Deep Learning Assignment 2 in FS 2022

Gradient Descent

Microsoft Forms Document:
https://forms.office.com/r/kkt8SK1HA8

Manuel Günther

Distributed: Friday, March 4, 2022
Discussed: Friday, March 11, 2022

The goal of this exercise is to gain experience with a basic technique of Deep Learning, i.e., gradient descent. A two-dimensional loss surface is created manually and gradient descent is implemented. Several runs of gradient descent from different starting locations will be performed. The loss surface is and the detected minima are plotted together in one 3D plot.

1 Gradient Descent

To learn the basics of gradient descent, we manually design a loss function $\mathcal{J}_{\vec{w}}$ with two weights $\vec{w} = (w_0, w_1)$ and many local minima. The loss function is given as:

$$\mathcal{J}_{\vec{w}} = w_0^2 + w_1^2 + 20 \cdot \sin(w_0) \cdot \cos(w_1) \tag{1}$$

1.1 Compute the Gradient

In order to perform gradient descent, first the gradient for this loss function needs to be computed.

Task 1 (theoretical question) Analytically compute the gradient for the loss function $\mathcal{J}_{\vec{w}}$. Provide both derivatives $\frac{\partial \mathcal{J}_{\vec{w}}}{\partial w_0}$ and $\frac{\partial \mathcal{J}_{\vec{w}}}{\partial w_1}$. Make use of the following derivative rules:

$$\frac{\partial \sin(x)}{\partial x} = \cos(x) \qquad \frac{\partial \cos(x)}{\partial x} = -\sin(x) \tag{2}$$

as well as the sum rule, the linear rule, the square rule, and the constant rule as given in the lecture.

Task 2 Implement the loss function in Python, which takes a given $\vec{w}$ and returns $\mathcal{J}_{\vec{w}}$ according to (1).

Task 3 Implement the gradient as a function in Python, which takes a given $\vec{w}$ and returns $\nabla_{\vec{w}}\mathcal{J}_{\vec{w}}$ according to the analytical result in Task 1.

Test 1 Analytically compute the loss and the gradient for $\vec{w} = (0, 0)^{\mathrm{T}}$. Call the loss function from Task 2 with $\vec{w} = (0, 0)^{\mathrm{T}}$. Call the gradient function from Task 3 with $\vec{w} = (0, 0)^{\mathrm{T}}$. Check that the returned values are correct.

1.2 Implement Gradient Descent

The procedure of gradient decent is the repeated application of two steps. First, the gradient of the loss $\nabla_{\vec{w}}\mathcal{J}_{\vec{w}}$ is computed based on the current value of the parameters $\vec{w}$. Second, the weights are updated by moving a small step into the direction of the negative gradient:

$$\vec{w} = \vec{w} - \eta \nabla_{\vec{w}}\mathcal{J}_{\vec{w}} \tag{3}$$

Optionally, the loss $\mathcal{J}_{\vec{w}}$ is computed to record the progress of the gradient descent. Finally, a criterion needs to be defined to decide when to stop the procedure.

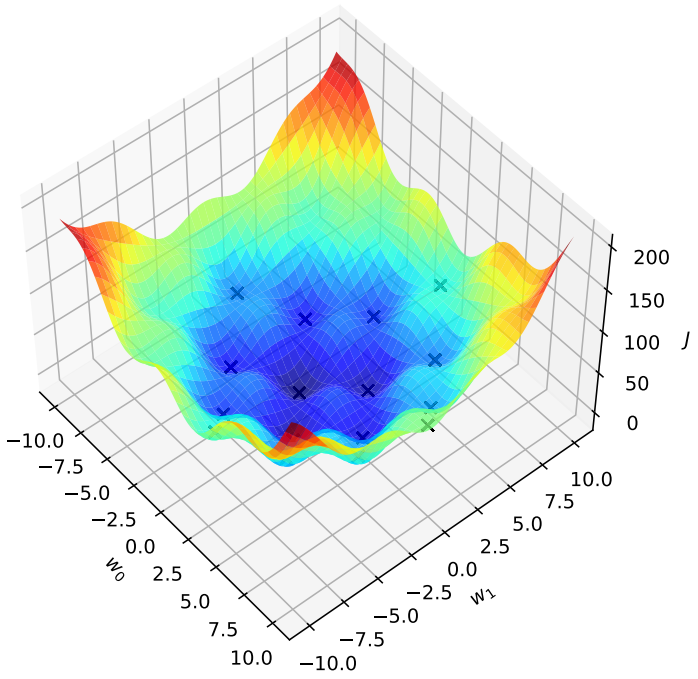

Figure 1: Exemplary plot resulting from Task 8.

Task 4 (theoretical question) Define a proper termination criterion. Which error cases might occur and need to be considered?

Task 5 Implement a function that performs the gradient descent. This function should take as parameters an initial weight vector $\vec{w}$ and a learning rate η, and make use of the gradient function implemented in Task 3 and, possibly, the loss function from Task 2. It should return the optimized weight vector $\vec{w}^*$. Incorporate the termination criterion designed in Task 4.

1.3 Evaluate Gradient Descent

Task 6 Call the gradient descent function from Task 5 1000 times with different randomly initialized weights $\vec{w} \in [-20, 20]^2$ and a learning rate of $\eta = 0.01$. Store the resulting optimized weight vectors in a list.

Test 2 (**difficult task**) Since our loss function has limited amount of minima (cf. Figure 1), check that all optimized weights fall into the same minima. Design and implement a method that counts the amount of minima into which our optimized weights fall. What happens if you run the gradient descent from Task 6 with a learning rate of $\eta = 0.1$?

Task 7 Find the global minimum of our error function by evaluating the obtained optimized weight vectors from Task 6. Print the minimum and its loss value.

1.4 Plot Error Surface and Points

Task 8 Plot the error surface of the loss function from (1). For each of the optimized weights from Task 6, plot a marker into the 3D plot. An example can be found in Figure 1.