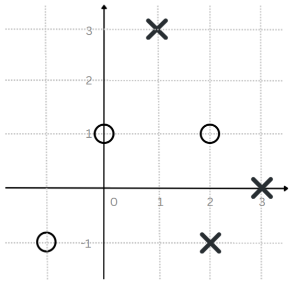
November 22nd, 2023.

Department of Electronics, Microelectronics,
Computer and Intelligent Systems

Deep Learning

midterm exam

1. Design a two-layer fully connected model that solves the problem shown in the image. The given problem contains five data points that need to be classified into two classes: circle and cross. Let the activation function of the hidden layer be a rectified linear unit (ReLU), and for the output layer, a sigmoid. Write all the parameters of the model and demonstrate that your model correctly classifies all the data under the assumption that the class boundary is defined by a probability of 50%.



2. (10 bodova)

   The input to the convolutional layer consists of single-channel data with dimensions 4×4. The convolutional layer has a kernel $\boldsymbol{W}$ with dimensions 2×2 and a bias b. The kernel $\boldsymbol{W}$ is defined by unknown parameters e and f as follows: $\boldsymbol{W} = \begin{bmatrix} e & e \\ f & f \end{bmatrix}$. The input to the layer is the input data $\boldsymbol{X}$, and the resulting output is $\boldsymbol{Y}$. Due to noise in the communication channel, some activations are unknown. Such activations are marked with a question mark (?). Their values are mutually independent, meaning they can be anything.

   $$\boldsymbol{X} = \begin{bmatrix} ? & ? & ? & ? \\ 1 & 2 & -1 & -2 \\ 3 & -1 & 1 & 2 \\ ? & 0 & 0 & ? \end{bmatrix} \qquad \boldsymbol{Y} = \begin{bmatrix} 7 & 2 & -5 \\ 6 & ? & 2 \\ 3 & -1 & -2 \end{bmatrix}$$

   Tasks:

   (a) Find the parameters of the convolutional kernel $\boldsymbol{W}$ and the bias value b.

   (b) Find the values of the first and last elements in the last row of the input data $\boldsymbol{X}$.

3. (10 points) Consider a classification model defined with a sequence of layers:

   $$\boldsymbol{h} = \mathrm{ReLU}(\boldsymbol{W} \cdot \boldsymbol{x} + \boldsymbol{b}) \quad (1)$$
   $$\boldsymbol{p}_1 = \mathrm{softmax}(\boldsymbol{W}_1 \cdot \boldsymbol{h} + \boldsymbol{b}_1) \quad (2)$$
   $$p_2 = \sigma(\boldsymbol{W}_2 \cdot \boldsymbol{h} + b_2) \quad (3)$$

   The model has two outputs: $\boldsymbol{p}_1$, which predicts the user's temperament (sanguine, choleric, melancholic, phlegmatic), and p_2, which predicts whether the user will incur a financial loss in the next month. The model's loss is the sum of the negative log-likelihoods of these two predictions.

   The inital values of the model parameters are as follows: $\boldsymbol{W} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$; $\boldsymbol{W}_1 = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}^\top$; $\boldsymbol{W}_2 = \begin{bmatrix} 1 & -1 \end{bmatrix}$; $\boldsymbol{b} = \boldsymbol{b}_1 = \boldsymbol{0}$; $b_2 = 0$. The model is fed with a vector representing a sanguine individual who will not incur a financial loss: $\boldsymbol{x} = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}^\top$.

   (a) Conduct a forward pass and determine the model prediction.

   (b) Calculate the loss value.

   (c) Determine the gradients with respect to $\boldsymbol{W}$.

4. Write a direct implementation of a function that performs the forward pass and calculates the loss for multi-class logistic regression using the `numpy` library. The function should take as input a single data point $\boldsymbol{x}$ with dimensions (D, 1), a natural number y in the range [1, C] indicating the correct class, and model parameters. Determine the dimensions of the model parameters tensor. Identify parts of your code where numerical errors could occur. Write a robust implementation that addresses these issues. Note: pay attention to the possibility of overflow in the exp and log functions.

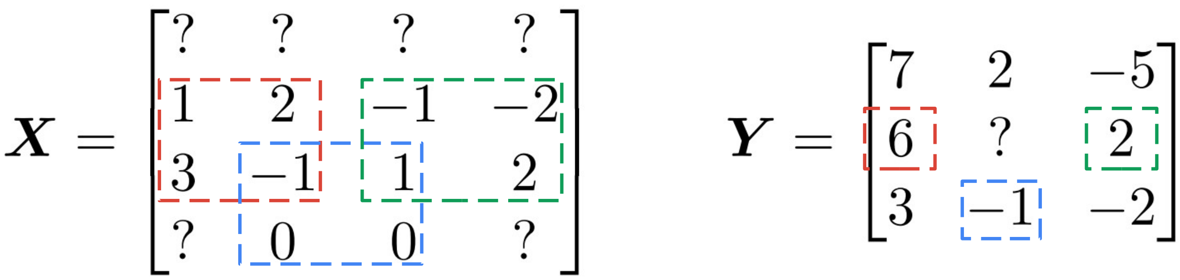
5. Consider a classification model defined with a sequence of layers:

$$\boldsymbol{s_1} = \text{conv1D}(\boldsymbol{x}, \boldsymbol{w_1}, b_1, \text{padding} = \text{"same"}) \tag{4}$$
$$\boldsymbol{h_1} = \text{ReLU}(\boldsymbol{s_1}) \tag{5}$$
$$s_2 = \boldsymbol{w_2}^T \boldsymbol{h_1} \tag{6}$$
$$p = \sigma(s_2) \tag{7}$$

Compute the gradients of the binary cross entropy with respect to the model parameters $\boldsymbol{w_1}$, b_1 and $\boldsymbol{w_2}$, if the input vector $\boldsymbol{x} = \begin{bmatrix} 1 & 1 & -1 & 1 \end{bmatrix}^T$ and the ground truth label $y = 0$. The initial parameters of the model are as follows: $\boldsymbol{w_1} = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}^T$, $b_1 = 0$, $\boldsymbol{w_2} = \begin{bmatrix} 2 & 1 & -1 & 1 \end{bmatrix}^T$,

6. Answer the following questions:
   (a) Draw a graph of training and generalization error with respect to model complexity.
   (b) What is overfitting? What is underfitting?
   (c) What is model bias? What is model variance?
   (d) What strategies may be used to prevent overfitting?

1. (a) 3 equations with 3 unknowns. We may look at three "complete" sections These are elements of output Y with the values 6, 2 and -1 at the positions (1, 0), (1, 2) i (2, 1). We get the



Slika 1: Pertinent positions.

following system of equations:

$$3e + 2f + b = 6 \tag{8}$$
$$-3e + 3f + b = 2 \tag{9}$$
$$0e + 0f + b = -1 \tag{10}$$

We get $b = -1$ from the last equation. We may that input that value in the first two equations:

$$3e + 2f = 7 \tag{11}$$
$$-3e + 3f = 3 \tag{12}$$
$$\tag{13}$$

Add them up:

$$5f = 10 \tag{14}$$
$$f = 2 \tag{15}$$

It then follows:

$$e = 1 \tag{16}$$

(b) First element

$$\begin{aligned} 3 \cdot 1 + (-1) \cdot 1 + x_{4,1} \cdot 2 + 0 - 1 &= 3 && (17)\\ 2x_{4,1} + 1 &= 3 && (18)\\ 2x_{4,1} &= 2 && (19)\\ x_{4,1} &= 1 && (20) \end{aligned}$$

Last element

$$\begin{aligned} 1 \cdot 1 + 2 \cdot 1 + 0 \cdot 2 + x_{4,4} \cdot 2 - 1 &= -2 && (21)\\ 2x_{4,4} + 2 &= -2 && (22)\\ 2x_{4,4} &= -4 && (23)\\ x_{4,4} &= -2 && (24) \end{aligned}$$

2. (a) Forward pass:

$$\boldsymbol{k} = \boldsymbol{W} \cdot \boldsymbol{x} + \boldsymbol{b} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\boldsymbol{h} = \mathrm{ReLU}(\boldsymbol{k}) = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\boldsymbol{s}_1 = \boldsymbol{W}_1 \cdot \boldsymbol{h} + \boldsymbol{b}_1 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 3 \\ 3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \\ 3 \\ 3 \end{bmatrix}$$

$$s_2 = \boldsymbol{W}_2 \cdot \boldsymbol{h} + b_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \begin{bmatrix} 3 \\ 3 \end{bmatrix} + 0 = 0$$

$$\boldsymbol{p}_1 = \mathrm{softmax}(\boldsymbol{s}_1) = \mathrm{softmax}(\begin{bmatrix} 3 \\ 3 \\ 3 \\ 3 \end{bmatrix}) = \begin{bmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{bmatrix}$$

$$p_2 = \sigma(s_2) = \sigma(0) = 0.5$$

(b) Loss

$$L_1 = -log(p_{sangvinik}) = -log(p_{1,0}) = -log(0.25)$$

$$L_2 = -log(p_{ne\_ide\_u\_minus}) = -log(1 - p_{ide\_u\_minus}) = -log(1 - p_2) = -log(0.5)$$

$$L = L_1 + L_2 = -[log(0.25) + log(0.5)] = -log(0.25 * 0.5) = -log(0.125) = -2.079$$

(c) Gradient with respect to $\boldsymbol{W}$

$$\frac{\partial L_1}{\partial \boldsymbol{s}_1} = (\boldsymbol{p}_1 - \boldsymbol{Y}_1^{OH})^T = \begin{bmatrix} 0.25 & 0.25 & 0.25 & 0.25 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} -0.75 & 0.25 & 0.25 & 0.25 \end{bmatrix}$$

$$\frac{\partial L_1}{\partial \boldsymbol{h}} = \frac{\partial L_1}{\partial \boldsymbol{s}_1}\frac{\partial \boldsymbol{s}_1}{\partial \boldsymbol{h}} = \begin{bmatrix} -0.75 & 0.25 & 0.25 & 0.25 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} == \begin{bmatrix} -0.25 & 0.25 \end{bmatrix}$$

$$\frac{\partial L_2}{\partial s_2} = p_2 - [[Y_2 = 1]] = p_2 - 0 = 0.5$$

$$\frac{\partial L_2}{\partial \boldsymbol{h}} = \frac{\partial L_2}{\partial s_2}\frac{\partial s_2}{\partial \boldsymbol{h}} = 0.5\begin{bmatrix} 1 & -1 \end{bmatrix} = \begin{bmatrix} 0.5 & -0.5 \end{bmatrix}$$

$$\frac{\partial L}{\partial \boldsymbol{h}} = \frac{\partial L_1}{\partial h} + \frac{\partial L_2}{\partial h} = \begin{bmatrix} -0.25 & 0.25 \end{bmatrix} + \begin{bmatrix} 0.5 & -0.5 \end{bmatrix} = \begin{bmatrix} 0.25 & -0.25 \end{bmatrix}$$

$$\frac{\partial L}{\partial \boldsymbol{k}} = \frac{\partial L}{\partial \boldsymbol{h}}\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{k}} = \begin{bmatrix} 0.25 & -0.25 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.25 & -0.25 \end{bmatrix}$$

$$\frac{\partial L}{\partial \boldsymbol{W}_{[1,:]}} = \frac{\partial L}{\partial \boldsymbol{s}}\frac{\partial \boldsymbol{s}}{\partial \boldsymbol{W}_{[1,:]}} = \begin{bmatrix} 0.25 & -0.25 \end{bmatrix}\begin{bmatrix} 1 & 2 & 3 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0.25 & 0.5 & 0.75 \end{bmatrix}$$

$$\frac{\partial L}{\partial \boldsymbol{W}_{[2,:]}} = \frac{\partial L}{\partial \boldsymbol{s}}\frac{\partial \boldsymbol{s}}{\partial \boldsymbol{W}_{[2,:]}} = \begin{bmatrix} 0.25 & -0.25 \end{bmatrix}\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 3 \end{bmatrix} = \begin{bmatrix} -0.25 & -0.5 & -0.75 \end{bmatrix}$$

$$\frac{\partial L}{\partial \boldsymbol{W}} = \begin{bmatrix} 0.25 & 0.5 & 0.75 \\ -0.25 & -0.5 & -0.75 \end{bmatrix}$$

```
import torch
x = torch.tensor([1, 2, 3.]).view(-1, 1)
W = torch.tensor([[1., 1, 0], [0, 0, 1]], requires_grad=True)
W1 = torch.tensor([[1., 0], [1, 0], [0, 1], [1,0]])
W2 = torch.tensor([[1., -1]])

h = torch.relu(W@x)
s1 = W1@h
s2 = W2@h
p1 = torch.softmax(s1, 1)
p2 = torch.sigmoid(s2)
L = -torch.log(p1[0]) - torch.log(1 - p2)
L.backward()
W.grad
```

3. code:

```
https://colab.research.google.com/drive/1_p0iZczX9n1HT2XUZIDEDcRjVwcK_8ka?usp=sharing
```