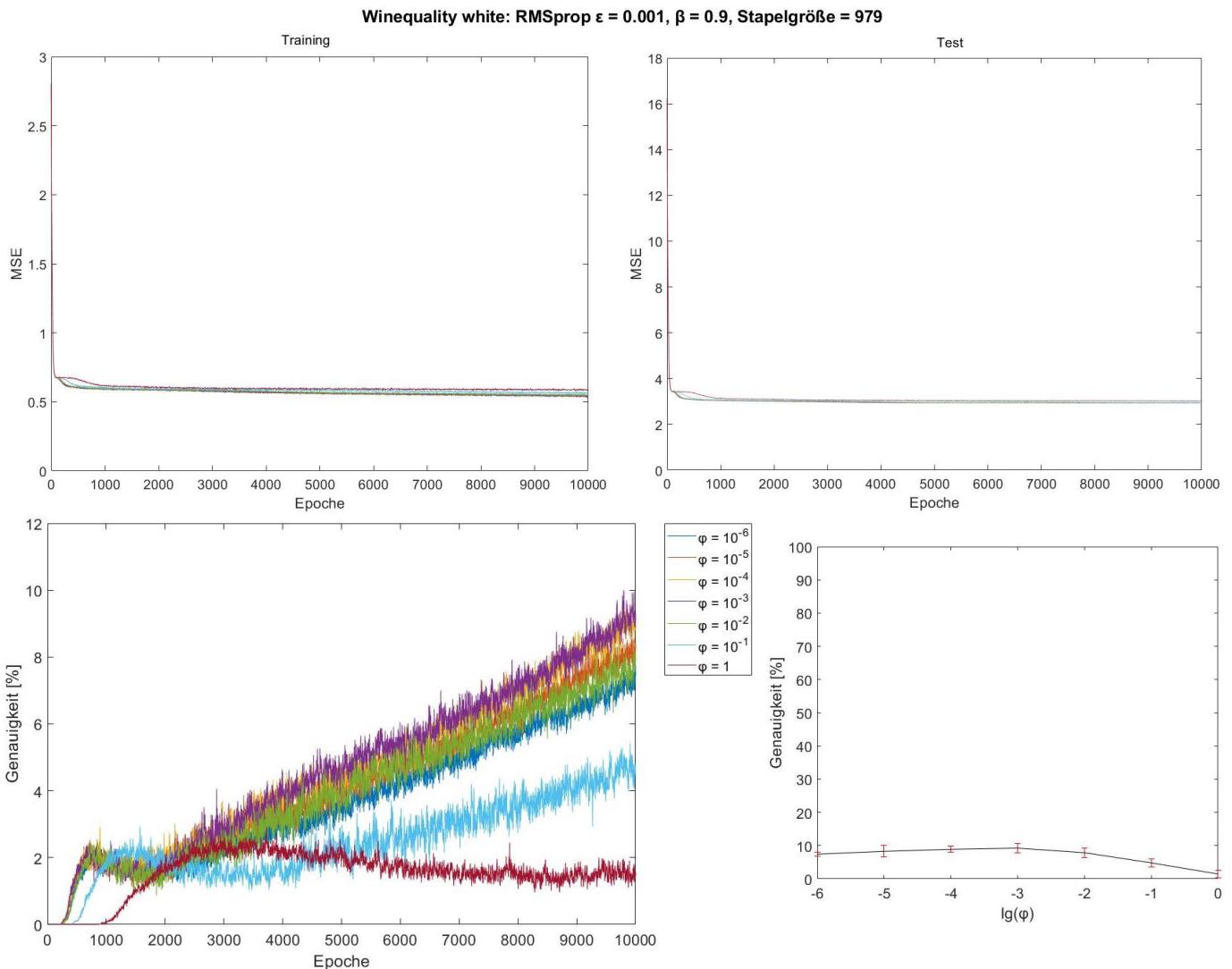


Abbildung 52: Verlauf des MSE der Trainings- (oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des logarithmierten Unschärfeefaktors dargestellt.

In Abbildung 52 kann man erkennen, dass zuerst der Lernerfolg des künstlichen neuronalen Netzwerkes mit steigendem Unschärfeefaktor steigt. Den größten Lernerfolg weist hierbei der Gradientenabstieg mittels eines Unschärfeefaktors von  $10^{-3}$  auf. Ab einem Unschärfeefaktor von  $10^{-2}$  beginnt sich der Unschärfeefaktor negativ auf den Gradientenabstieg auszuwirken, sodass im Mittel schlechtere Genauigkeitswerte mit steigendem Unschärfeefaktor erreicht werden.

#### **4.8.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel**

Dieses Experiment soll den Effekt der Größe der Mini-Stapel von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Wine Quality White-Datensatzes analysieren. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 53 dargestellt.

## 4. Experimentelle Untersuchungen

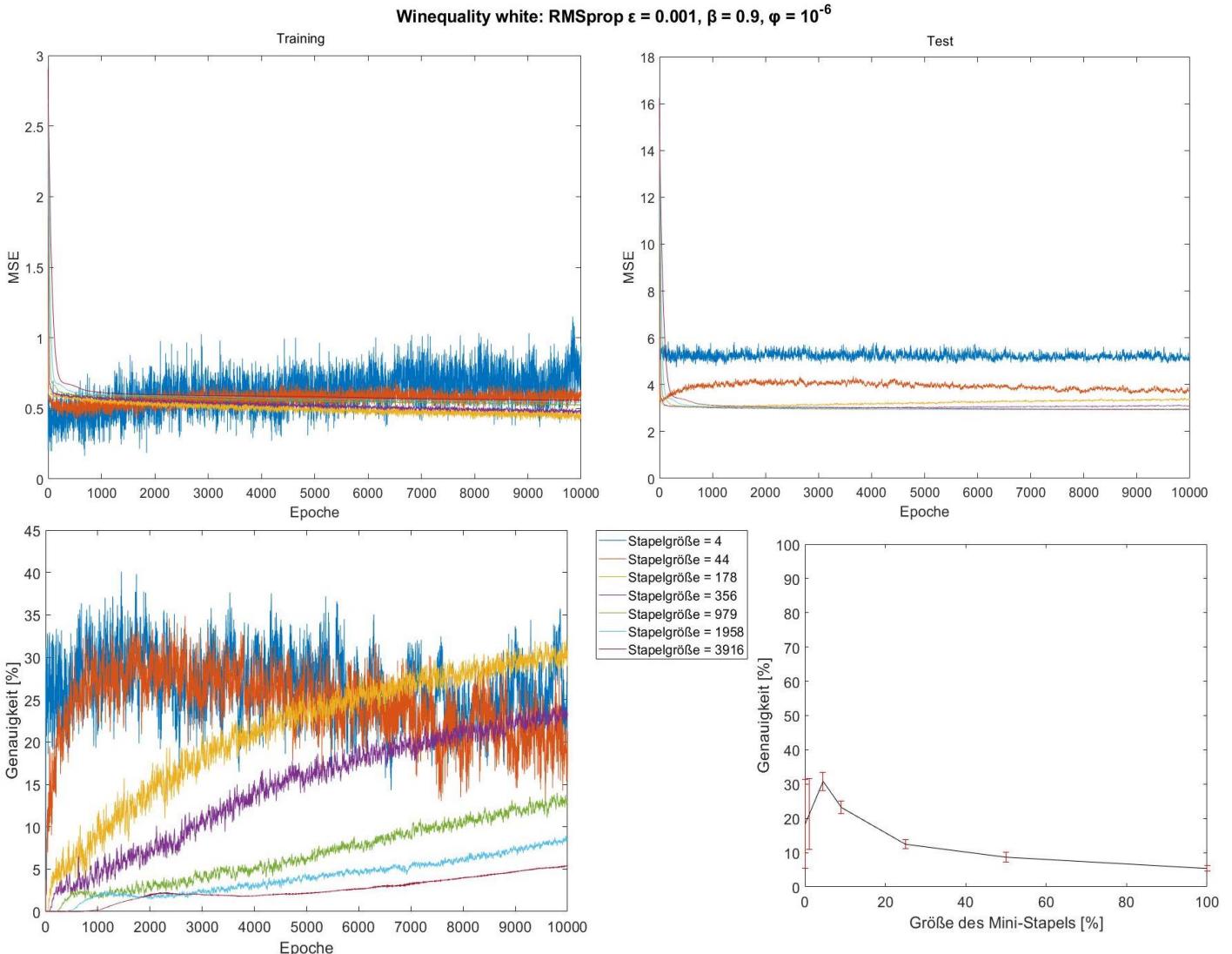


Abbildung 53: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Größe des Mini-Stapels im Verhältnis zu der gesamten Anzahl der Trainingsbeispiele dargestellt.

Aus Abbildung 53 folgt, dass mittels eines Gradientenabstiegs mit einer Mini-Stapelgröße von 4 und 44 der größte Lernerfolg des künstlichen neuronalen Netzwerkes erzielt werden kann. Bei dem Verlauf der Mini-Stapelgröße von 4 treten sogar bei einer Epoche von etwa 1500 mehrere Spitzen der Genauigkeitswerte auf, die ein Maximum der erreichten Genauigkeitswerte darstellen. Erwähnenswert ist hierbei, dass eine Mini-Stapelgröße von 1 einem stochastischen Gradientenabstieg von RMSprop gleichen würde, wohingegen eine Mini-Stapelgröße von 3916 einem Stapel-Gradientenabstieg von RMSprop gleichen würde. Mittels einer Mini-Stapelgröße von 178 lassen sich, ohne starke Oszillationen der Verläufe, die größten Genauigkeitswerte erreichen. Aus der genannten Abbildung lässt sich außerdem erkennen, dass ab der Mini-Stapelgröße von 356, aufgrund nur weniger Gewichtsaktualisierungen innerhalb einer Epoche, die niedrigsten Genauigkeitswerte erzielt werden.

Die Verläufe der Mini-Stapelgrößen ab 178 erreichen jedoch innerhalb der 10000 Epochen noch keinen Plateauwert also sind durchaus bessere Ergebnisse denkbar, wenn man das künstliche neuronale Netzwerk über die 10000 Epochen hinaus trainiert.

#### 4.8.5. Vergleich der Gradientenabstiegsalgorithmen

Hier soll der Lernerfolg des künstlichen neuronalen Netzwerkes mittels des Gradientenabstiegsalgorithmus RMSprop mit dem Lernerfolg des künstlichen neuronalen Netzwerkes mittels der Gradientenabstiegsalgorithmen BP und Rprop verglichen werden. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 54 dargestellt.

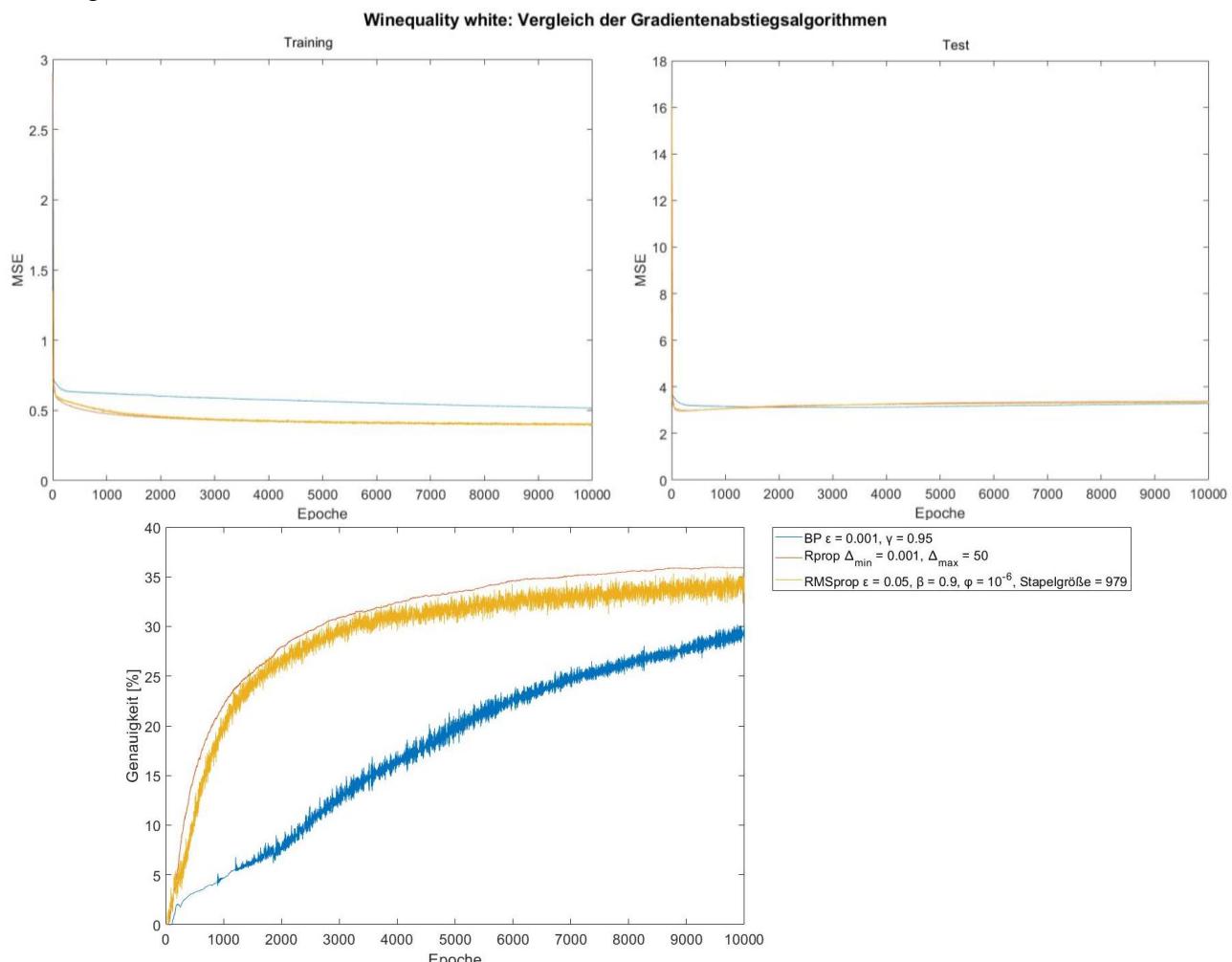


Abbildung 54: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten) des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop.

#### 4. Experimentelle Untersuchungen

Abbildung 54 zeigt, dass mittels Rprop der größte Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des Wine Quality White-Datensatzes erreicht werden kann. Das künstliche neuronale Netzwerk passt sich mittels RMSprop und Rprop nahezu gleichmäßig effizient an die Trainingsbeispiele an, was wiederum darauf deutet, dass Rprop ein günstigeres lokales Minimum im Hyperraum der Kostenfunktion in Folge des Gradientenabstiegs erreicht. Unter Rücksichtnahme von Abbildung 52 und Abbildung 53 kann man feststellen, dass durch die Wahl geeigneter Parameter der Lernerfolg von RMSprop gesteigert werden kann. Dadurch könnte RMSprop unter Umständen auch bessere Ergebnisse als Rprop erzielen. Der Gradientenabstieg mittels BP fällt unter Verwendung des Wine Quality White-Datensatzes am schlechtesten aus, erreicht jedoch innerhalb der 10000 Epochen keinen Plateauwert der Genauigkeitswerte. Dementsprechend sind bessere Genauigkeitswerte von BP möglich. In Abbildung 55 wird die erreichte Genauigkeit nach 10000 Epochen, die durchschnittliche Genauigkeit der letzten 1000 Epochen und die maximal erreichte Genauigkeit innerhalb der 10000 Epochen des jeweiligen Gradientenabstiegsalgorithmus übersichtlich dargestellt.

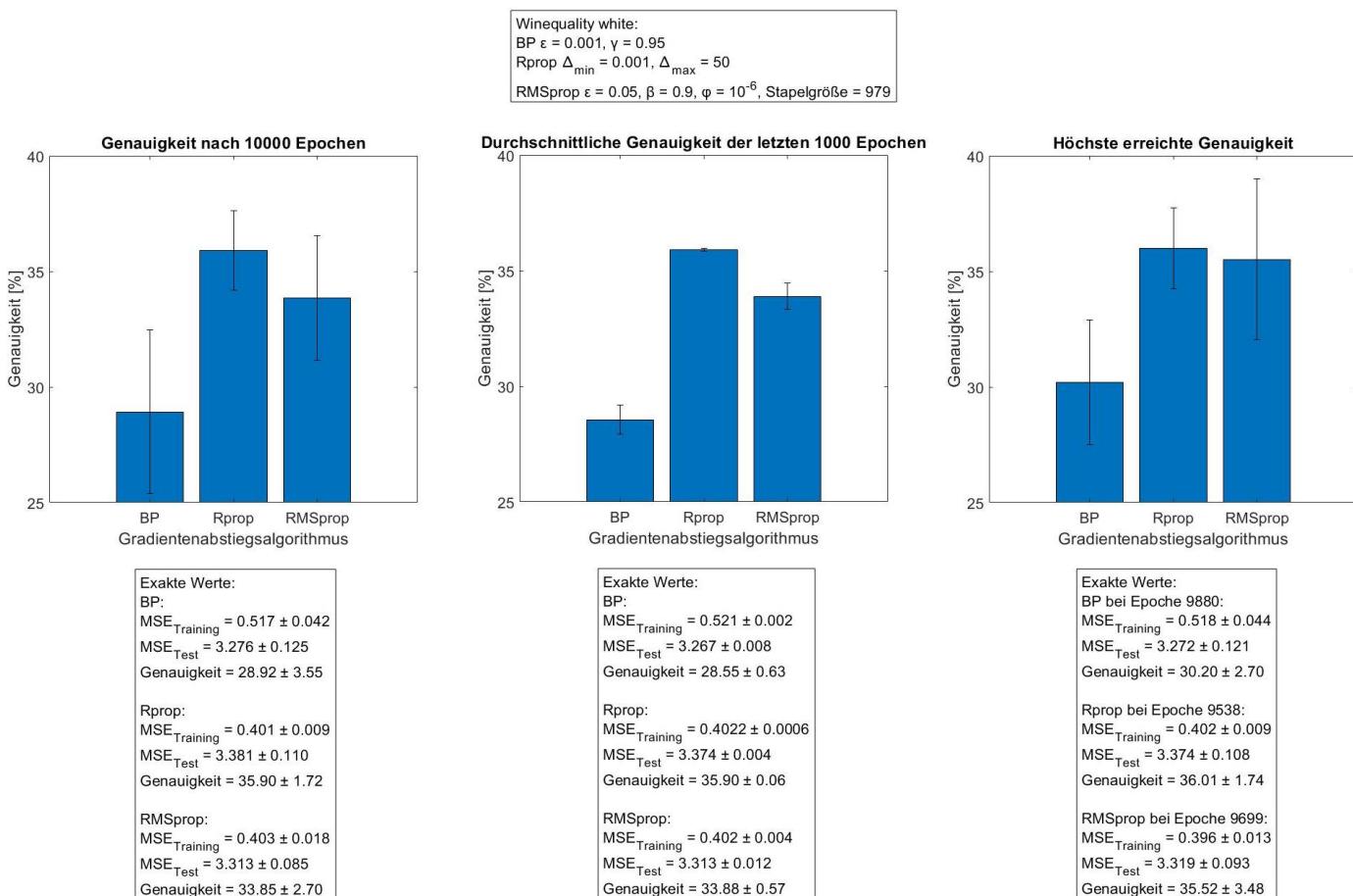


Abbildung 55: Übersicht der nach 10000 Epochen erreichten Genauigkeit (links), der durchschnittlichen Genauigkeit der letzten 1000 Epochen (mittig) und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen (rechts) des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes von BP, Rprop und RMSprop.

## 4.9. MAGIC Gamma Telescope

Bei dem MAGIC Gamma Telescope-Datensatz des UCI Machine Learning Repository handelt es sich um ein Klassifikationsproblem [29]. Der Datensatz besitzt insgesamt 19020 Beispiele, die auf 15216 Trainingsbeispiele und 3804 Testbeispiele aufgeteilt wurden. In dem Datensatz existieren 2 mögliche Klassen, die ausgewertet werden können. Bei den möglichen Klassen handelt es sich darum, ob es sich bei den vom Teleskop detektierten Cherenkov Photonen um ein tatsächliches Signal, ausgelöst durch Primäre Gammastrahlen, oder ob es sich um ein Signal der Hintergrundstrahlung, ausgelöst durch kosmische Strahlung der oberen Atmosphäre, handelt – g (gamma, signal), h (hadron, background). In den 19020 Beispielen des Datensatzes existieren insgesamt 12332 Beispiele für g und 6688 Beispiele für h. Jedes Beispiel besitzt insgesamt 10 Attribute, die als Eingabewerte für das künstliche neuronale Netzwerk zur Ermittlung des Signals dienen. Diese sind in Tabelle 11 aufgelistet.

Tabelle 11: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des MAGIC Gamma Telescope-Datensatzes.

Attribut	Information	Wertebereich
A <sub>1</sub>	fLength: Hauptachse der Ellipse in mm	[4.2835, 334.177]
A <sub>2</sub>	fWidth: Nebenachse der Ellipse in mm	[0, 256.382]
A <sub>3</sub>	fSize: Dekadischer Logarithmus der Summe des Inhalts aller Pixel in #phot	[1.9413, 5.3233]
A <sub>4</sub>	fConc: Verhältnis der Summe zweier höchster Pixel über fSize	[0.0131, 0.893]
A <sub>5</sub>	fConc1: Verhältnis des höchsten Pixels zu fSize	[0.0003, 0.6742]
A <sub>6</sub>	fAsym: Abstand vom höchsten Pixel zur Mitte, projiziert auf die Hauptachse in mm	[-457.9161, 575.2407]
A <sub>7</sub>	fM3Long: Dritte Wurzel des dritten Moments entlang der Hauptachse	[-331.78, 238.321]
A <sub>8</sub>	fM3Trans: Dritte Wurzel des dritten Moments entlang der Nebenachse in mm	[-205.8947, 179.851]
A <sub>9</sub>	fAlpha: Winkel der Hauptachse mit Vektor zum Ursprung in °	[0, 90]
A <sub>10</sub>	fDist: Abstand vom Ursprung zum Zentrum der Ellipse in mm	[1.2826, 495.561]

#### 4. Experimentelle Untersuchungen

Es wurde darauf geachtet, dass der Datensatz zufallsgeneratorbasiert derart gemischt wurde, dass 9860 Beispiele für g und 5356 Beispiele für h in den Trainingsbeispielen vorhanden sind. Folglich befanden sich 2465 Beispiele für g und 1339 Beispiele für h in den Testbeispielen. Für den MAGIC Gamma Telescope-Datensatz wurden künstliche neuronale Netzwerke gemäß (3) mit zwei verborgenen Schichten mittels des n++-Simulatorkerns erstellt. Dementsprechend befanden sich in der Eingabeschicht 10 Eingabeneuronen, in den verborgenen Schichten jeweils 9 verborgene Neuronen und in der Ausgabeschicht 2 Ausgabeneuronen. Als Aktivierungsfunktionen der verborgenen Neuronen und der Ausgabeneuronen wurde die Sigmoidfunktion verwendet. Jedes Ausgabeneuron repräsentiert die Wahrscheinlichkeit der Eingabewerte ein durch primäre Gammastrahlen ausgelöstes Signal oder ein durch Hintergrundstrahlung ausgelöste Signal zu sein.

##### **4.9.1. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Lernrate**

Dieses Experiment soll den Effekt der Lernrate von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des MAGIC Gamma Telescope-Datensatzes ermitteln. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 56 dargestellt.

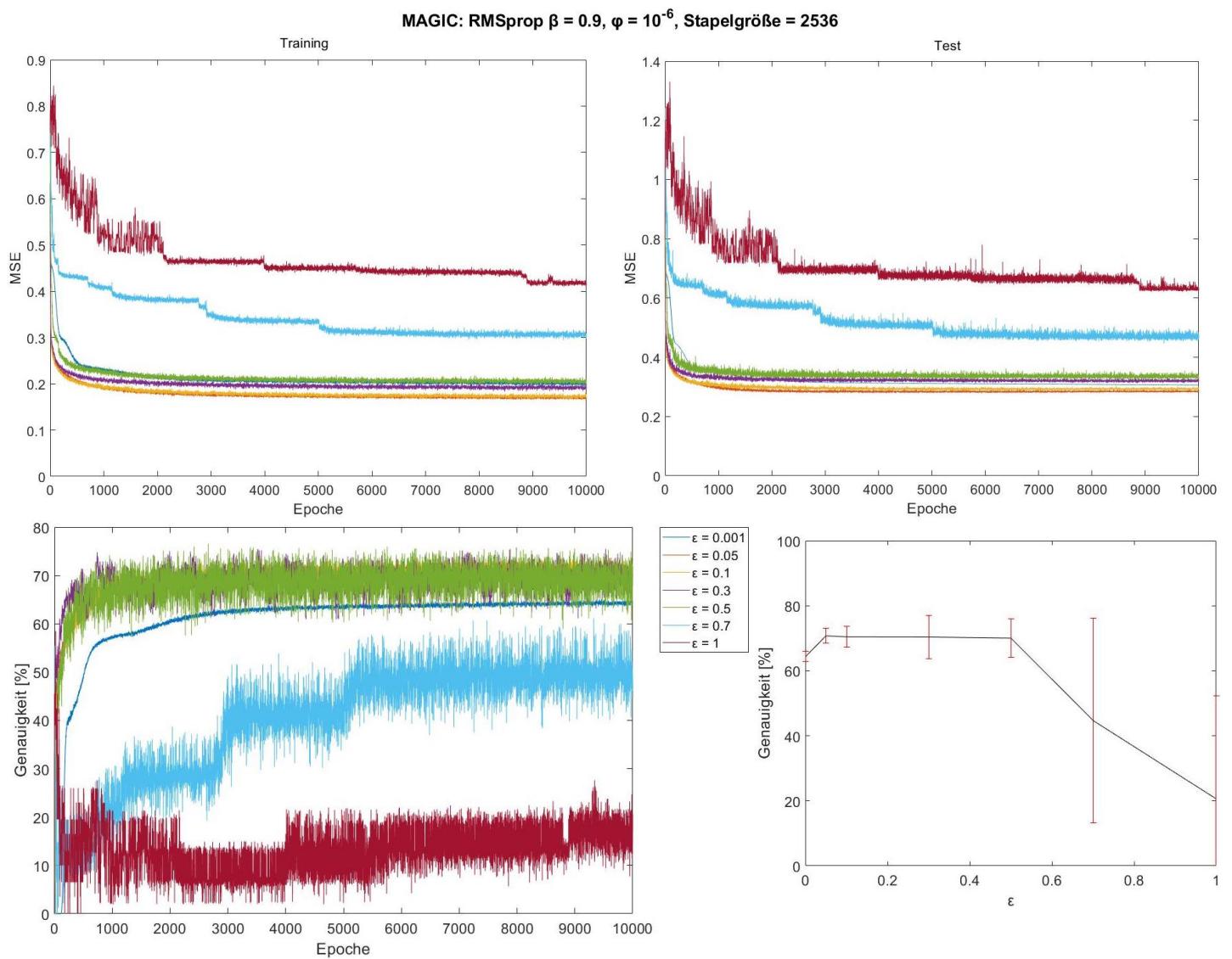


Abbildung 56: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Lernrate dargestellt.

In Abbildung 56 kann man sehen, dass der Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des MAGIC Gamma Telescope-Datensatzes zuerst mit steigender Lernrate steigt. Ab etwa einer Lernrate von 0.3 beginnen die MSE-Werte jedoch im Mittel zu höheren Plateauwerten zu steigen und der Genauigkeitswert zu sinken. Zusätzlich nimmt zunehmend die Stärke der Oszillation der Verläufe zu. Die höchsten Genauigkeitswerte werden mit einer Lernrate von 0.05, 0.1, 0.3 und 0.5 erreicht. Mittels einer Lernrate von 0.001 erreicht man ab etwa der Epoche 3000 bereits bei etwa 63 % einen Plateauwert. Mit einer Lernrate von 0.7 und 1 erreicht man den schlechtesten Lernerfolg. Dies ist auf die zu große Schrittweite des Gradientenabstiegs zurückzuführen.

## 4. Experimentelle Untersuchungen

### 4.9.2. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Vergessensfaktors

Dieses Experiment soll den Effekt des Vergessensfaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des MAGIC Gamma Telescope-Datensatzes erforschen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 57 dargestellt.

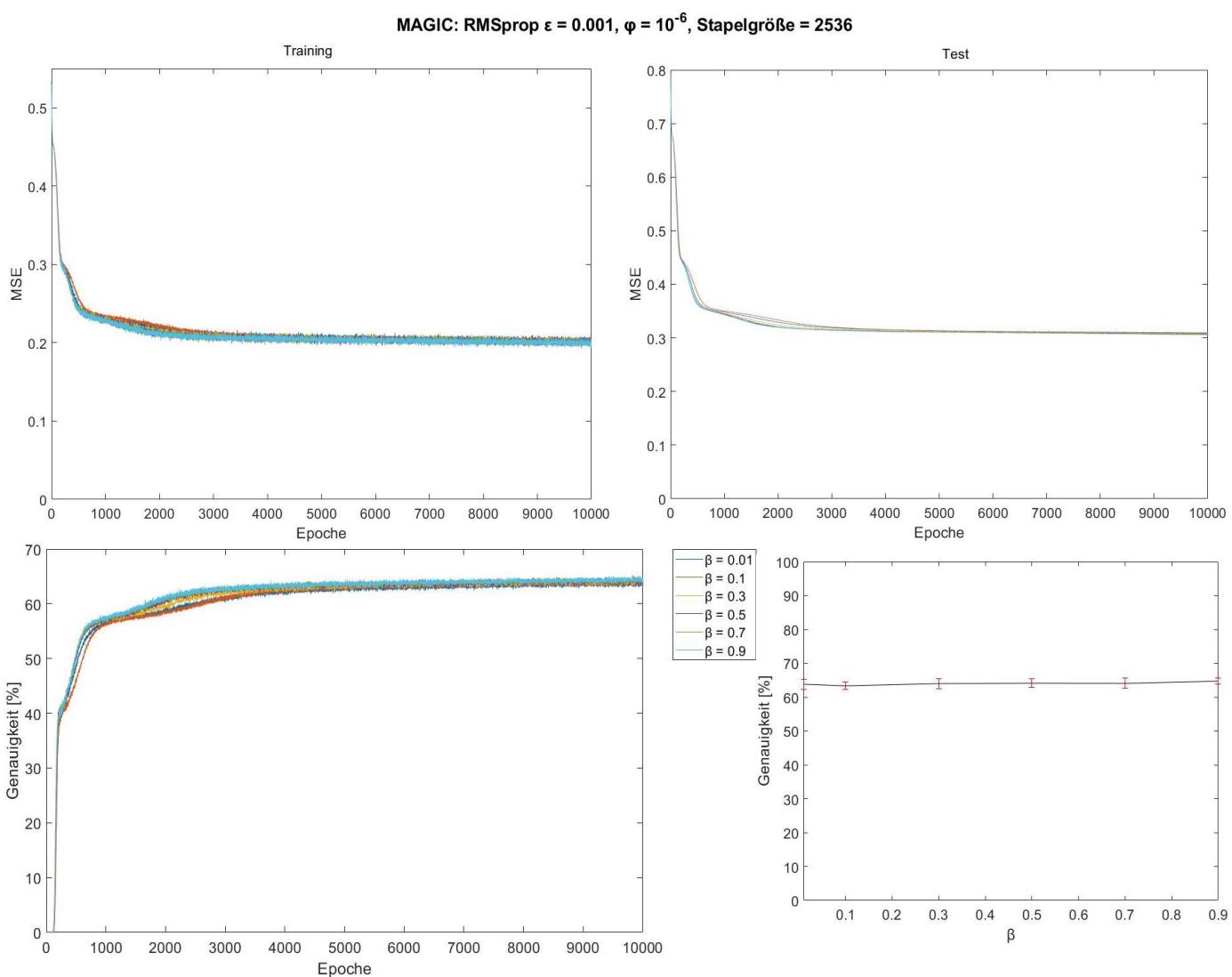


Abbildung 57: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des Vergessensfaktors dargestellt.

In Abbildung 57 ist zu sehen, dass mit steigendem Vergessensfaktor der Lernerfolg des künstlichen neuronalen Netzwerks unter Verwendung des MAGIC Gamma Telescope-Datensatzes steigt. Da bei einem kleinen Vergessensfaktor Gradienten vorheriger Mini-Stapel weniger Einfluss auf den gleitenden Durchschnitt haben, könnte bei der Epoche von etwa 300 der Gradientenabstiegsalgorithmus leicht andere Routen entlang des Hyperraums der Kostenfunktion, hin zu schlechteren lokalen Minima, nehmen. Im Gegenteil dazu, ist bei einem großen Vergessensfaktor der Einfluss der Gradienten vorheriger Mini-Stapel auf den gleitenden Durchschnitt größer, wodurch im Vergleich zu den kleineren Vergessensfaktoren günstigere Minima erreicht werden. Allerdings konvergieren die Verläufe der unterschiedlichen Vergessensfaktoren zu nahezu einem Plateauwert und damit zu dem gleichen oder einem ähnlichen lokalen Minimum.

#### **4.9.3. Gradientenabstiegsverfahren mittels RMSprop unter Variation des Unschärfeefaktors**

Dieses Experiment soll den Effekt des Unschärfeefaktors von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des MAGIC Gamma Telescope-Datensatzes untersuchen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 58 dargestellt.

#### 4. Experimentelle Untersuchungen

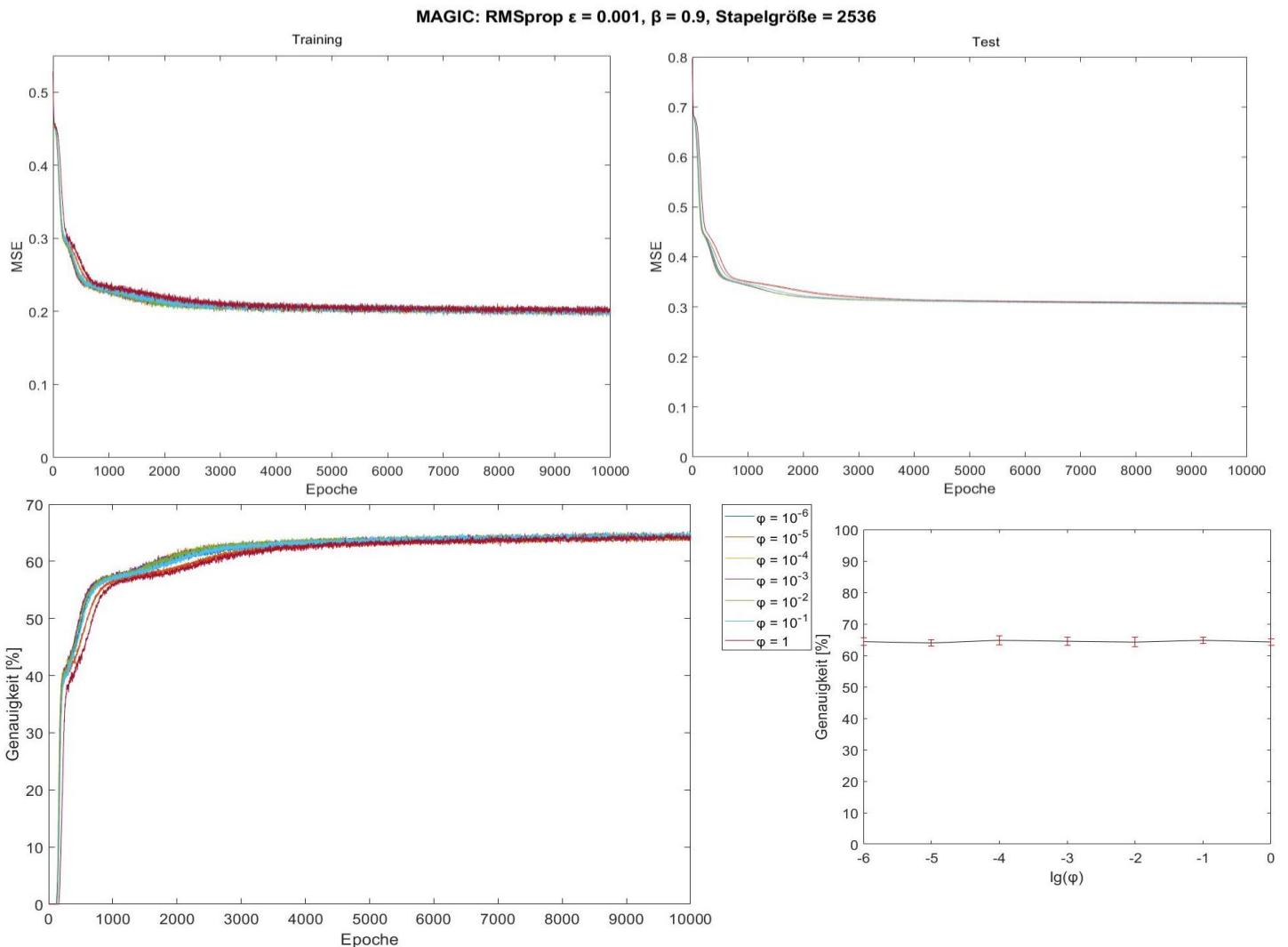


Abbildung 58: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit des logarithmierten Unschärfeefaktors dargestellt.

Aus Abbildung 58 kann man folgern, dass der Unschärfeefaktor nur einen geringen Effekt auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des MAGIC Gamma Telescope-Datensatzes hat. Es erscheint jedoch, dass der Unschärfeefaktor den Gradientenabstieg mittels RMSprop mit steigendem Wert verlangsamt, da die Verläufe der MSE-Werte sowie der Genauigkeitswerte nahezu zu den gleichen Plateauwerten konvergieren.

#### 4.9.4. Gradientenabstiegsverfahren mittels RMSprop unter Variation der Größe der Mini-Stapel

Dieses Experiment soll den Effekt der Größe der Mini-Stapel von RMSprop auf den Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des MAGIC Gamma Telescope-Datensatzes überprüfen. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 59 dargestellt.

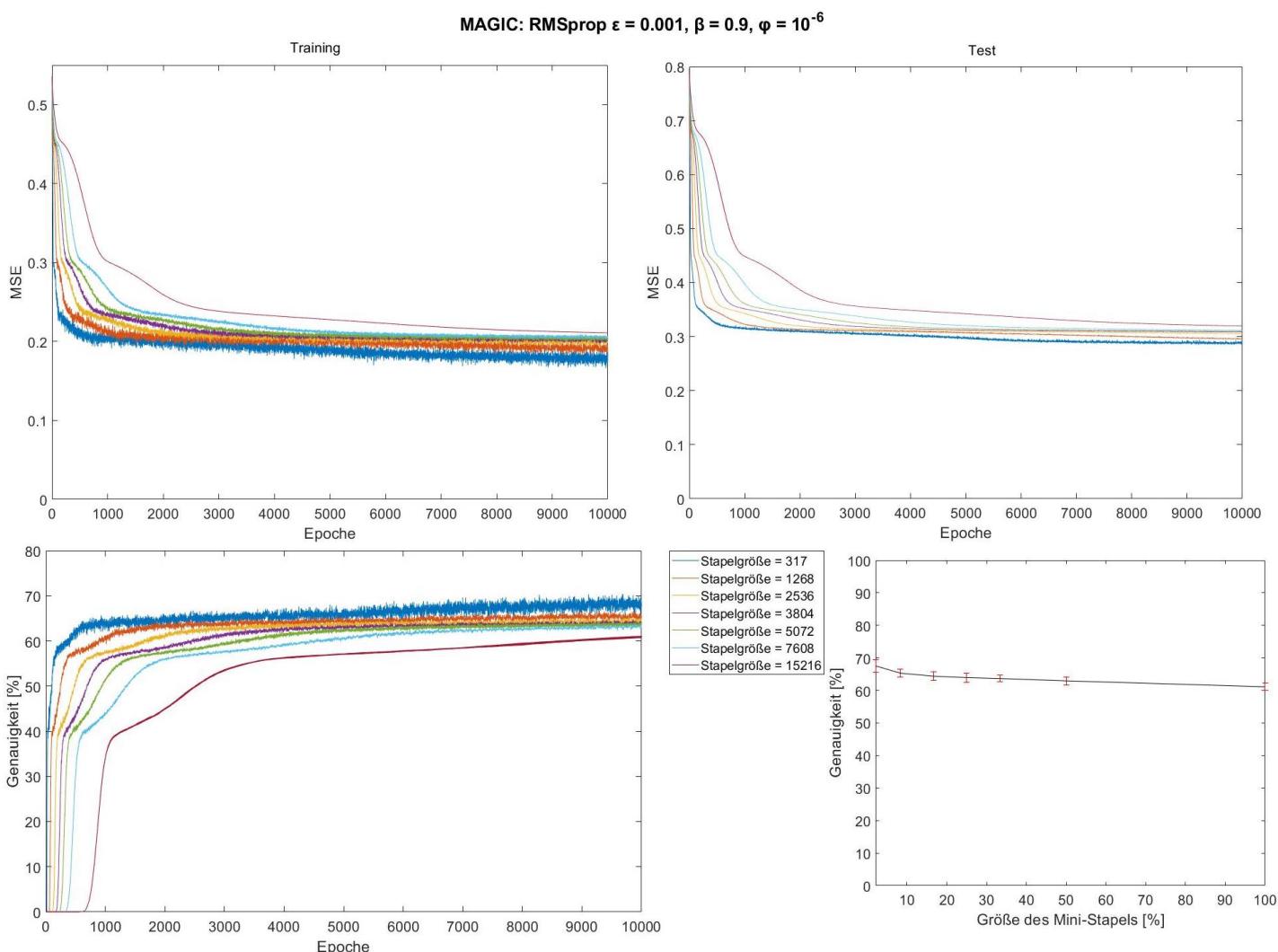


Abbildung 59: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten links) des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels. Unten rechts ist die erreichte Genauigkeit nach 10000 Epochen Training in Abhängigkeit der Größe des Mini-Stapels im Verhältnis zu der gesamten Anzahl der Trainingsbeispiele dargestellt.

Aus Abbildung 59 kann man deutlich erkennen, dass der Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des MAGIC Gamma Telescope-Datensatzes mit steigender Größe des Mini-Stapels im Mittel sinkt. Erwähnenswert ist hierbei, dass eine Mini-Stapelgröße von 1 einem

#### 4. Experimentelle Untersuchungen

stochastischen Gradientenabstieg von RMSprop gleichen würde, wohingegen eine Mini-Stapelgröße von 15216 einem Stapel-Gradientenabstieg von RMSprop gleichen würde. Dementsprechend erreicht man durch eine Mini-Stapelgröße von 317 innerhalb von etwa 800 Epochen die höchsten Genauigkeitswerte. Der Verlauf der Mini-Stapelgröße von 317 scheint sogar noch keinen Plateauwert erreicht zu haben, wodurch sogar womöglich noch bessere Genauigkeitswerte möglich sein könnten. Alle anderen Verläufe der Mini-Stapel scheinen nahezu ihre Plateauwerte erreicht zu haben. Außerdem kann man aus den genannten Abbildungen erkennen, dass mit steigender Mini-Stapelgröße der Gradientenabstieg mittels RMSprop verlangsamt wird. Dies ist jedoch auf die Anzahl an Gewichtsaktualisierungen innerhalb einer Epoche, die mit steigender Größe der Mini-Stapel sinkt, zurückzuführen.

##### **4.9.5. Vergleich der Gradientenabstiegsalgorithmen**

Hier soll der Lernerfolg des künstlichen neuronalen Netzwerkes mittels des Gradientenabstiegsalgorithmus RMSprop mit dem Lernerfolg des künstlichen neuronalen Netzwerkes mittels der Gradientenabstiegsalgorithmen BP und Rprop verglichen werden. Dazu wurde der Verlauf der ermittelten mittleren quadratischen Abweichung der Trainings- und der Testbeispiele gemäß (19) sowie die Genauigkeit des künstlichen neuronalen Netzwerkes innerhalb jeder Epoche in Abbildung 60 dargestellt.

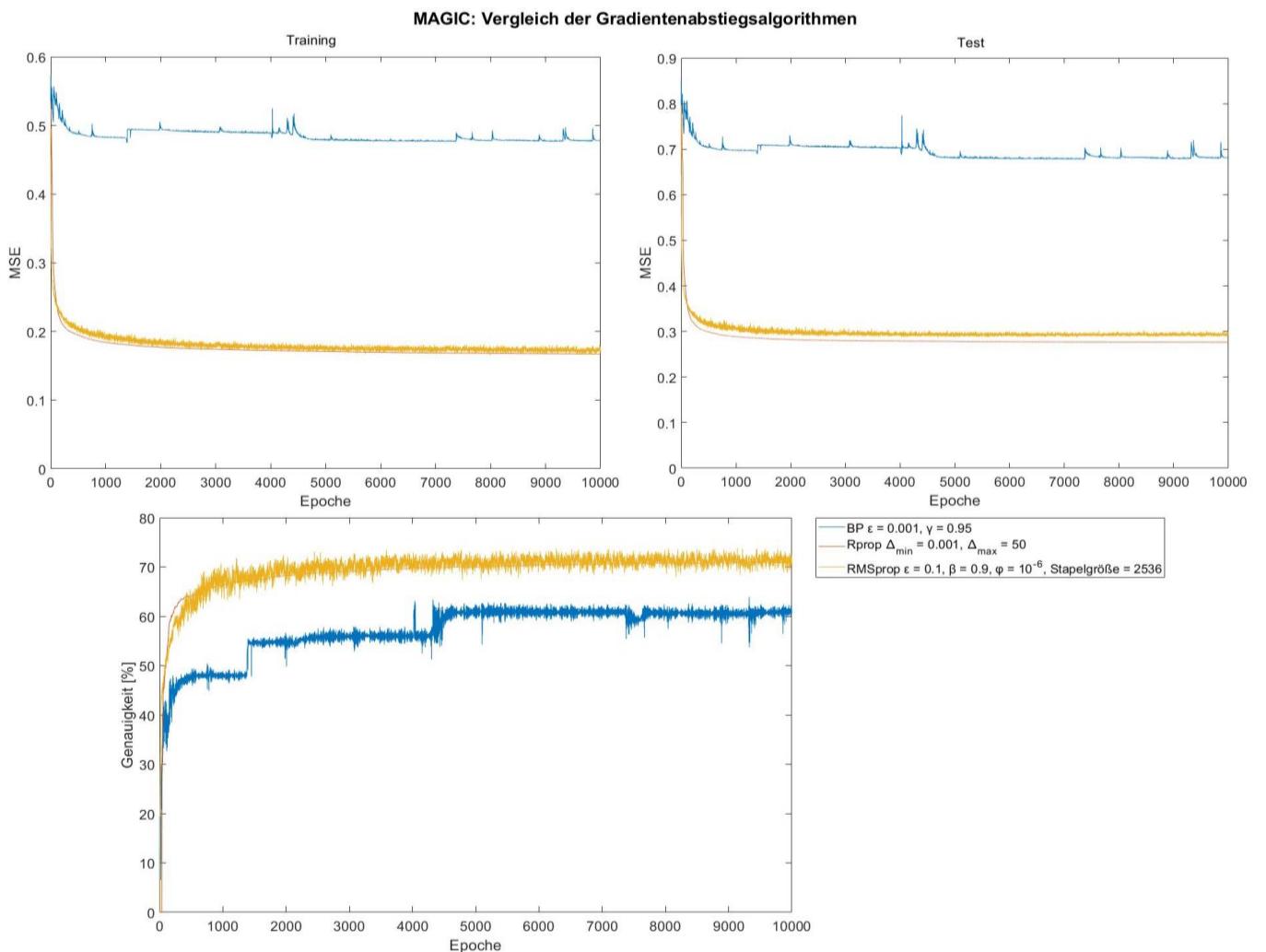


Abbildung 60: Verlauf des MSE der Trainings-(oben links) und der Testbeispiele (oben rechts) sowie der Genauigkeit (unten) des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop.

In Abbildung 60 ist zu erkennen, dass RMSprop und Rprop effizient innerhalb weniger Epochen einen hohen Genauigkeitswert erreichen. Obwohl mittels Rprop niedrigere MSE-Werte der Trainings- und Testbeispiele erreicht werden, erreicht der Gradientenabstieg mittels RMSprop höhere Genauigkeitswerte als Rprop. Mittels BP wird der schlechteste Lernerfolg des künstlichen neuronalen Netzwerkes unter Verwendung des MAGIC Gamma Telescope-Datensatzes ermittelt. Obwohl es sich bei BP um einen Stapel-Gradientenabstieg handelt, kommt es zu Oszillationen der Verläufe. Dies deutet darauf hin, dass BP Schwierigkeiten hat durch den Hyperraum der Kostenfunktion zu einem lokalen Minimum zu gelangen. Nichtsdestotrotz erreicht man innerhalb von 10000 Epochen mit BP den niedrigsten Genauigkeitswert und die höchsten MSE-Werte für Trainings- und Testbeispiele aller Gradientenabstiegsalgorithmen. RMSprop hingegen erreicht den höchsten Genauigkeitswert. In Abbildung 61 wird die erreichte Genauigkeit nach 10000 Epochen, die durchschnittliche Genauigkeit

## 4. Experimentelle Untersuchungen

der letzten 1000 Epochen und die maximal erreichte Genauigkeit innerhalb der 10000 Epochen des jeweiligen Gradientenabstiegsalgorithmus übersichtlich dargestellt.

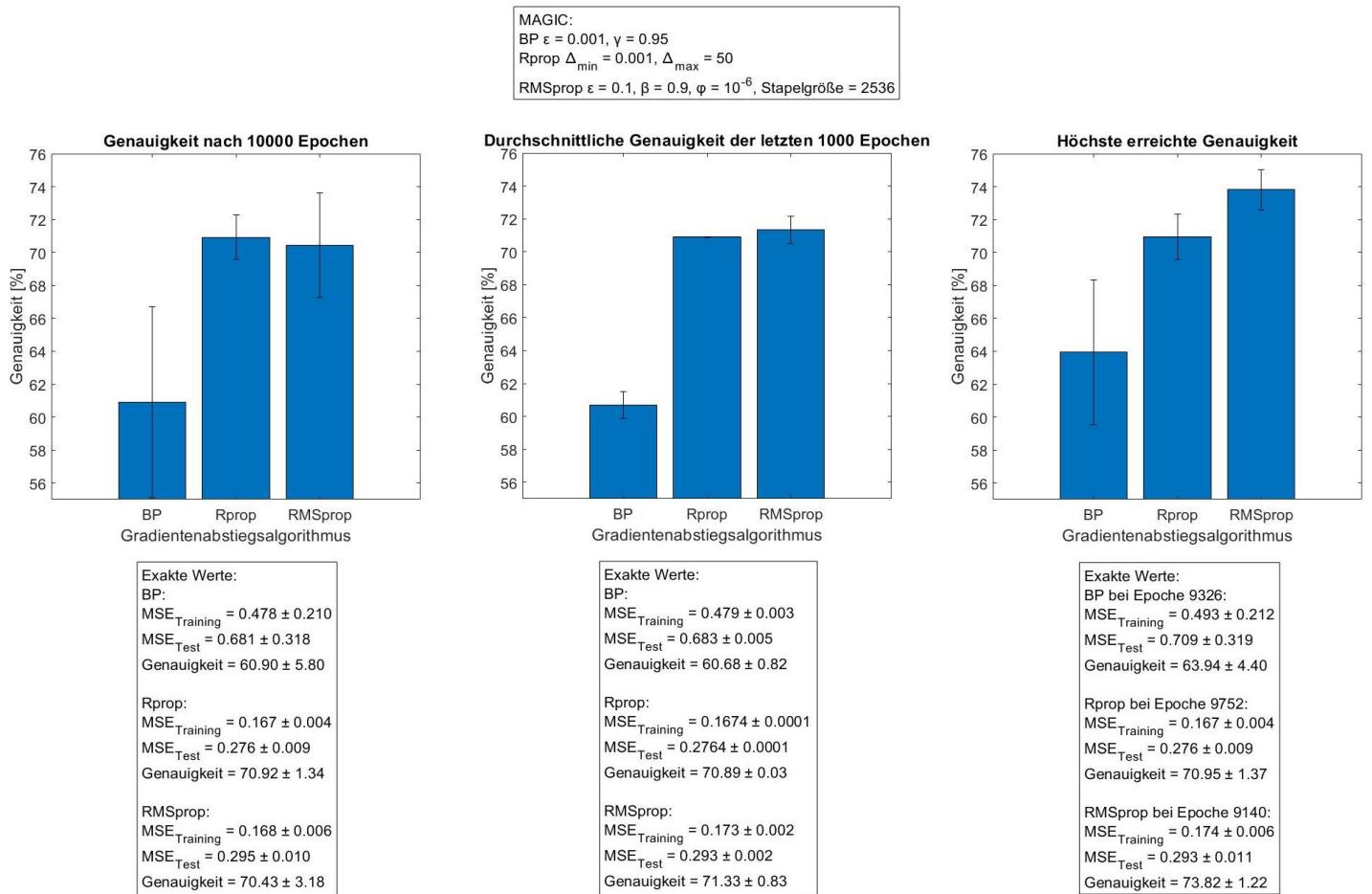


Abbildung 61: Übersicht der nach 10000 Epochen erreichten Genauigkeit (links), der durchschnittlichen Genauigkeit der letzten 1000 Epochen (mittig) und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen (rechts) des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes von BP, Rprop und RMSprop.

## 4.10. Zusammenfassung der experimentellen Untersuchungen

Das Verhalten des Gradientenabstiegsalgorithmus RMSprop in Bezug auf seine Parameter wurde anhand von insgesamt 9 Datensätzen untersucht. Zusätzlich wurde der Lernerfolg von RMSprop mit dem Lernerfolg von BP und Rprop verglichen. Die ermittelten Ergebnisse dessen wurden in Tabelle 12 übersichtlich aufgelistet. Dazu wurde für Klassifikationsaufgaben die ermittelte Genauigkeit nach 10000 Epochen, die durchschnittliche Genauigkeit der letzten 1000 Epochen sowie die maximal erreichte Genauigkeit innerhalb der 10000 Epochen jedes Gradientenabstiegsalgorithmus gegenübergestellt. Für Regressionsaufgaben wurde in entsprechender Weise der mittlere absolute prozentuale Fehler dargestellt. Der genannten Tabelle kann man entnehmen, dass RMSprop in 5 von

9 Datensätzen die besten Ergebnisse erzielt hat. Beschränkt man sich lediglich auf die Klassifikationsaufgaben so lässt sich erkennen, dass RMSprop in 5 von 7 Klassifikationsaufgaben den größten Lernerfolg aufwies. In Regressionsaufgaben hingegen hat RMSprop ebenfalls gute Ergebnisse erzielt, jedoch wurden diese von Rprop übertroffen. BP hat in allen Datensätzen, bis auf den QSAR Biodegradation-Datensatz, am schlechtesten abgeschnitten. Für Rprop konnte der Effekt der Überanpassung bei dem Iris-, dem QSAR Biodegradation- und dem Real Estate Valuation-Datensatz in hohem Maße wahrgenommen werden, da die maximal erreichte Genauigkeit bzw. der kleinste erreichte mittlere absolute prozentuale Fehler sehr stark von dem ermittelten Durchschnitt der letzten 1000 Epochen der jeweiligen Kenngröße abweicht. Dies ist beispielsweise in Abbildung 12, Abbildung 24 und Abbildung 36 an dem Anstieg des Verlaufs der MSE-Werte der Testbeispiele und dem gleichzeitigen Abfall der Genauigkeit bzw. dem Anstieg des mittleren absoluten prozentualen Fehlers ersichtlich. Dementsprechend ist für Rprop die maximal erreichte Genauigkeit bzw. der kleinste erreichte mittlere absolute prozentuale Fehler die wichtigste Kenngröße zur Bewertung des Lernerfolgs. Für RMSprop trat der Effekt der Überanpassung nicht bzw. nur im geringen Maße auf, jedoch können im Verlauf von RMSprop, abhängig von der gewählten Lernrate bzw. der Größe der Mini-Stapel, Oszillationen beobachtet werden. Aufgrund dessen stellt die durchschnittliche Genauigkeit bzw. der durchschnittliche mittlere absolute prozentuale Fehler der letzten 1000 Epochen die wichtigste Kenngröße für RMSprop dar. Zusätzlich lassen sich mittels der maximal erreichten Genauigkeit bzw. des kleinsten mittleren absoluten prozentualen Fehlers innerhalb der 10000 Epochen günstige Oszillationen von RMSprop ermitteln.

## 4. Experimentelle Untersuchungen

---

Tabelle 12: Zusammenfassung der Ergebnisse der experimentellen Untersuchungen. Der Gradientenabstiegsalgorithmus mit den besten Ergebnissen wurde hervorgehoben. K(X) = Klassifizierungsaufgabe mit X möglichen Klassen. R = Regressionsaufgabe. Acc<sub>10000</sub> = Genauigkeit (K) bzw. Mittlerer absoluter prozentualer Fehler (R) nach 10000 Epochen Training. Acc<sub>Mean</sub> = Durchschnittliche Genauigkeit (K) bzw. Durchschnittlicher mittlerer absoluter prozentualer Fehler (R) der letzten 1000 Epochen. Acc<sub>Max</sub> = Maximal erreichte Genauigkeit (K) bzw. kleinster erreichter mittlerer absoluter prozentualer Fehler (R) innerhalb der 10000 Epochen. Die experimentellen Bedingungen von BP, Rprop und RMSprop sind dem jeweiligen Kapitel des Datensatzes zu entnehmen.

Datensatz	Beispiele	Anzahl der Attribute	Aufgabe	BP			Rprop			RMSprop		
				Acc <sub>10000</sub> [%]	Acc <sub>Mean</sub> [%]	Acc <sub>Max</sub> [%]	Acc <sub>10000</sub> [%]	Acc <sub>Mean</sub> [%]	Acc <sub>Max</sub> [%]	Acc <sub>10000</sub> [%]	Acc <sub>Mean</sub> [%]	Acc <sub>Max</sub> [%]
Iris	150	4	K(3)	94.33 ± 3.67	94.33 ± 0	94.33 ± 3.67	93.67 ± 4.82	93.43 ± 0.28	95.67 ± 3.00	<b>96.00 ± 3.59</b>	<b>95.70 ± 0.59</b>	<b>97.33 ± 2.49</b>
Balance Scale	625	4	K(3)	92.48 ± 2.46	92.12 ± 0.18	92.80 ± 2.37	94.16 ± 3.74	94.08 ± 0.11	94.24 ± 3.78	<b>94.96 ± 1.19</b>	<b>95.23 ± 0.31</b>	<b>96.24 ± 1.60</b>
QSAR Biodegradation	1050	41	K(2)	85.43 ± 1.97	85.35 ± 0.11	85.71 ± 1.99	83.33 ± 2.24	83.37 ± 0.14	84.10 ± 2.77	<b>85.19 ± 2.07</b>	<b>85.50 ± 0.50</b>	<b>87.10 ± 2.48</b>
Car Evaluation	1725	6	K(4)	90.12 ± 6.12	90.09 ± 0.02	90.12 ± 6.13	<b>97.25 ± 1.40</b>	<b>97.32 ± 0.08</b>	<b>97.54 ± 1.32</b>	97.13 ± 1.69	97.13 ± 0.27	97.86 ± 0.95
Wine Quality red	1595	11	K(11)	49.09 ± 3.47	48.42 ± 0.37	49.47 ± 3.40	53.86 ± 2.69	53.80 ± 0.05	54.14 ± 2.06	<b>56.05 ± 2.70</b>	<b>56.14 ± 0.49</b>	<b>57.59 ± 2.13</b>
Wine Quality white	4895	11	K(11)	28.92 ± 3.55	28.55 ± 0.63	30.20 ± 2.70	<b>35.90 ± 1.72</b>	<b>35.90 ± 0.06</b>	<b>36.01 ± 1.74</b>	33.85 ± 2.70	33.88 ± 0.57	35.52 ± 3.48
MAGIC Gamma Telescope	19020	10	K(2)	60.90 ± 5.80	60.68 ± 0.82	63.94 ± 4.40	70.92 ± 1.34	70.89 ± 0.03	70.95 ± 1.37	<b>70.43 ± 3.18</b>	<b>71.33 ± 0.83</b>	<b>73.82 ± 1.22</b>
Yacht Hydrodynamics	305	6	R	1746.98 ± 466.00	1746.98 ± 0	155.04 ± 68.87	<b>22.84 ± 8.89</b>	<b>23.42 ± 0.30</b>	<b>22.81 ± 8.85</b>	25.82 ± 9.74	26.21 ± 1.57	21.94 ± 6.76
Real Estate Valuation	410	6	R	35.84 ± 3.08	35.84 ± 0	33.26 ± 2.35	<b>18.11 ± 3.19</b>	<b>18.12 ± 0.01</b>	<b>15.40 ± 2.17</b>	19.15 ± 3.01	19.28 ± 0.64	17.22 ± 2.23

## 5. Diskussion der Ergebnisse

Ziel der Bachelorarbeit war es, den Gradientenabstiegsalgorithmus RMSprop in den n++-Simulatorkern zu implementieren und anschließend mit Hilfe dessen vergleichende Untersuchungen zwischen RMSprop, Rprop und BP hinsichtlich des Lernerfolgs durchzuführen. Ferner sollte der Effekt der einzelnen Parameter von RMSprop und der Größe der Mini-Stapel, die zur Berechnung des Gradienten hinzugezogen werden, auf den Lernerfolg des jeweiligen künstlichen neuronalen Netzwerkes untersucht werden. Die Prämissen der vergleichenden Untersuchungen zwischen den Gradientenabstiegsalgorithmen war, dass BP keine Anpassung der Lernrate für individuelle Gewichte des künstlichen neuronalen Netzwerkes durchführt, wodurch BP insbesondere bei flachen Ebenen und steilen Tälern an seine Grenzen stößt, und Rprop aufgrund dessen, dass Rprop mittels eines stochastischen Gradientenabstiegs bzw. Mini-Stapel Gradientenabstiegs nicht verwendet werden kann, bei großen Datensätzen an seine Grenzen stößt. Der Gedanke dahinter war, dass Rprop, durch die vielen Sprünge des Gradienten eines stochastischen Gradientenabstiegs, die Aktualisierungswerte der einzelnen Gewichte schlecht an die Problemstellung anpassen kann, sodass eine Konvergenz für Rprop in einem Minimum der Kostenfunktion unter Verwendung eines stochastischen Gradientenabstiegs erheblich erschwert wird.

Aus der Zusammenfassung der experimentellen Untersuchungen in Tabelle 12 hat sich ergeben, dass mittels RMSprop in 5 von 9 verwendeten Datensätzen die besten Ergebnisse erzielt werden konnten. In den übrigen 4 Fällen, in denen Rprop die besten Lernerfolge erzielt hatte, wurden mittels RMSprop unter den gegebenen experimentellen Bedingungen durchaus gute Lernerfolge erreicht. Daraus resultiert, dass RMSprop die Fähigkeit besitzt die Robustheit von Rprop, durch die Anpassung der Lernrate der einzelnen Gewichte mit Hilfe der Wurzel des gleitenden Durchschnitts des quadrierten Gradienten, anzunähern oder sogar zu überbieten und bietet zudem die Möglichkeit, durch den Mini-Stapel Gradientenabstieg, künstliche neuronale Netzwerke effizient an sehr großen Datensätzen anzutrainieren. Durch die Verwendung des Mini-Stapel Gradientenabstiegs wird folglich verhindert, dass sehr große Datensätze für einen Stapel-Gradientenabstieg vollständig in den Arbeitsspeicher geladen werden müssen und verringert gleichzeitig, durch die Aktualisierung der Gewichte je Mini-Stapel, die Zeit zum Trainieren des künstlichen neuronalen Netzwerks. RMSprop zeigte in den experimentellen Untersuchungen, im Gegenteil zu Rprop, keine bzw. sehr schwache Anzeichen von Überanpassung an die Trainingsbeispiele. Dies konnte vor allem an den Gewichten der künstlichen neuronalen Netzwerke, nachdem diese 10000 Epochen lang trainiert wurden, beobachtet werden. Während Rprop am Ende der 10000 Epochen sehr hohe Gewichtswerte aufzeigte, konnten bei

## 5. Diskussion der Ergebnisse

RMSprop keine hohen Gewichtswerte festgestellt werden. Die Überanpassung an die Trainingsbeispiele konnte man ebenfalls an dem Verlauf der MSE-Werte der Testbeispiele erkennen. Hier zeigte RMSprop in den experimentellen Untersuchungen keinen Anstieg der MSE-Werte der Testbeispiele unter gleichzeitigem Abfall der Genauigkeit des künstlichen Netzwerkes auf.

Aus den experimentellen Untersuchungen des Effekts der Lernrate auf den Lernerfolg von RMSprop konnte festgestellt werden, dass eine Lernrate ab einem Wert von etwa 0.3 einen negativen Effekt auf den Lernerfolg des künstlichen neuronalen Netzwerkes ausübt. Dies zeigte sich vor allem durch die Steigerung der Intensität der Oszillationen in den Verläufen der MSE-Werte der Trainings- sowie der Testbeispiele und dem Verlauf der Genauigkeit des künstlichen neuronalen Netzwerkes, wodurch im Mittel der Lernerfolg des künstlichen neuronalen Netzwerkes sank. Der Grund dafür liegt womöglich darin, dass die Lernrate der einzelnen Gewichte, und damit die Schrittweite des Gradientenabstiegs, ab einem Wert von etwa 0.3 schlechter durch die Wurzel des gleitenden Durchschnitts des quadrierten Gradienten angepasst werden kann, sodass eine Konvergenz in einem Minimum deutlich erschwert wird. Gleichzeitig konnte beobachtet werden, dass der von G. Hinton empfohlene Wert der Lernrate von 0.001 einen nahezu stabilen Gradientenabstieg gewährleistet, in dem das Ausmaß der Oszillationen weitgehend minimiert wird, jedoch erreichte der Lernerfolg des künstlichen neuronalen Netzwerkes unter dieser Lernrate dadurch innerhalb der 10000 Epochen oftmals keinen Plateauwert. Dies konnte zum Beispiel in Abbildung 20, Abbildung 26, Abbildung 32, Abbildung 38, Abbildung 44 und Abbildung 50 beobachtet werden. In einer experimentellen Untersuchung ermöglichte ein geringer Wert der Lernrate jedoch einen höheren Lernerfolg des künstlichen neuronalen Netzwerkes. Dies konnte in Abbildung 14 beobachtet werden. Folglich könnten kleinere Lernraten dafür sorgen, dass bessere lokale Minima im Hyperraum der Kostenfunktion erreicht werden können als bei größeren Lernraten. Gleichzeitig könnten jedoch kleinere Minima die Möglichkeit bieten, aufgrund der kleinen Schrittweite, in schlechteren Minima zu konvergieren, was beispielsweise in Abbildung 56 beobachtet werden konnte. Dementsprechend muss der Benutzer von RMSprop eine Abwägung treffen, ob man das künstliche neuronale Netzwerk über viele Epochen hinweg mit einer kleinen Lernrate trainiert, um potenziell ein günstiges lokales Minimum zu erreichen, oder ob man mit einer höheren Lernrate das Training des künstlichen neuronalen Netzwerkes weitgehend verkürzt und dabei eventuell auf das beste Lernergebnis verzichtet. Dazu müssten weitere experimentelle Untersuchungen, in dem die künstlichen neuronalen Netzwerke über die 10000 Epochen hinaus mit einer kleinen Lernrate trainiert werden, durchgeführt werden und untersuchen, ob die geringen Lernraten zu besseren Lernerfolgen führen oder ob dieselben Lernerfolge erzielt werden wie mit höheren Lernraten.

Aus den experimentellen Untersuchungen des Effekts des Vergessensfaktors auf den Lernerfolg von RMSprop konnte kein eindeutiger Trend erkannt werden. In einem Teil der Untersuchungen konnte festgestellt werden, dass mit einem steigenden Vergessensfaktor bessere Lernerfolge der künstlichen neuronalen Netzwerke innerhalb der 10000 Epochen erzielt werden konnten. Dies konnte man beispielsweise in Abbildung 15, Abbildung 33, Abbildung 45 und Abbildung 51 gut feststellen. In einer Untersuchung konnte man innerhalb der 10000 Epochen sogar das Gegenteil, also dass mit einem steigenden Vergessensfaktor schlechtere Lernerfolge erzielt wurden, feststellen. Dies kann man in Abbildung 27 erkennen. In den restlichen Untersuchungen hatte der Vergessensfaktor keinen oder nur einen sehr geringen Einfluss auf den Lernerfolg des künstlichen neuronalen Netzwerkes. Beobachten kann man diese Fälle in Abbildung 9, Abbildung 39 und Abbildung 57. Was man jedoch in fast allen experimentellen Untersuchungen des Effekts des Vergessensfaktors erkennen konnte ist, dass die einzelnen Verläufe der MSE-Werte der Trainings- bzw. der Testbeispiele und der Genauigkeitswerte anfangs dem selben Verlauf folgen bis an einem bestimmten Punkt des Trainings, abhängig von dem jeweiligen Vergessensfaktor, es zu einer Spaltung in mehrere Verläufe führt. In Anbetracht dessen, dass bei einem kleinen Vergessensfaktor Gradienten vorheriger Mini-Stapel weniger Einfluss auf den gleitenden Durchschnitt des quadrierten Gradienten haben und bei einem großen Vergessensfaktor der Einfluss der Gradienten vorheriger Mini-Stapel auf den gleitenden Durchschnitt des quadrierten Gradienten größer ist, kann man vermuten, dass an den beschriebenen Stellen des Trainings eine unterschiedliche Route entlang des Hyperraums der Kostenfunktion eingeschlagen wird. Diesen Effekt kann man beispielsweise in Abbildung 21, Abbildung 27, Abbildung 45, Abbildung 51 und Abbildung 57 gut wahrnehmen. Hierbei muss beachtet werden, dass oftmals ein hoher Vergessensfaktor eine Spaltung des Verlaufs zu niedrigeren MSE-Werten der Trainings- und Testbeispiele und zu höheren Genauigkeitswerten führt. Nichtsdestotrotz konvergierten die Verläufe der einzelnen Vergessensfaktoren bei einzelnen experimentellen Untersuchungen innerhalb der 10000 Epochen, wie beispielsweise bei Abbildung 9 und Abbildung 57, zu einem Verlauf, was wiederum den Anschein macht, dass der Vergessensfaktor keinen Einfluss auf den Lernerfolg, hinsichtlich der Findung besserer lokaler Minima, hat. Aus den experimentellen Untersuchungen lässt sich demnach vermuten, dass die unterschiedlichen Routen entlang des Hyperraums in diesen Fällen zu dem gleichen oder ähnlichen Minimum der Kostenfunktion führen. Letztlich müssen weitere experimentelle Untersuchungen durchgeführt werden, die untersuchen wie sich die Verläufe der einzelnen Vergessensfaktoren der jeweiligen Datensätze, bei denen innerhalb der 10000 Epochen die Verläufe nicht zu einem Plateauwert konvergierten, über die 10000 Epochen hinaus verhalten. Es ist zu beweisen, dass in diesen Fällen die Routen zu unterschiedlichen Minima im Hyperraum der Kostenfunktion und somit zu starken Unterschieden im Lernerfolg des künstlichen neuronalen Netzwerkes führen, sodass eindeutige Schlüsse des Effekt des Vergessensfaktors auf den Lernerfolg gezogen werden können.

## 5. Diskussion der Ergebnisse

---

Aus den experimentellen Untersuchungen des Effekts des Unschärfeefaktors auf den Lernerfolg von RMSprop konnte man beobachten, dass ein Unschärfeefaktor ab dem Wert von etwa  $10^{-3}$  sich oftmals negativ auf den Lernerfolg des künstlichen neuronalen Netzwerkes auswirkte. Folglich sanken die Werte der Genauigkeit und stiegen die MSE-Werte für Trainings- und Testbeispiele mit einem steigenden Unschärfeefaktor im Mittel zu niedrigeren Plateauwerten der Genauigkeitswerte bzw. zu höheren Plateauwerten der MSE-Werte. Dies kann man beispielsweise in Abbildung 10, Abbildung 28, Abbildung 34, Abbildung 40 und Abbildung 46 erkennen. In zwei experimentellen Untersuchungen sorgte ein steigender Unschärfeefaktor zu einem gewissen Maß dafür, dass der Lernerfolg des künstlichen neuronalen Netzwerkes innerhalb der 10000 Epochen im Vergleich zu anderen Unschärfeefaktoren verbessert wird. Beobachten kann man diese Fälle in Abbildung 22 und Abbildung 52. In zwei experimentellen Untersuchungen, die man in Abbildung 16 und Abbildung 58 erkennen kann, hatte der Unschärfeefaktor am Ende der 10000 Epochen sogar keinen eindeutigen Einfluss auf den Lernerfolg des künstlichen neuronalen Netzwerkes. Rückblickend wurde der Unschärfeefaktor in RMSprop dazu eingeführt, um zu verhindern, dass man die Lernrate nicht durch einen ungültigen Wert von 0 des gleitenden Durchschnitts des quadrierten Gradienten teilt. Durch die Addition eines zunehmend großen Unschärfeefaktors mit dem gleitenden Durchschnitt des quadrierten Gradienten wird gemäß (17) der durch die Lernrate bestimmte Bruchteil des Gradienten, um welches das jeweilige Gewicht aktualisiert wird, insgesamt verkleinert. Dies sollte zur Folge haben, dass der Gradientenabstieg mit zunehmend großem Unschärfeefaktor im Allgemeinen verlangsamt werden sollte. Diese Vermutung wird beispielsweise in den experimentellen Untersuchungen oftmals durch die mit steigendem Unschärfeefaktor verstärkte Verschiebung der einzelnen Verläufe hin zu höheren Epochen belegt. Dies kann man beispielsweise in Abbildung 10, Abbildung 28 und Abbildung 52 erkennen. Auffällig ist hierbei, dass dieser Effekt nur bei den Klassifikationsproblemen verstärkt auftrat. Der Grund hierfür könnte sein, dass bei Regressionsproblemen die Ergebnismenge nicht auf einen Zahlenwert zwischen 0 und 1 beschränkt ist, wodurch der Einfluss des Unschärfeefaktors unter Umständen vernachlässigbar klein wird. Dies wird zum Beispiel durch die teilweise identischen Verläufe der Genauigkeitswerte und der MSE-Werte der Testbeispiele in Abbildung 16 und Abbildung 21 belegt. Bei dem Real Estate Valuation-Datensatz, der die größte Ergebnismenge aller Datensätze enthielt, erscheint es sogar, dass der Unschärfeefaktor einen beschleunigenden Effekt auf den Gradientenabstieg ausübte. Da die Schrittweite des Gradientenabstiegs mit zunehmendem Unschärfeefaktor verkleinert wird, kann zusätzlich die Möglichkeit bestehen, dass andere Routen entlang des Hyperraums der Kostenfunktion in Folge des Gradientenabstiegs mittels RMSprop genutzt werden. Somit können unter Umständen abhängig vom Unschärfeefaktor bessere lokale Minima der Kostenfunktion erreicht werden. Gleichzeitig besteht ebenfalls die Gefahr, dass aufgrund der kleineren Schrittweite des Gradientenabstiegs, es zu einer frühzeitigen Konvergenz in vergleichsweise

schlechteren Minima der Kostenfunktion führen kann. Dies könnte vermutlich die voneinander abweichenden Plateauwerte der Verläufe der unterschiedlichen Unschärfe faktoren in Abbildung 22 erklären. Konkludierend sollte man meiner Meinung nach den Unschärfe faktor möglichst klein halten und keine Änderung dessen vornehmen. Wenn man kleinere Schrittweiten in Folge des Gradientenabstiegs mittels RMSprop verwenden möchte, sollte dies anhand der Lernrate variiert werden.

Die experimentellen Untersuchungen hinsichtlich des Effekts der Größe der Mini-Stapel haben ergeben, dass mittels einer Größe des Mini-Stapels von etwa 5-15 % der gesamten Trainingsbeispiele innerhalb der 10000 Epochen der größte Lernerfolg erreicht werden kann. Dies kann man beispielsweise in Abbildung 29, Abbildung 35, Abbildung 47, Abbildung 53 und Abbildung 59 beobachten. Durch die Größe des Mini-Stapels von RMSprop wird die Art des Gradientenabstiegs reguliert. Verwendet man Mini-Stapel mit der Größe von 1, gleicht RMSprop einem rein stochastischen Gradientenabstieg. Dies hat zur Folge, dass die Stärke der Oszillationen im Verlauf des Genauigkeitswerts sowie der MSE-Werte der Trainings- und Testbeispiele während des Gradientenabstiegs, aufgrund der hohen Anzahl an Gewichtsaktualisierungen, mit sinkender Größe des Mini-Stapels zunimmt. Dieser Effekt konnte in allen experimentellen Untersuchungen beobachtet werden. Wird eine zu geringe Größe des Mini-Stapels gewählt, kann unter Umständen, aufgrund der hohen Anzahl an Aktualisierungen gemäß der zufällig gewählten Trainingsbeispiele und damit den häufigen Sprüngen zu unterschiedlichen Stellen des Hyperraums, eine Konvergenz in einem Minimum deutlich erschwert werden. Dadurch werden im Mittel niedrigere Plateauwerte der Genauigkeitswerte sowie der MSE-Werte der Trainings- und Testbeispiele erreicht. Dies kann beispielsweise in Abbildung 11, Abbildung 29, Abbildung 35, Abbildung 41, Abbildung 47 und Abbildung 53 erkannt werden. Verwendet man hingegen nur einen einzigen Mini-Stapel mit den gesamten Trainingsbeispielen, gleicht RMSprop wiederum einem Stapel-Gradientenabstieg. Dadurch, dass man zur Berechnung des Gradienten alle Trainingsbeispiele berücksichtigt und dementsprechend nur eine Gewichtsaktualisierung in einer Epoche durchführt, ist die Stärke der Oszillationen im Verlauf der Genauigkeitswerte sowie der MSE-Werte der Trainings- und Testbeispiele in diesem Fall minimal. Folglich sinkt die Stärke der Oszillationen der Verläufe mit steigender Größe des Mini-Stapels, was ebenfalls in allen experimentellen Untersuchungen beobachtet werden konnte. Gleichzeitig wird dadurch jedoch der Gradientenabstieg, aufgrund der sinkenden Anzahl an Gewichtsaktualisierungen innerhalb einer Epoche, deutlich verlangsamt und erreicht oftmals vergleichsweise schlechtere Minima, da weniger Sprünge im Hyperraum der Kostenfunktion stattfinden. Beobachten lässt sich dies in Abbildung 17, Abbildung 23, Abbildung 35, Abbildung 41, Abbildung 47, Abbildung 53 und vor allem Abbildung 59. Beim MAGIC Gamma Telescope-Datensatz, der die meisten Trainingsbeispiele

## **6. Fazit**

---

aller experimentellen Untersuchungen aufweist, hat RMSprop sogar mit einer Stapelgröße von etwa 2 % der gesamten Trainingsbeispiele den größten Lernerfolg erzielt und gleichzeitig einen vergleichsweise stabilen Verlauf des Gradientenabstiegs gewährleistet. Dies wiederum zeigt, welches Potenzial RMSprop unter Verwendung von sehr großen Datensätzen mit sich bringt.

## **6. Fazit**

Der Gradientenabstiegsalgorithmus RMSprop wurde in den n++-Simulatorkern integriert und erfolgreich an ausgewählten Problemstellungen verwendet. An diesen Problemstellungen konnte die Effizienz von RMSprop hinsichtlich des Lernerfolgs untersucht und mit den Gradientenabstiegsalgorithmen BP und Rprop verglichen werden. Dabei hat sich herausgestellt, dass RMSprop in den meisten Problemstellungen den besten Lernerfolg erzielen konnte. Ferner wurde der Effekt der einzelnen Parameter und die Größe der verwendeten Mini-Stapel auf den Lernerfolg untersucht. Hierbei konnte festgestellt werden, dass der Lernerfolg von RMSprop von den jeweiligen verwendeten Parametern bzw. der Größe der Mini-Stapel abhängig ist. Bevor man sich jedoch auf die Suche nach den günstigsten experimentellen Bedingungen begibt, um den erreichten Lernerfolg zu maximieren, sollte man jedoch den Rahmen der Suche eingrenzen. Hierzu konnten mit Hilfe dieser Bachelorarbeit wichtige Erkenntnisse gewonnen werden. Für zukünftige vergleichende Untersuchungen von RMSprop wären Problemstellungen mit weitaus mehr Attributen und Trainingsbeispielen höchst interessant. Außerdem könnte das Verhalten von RMSprop, unter ausgewählten experimentellen Bedingungen, über 10000 Epochen hinaus untersucht werden.

## 7. Tabellenverzeichnis

Tabelle 1: Aktivierungsfunktionen mit den dazugehörigen Formeln, Ableitungen und Wertebereichen .....	7
Tabelle 2: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Iris-Datensatzes .....	30
Tabelle 3: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Yacht Hydrodynamics-Datensatzes .....	39
Tabelle 4: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Real Estate Valuation-Datensatzes .....	48
Tabelle 5: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Balance Scale-Datensatzes .....	57
Tabelle 6: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des QSAR Biodegradation-Datensatzes .....	66
Tabelle 7: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Car Evaluation-Datensatzes .....	76
Tabelle 8: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute nach Anpassung des Car Evaluation-Datensatzes .....	76
Tabelle 9: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Wine Quality Red-Datensatzes .....	85
Tabelle 10: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des Wine Quality White-Datensatzes .....	96
Tabelle 11: Übersichtliche Darstellung der Informationen und Wertebereiche der Attribute des MAGIC Gamma Telescope-Datensatzes .....	105
Tabelle 12: Zusammenfassung der Ergebnisse der experimentellen Untersuchungen .....	116

## 8. Formelverzeichnis

(1) Berechnung der gewichteten Summe anhand der Eingabewerte, den zugehörigen Gewichten und dem Schwellenwert mit dem zugehörigen Gewicht .....	6
(2) Berechnung des Ausgabewerts des einfachen Perzeptrons anhand der gewichteten Summe und der Aktivierungsfunktion .....	7
(3) Daumenregel zur Ermittlung der Anzahl an verborgenen Neuronen in der verborgenen Schicht aus der Anzahl der Neuronen in der Eingabeschicht und der Anzahl der Neuronen in der Ausgabeschicht .....	10
(4) Berechnung der Eingabe des $k$ -ten Neurons der $l$ -ten Schicht.....	11
(5) Berechnung der Aktivierung des $k$ -ten Neurons der $l$ -ten Schicht.....	11
(6) Berechnung der mittleren quadratischen Abweichung aller Ausgabeneuronen .....	12
(7) Darstellung der mittleren quadratischen Abweichung aller Ausgabeneuronen in Abhängigkeit zu den Aktivierungen der letzten verborgenen Schicht.....	12
(8) Berechnung des Gradienten von $C$ in $w_{k,j}^{(l-1)}$ aus dem Produkt des Gradienten von $C$ in $a_k^{(l)}$ und dem Gradienten von $a_k^{(l)}$ in $w_{k,j}^{(l-1)}$ .....	13
(9) Berechnung des Gradienten von $C$ in $a_k^{(l)}$ .....	13
(10) Berechnung des Gradienten von $a_k^{(l)}$ in $z_k^{(l)}$ .....	13
(11) Berechnung des Gradienten von $z_k^{(l)}$ in $w_{k,j}^{(l-1)}$ .....	13
(12) BP. Berechnung des neuen Gewichtswertes für die gewichtete Kante des $j$ -ten Neurons zum $k$ -ten Neuron.....	18
(13) Schwungbasierte BP. Berechnung des neuen Gewichtswertes für die gewichtete Kante des $j$ -ten Neurons zum $k$ -ten Neuron.....	19

(14) Anpassungen des individuellen Aktualisierungswertes $\Delta_{k,j}^{(t)}$ der gewichteten Kante des $j$ -ten Neurons zum $k$ -ten Neuron.....	20
(15) Rprop. Berechnung des neuen Gewichtswertes für die gewichtete Kante des $j$ -ten Neurons zum $k$ -ten Neuron .....	20
(16) Berechnung des gleitenden Durchschnitts des quadrierten Gradienten von $C$ in $w_{k,j}$ .....	21
(17) RMSprop. Berechnung des neuen Gewichtswertes für die gewichtete Kante des $j$ -ten Neurons zum $k$ -ten Neuron.....	22
(18) Normalisierung des Eingabewertes mittels des kleinstmöglichen Eingabewertes des größtmöglichen Eingabewertes .....	27
(19) Berechnung der mittleren quadratischen Abweichung innerhalb einer Epoche .....	28
(20) Berechnung der totalen Quadratsumme der Ausgabeneuronen .....	28
(21) Berechnung des mittleren absoluten prozentualen Fehlers des Ausgabeneurons .....	28

## 9. Abbildungsverzeichnis

Abbildung 1: Vereinfachte Darstellung eines einfachen Perzeptrons .....	6
Abbildung 2: Übersicht ausgewählter Aktivierungsfunktionen .....	8
Abbildung 3: Aufbau eines künstlichen neuronalen Netzwerkes.....	9
Abbildung 4: Räumliche Darstellung einer von zwei Gewichten abhängigen Kostenfunktion.....	14
Abbildung 5: Quellcode-Ausschnitt der set_update_f Funktion des n++-Simulatorkerns .....	24
Abbildung 6: Quellcode-Ausschnitt der rmsprop_init Funktion des n++-Simulatorkerns.....	25
Abbildung 7: Quellcode-Ausschnitt der rmsprop_update Funktion des n++-Simulatorkerns.....	26
Abbildung 8: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Iris-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate.....	31
Abbildung 9: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Iris-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors .....	33
Abbildung 10: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Iris-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfe faktors .....	34
Abbildung 11: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Iris-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels .....	35
Abbildung 12: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Iris-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop.....	37
Abbildung 13: Übersicht der nach 10000 Epochen erreichten Genauigkeit, der durchschnittlichen Genauigkeit der letzten 1000 Epochen und der maximal erreichten Genauigkeit innerhalb der 10000	

## 9. Abbildungsverzeichnis

Epochen des künstlichen neuronalen Netzwerkes des Iris-Datensatzes von BP, Rprop und RMSprop .....	38
Abbildung 14: Verlauf des MSE der Trainings- und der Testbeispiele sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele des Yacht Hydrodynamics-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate .....	40
Abbildung 15: Verlauf des MSE der Trainings- und der Testbeispiele sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele des Yacht Hydrodynamics-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors .....	41
Abbildung 16: Verlauf des MSE der Trainings- und der Testbeispiele sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele des Yacht Hydrodynamics-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors .....	43
Abbildung 17: Verlauf des MSE der Trainings- und der Testbeispiele sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele des Yacht Hydrodynamics-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels .....	44
Abbildung 18: Verlauf des MSE der Trainings- und der Testbeispiele sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele des Yacht Hydrodynamics-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop .....	46
Abbildung 19 Übersicht des nach 10000 Epochen erreichten mittleren absoluten prozentualen Fehlers, des durchschnittlichen mittleren absoluten prozentualen Fehlers der letzten 1000 Epochen und des kleinsten erreichten mittleren absoluten prozentualen Fehlers innerhalb der 10000 Epochen des Yacht Hydrodynamics-Datensatzes von BP, Rprop und RMSprop .....	47
Abbildung 20: Verlauf des MSE der Trainings- und der Testbeispiele sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele des Real Estate Valuation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate .....	49
Abbildung 21: Verlauf des MSE der Trainings- und der Testbeispiele sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele des Real Estate Valuation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors .....	50
Abbildung 22: Verlauf des MSE der Trainings- und der Testbeispiele sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele des Real Estate Valuation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors .....	52

## **9. Abbildungsverzeichnis**

---

Abbildung 23: Verlauf des MSE der Trainings- und der Testbeispiele sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele des Real Estate Valuation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels .....	53
Abbildung 24: Verlauf des MSE der Trainings- und der Testbeispiele sowie des mittleren absoluten prozentualen Fehlers der Testbeispiele des Real Estate Valuation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop. ....	55
Abbildung 25: Übersicht des nach 10000 Epochen erreichten mittleren absoluten prozentualen Fehlers, des durchschnittlichen mittleren absoluten prozentualen Fehlers der letzten 1000 Epochen und des kleinsten erreichten mittleren absoluten prozentualen Fehlers innerhalb der 10000 Epochen des Real Estate Valuation-Datensatzes von BP, Rprop und RMSprop. ....	56
Abbildung 26: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate .....	58
Abbildung 27: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors .....	59
Abbildung 28: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors .....	61
Abbildung 29: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels .....	62
Abbildung 30: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop .....	64
Abbildung 31: Übersicht der nach 10000 Epochen erreichten Genauigkeit, der durchschnittlichen Genauigkeit der letzten 1000 Epochen und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen des künstlichen neuronalen Netzwerkes des Balance Scale-Datensatzes von BP, Rprop und RMSprop. ....	65

## 9. Abbildungsverzeichnis

---

Abbildung 32: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate .....	68
Abbildung 33: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors .....	69
Abbildung 34: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors .....	71
Abbildung 35: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels .....	72
Abbildung 36: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop .....	74
Abbildung 37: Übersicht der nach 10000 Epochen erreichten Genauigkeit, der durchschnittlichen Genauigkeit der letzten 1000 Epochen und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen des künstlichen neuronalen Netzwerkes des QSAR Biodegradation-Datensatzes von BP, Rprop und RMSprop .....	75
Abbildung 38: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate .....	78
Abbildung 39: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors .....	79
Abbildung 40: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors .....	80

## 9. Abbildungsverzeichnis

---

Abbildung 41: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels .....	82
Abbildung 42: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop .....	83
Abbildung 43: Übersicht der nach 10000 Epochen erreichten Genauigkeit, der durchschnittlichen Genauigkeit der letzten 1000 Epochen und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen des künstlichen neuronalen Netzwerkes des Car Evaluation-Datensatzes von BP, Rprop und RMSprop .....	84
Abbildung 44: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate.....	87
Abbildung 45: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors .....	89
Abbildung 46: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors .....	90
Abbildung 47: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels .....	92
Abbildung 48: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop .....	94
Abbildung 49: Übersicht der nach 10000 Epochen erreichten Genauigkeit, der durchschnittlichen Genauigkeit der letzten 1000 Epochen und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen des künstlichen neuronalen Netzwerkes des Wine Quality Red-Datensatzes von BP, Rprop und RMSprop .....	95

---

## 9. Abbildungsverzeichnis

Abbildung 50: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate .....	97
Abbildung 51: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors .....	99
Abbildung 52: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors .....	100
Abbildung 53: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels .....	102
Abbildung 54: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop .....	103
Abbildung 55: Übersicht der nach 10000 Epochen erreichten Genauigkeit, der durchschnittlichen Genauigkeit der letzten 1000 Epochen und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen des künstlichen neuronalen Netzwerkes des Wine Quality White-Datensatzes von BP, Rprop und RMSprop .....	104
Abbildung 56: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Lernrate .....	107
Abbildung 57: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Vergessensfaktors .....	108
Abbildung 58: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation des Unschärfeefaktors .....	110

## 9. Abbildungsverzeichnis

---

Abbildung 59: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels RMSprop unter Variation der Größe des Mini-Stapels .....	111
Abbildung 60: Verlauf des MSE der Trainings- und der Testbeispiele sowie der Genauigkeit des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes in Folge des Gradientenabstiegsverfahrens mittels BP, Rprop und RMSprop .....	113
Abbildung 61: Übersicht der nach 10000 Epochen erreichten Genauigkeit, der durchschnittlichen Genauigkeit der letzten 1000 Epochen und der maximal erreichten Genauigkeit innerhalb der 10000 Epochen des künstlichen neuronalen Netzwerkes des MAGIC Gamma Telescope-Datensatzes von BP, Rprop und RMSprop.....	114

## 10. Literaturverzeichnis

- [1] F. Rosenblatt, „The perceptron: A probabilistic model for information storage and organization in the brain,“ *Psychological review*, Bd. 65, Nr. 6, pp. 386-408, 1 November 1958.
- [2] L. J. Lancashire, C. Lemetre und G. R. Ball, „An introduction to artificial neural networks in bioinformatics--application to complex microarray and mass spectrometry datasets in cancer studies,“ *Briefings in Bioinformatics*, Bd. 10, Nr. 3, pp. 315-329, 10 Mai 2009.
- [3] D. Zhang, Q. Jiang und X. Li, „Application of Neural Networks in Financial Data Mining,“ *Conference: International Conference on Computational Intelligence*, 17-19 Dezember 2004.
- [4] Y. LeCun, Y. Bengio und G. Hinton, „Deep learning,“ *Nature*, Bd. 521, Nr. 7553, pp. 436-444, 28 Mai 2015.
- [5] S. Ruder, „An overview of gradient descent optimization,“ *CoRR abs/1609.04747*, 15 Juni 2017.
- [6] C. Florescu und C. Igel, „Resilient Backpropagation (Rprop) for Batch-learning in TensorFlow,“ *6th International Conference on Learning Representations*, 12 Februar 2018.
- [7] A. Dey, „Machine Learning Algorithms: A Review,“ *International Journal of Computer Science and Information Technologies*, Bd. 7, Nr. 3, pp. 1174-1179, 3 Mai 2016.

- [8] Y. Bengio, A. Courville und P. Vincent, „Representation Learning: A Review and New Perspectives,“ *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Bd. 35, Nr. 8, pp. 1798-1828, 7 März 2013.
- [9] F. Musumeci, C. Rottandi, A. Nag, I. Macaluso, D. Zibar, M. Ruffini und M. Tornatore, „An Overview on Application of Machine Learning,“ *IEEE Communications Surveys & Tutorials*, Bd. 21, Nr. 2, pp. 1383-1408, 8 November 2018.
- [10] S. Otte, „Künstliche neuronale Netze - Das Perzeptron,“ [Online]. Available: <https://www.cs.hs-rm.de/~panitz/prog3WS08/perceptron.pdf>. [Zugriff am 27 August 2020].
- [11] M. Nielsen, „Using neural nets to recognize handwritten digits,“ [Online]. Available: <http://neuralnetworksanddeeplearning.com/chap1.html>. [Zugriff am 27 August 2020].
- [12] Z. Yanling, D. Bimin und W. Zhanrong, „Analysis and study of perceptron to solve XOR problem,“ *The 2nd International Workshop on Autonomous Decentralized System*, 7 November 2002.
- [13] J. Heaton, „Heaton Research - The Number of Hidden Layers,“ [Online]. Available: <https://www.heatonresearch.com/2017/06/01/hidden-layers.html>. [Zugriff am 10 September 2020].
- [14] „Feed-Forward Netze,“ [Online]. Available: <https://swl.htwsaar.de/lehre/ws17/ds/slides/2017-vl-ds-kap6-2-neuronale-netze.pdf>. [Zugriff am 11 September 2020].

- [15] S. F. Crone, „Training artificial neural networks for time series prediction using asymmetric cost functions,“ *Proceedings of the 9th International Conference on Neural Information Processing*, 18-22 November 2002.
- [16] J. Schmitz, „Grundlagen Neuronaler Netze,“ [Online]. Available: [https://www5.in.tum.de/lehre/seminare/datamining/ss17/paper\\_pres/12\\_nn\\_basics/Proseminar\\_Datamining\\_Julian\\_Schmitz.pdf](https://www5.in.tum.de/lehre/seminare/datamining/ss17/paper_pres/12_nn_basics/Proseminar_Datamining_Julian_Schmitz.pdf). [Zugriff am 20 September 2020].
- [17] M. Riedmiller und H. Braun, „A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm,“ *IEEE International Conference on Neural Networks*, 28 März 1993.
- [18] J. Chen, „An updated overview of recent gradient descent algorithms,“ [Online]. Available: <https://johnchenresearch.github.io/demon/>. [Zugriff am 18 September 2020].
- [19] G. Hinton, „Overview of mini-batch gradient descent,“ [Online]. Available: [https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture\\_slides\\_lec6.pdf](https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf). [Zugriff am 20 September 2020].
- [20] „UCI Machine Learning Repository,“ [Online]. Available: <https://archive.ics.uci.edu/ml/index.php>. [Zugriff am 28 September 2020].
- [21] M. Puheim und L. Madarász, „Normalization of inputs and outputs of neural network based robotic arm controller in role of inverse kinematic model,“ *2014 IEEE 12th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*, 23-25 Januar 2014.
- [22] „Iris Data Set,“ [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/Iris>. [Zugriff am 17 Juli 2020].

## 10. Literaturverzeichnis

---

- [23] „Yacht Hydrodynamics Data Set,“ [Online]. Available: <http://archive.ics.uci.edu/ml/datasets/yacht+hydrodynamics>. [Zugriff am 25 August 2020].
- [24] „Real Estate Valuation Data Set,“ [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/Real+estate+valuation+data+set>. [Zugriff am 31 August 2020].
- [25] „Balance Scale Data Set,“ [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/Balance+Scale>. [Zugriff am 31 August 2020].
- [26] „QSAR Biodegradation Data Set,“ [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/QSAR+biodegradation>. [Zugriff am 27 Juli 2020].
- [27] „Car Evaluation Data Set,“ [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/Car+Evaluation>. [Zugriff am 31 August 2020].
- [28] „Wine Quality Data Set,“ [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/wine+quality>. [Zugriff am 19 Juli 2020].
- [29] „MAGIC Gamma Telescope Data Set,“ [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/MAGIC+Gamma+Telescope>. [Zugriff am 31 August 2020].

## **Eidesstattliche Erklärung**

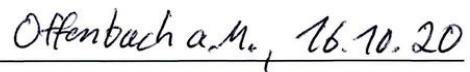
Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit selbstständig, ohne unerlaubte fremde Hilfe angefertigt und andere als die angegebenen Quellen und Hilfsmittel nicht benutzt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Stellen sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keinem anderen Prüfungsamt vorgelegt und auch nicht veröffentlicht.

---



Brening, Artur

---



Ort, Datum