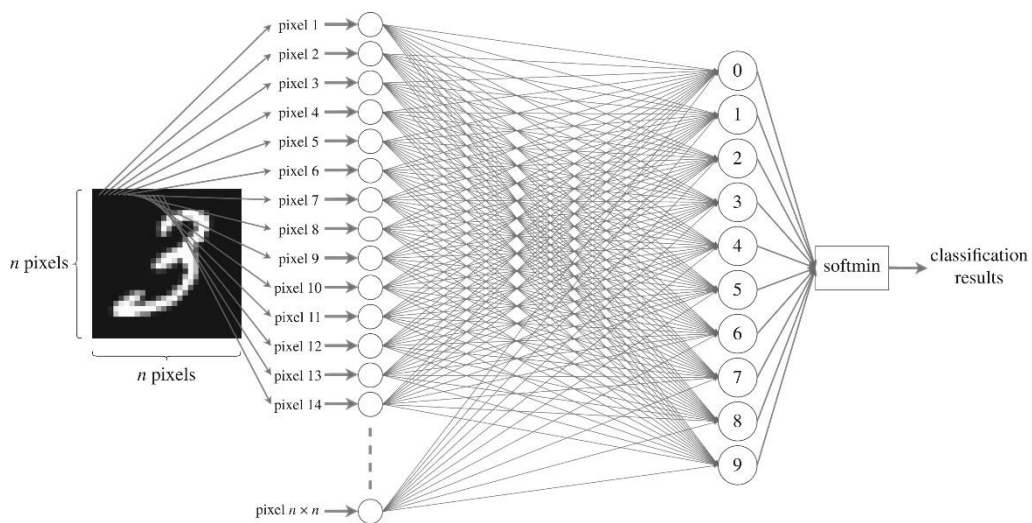


Protokoll

Klassifizierung handgeschriebener Ziffern



Quelle: <https://royalsocietypublishing.org/doi/10.1098/rsta.2019.0163>

Datum:

Name	Matrikelnr.	Punkte Protokoll
Dustin Hanusch	4844370	/20

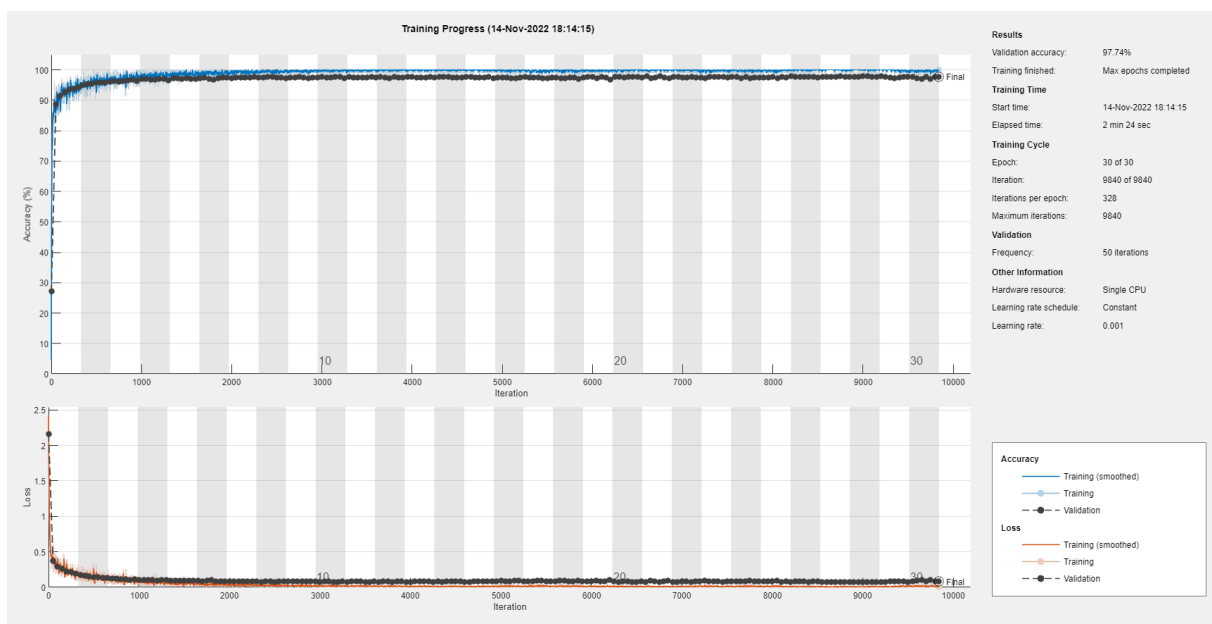
Aufgabe 1: Training eines Neuronalen Netzes zur Ziffernklassifizierung

Entnehmen Sie der Versuchsanleitung die zur Lösung der Aufgabe notwendigen Schritte 1 (Laden des Trainings- und Validierungsdatensatzes) und 2 (Erstellung einer geeigneten Netzarchitektur).

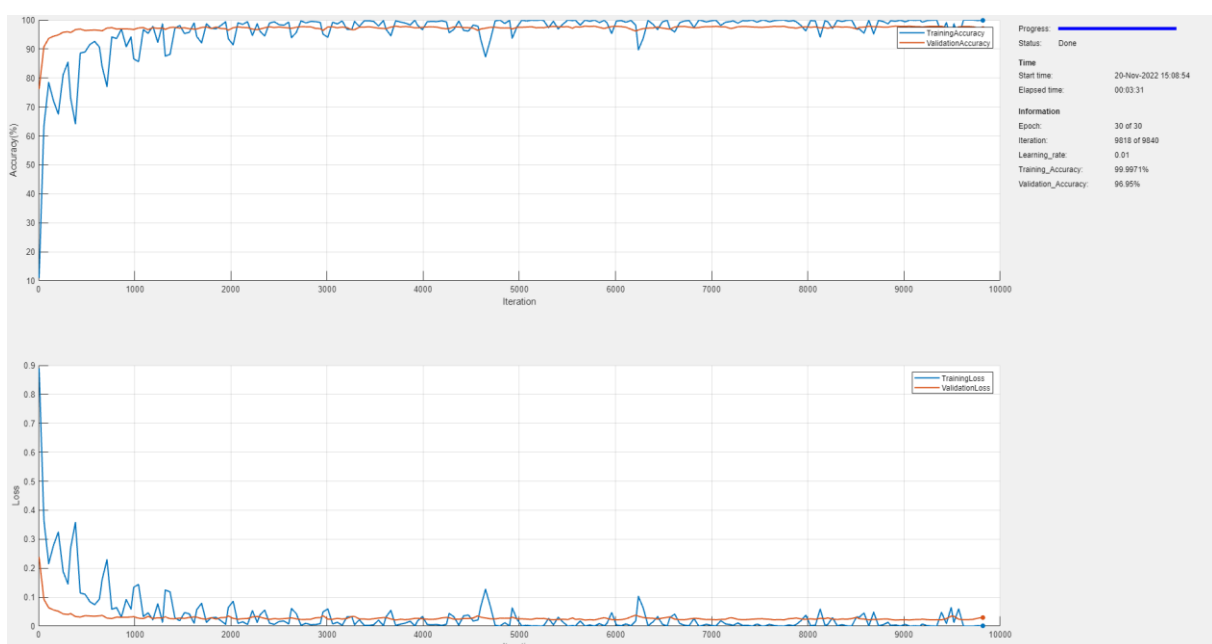
Stellen Sie für beide Trainingsvorgänge (automatisierte und benutzerdefinierte Trainingschleife) die Trainingskurven $\text{loss} = f(\text{iteration})$ und $\text{accuracy} = f(\text{iteration})$ in einem Diagramm dar.

Diagramm (5P):

1.1 Automatisiertes Training



1.2 Benutzerdefiniertes Training



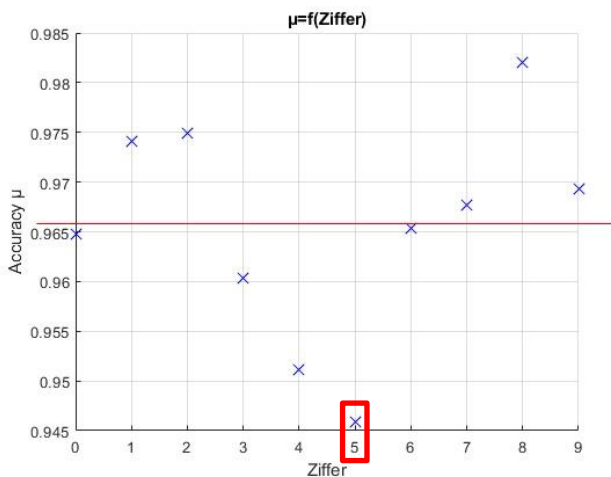
Beschreibung (1P):

Verwenden Sie für alle nachfolgenden Aufgaben die benutzerdefinierte Trainingsschleife!

Aufgabe 2: Klassifizierungsgenauigkeit in Abhängigkeit der eingelesenen Ziffer

Sortieren Sie hierfür die Testdaten nach den jeweiligen Ziffern und ermitteln Sie für jede Ziffer die mittlere Klassifizierungsgenauigkeit über den Mittelwert. Stellen Sie das Ergebnis in einem Diagramm $\mu = f(\text{Ziffer})$ dar. Welche Ziffer kann am besten erkannt werden? Markieren Sie diese im Diagramm!

Diagramm (3P):



Beschreibung (1P):

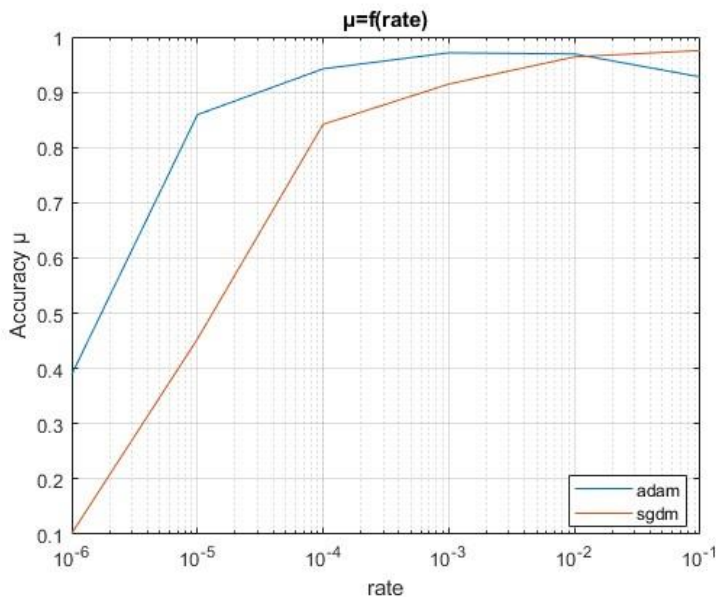
Im Diagramm wird die mittlere Genauigkeit der einzelnen Ziffern dargestellt, welche durch das Auswerten von Testdaten durch das Neuronale Netz errechnet worden ist.

Die fünf wird in diesem Fall mit einer mittleren Genauigkeit von ca. 94,6 % am schlechtesten erkannt. Die Genauigkeit aller Ziffern im Mittel beträgt ca. 96,7%, wobei die Acht mit 98,2% am besten erkannt wird.

Aufgabe 3: Klassifizierungsgenauigkeit in Abhängigkeit von Lernrate und Optimierungsalgorithmus

Vergleichen Sie nun die beiden Optimierer *adam* und *sgdm* mit mindestens 6 verschiedenen *learning rates* zwischen 10^{-6} und 10^{-1} . Stellen Sie die erreichte mittlere Klassifizierungsgenauigkeit in Abhängigkeit von der *learning rate* $\mu = f(\text{rate})$ dar und verwenden Sie für die *learning rate* eine logarithmische Achse.

Diagramm (4P):



Beschreibung (1P):

Im Diagramm wurden auf der X-Achse die getesteten learning rates

(10^{-6} , 10^{-5} , 10^{-4} , 10^{-3} , 10^{-2} , 10^{-1}) abgetragen und die mittlere Genauigkeit auf der Y-Achse.

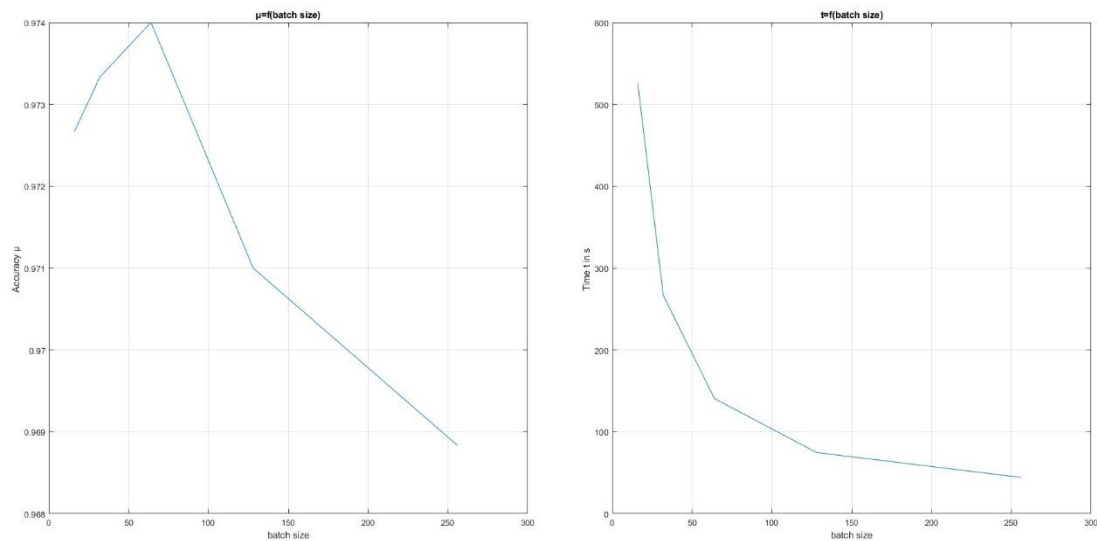
Die bei diesem Versuch verwendete Batch Size beträgt 128 und die Epochenanzahl ist 10.

Bei diesen Einstellungen erreicht der adam Solver die beste Präzision von 97,1% bei einer Learning rate von 10^{-3} . Der sgdm Solver hingegen erreicht hierbei mit 97,6% eine geringfügig höhere Genauigkeit bei einer Learning rate von 10^{-1} .

Aufgabe 4: Klassifizierungsgenauigkeit und Trainingszeit in Abhängigkeit der Größe des Mini-Batch

Testen Sie mindestens 5 verschiedene Größen des Mini-Batch zwischen 16 und 256. Nutzen Sie den solver *adam*. Wählen Sie aus der vorherigen Aufgabe eine *learning rate* aus, bei der der Trainingsprozess zu einer hohen Klassifizierungsgenauigkeit konvergiert. Stellen Sie die mittlere Klassifizierungsgenauigkeit und die benötigte Trainingszeit in Abhängigkeit von der Größe des Mini-Batches dar: $\mu = f(\text{batch size})$ und $t = f(\text{batch size})$.

Diagramme (3P):



Beschreibung (1P):

Die verwendeten Hyperparameter sind:

Batch Size = [16 32 64 128 256]

Learning rate = 10^{-3}

Epochenanzahl = 10

In dem rechten Diagramm ist die Batch Size gegenüber der benötigten Lernzeit abgetragen. Zu erkennen ist dabei, dass die Zeit mit geringer Batchsize exponentiell steigt.

Das linke Diagramm stellt zum rechten passenden die Accuracy zu den verschiedenen Größen der Batches dar. Die höchste Genauigkeit mit 97,4% erreicht die Batch Size von 64.

Aufgabe 5: Diskussion (1P)

Leiten Sie aus den Trainingsergebnissen Zusammenhänge zwischen den untersuchten Hyperparametern und der Trainingszeit, sowie der erreichten Klassifizierungsgenauigkeit ab. Formulieren Sie hierfür eine kurze Diskussion (Stichpunkte erlaubt):

- eine zu große sowie zu kleine Lernrate kann für eine Schlechte Präzision sorgen, in meinem Versuch haben die mittleren Lernraten die besten Ergebnisse erzielt
- die Trainingszeit steigt exponentiell bei fallender Batch Size
- die Klassifizierungsgenauigkeit steigt in derer Regel bei kleinerer Batch Size
- Steigt die Epochenzahl so wir sowohl Klassifizierungsgenauigkeit als auch Trainingszeit größer
- Der Einfluss der PC-Hardware ist ebenfalls Signifikant. Ein schneller Prozessor, aber besonders eine performante Grafikkarte reduzieren die Lernzeit erheblich.