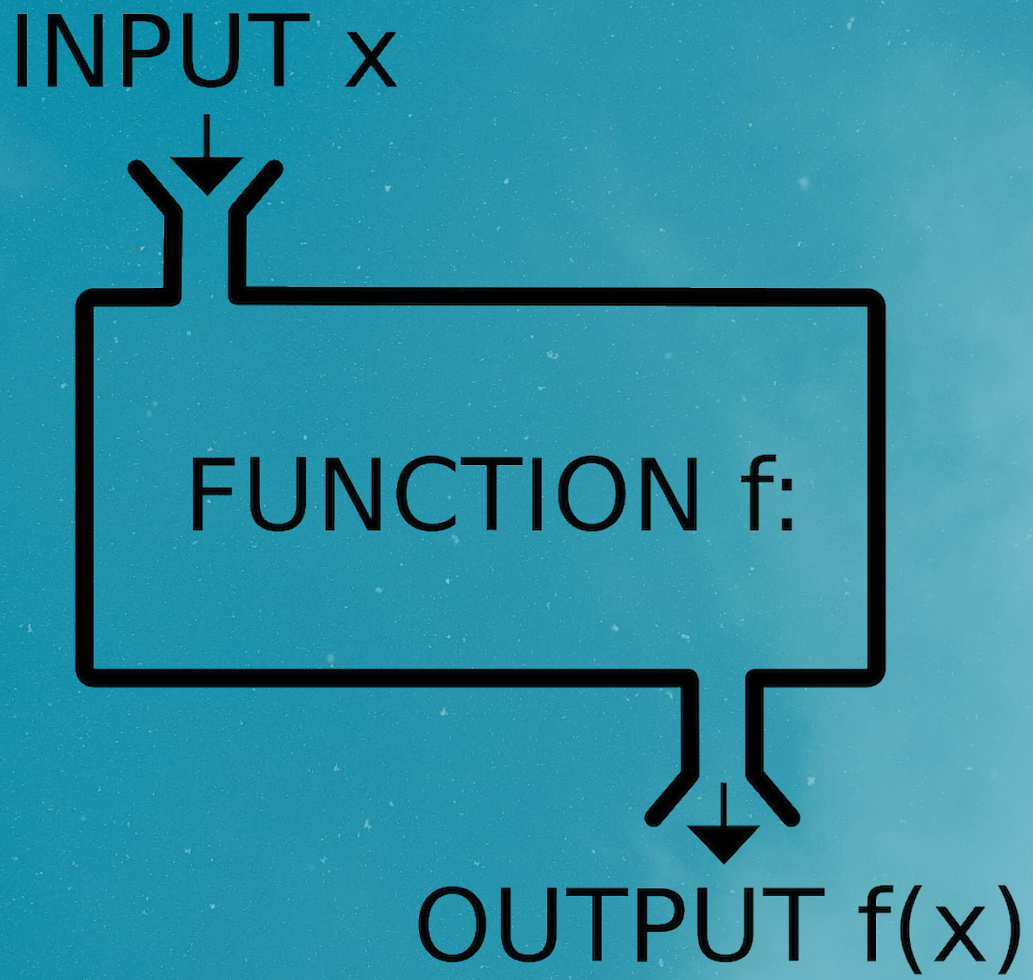
KI & Robotik

Norman Koch


ScaDS.AI
DRESDEN LEIPZIG


INPUT x
FUNCTION f:
OUTPUT f(x)

$$f(x) = x^2$$

```
def x_quadrat (x):
    return x ** 2
```

x	$f(x) = x^2$
-10	100
-9	81
-8	64
-7	49
-6	36
-5	25
-4	16
-3	9
-2	4
-1	1
0	0
1	1
2	4
3	9
4	16
5	25
6	36
7	49
8	64
9	81
10	100

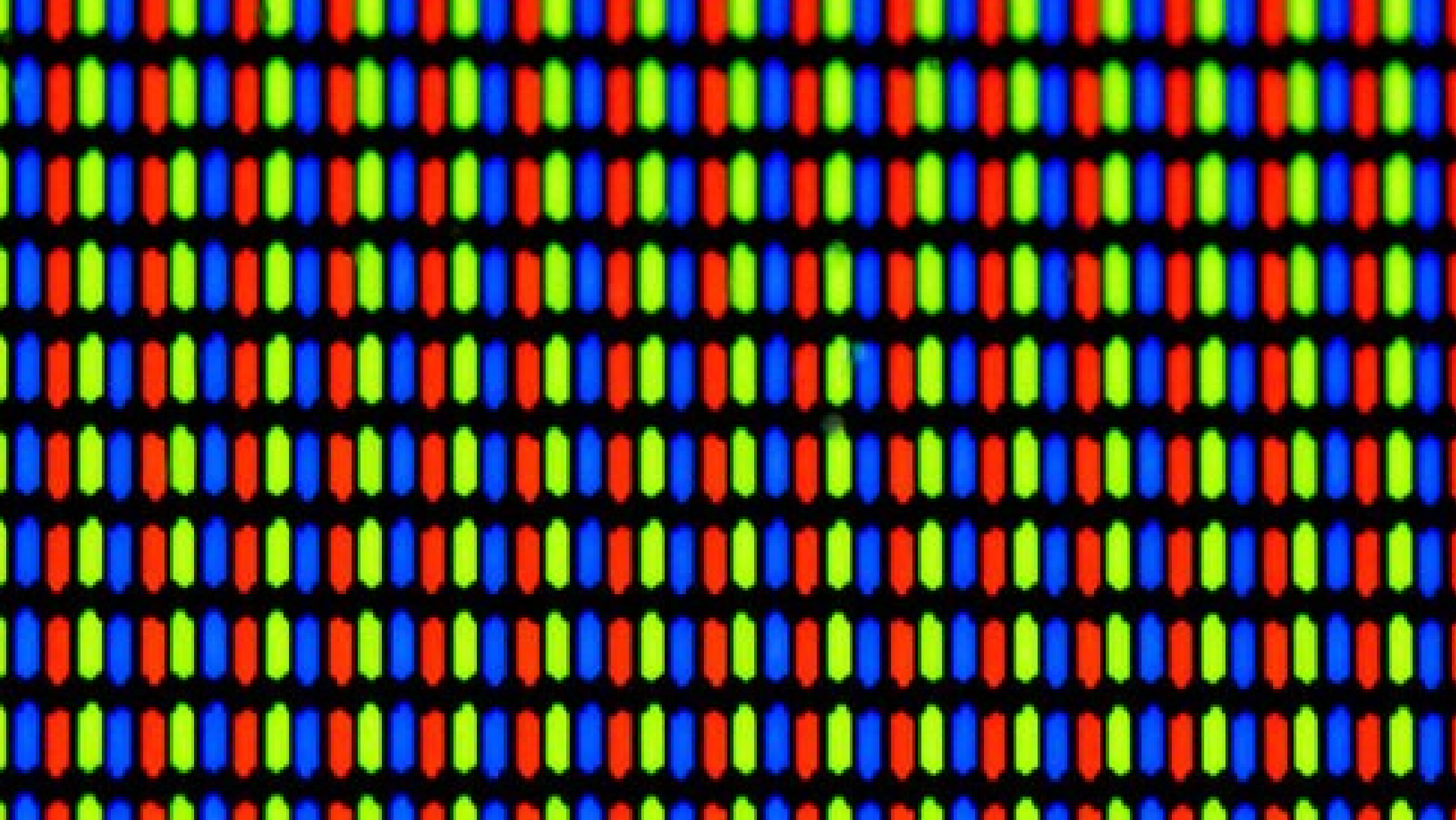
Problem:

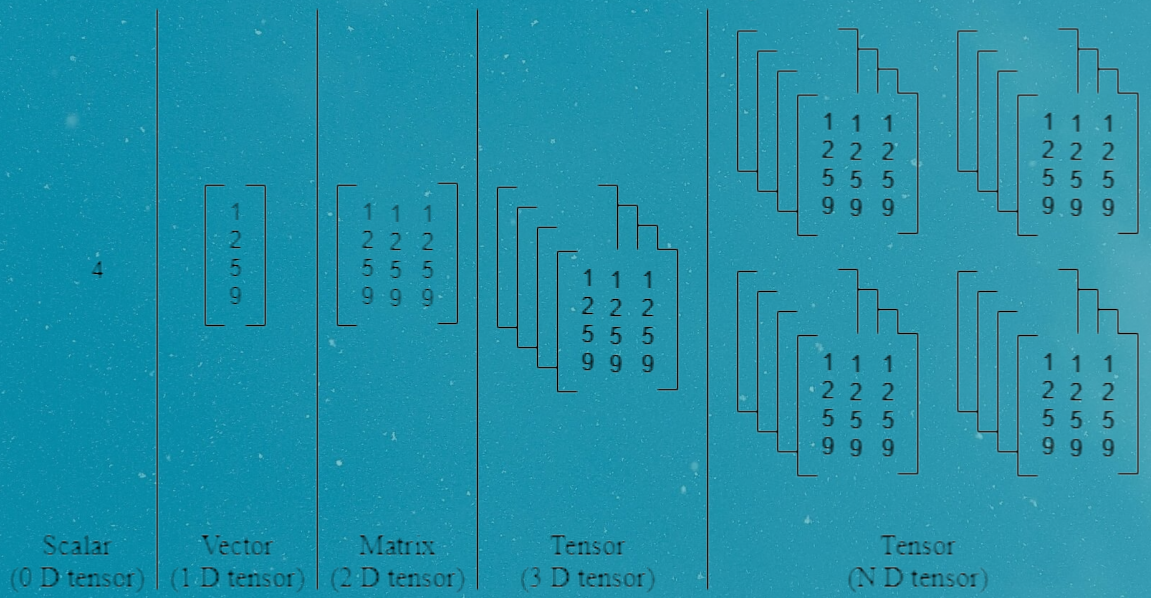
Klassische Algorithmen sind exakt, aber nicht für alle Probleme geeignet

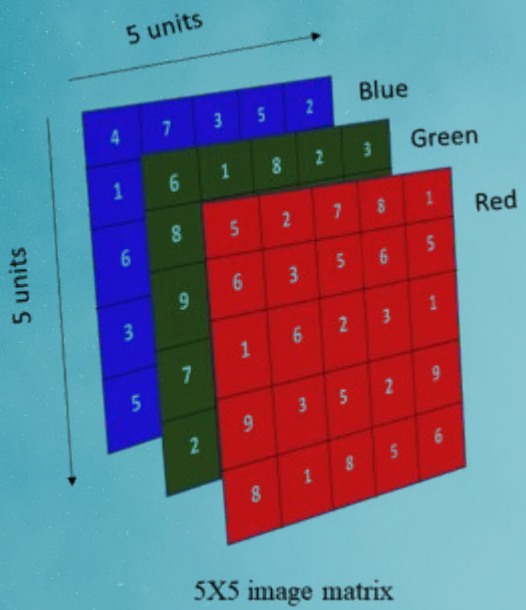
KIs sind nicht exakt

Mit KI: $4^2 \approx 16$, aber vielleicht auch 15 oder 17

Wie würde man klassisch eine Katze in Bildern erkennen?

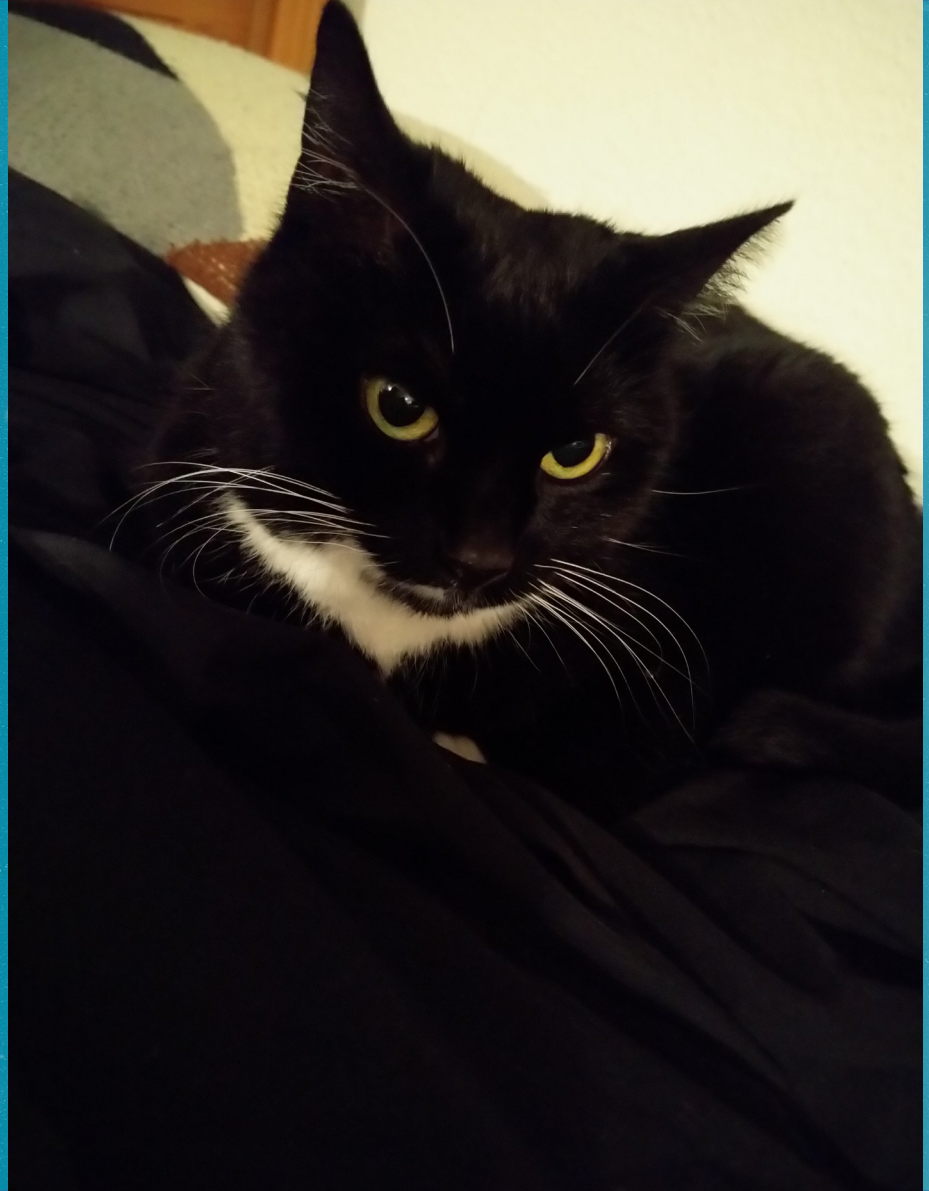






5X5 image matrix

$$X =$$

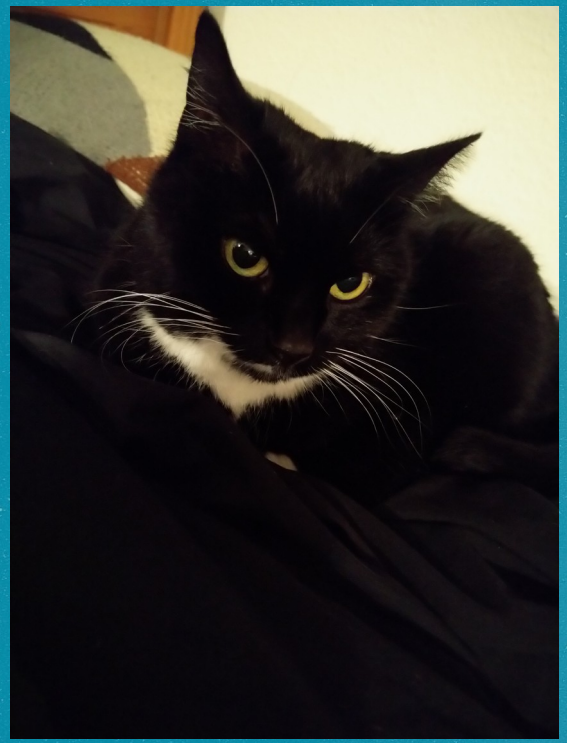


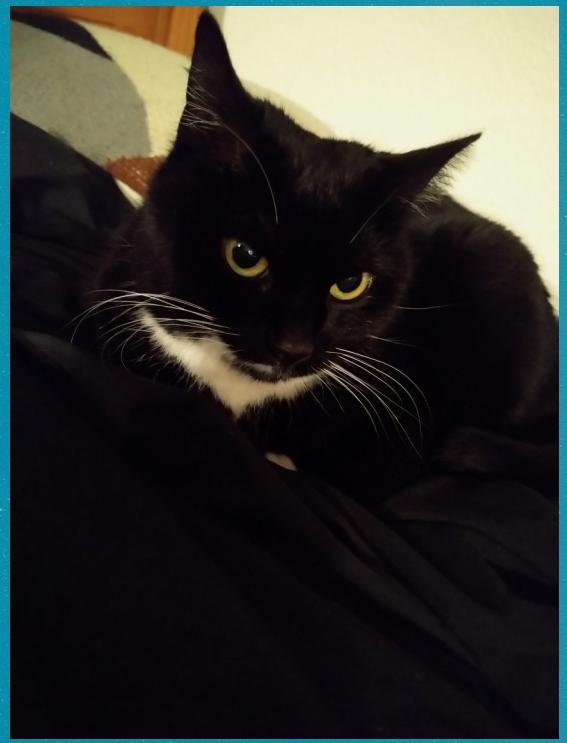
$$=$$

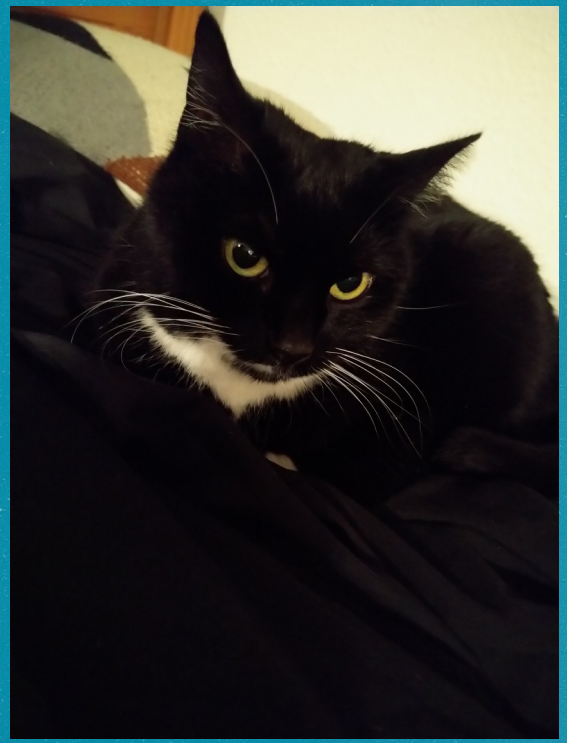
$$R = \begin{pmatrix}
217 & 218 & 216 & 220 & 218 & \dots & 7 & 8 & 4 & 8 & 9 \\
219 & 219 & 216 & 214 & 222 & \dots & 8 & 8 & 5 & 5 & 4 \\
215 & 218 & 218 & 225 & 220 & \dots & 5 & 4 & 5 & 6 & 7 \\
221 & 217 & 216 & 223 & 218 & \dots & 9 & 7 & 7 & 7 & 7 \\
217 & 218 & 219 & 220 & 220 & \dots & 8 & 6 & 6 & 7 & 8 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
49 & 51 & 53 & 49 & 50 & \dots & 13 & 12 & 11 & 13 & 11 \\
46 & 51 & 50 & 45 & 51 & \dots & 10 & 11 & 11 & 11 & 11 \\
50 & 51 & 41 & 43 & 53 & \dots & 11 & 11 & 11 & 11 & 12 \\
46 & 47 & 53 & 43 & 46 & \dots & 12 & 11 & 11 & 11 & 12 \\
44 & 49 & 42 & 42 & 51 & \dots & 6 & 11 & 12 & 10 & 12
\end{pmatrix}$$

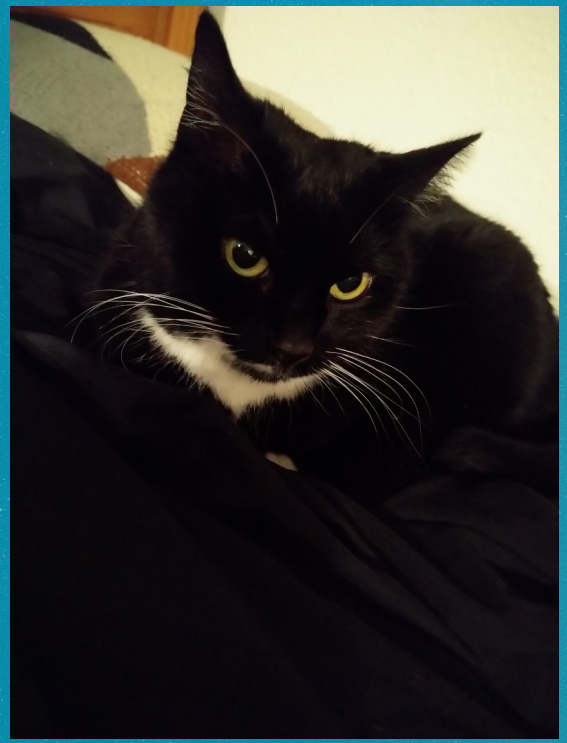
$$G = \begin{pmatrix}
206 & 207 & 209 & 213 & 211 & \dots & 6 & 7 & 3 & 7 & 8 \\
208 & 208 & 209 & 207 & 215 & \dots & 7 & 7 & 4 & 4 & 3 \\
207 & 210 & 210 & 217 & 214 & \dots & 4 & 3 & 4 & 5 & 6 \\
213 & 210 & 208 & 216 & 212 & \dots & 8 & 6 & 6 & 6 & 6 \\
210 & 211 & 213 & 215 & 214 & \dots & 7 & 5 & 5 & 6 & 7 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
29 & 31 & 33 & 29 & 30 & \dots & 11 & 10 & 9 & 11 & 9 \\
28 & 33 & 32 & 27 & 31 & \dots & 8 & 9 & 9 & 9 & 9 \\
32 & 33 & 25 & 27 & 33 & \dots & 9 & 9 & 9 & 9 & 10 \\
30 & 31 & 37 & 27 & 26 & \dots & 10 & 9 & 9 & 9 & 10 \\
28 & 33 & 26 & 26 & 31 & \dots & 4 & 9 & 10 & 8 & 10
\end{pmatrix}$$

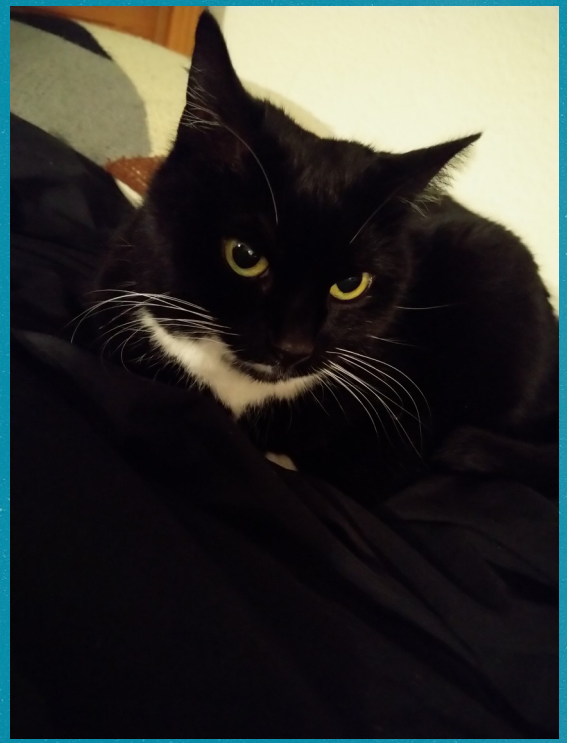
$$B = \begin{pmatrix}
184 & 185 & 183 & 187 & 185 & \dots & 11 & 12 & 8 & 12 & 13 \\
186 & 186 & 183 & 181 & 189 & \dots & 12 & 12 & 9 & 9 & 8 \\
184 & 187 & 187 & 194 & 190 & \dots & 9 & 8 & 9 & 10 & 11 \\
190 & 184 & 185 & 190 & 188 & \dots & 13 & 11 & 11 & 11 & 11 \\
184 & 183 & 187 & 186 & 188 & \dots & 12 & 10 & 10 & 12 & 13 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
28 & 30 & 34 & 30 & 31 & \dots & 14 & 13 & 12 & 14 & 12 \\
28 & 33 & 32 & 27 & 32 & \dots & 11 & 12 & 12 & 12 & 12 \\
32 & 33 & 25 & 27 & 34 & \dots & 12 & 12 & 12 & 12 & 13 \\
30 & 31 & 37 & 27 & 27 & \dots & 13 & 12 & 12 & 12 & 13 \\
28 & 33 & 26 & 26 & 32 & \dots & 7 & 12 & 13 & 11 & 13
\end{pmatrix}$$

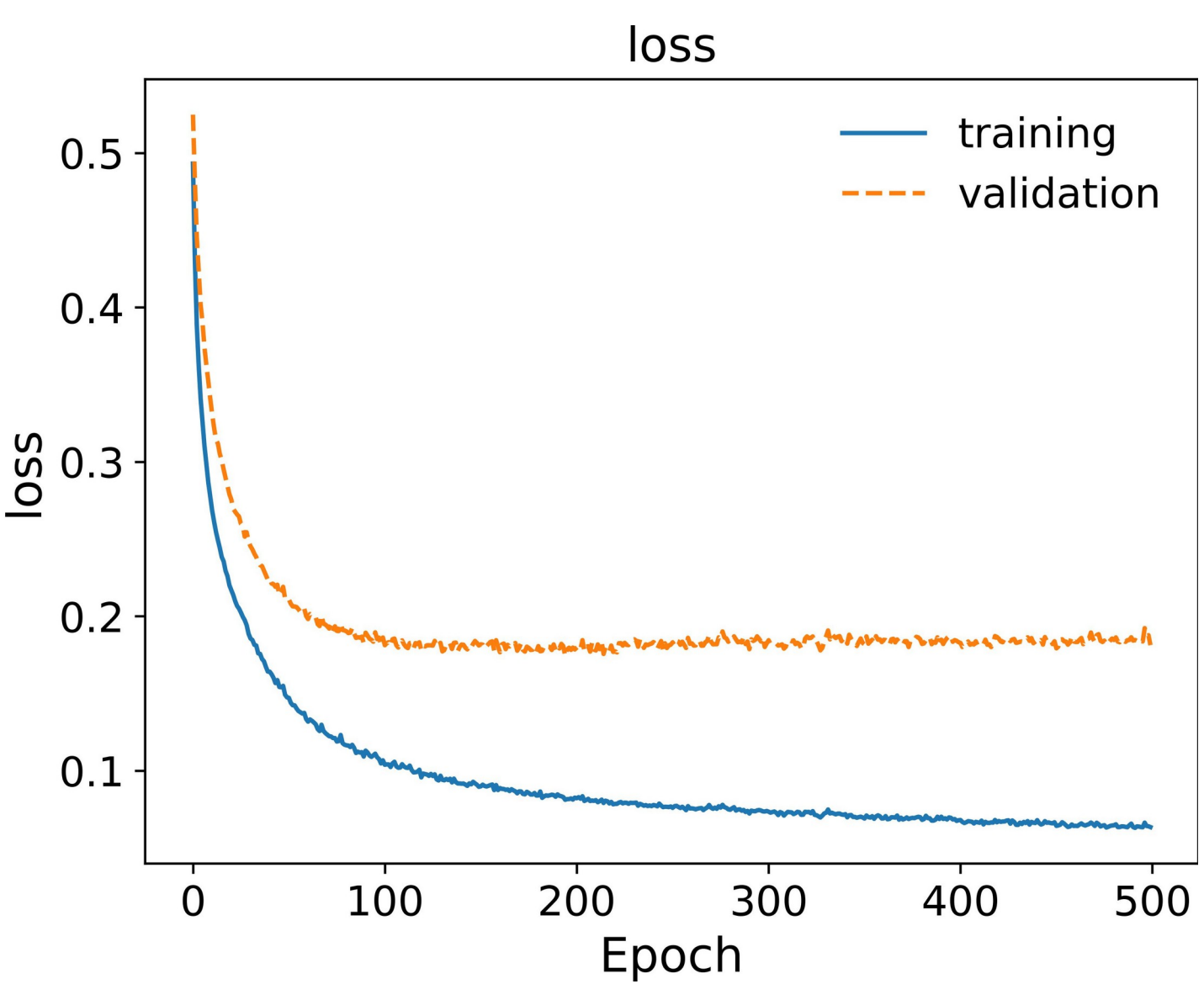
F() = $\begin{bmatrix} \text{Katze} & \text{Hund} & \text{Vogel} & \text{Fisch} & \text{Maus} \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$

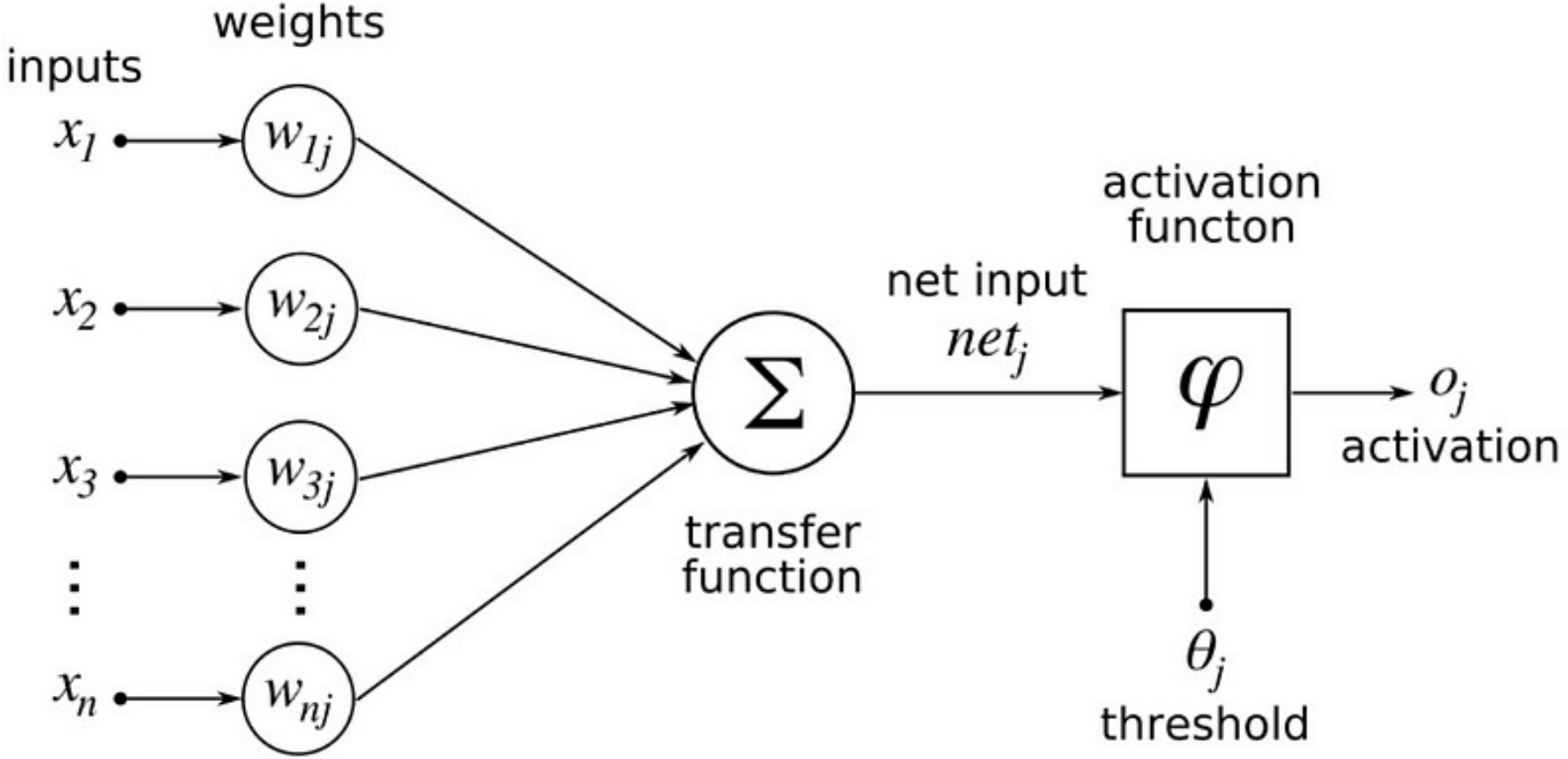
F() = $\begin{bmatrix} \text{Katze} & \text{Hund} & \text{Vogel} & \text{Fisch} & \text{Maus} \\ 0.20 & 0.19 & 0.21 & 0.18 & 0.22 \end{bmatrix}$

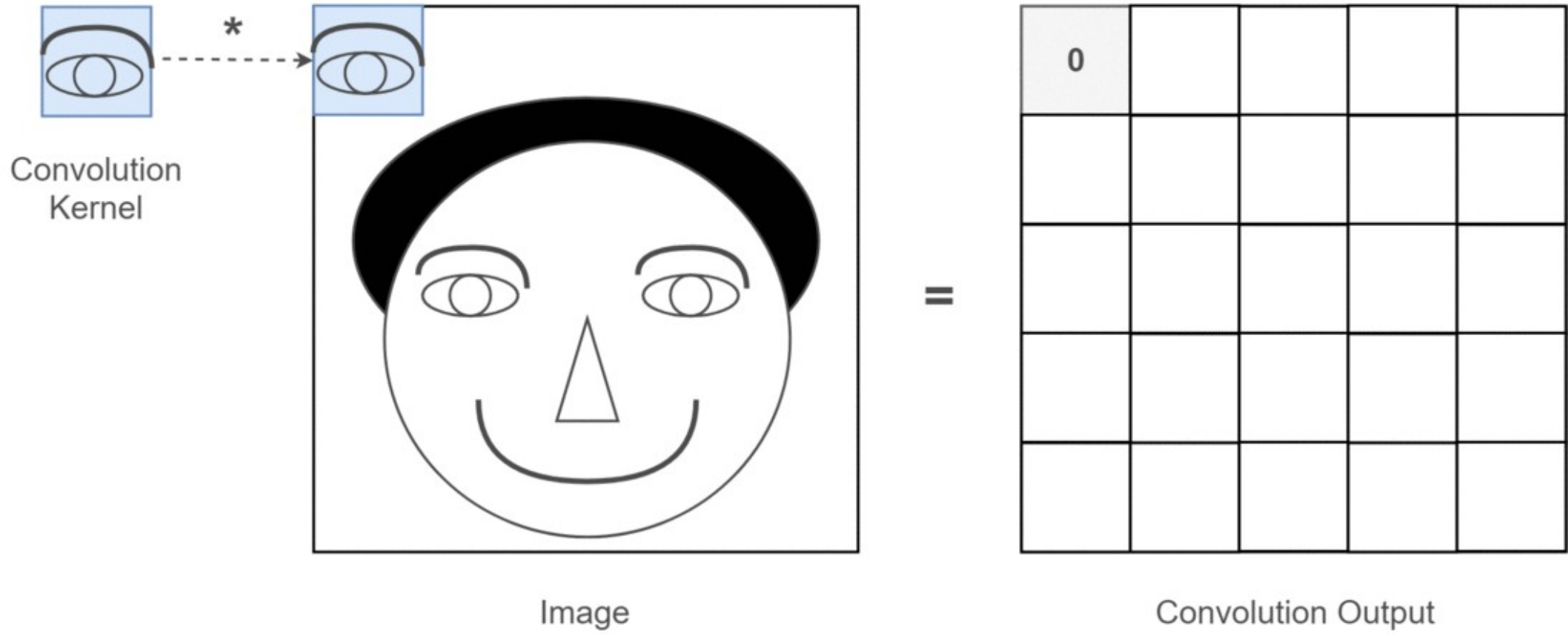
F() = $\begin{bmatrix} \text{Katze} & \text{Hund} & \text{Vogel} & \text{Fisch} & \text{Maus} \\ 0.25 & 0.15 & 0.20 & 0.20 & 0.20 \end{bmatrix}$

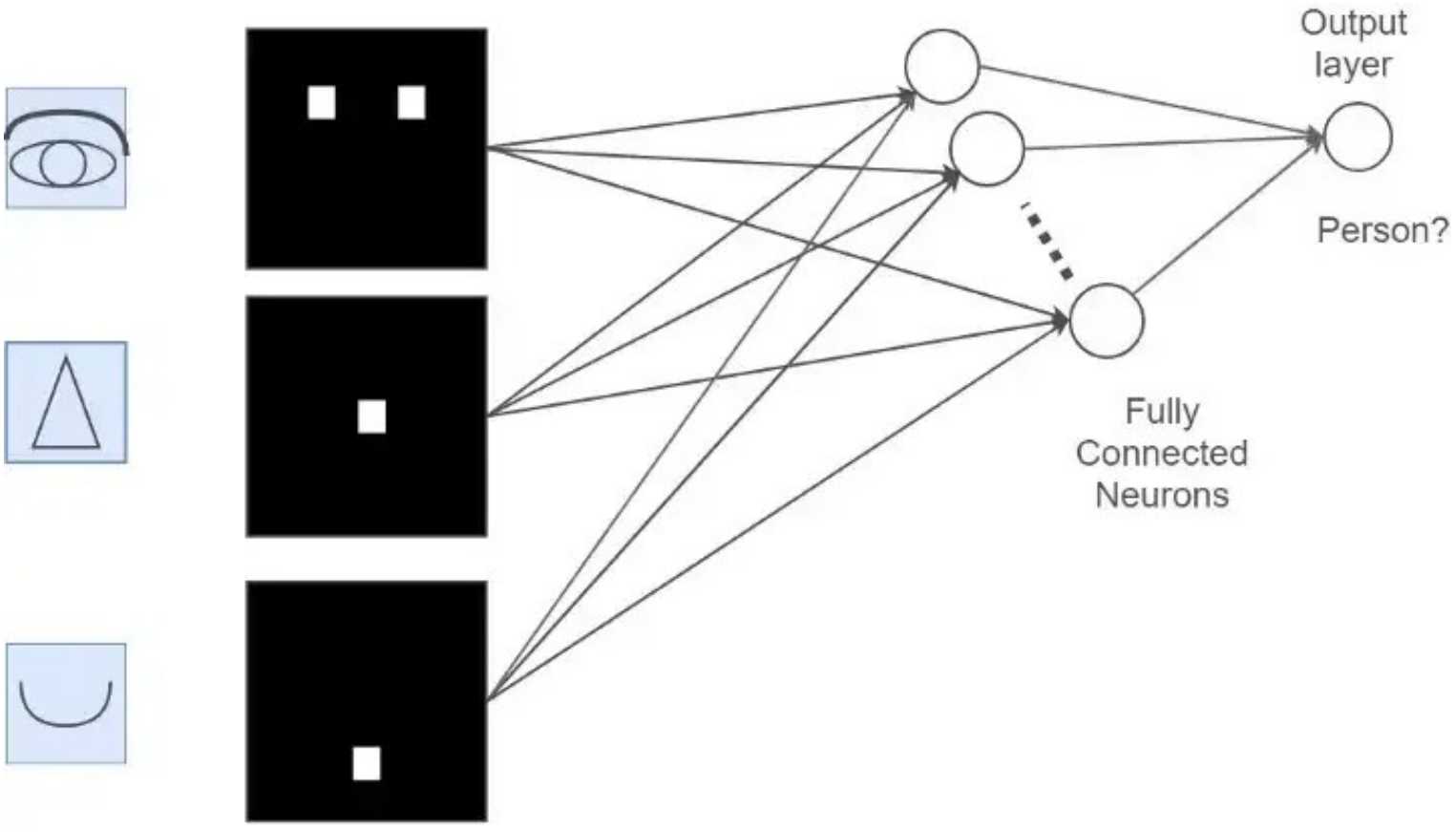
F() = $\begin{bmatrix} \text{Katze} & \text{Hund} & \text{Vogel} & \text{Fisch} & \text{Maus} \\ 0.5 & 0.12 & 0.15 & 0.12 & 0.11 \end{bmatrix}$

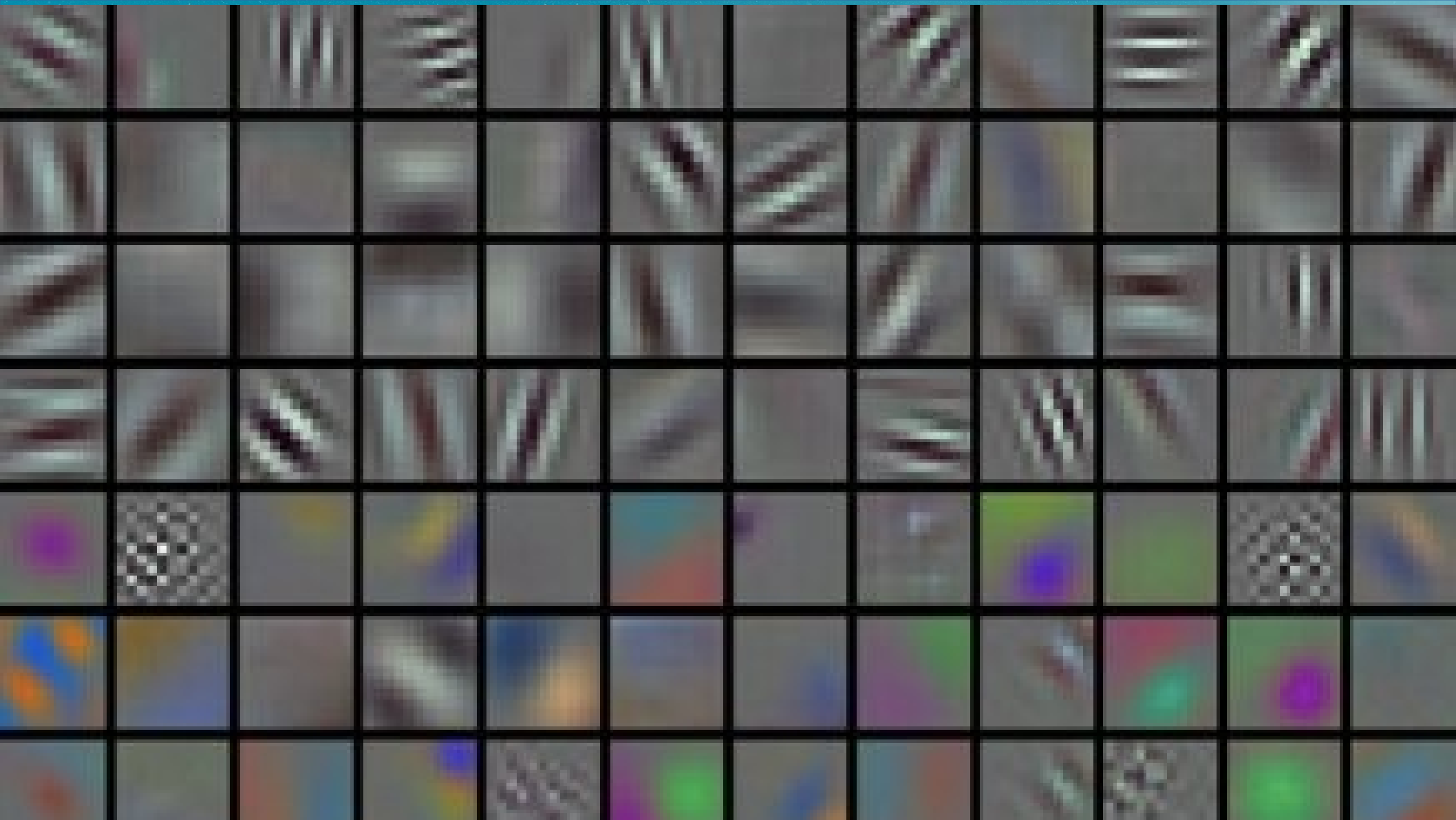
F() = $\begin{bmatrix} \text{Katze} & \text{Hund} & \text{Vogel} & \text{Fisch} & \text{Maus} \\ 0.80 & 0.07 & 0.03 & 0.02 & 0.10 \end{bmatrix}$

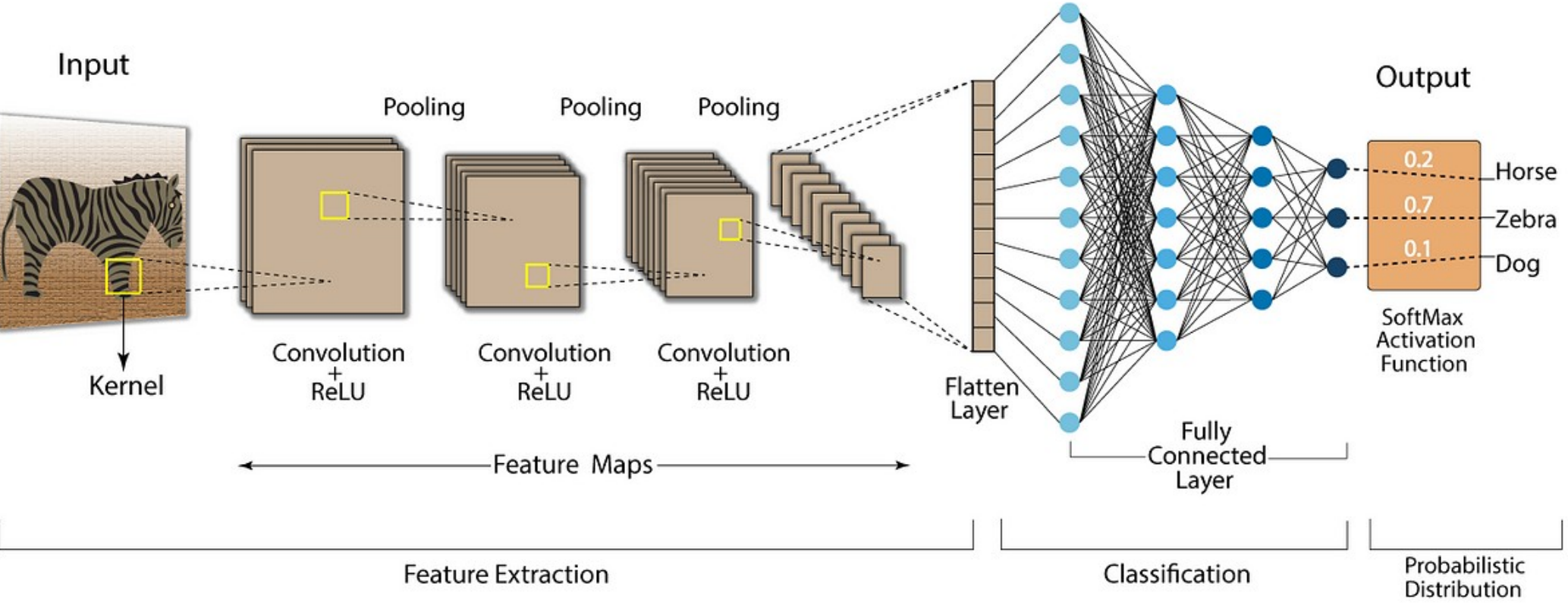

loss
training
validation
0.5
0.4
0.3
0.2
0.1
loss
0
100
200
300
400
500
Epoch

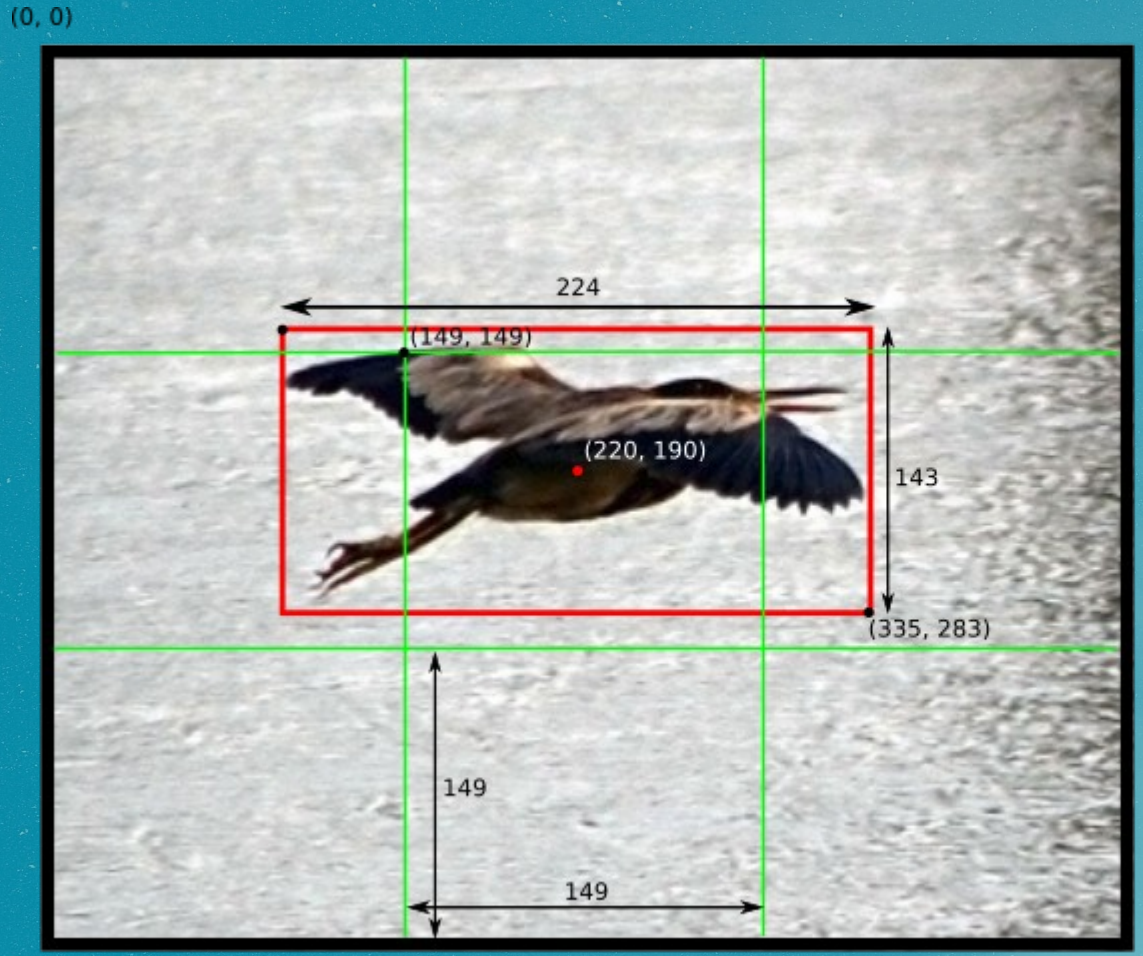

weights
inputs
x_1
x_2
x_3
x_n
w_{1j}
w_{2j}
w_{3j}
w_{nj}
Σ
transfer
function
net input
net_j
activation
functon
φ
o_j
activation
θ_j
threshold


*
Convolution
Kernel
=
0
Image
Convolution Output


Output
layer
Person?
Fully
Connected
Neurons
Convolution Outputs
corresponding to the
shown filters




Input
Pooling
Pooling
Pooling
Output
0.2
Horse
0.7
Zebra
0.1
Dog
SoftMax
Activation
Function
Kernel
Convolution
+
ReLU
Convolution
+
ReLU
Convolution
+
ReLU
Flatten
Layer
Fully
Connected
Layer
Feature Maps
Feature Extraction
Classification
Probabilistic
Distribution

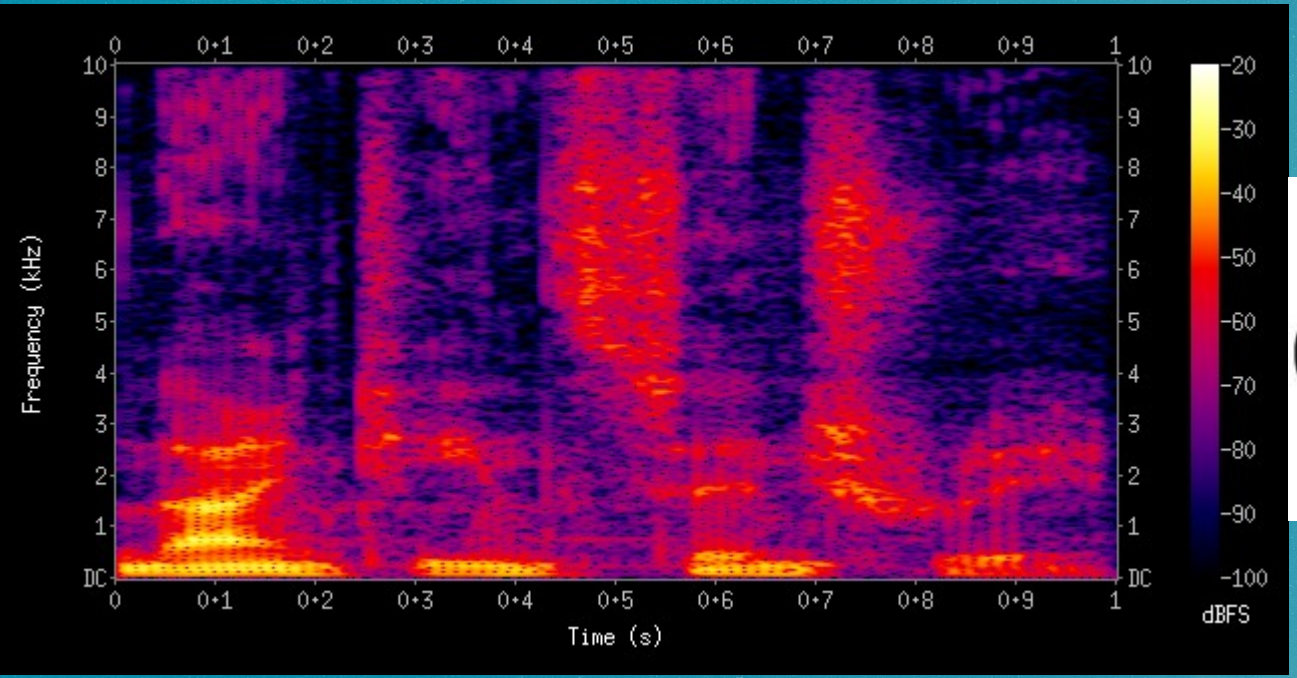


x = (220-149) / 149 = 0.48

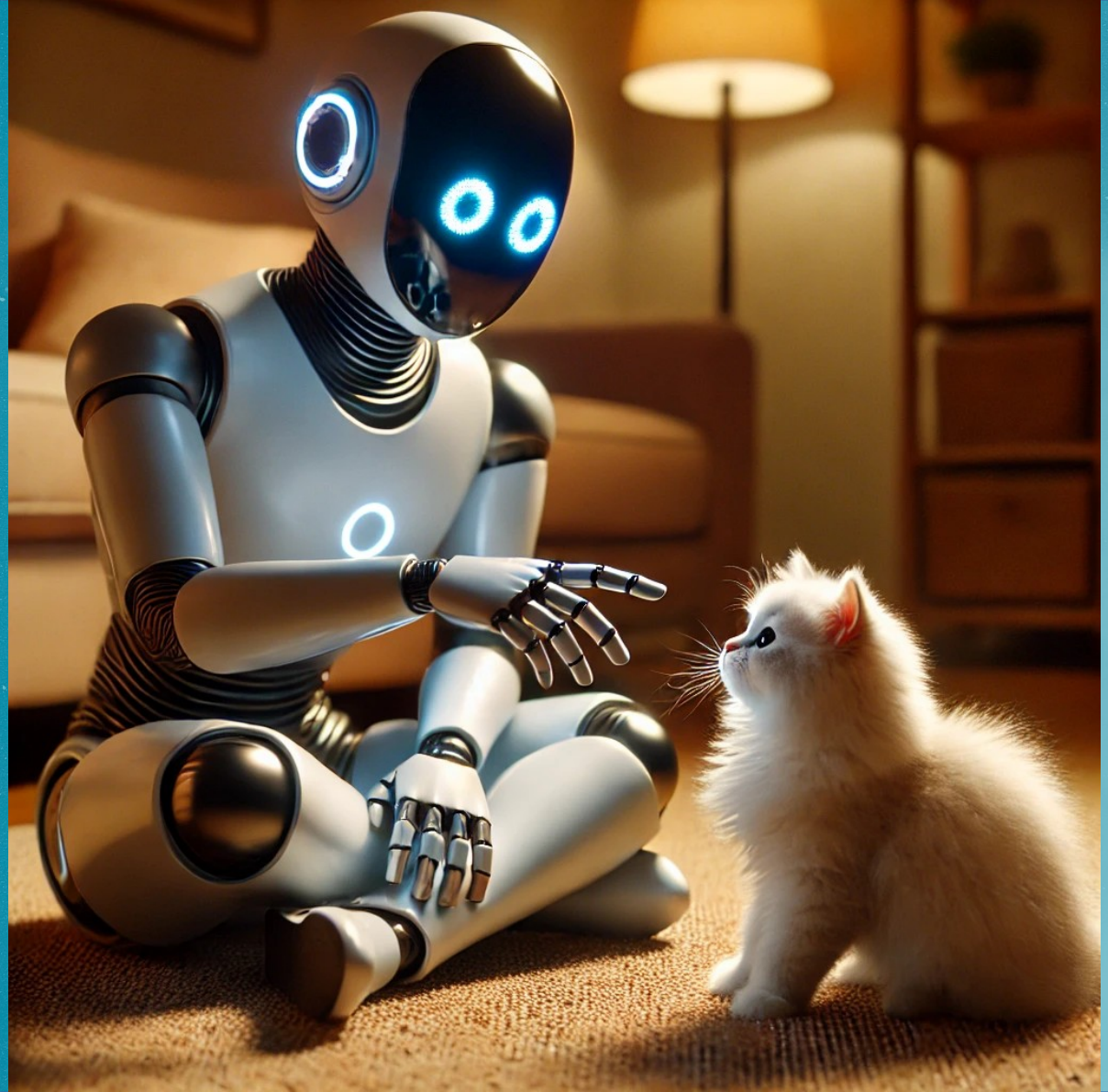
y = (190-149) / 149 = 0.28

w = 224 / 448 = 0.50

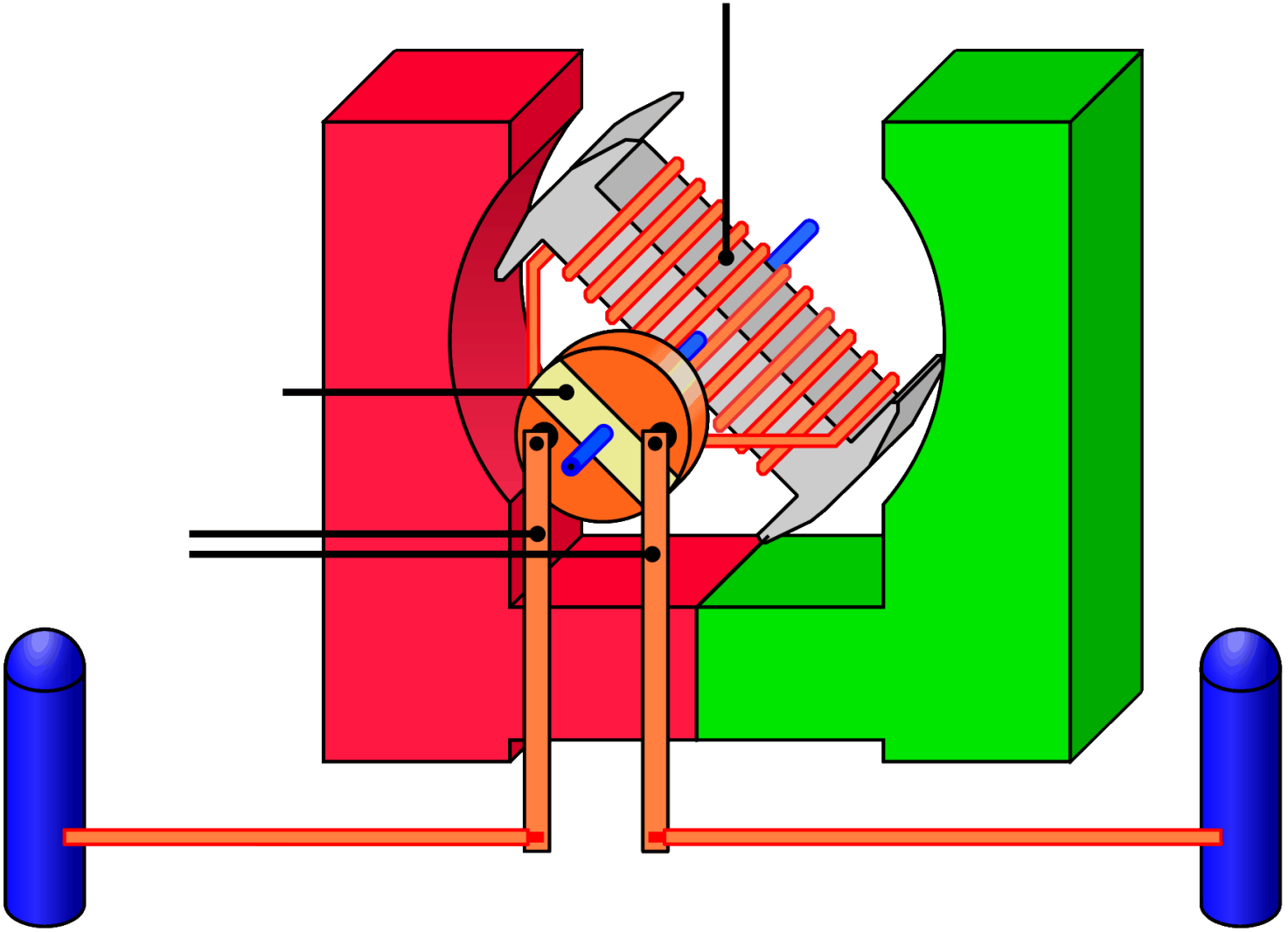
h = 143 / 448 = 0.32

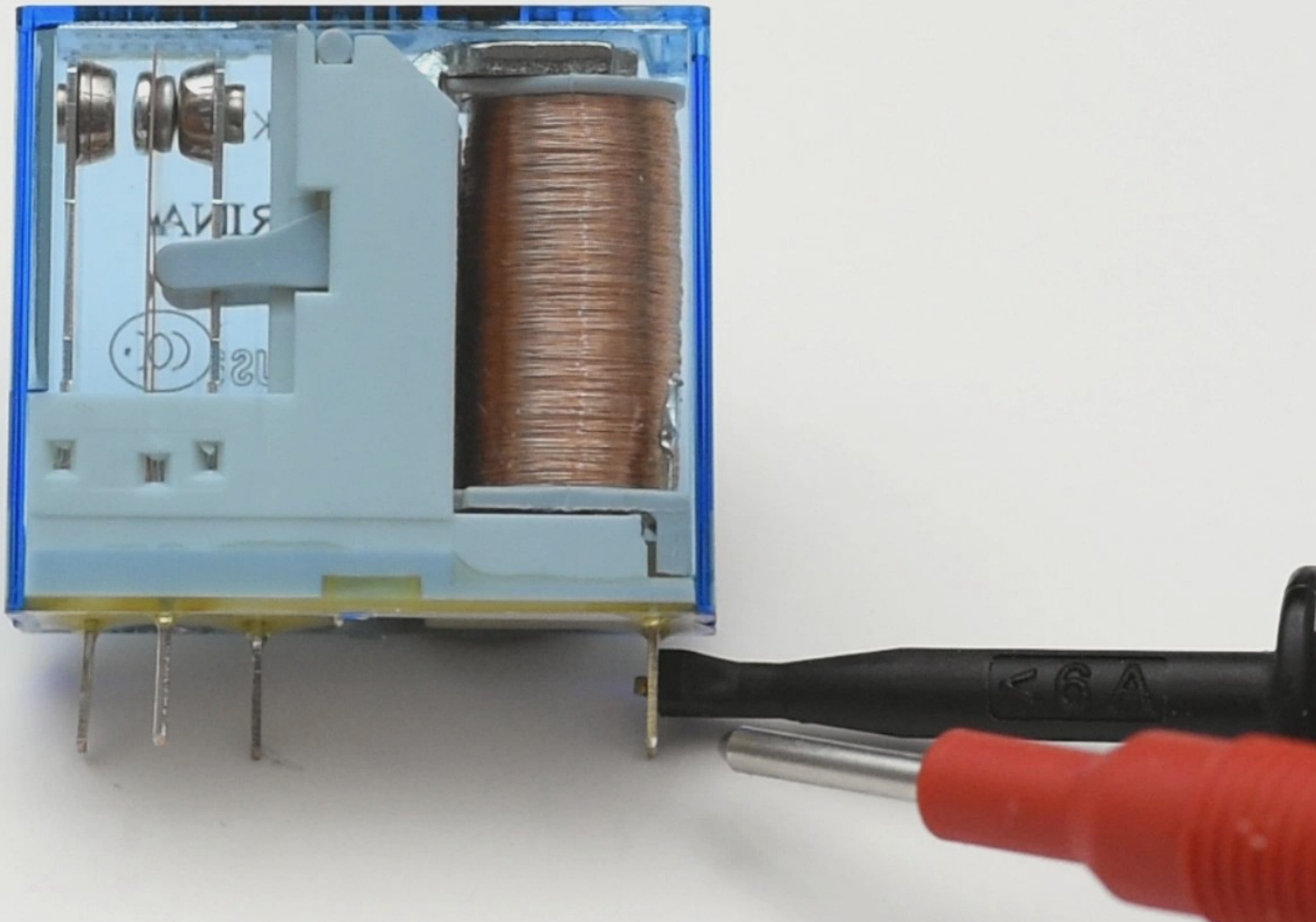

0
0·1
0·2
0·3
0·4
0·5
0·6
0·7
0·8
0·9
1
10
9
8
7
6
5
4
3
2
1
DC
Frequency (kHz)
Time (s)
-20
-30
-40
-50
-60
-70
-80
-90
-100
dBFS

ChatGPT

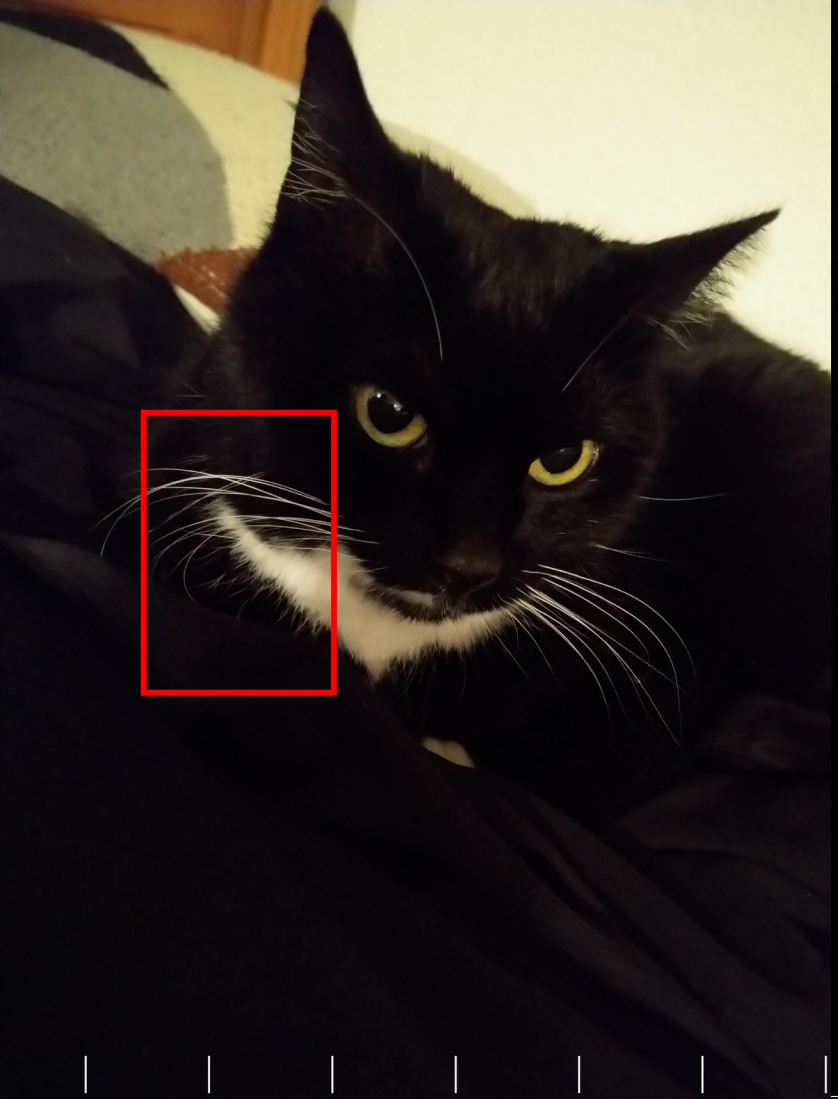



I've made a robot that screams!
AAAAAAAAAA
WHY?
AAAAAAAAA
AAAAAAAAA






ARDUINO
UNO R4
WIFI
DIGITAL (PWM~)
ANALOG IN
POWER
RESET



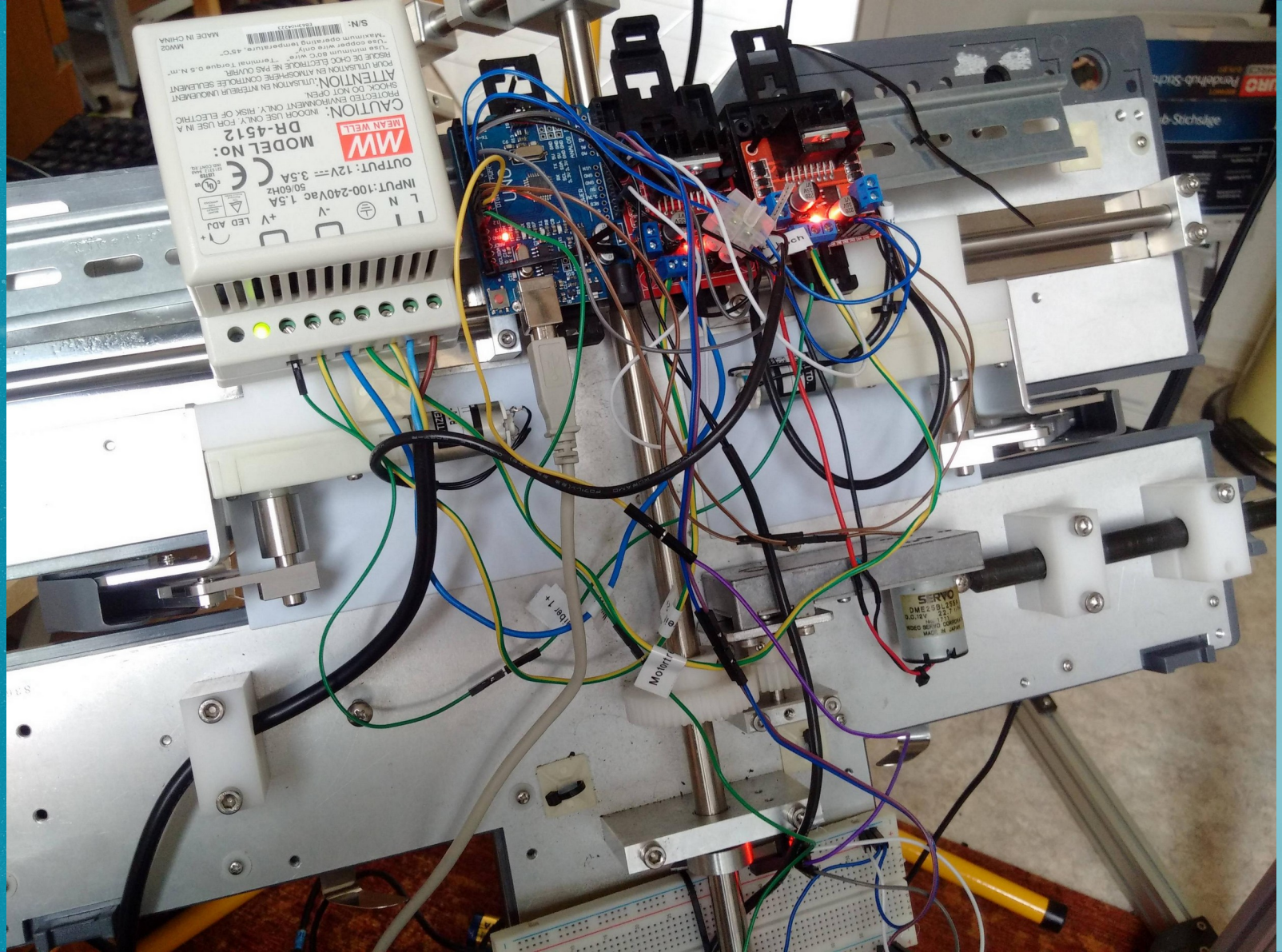

```arduino
int motorPin = 9;  // Beispiel Pin für den Motor

void setup() {
        pinMode(motorPin, OUTPUT);
        Serial.begin(9600);
}

void loop() {
        int angle = stroke_here();

        // Bewege den Motor, bis stroke_here() 0 zurückgibt
        if (angle != 0) {
                if (angle > 0) {
                        // Motor nach links bewegen (positive Winkel)
                        analogWrite(motorPin, 255);
                } else {
                        // Motor nach rechts bewegen (negative Winkel)
                        analogWrite(motorPin, 0);
                }
        } else {
                stroke();
        }

        delay(500);  // Kurze Pause
}
```


MEAN WELL
DR-4512
SERVO

asanAI

Thank you!