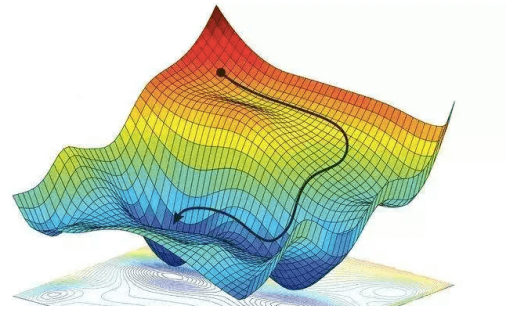


Assignment

Gradient Descent



Das Ziel dieser Aufgabe besteht darin, dass Sie ein grundlegendes Verständnis für numerische Näherungsverfahren in höheren Dimensionen erlangen, insbesondere für den Gradient Descent und dessen praktische Anwendung. Hierfür sollen Sie ein Jupyter Notebook erstellen und das MNIST Dataset laden und erkunden. Anschließend sollen Sie ein neuronales Netzwerk erstellen und trainieren, um die Bilder korrekt zu klassifizieren. Es dürfen nur die angegebenen Python Pakete verwendet werden.

Ziel dieser Aufgabe ist nicht nur, Ihre mathematischen Kenntnisse in der Praxis unter Beweis zu stellen, sondern auch die entsprechende Kommunikation, Argumentation und Präsentation Ihrer Ergebnisse. Ihre Abgaben sollen also nicht nur **mathematisch korrekt**, sondern auch **leicht verständlich** und **reproduzierbar** sein. Genauere Angaben zu den Erwartungen an die Abgabe finden Sie in den Auswertungskriterien, welche auch die Grundlage der Bewertung darstellen. Dokumentieren Sie ihren Arbeitsfortschritt und Erkenntnisgewinn in Form eines Lerntagebuchs, um Lernfortschritte, Schwierigkeiten und Erkenntnisse festzuhalten.

Die folgenden Aufgabenstellungen präzisieren die einzelnen Bearbeitungsschritte und geben die Struktur des Notebooks vor.

Erlaubte Pakete: numpy, matplotlib, Built-in Pakete (time, sys, math, ...), torchvision für Aufgabe 1

Nicht erlaubte Pakete: Alle anderen 3rd-Party Pakete, insbesondere PyTorch, Tensorflow, Keras, scikit-learn, torchvision ausserhalb von Aufgabe 1.

Lesen Sie als erstes sorgfältig die Aufgabenstellung durch und schreiben Verständnisfragen und unbekannte Begriffe heraus. Dann beschäftigen Sie sich mit der Idee Neuronaler Netzwerke und des Gradient Descent, zB mit den Ressourcen weiter unten, und beantworten diese Fragen. Nachdem Sie sich ein grundsätzliches Verständnis angeeignet haben, lesen Sie sorgfältig die Bewertungskriterien. Erst danach sollten Sie mit dem Coden beginnen.

Aufgabe 1.

Laden Sie das MNIST-Dataset (Training und Test) mithilfe des torchvision-Pakets (Verwenden Sie das torchvision Paket nur für diese Aufgabe) und verwenden Sie matplotlib, um sich einen Überblick über die Daten zu verschaffen. Beschreiben Sie die grundlegenden Eigenschaften des Datensets, z.B. wie viele und welche Daten es enthält und wie diese verteilt sind.

Aufgabe 2.

Erstellen Sie eine Klasse für ein lineares Layer mit beliebig vielen Knoten. Implementieren Sie darin getrennte Methoden für Forward-Pass, Backward-Pass und Parameter-Update mithilfe von numpy. Schreiben Sie geeignete Unittests, um die Funktionsweise dieser Funktionen zu prüfen. Schreiben Sie insbesondere einen expliziten Test, für ein Layer mit 2 Knoten, welches als Input 2 Datensätzen zu je zwei 2 floats erhält. Wählen Sie dazu unterschiedliche feste Werte für Input, initiale Gewichte und Lernrate. Dann berechnen sie von Hand die Ergebnisse von Forward, Backward und Update und testen damit ihre Implementation. Legen Sie die Berechnung der Ergebnisse ihrer Lösung bei.

Aufgabe 3.

Erstellen Sie ein neuronales Netzwerk mit 3 Hidden Layern mit gleicher Anzahl Knoten und einem Output Layer mit 10 Knoten. Verwenden Sie dazu die in Aufgabe 2 implementierte Klasse. Das Ziel ist die korrekte Klassifizierung aller Ziffern. Bereiten Sie die Trainingsloop und alles dafür benötigte vor, um das Netzwerk darauf zu trainieren. Das heisst, jeder Output Knoten wird einer Ziffer zugeordnet; der Output soll 1 für diese Ziffer und 0 für alle anderen Ziffern sein.

Das Netzwerk soll auf den Trainingsdaten trainieren und auf den Testdaten evaluiert werden. Achten Sie beim Training darauf, nicht auf dem gesamten Datensatz gleichzeitig zu trainieren, sondern stückeln Sie diesen in kleine Portionen (batches), auf denen nacheinander trainiert wird. Erläutern Sie kurz, warum das nötig ist.

Verwenden Sie eine geeignete Kosten-Funktion sowie Evaluations-Funktion und geben Sie deren mathematische Definition an. Begründen Sie Ihre Wahl dieser Funktion und diskutieren Sie kurz eine weitere Option für Kosten und Evaluation mit einer Abwägung der Vor- und Nachteile.

Aufgabe 4.

Trainieren Sie das Netzwerk mit verschiedenen Lernraten (0.01 - 1) und Größen der Hidden Layer (4, 8, 16). Verfolgen Sie während des Trainings die Entwicklung der Kosten- und Evaluations-Funktionen sowohl auf Trainings- als auch auf Testdaten. Interpretieren Sie die Ergebnisse des Netzwerks und entscheiden Sie, welche Wahl von Lernrate und Hidden Layer-Größe die Beste ist. Begründen Sie Ihre Wahl.

(Hinweis: Ziel ist es ein prinzipiell funktionierendes Modell zu entwickeln und den Trainingsvorgang kennenzulernen. Es ist nicht das Ziel das Modell zu optimieren.)

Künstliche Intelligenz:

Für diese Aufgaben dürfen KI Tools wie ChatGPT oder Github Copilot als Ressource genutzt werden, um Fragen zu stellen oder bei Problemen Unterstützung zu erhalten. Voraussetzung ist, dass Sie transparent kommunizieren, wo und wie Sie diese Tools eingesetzt haben und welche Verbesserungen nötig waren. Legen Sie Ihrer Abgabe einen Screenshot eines relevanten Chats bei, der Ihren Umgang mit KI zeigt, und verfassen sie eine kurze Reflexion über Ihren Umgang mit KI: Wo hat es geholfen? Wo war es hinderlich?

Zur Nutzung von ChatGPT und anderen KI-Tools liegt ein weiteres Dokument bei.

Eigenverantwortung

Diese Aufgabe zählt als unbegleitetes Selbststudium. Es wird also von Ihnen erwartet, sich die nötigen Informationen und Verständnis eigenständig anzueignen, sowie diesen Prozess zu planen. Ich empfehle Ihnen, sich jede Woche zumindest 1-2 Stunden mit dieser Aufgabe zu beschäftigen und die Bearbeitung nicht aufzuschieben. Als Dozent stehe ich Ihnen auf Anfrage (im Unterricht und per Mail) gerne als Ressource zur Verfügung, zB mit individuellem Feedback, Erläuterungen und Diskussion. Kommen Sie gerne einfach auf mich zu.

Hilfreiche Ressourcen:

Mehr zu Gradient Descent und Neuronalen Netzwerken

Neuronale Netzwerke, erklärt von 3blue1brown

Jupyter Notebook Cloud Service: Google Colab

Tutorials: Build Neural Network in Numpy:

medium.com, towardsdatascience.com 1, towardsdatascience.com 2 kaggle.com,