Machine Learning 3
Refugeeks / intoCODE
2022

Torben Laue, M.Sc.
Florian Kluger, M.Sc.

Fully-Connected Netzwerk

Für die folgenden Aufgaben haben Sie zweidimensionale (2D) Eingabe-Daten **x** und Ground-Truth Ausgabe-Daten **y** gegeben. Jeder Punkt in den Daten gehört zu einer von **zwei Klassen**. Die Ground-Truth **y** enthält für jeden Punkt die korrekte Klassen-ID (0 oder 1). Diese Daten sind in Abbildung 1 dargestellt.

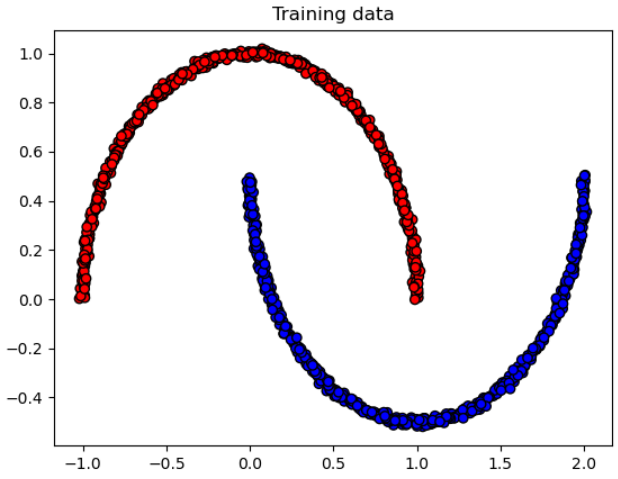

Abbildung 1: Daten

Der Datensatz ist aufgeteilt in ein **Trainings-Set** mit 1024 Punkten, und ein **Validation-Set** mit 128 Punkten.

1 Neuronales Netz Definieren

Verwenden Sie für diese Aufgabe die Datei `network.py`.

Vervollständigen Sie die Klasse `NeuralNetwork` und implementieren Sie ein neuronales Netz bestehend aus:

1. Drei Fully-Connected-Layer (torch.nn.Linear)
2. Das erste Layer hat eine Input-Dimension von 2
3. Das zweite Layer hat Input- und Output-Dimensionen von je 16

4. Das dritte Layer hat eine Output-Dimension von 2
5. Nach Layer 1 und 2 folgt eine ReLU-Aktivierungsfunktion (torch.nn.ReLU)

Definieren Sie zuerst die einzelnen Layer in der `__init__`-Funktion. Implementieren Sie dann den Forward-Pass in der `forward`-Funktion.

Wenn Sie alles richtig gemacht haben, sollte beim Ausführen des Skriptes der Text SSieht gut ausërscheinen.

2 Training

Nun sollen Sie Ihr neuronales Netz auf die oben gegebenen Daten trainieren. Verwenden Sie für diese Aufgabe die Datei `train.py`.

Schauen Sie sich den vorgegebenen Code zuerst genau an! Dataset und Dataloader und sind bereits gegeben.

Implementieren Sie nun folgende Teile, die noch fehlen:

2.1 Netz und Optimizer

(a) Erstellen Sie eine Instanz ihres neuronalen Netzes.

(b) Erstellen Sie eine Instanz der Klasse `torch.nn.CrossEntropyLoss`.

(c) Erstellen Sie eine Instanz des Adam-Optimizers, mit dem Sie Ihr Netz trainieren werden.

2.2 Trainings-Schleife

(a) Versetzen Sie ihr Netz zu Beginn der Epoche in den Trainingsmodus.

(b) Normalisieren Sie die Daten `x` mit dem Mittelwert `train_mean` und der Standardabweichung `train_std`.

(c) Forward-Pass: Bestimmen Sie `y_estimate` mithilfe Ihres neuronalen Netzes

(d) Berechnen Sie den Cross-Entropy-Loss.

2.3 Validation

Nach jeder Epoche sollen Sie nun die Genauigkeit (Accuracy) Ihres neuronalen Netzes auf dem Validation-Datensatz testen.

(a) Versetzen Sie ihr Netz in den Evaluations-Modus.

(b) Normalisieren Sie die Daten `x` mit dem Mittelwert `train_mean` und der Standardabweichung `train_std`.

(c) Forward-Pass: Bestimmen Sie `y_estimate` mithilfe Ihres neuronalen Netzes

(d) Bestimmen Sie aus `y_estimate` die zugehörigen Klassen-IDs `y_estimate_ids` mithilfe der Funktion `torch.argmax(...)`.

(e) Berechnen Sie die Genauigkeit: wieviel Prozent der IDs in `y_estimate_ids` stimmen mit den Ground-Truth-IDs in `y` überein?

Bonus-Aufgabe: Finden Sie heraus, wie Sie die Gewichte (Parameter) Ihres neuronalen Netzes auf der Festplatte speichern können. Speichern Sie am Ende jeder Epoche einen *Checkpoint* Ihres neuronalen Netzes.