Laborversuch 4

Versuch	**Deep Convolutional Neural Networks**
Fach	Intelligente Lernende Systeme
Semester	WS 2021/22
Fachsemester	CPS 5 und ITS5/WIN5
Labortermine	13.01.2022
	20.01.2022
Abgabe bis spätestens	28.01.2022 (keine Pflicht)

Versuchsteilnehmer

Name: Vorname:

Semester: Matrikelnummer:

Bewertung des Versuches

Aufgabe:	1	2	3	
Punkte maximal:	20	40	20	
Punkte erreicht:				
Gesamtpunktezahl:	/80	Note:		Zeichen:

Anmerkungen:

Aufgabe 1: (5+5+5+5 = 20 Punkte)

Thema: Verbesserte Optimierungsverfahren

a) Versuchen Sie die quadratische Fehlerfunktion $E(\mathbf{w}) := \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{A}\mathbf{w} - \mathbf{b}^{\mathrm{T}}\mathbf{w}$ für

$$\mathbf{A} := \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}, \qquad \mathbf{b} := \begin{pmatrix} 1 \\ -2 \end{pmatrix} \mathbf{w}$$

mit Hilfe von Stochastischen Gradientenabstieg (SGD) zu minimieren. Gehen Sie dabei vom Anfangsgewicht $\mathbf{w}(0) := \begin{pmatrix} 2 & 1 \end{pmatrix}^{\mathrm{T}}$ aus.

Hinweis: Sie können hierzu das Programmgerüst `V4A1_Optimizer.py` verwenden. Stellen Sie die Lernrate optimal ein, sodass Sie in möglichst wenig Lernschritten $\tau_{\min}$ das Minimum erreichen (Kriterium sei $\tau_{\min} := \arg\min_{\tau \geq 1} ||\mathbf{w}(\tau) - \mathbf{w}_{\min}|| < 1e-8$).

b) Versuchen Sie dann mindestens zwei alternative Optimierer zu implementieren (siehe Progammgerüst), z.B. Momentum und ADAM. Optimieren Sie wieder und vergleichen Sie mit SGD von (a). Welcher Optimierer ist am besten (d.h. erreicht am schnellsten das Minimum)?

c) Verändern Sie dann $\mathbf{A}$ und $\mathbf{b}$ derart, dass eine "tiefe Rinne" entsteht (z.B. $\mathbf{A}$ Diagonalmatrix mit stark unterschiedlichen Diagonalelementen). Optimieren und vergleichen Sie wieder!

Aufgabe 2: (20+5+10+5 = 40 Punkte)

Thema: Implementierung Allgemeiner Backpropagation-Algorithmus

a) Implementieren Sie den Allgemeinen Backpropagation-Algorithmus von Satz 6.1 (SB6) in Python/Numpy:

- Es soll möglich sein ein geschichtetes vorwärtsgerichtetes Neuronales Netzwerk mit beliebiger Graph-Struktur zu trainieren. Sie können als Ausgangspunkt das MLP-Modell des vorigen Praktikums-Versuchs verwenden. Benutzen Sie die vektorwertige Darstellung von Satz 6.1.
- Implementieren Sie jeweils Klassen für Layers l und Verbindungen (l, l') zwischen den Layern. Jede Layer l soll Numpy-Arrays für die Zustandsvariablen dendritische Potentiale $\mathbf{a}^{(l)}$, Feuerraten $\mathbf{z}^{(l)}$, Fehlerpotentiale $\boldsymbol{\alpha}^{(l)}$ und Fehlersignale $\boldsymbol{\delta}^{(l)}$ haben. Jede Layer soll außerdem Bias-Gewichte $\mathbf{b}^{(l)}$ und jede Verbindung eine Gewichts-Matrix $\mathbf{W}^{(l,l')}$ besitzen.
- Testen Sie Ihre Implementierung. Überprüfen Sie insbesondere ob die Gradienten welche der Backprop-Algorithmus berechnet mit den numerischen Gradienten (über die Differenzenquotienten) übereinstimmen.

b) Implementieren Sie zusätzlich zu stochastischem Gradientenabstieg die Optimierungsverfahren Momentum und ADAM von der vorigen Aufgabe in Ihre Implementierung des Allgemeinen Backpropagation-Algorithmus von (a).

c) Versuchen Sie dann mit einem geeigneten Netzwerk die Klassifikation von Wald-Typen von Satellitenaufnahmen von Versuch 3 so gut wie möglich zu lösen.

d) Vergleichen Sie bezüglich Trainings/Test-Geschwindigkeit sowie Klassifikationsergebnissen mit Ihren Ergebnissen von Versuch 3 (Auf.4 und 5).

Hinweis: Da es sehr viel Aufwand ist Aufgabenteile (a) und (b) selbständig zu lösen, können Sie stattdessen auch das Python-Modul `V4A2_BackpropGraphNN_withOptimizer.py` verwenden. Versuchen Sie den Aufbau des Moduls und den Python-Code (einigermaßen) zu verstehen. Lösen Sie damit Aufgabenteile (c) und (d).

Aufgabe 3: (20 Punkte)

Thema: MNIST mit Keras und/oder PyTorch

Implementieren Sie mit Hilfe von Keras und/oder PyTorch ein geeignetes Netzwerk, um das MNIST-Datenset aus handgeschriebenen Ziffern zu klassifizieren. Sie können dazu CNN-Layers, Max-Pool-Layers, Fully-Connected Layers, und weitere Layer-Typen verwenden (z.B. Dropout). Optimieren Sie das Netzwerk um die Klassifikationsleistung zu maximieren.

Hinweis: Sie können das Datenset in Python z.B. mit Hilfe von Keras laden:

```
from keras.datasets import mnist
(X_train, T_train), (X_test, T_test) = mnist.load_data()
```