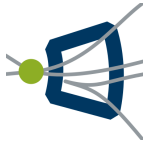


Data Analysis Block Course: Statistics and Machine learning



Exercise: Adaptive Moment Estimation (Adam) and convolutional neural networks (CNNs)



Prof. Dr. Mikhail Mikhaseenko

Assistants: Hendric Jonas, Christoph Tönnis

29 September 2026

Indico schedule

RUB Moodle

Part 1: Adaptive Moment Estimation (Adam)

The problem and update rules

Minimize the function

$$f(x) = (x - 3)^2 \quad f'(x) = 2(x - 3) \quad (1)$$

Its minimum is at $x^* = 3$. Start from $x_0 = 0$, with $m_0 = v_0 = 0$, and use

$$\alpha = 0.1 \quad \beta_1 = 0.9 \quad \beta_2 = 0.999 \quad \epsilon = 10^{-8} \quad (2)$$

At iteration t , calculate the following quantities in this order:

$$\begin{aligned} g_t &= f'(x_{t-1}) \\ m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \\ \hat{m}_t &= \frac{m_t}{1 - \beta_1^t} \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \\ x_t &= x_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \end{aligned} \quad (3)$$

Here m_t tracks gradients and v_t tracks squared gradients. The hats denote bias-corrected values. Keep at least six decimal places in intermediate calculations; round the reported results at the end. The small stabilizer ϵ is added **outside** the square root.

Exercise 1 - Iteration 1: guided calculation

Starting from $x_0 = m_0 = v_0 = 0$, calculate:

1. **Gradient.** Evaluate $g_1 = 2(x_0 - 3)$.
2. **First moment.** Use $m_1 = 0.9m_0 + 0.1g_1$.
3. **Second moment.** Use $v_1 = 0.999v_0 + 0.001g_1^2$.
4. **Bias correction.** Calculate $\hat{m}_1 = m_1/(1 - 0.9)$ and $\hat{v}_1 = v_1/(1 - 0.999)$.
5. **Parameter update.** Calculate

$$x_1 = x_0 - 0.1 \frac{\hat{m}_1}{\sqrt{\hat{v}_1} + 10^{-8}}$$

Record your answers:

Quantity	Iteration 1
g_1	
m_1	
v_1	
$\hat{m}_1$	
$\hat{v}_1$	
x_1	

Exercise 2 - Iteration 2

Use your previous values for x_1, m_1, v_1 . Repeat the five steps:

1. Calculate $g_2 = 2(x_1 - 3)$.
2. Calculate $m_2 = 0.9m_1 + 0.1g_2$.
3. Calculate $v_2 = 0.999v_1 + 0.001g_2^2$.
4. Correct the bias using $1 - 0.9^2$ and $1 - 0.999^2$.
5. Calculate $x_2 = x_1 - 0.1\hat{m}_2/(\sqrt{\hat{v}_2} + 10^{-8})$.

Quantity	Iteration 2
g_2	
m_2	
v_2	
$\hat{m}_2$	
$\hat{v}_2$	
x_2	

Exercise 3 - Iteration 3

Perform the third iteration using the same update rules, now with the bias-correction denominators $1 - 0.9^3$ and $1 - 0.999^3$. Complete the table.

Quantity	Iteration 3
x_2	
g_3	
m_3	
v_3	
$\hat{m}_3$	
$\hat{v}_3$	
x_3	

Part 2: Convolutional neural networks

Exercise 4: Images are numbers

A computer does not see an image in the same way that we do. A grayscale image can be represented as a matrix of numbers. For example, consider the following 5×5 grayscale image:

$$X = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Here, 0 represents a dark pixel and 1 represents a bright pixel. The matrix therefore describes a simple bright pattern in the centre of the image.

A colour image usually contains three channels: red, green, and blue. A colour image with height H and width W can therefore be represented as an array with dimensions $H \times W \times 3$.

Task 1

Suppose a grayscale image has a resolution of 28×28 pixels.

1. How many pixel values does the image contain?
2. How many values would a 28×28 RGB image contain?
3. Why might it be useful for a neural network to preserve the spatial arrangement of the pixel values?

Exercise 5: Looking for patterns with a filter

Instead of processing every pixel independently, a CNN looks for local patterns. One way to do this is with a small matrix called a **filter**, **kernel**, or **convolution kernel**.

Consider the filter

$$K = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}.$$

We place this filter over a small region of an image and multiply corresponding entries. The resulting values are then added together to produce a single output value.

For example, suppose the image region is

$$P = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

The element-wise multiplication gives

$$P \odot K = \begin{bmatrix} 0(-1) & 0(-1) & 0(-1) \\ 0(0) & 1(0) & 1(0) \\ 1(1) & 1(1) & 1(1) \end{bmatrix}.$$

Adding all entries gives

$$0 + 0 + 0 + 0 + 0 + 0 + 1 + 1 + 1 = 3.$$

The filter therefore produces the output value 3 for this particular image region.

The same calculation can be performed at different positions in the image. In this way, one filter produces a collection of output values called a **feature map**.

Task 2

Calculate the result for the following image patch using the same filter:

$$P = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Show the element-wise multiplication and calculate the final sum.

Task 3

Now calculate the result for

$$P = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Compare your result with Task 2.

What does the difference between the two results tell you about the pattern detected by the filter?

Exercise 6: From one filter to a feature map - 15 minutes

Let us now apply a filter to a complete image. Consider the following 5×5 image:

$$X = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the filter

$$K = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}.$$

For simplicity, we use a stride of 1 and no padding.

The filter starts in the upper-left corner of the image and considers the first 3×3 region:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

The first output value is

$$\begin{aligned} y_{1,1} &= (0 \cdot 1) + (0 \cdot 1) + (0 \cdot 1) \\ &\quad + (0 \cdot 0) + (1 \cdot 0) + (1 \cdot 0) \\ &\quad + (0 \cdot (-1)) + (1 \cdot (-1)) + (1 \cdot (-1)) \\ &= -2. \end{aligned}$$

The filter is then moved one pixel to the right and the calculation is repeated. The process continues across the image and then down to the next row.

Since the input has size 5×5 and the filter has size 3×3 , the output has size

$$(5 - 3 + 1) \times (5 - 3 + 1) = 3 \times 3.$$

This output is the feature map produced by the filter.

Task 4

Calculate the complete feature map. The first value has already been calculated for you.

$$F = \begin{bmatrix} -2 & ______ & ______ \\ ______ & ______ & ______ \\ ______ & ______ & ______ \end{bmatrix}.$$

For each position, identify the corresponding 3×3 image region and calculate the sum of the element-wise products.

Task 5

Look at the values in your feature map.

- Which image regions produce large positive values?
 - Which regions produce large negative values?
 - Based on these results, what kind of visual pattern is the filter detecting?
-

Exercise 7: Different filters detect different features - 10 minutes

A CNN does not normally use just one filter. It learns many different filters, and different filters can respond to different visual patterns.

For example, consider the following three filters:

$$K_1 = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \quad K_2 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix},$$

and

$$K_3 = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}.$$

The first filter responds strongly to certain horizontal intensity changes, while the second responds to vertical intensity changes. The third can respond to diagonal or corner-like patterns.

The important point is that the network does not need to be explicitly told which features to look for. During training, the values inside the filters are learned from the training data.

In an early CNN layer, some filters may learn to respond to edges, corners, simple textures, or changes in brightness. Later layers can combine these simpler patterns into more complex representations.

For example, imagine that the input is a photograph of a cat. An early layer might detect edges. A later layer could combine several edges into a representation of a shape such as an ear. Further layers can combine information about different shapes and textures to form representations that are useful for recognizing the object.

This gives us an intuitive progression from pixels to simple features, more complex features, object parts, and finally object-level representations. This description is only an intuition; the representations learned by real CNNs do not necessarily correspond to these categories in such a simple way.

Task 6

Give one example of:

- a simple visual feature that an early CNN layer might detect;
 - a more complex visual feature that a later layer might represent.
-

Exercise 8: Putting the CNN together - 10 minutes

A simplified CNN consists of several types of operations. An image is first processed by convolutional layers, which produce feature maps. Activation functions are then applied to introduce non-linearity. Pooling may be used to reduce the spatial dimensions of the feature maps. Further convolutional layers can then construct more complex representations, which are eventually used to make a prediction.

1. Convolution

A convolution applies several learned filters to the input. If three filters are used, they produce three different feature maps:

$$X \longrightarrow \begin{cases} F_1 & \text{from filter } K_1, \\ F_2 & \text{from filter } K_2, \\ F_3 & \text{from filter } K_3. \end{cases}$$

The filters contain numerical parameters that are adjusted during training.

2. Activation

A common activation function is the ReLU function:

$$\text{ReLU}(x) = \max(0, x).$$

For example,

$$[-2, 3, -1, 5]$$

becomes

$$[0, 3, 0, 5].$$

The activation function introduces non-linearity, which allows the network to learn more complex relationships.

3. Pooling

Pooling reduces the spatial dimensions of a feature map. For example, 2×2 max pooling selects the largest value from each 2×2 region:

$$\begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \longrightarrow 4.$$

Pooling can therefore reduce the amount of computation while retaining strong responses.

4. Multiple layers

A simplified view of a CNN is:

pixels $\longrightarrow$ local features $\longrightarrow$ more complex features $\longrightarrow$ object representation $\longrightarrow$ prediction.

The filters in the network are learned during training. For example, if a CNN is trained to distinguish cats from dogs, the training process adjusts the filter values so that the resulting representations become useful for making that distinction.

Task 7

Complete the following statement in your own words:

A convolutional neural network is useful for images because _____

Write two or three sentences. Try to include the ideas of local patterns, filters, feature maps, and learned representations.