Zpracování obrazu pomocí neuronových sítí

Blok 2: Konvoluční neuronové sítě a jejich využití

Michal Vašinek

25.4.2025

Úlohy počítačového vidění

KLASIFIKACE OBRÁZKŮ

KOŤÁTKO?

PŘENOS STYLU

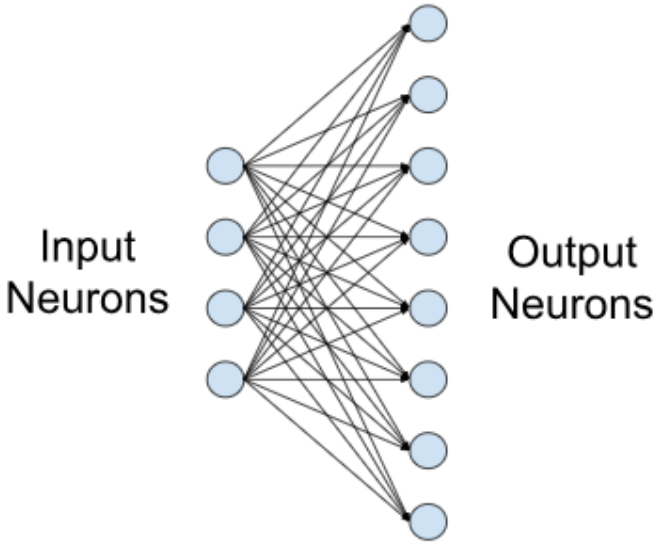
DETEKCE OBJEKTŮ

Učení na velkých obrázcích

64 x 64 x 3 = 12288

1000 x 1000 x 3 = 3 miliony

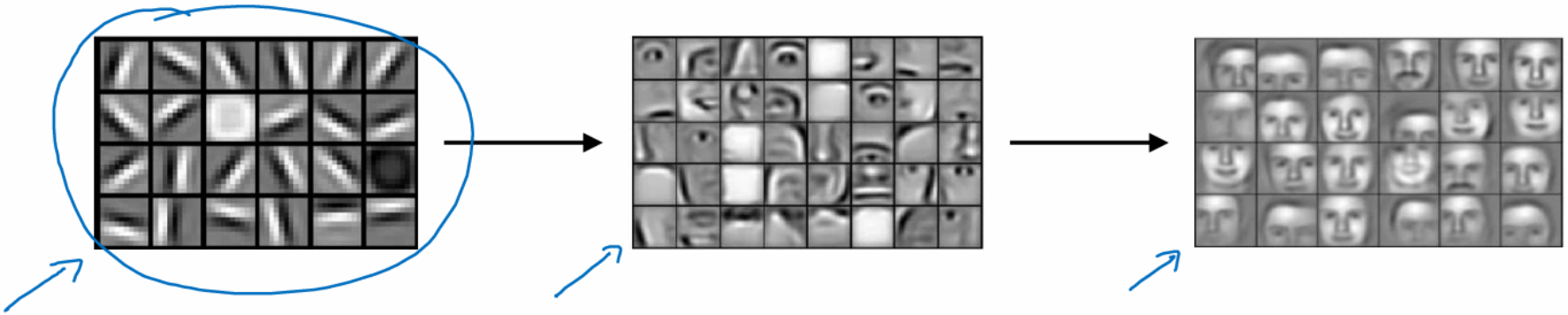
vstup - 12 288 hodnot pixelů
počet neuronů - 1000
počet vah - 12 milionů



vstup - 3 miliony hodnot pixelů
počet neuronů - 1000
počet vah - 3 miliardy

Skládání vzorů

- Základem je rozpoznávání malých vzorů
- Skládání vzorů do větších celků
- Využití hierarchické struktury neuronových sítí



Konvoluce - detekce hran

- Konvoluce jako operace pro detekci vzorů
- Využití filtrů(kernel) pro extrakci rysů.

3	2	1	5	6	7
0	1	4	2	3	8
9	6	2	1	7	5
5	3	8	6	4	2
1	7	0	3	9	0
4	2	5	8	1	6

Vstup

$*$

1	0	-1
1	0	-1
1	0	-1

Filtr

$=$

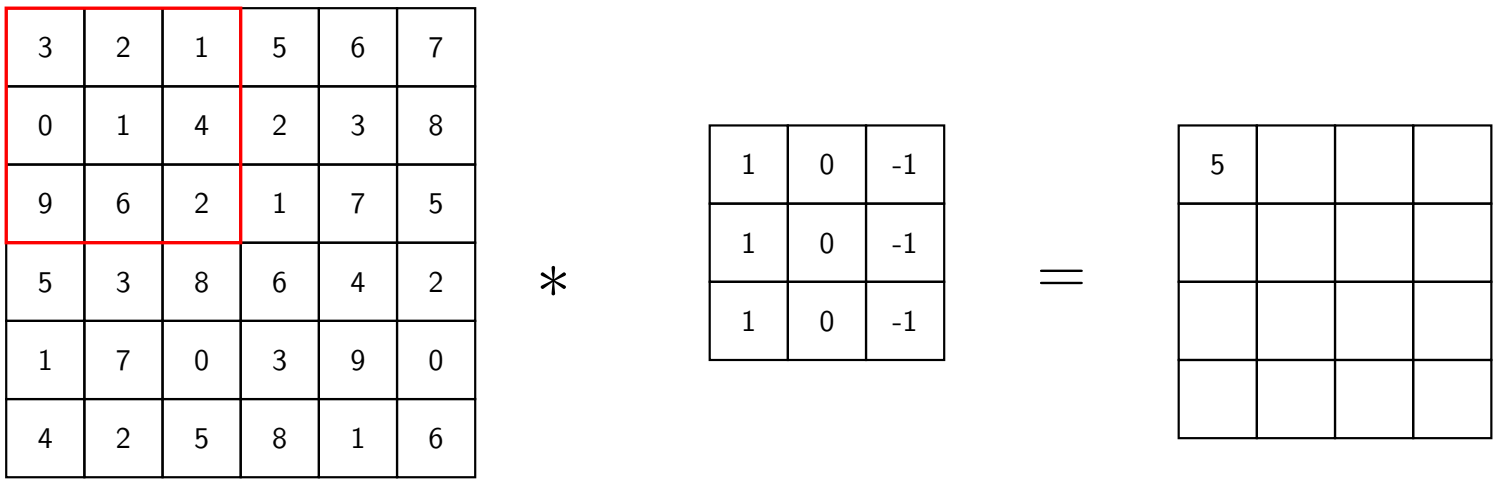
Výstup

Obrázek: Konvoluce s 3x3 filtrem na 6x6 vstupní matici.

Konvoluce - detekce hran

- Konvoluce jako operace pro detekci vzorů
- Využití filtrů(kernel) pro extrakci rysů.

$$3 * 1 + 9 * 1 - 1 * 1 - 1 * 4 - 1 * 2 = 5$$



Obrázek: Konvoluce s 3x3 filtrem na 6x6 vstupní matici.

Konvoluce - detekce hran

- Často se tato operace označuje jako Conv2D.

3	2	1	5	6	7
0	1	4	2	3	8
9	6	2	1	7	5
5	3	8	6	4	2
1	7	0	3	9	0
4	2	5	8	1	6

Vstup

$*$

1	0	-1
1	0	-1
1	0	-1

Filtr

$=$

5	1	-9	-12
0	1	0	-6
5	6	-10	3
-3	-5	-1	9

Výstup

Obrázek: Konvoluce s 3x3 filtrem na 6x6 vstupní matici.

Konvoluce - detekce hran

Google Colab - K vyzkoušení

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

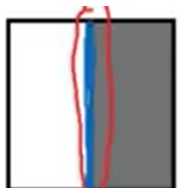
$*$

1	0	-1
1	0	-1
1	0	-1

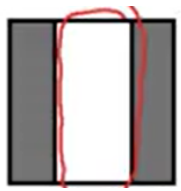
$=$

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

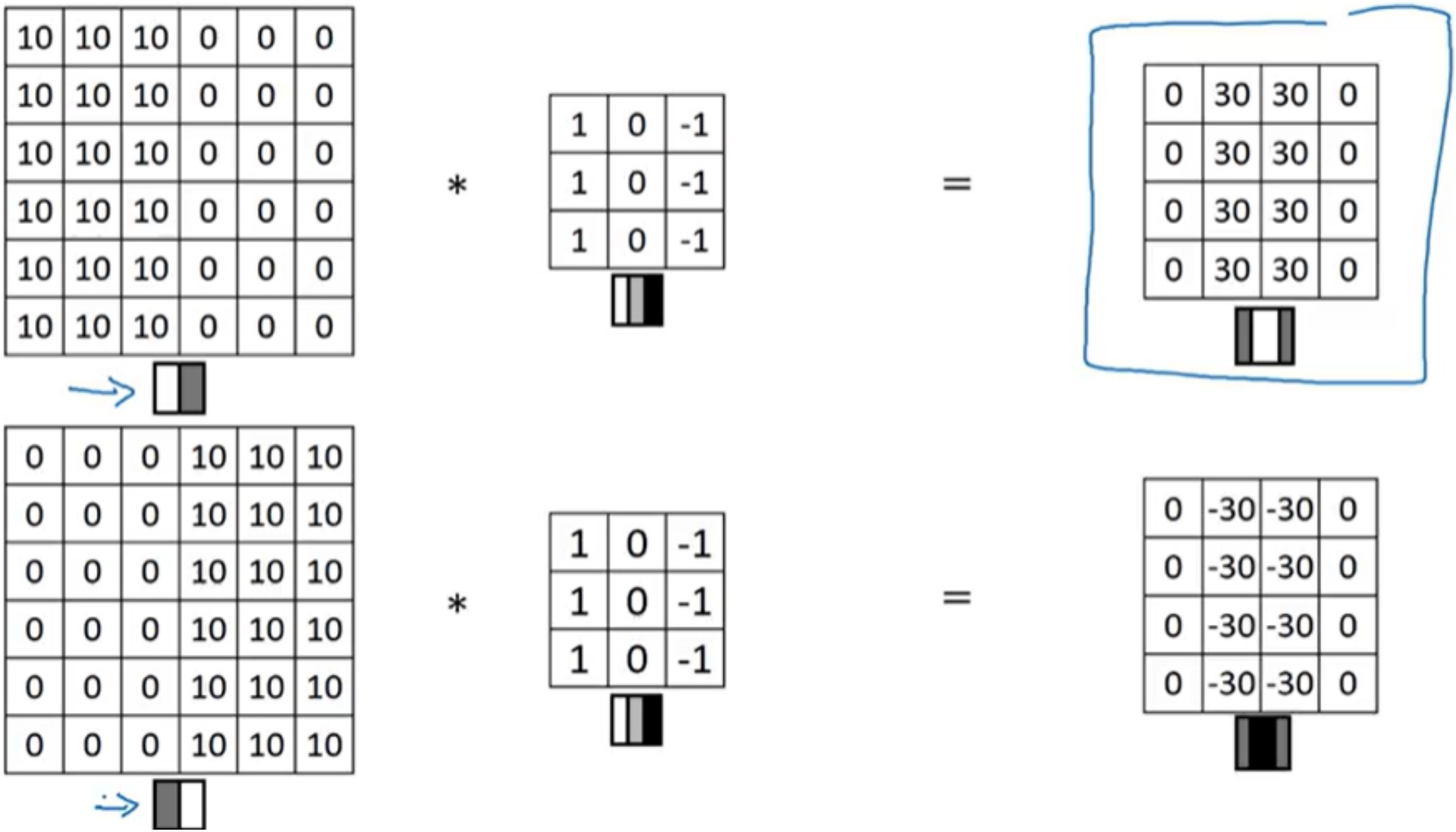
Vstup | Filtr | Výstup



$*$



Konvoluce - horizontální hrany



Konvoluce - horizontální hrany

1	0	-1
1	0	-1
1	0	-1

Vertical

1	1	1
0	0	0
-1	-1	-1

Horizontal

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10

*

1	1	1
0	0	0
-1	-1	-1

=

0	0	0	0
30	10	-10	-30
30	10	-10	-30
0	0	0	0

Konvoluce - obecný princip učení

- Různé filtry detekují různé vzory.
- Konvoluční vrstvy se učí vhodné filtry pro detekci konkrétních vzorů.

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

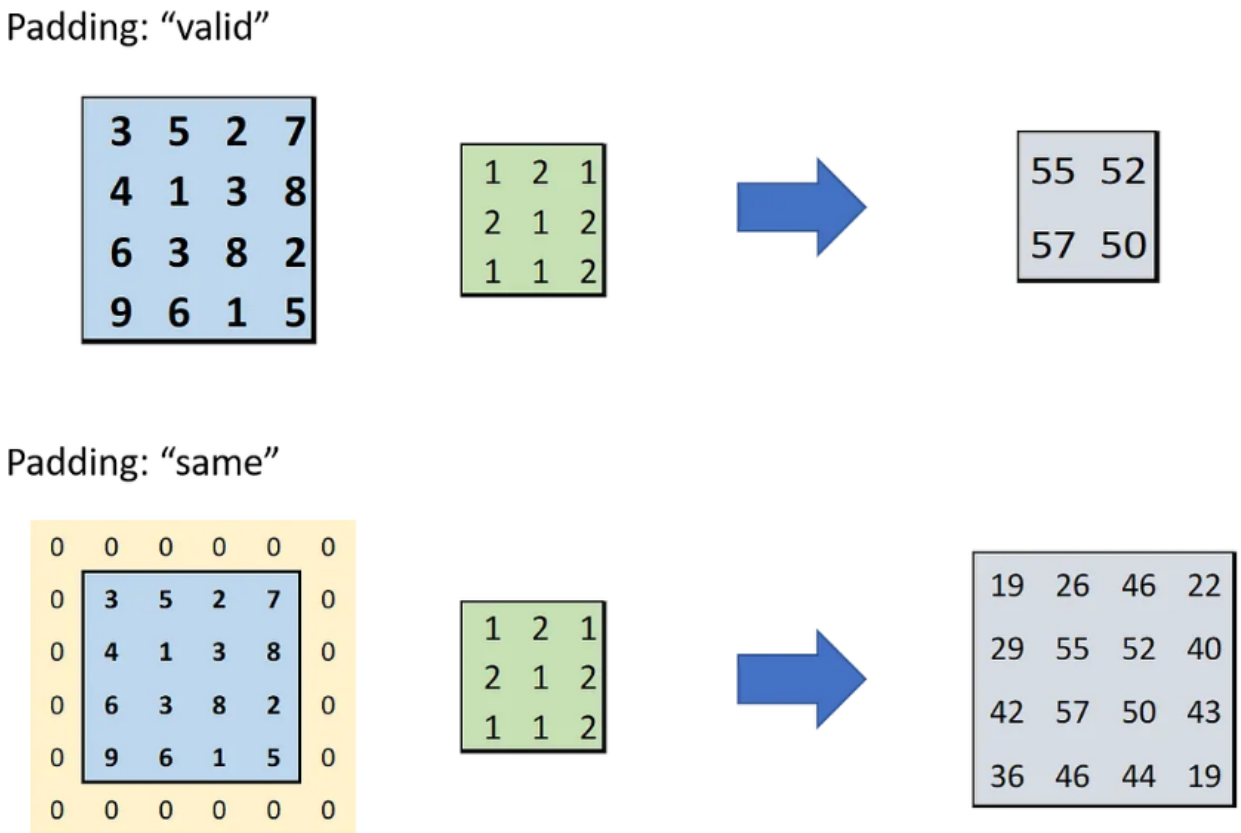
w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Důsledky konvoluce

- Viděli jsem, že pro vstupní obrázek o velikosti 6x6 a filtr 3x3 dostaneme výstup 4x4.
- Obecně pro obrázek s rozměrem $n \times n$ a filtr $k \times k$ dostaneme výstup $(n - k + 1) \times (n - k + 1)$.
- S každou konvolucí zmenšujeme rozměr obrázku.
- Okraje obrázku se účastní menšího počtu konvolucí a ztrácíme informaci z hran obrázku.

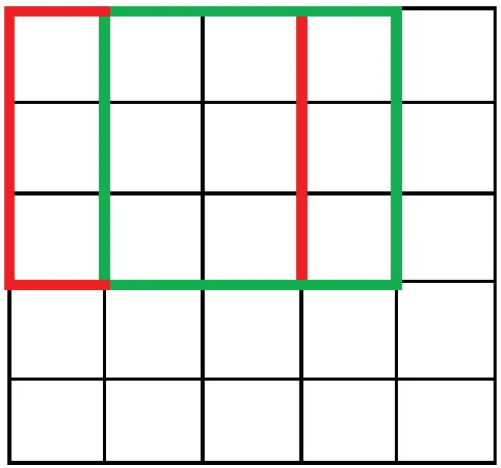
Padding - typy

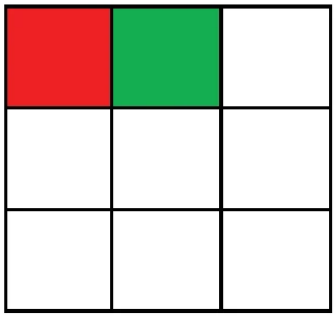
- Padding – přidání okrajů k obrázku.
- Většinou se používá padding $= 1$, tedy přidání jednoho pixelu okolo celého obrázku.
- Výstupní rozměr je pak stejný jako vstupní.

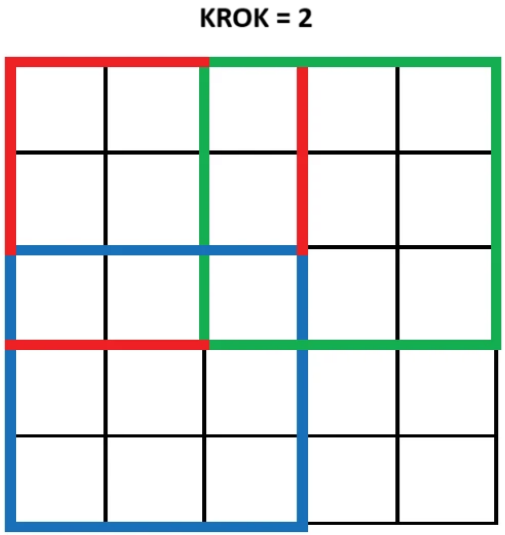


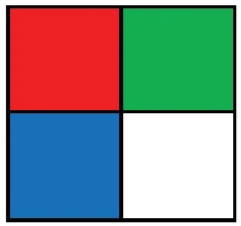
Kroková konvoluce

- Kroková konvoluce – posun filtru o více než 1 pixel.
- Zmenšuje výstupní rozměr.
- Větší krok = menší výstupní rozměr.





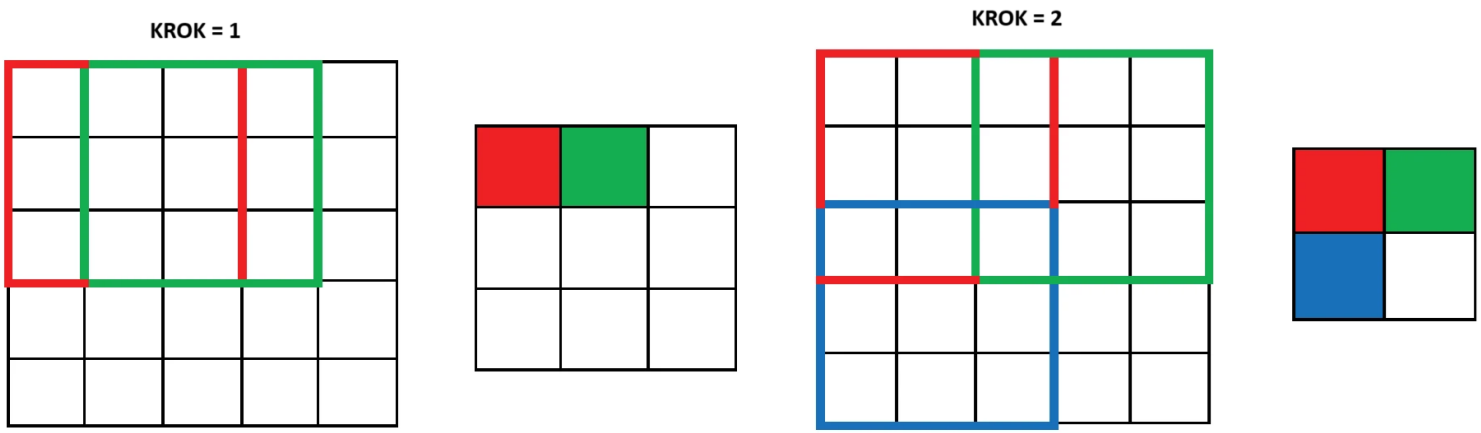




Kroková konvoluce

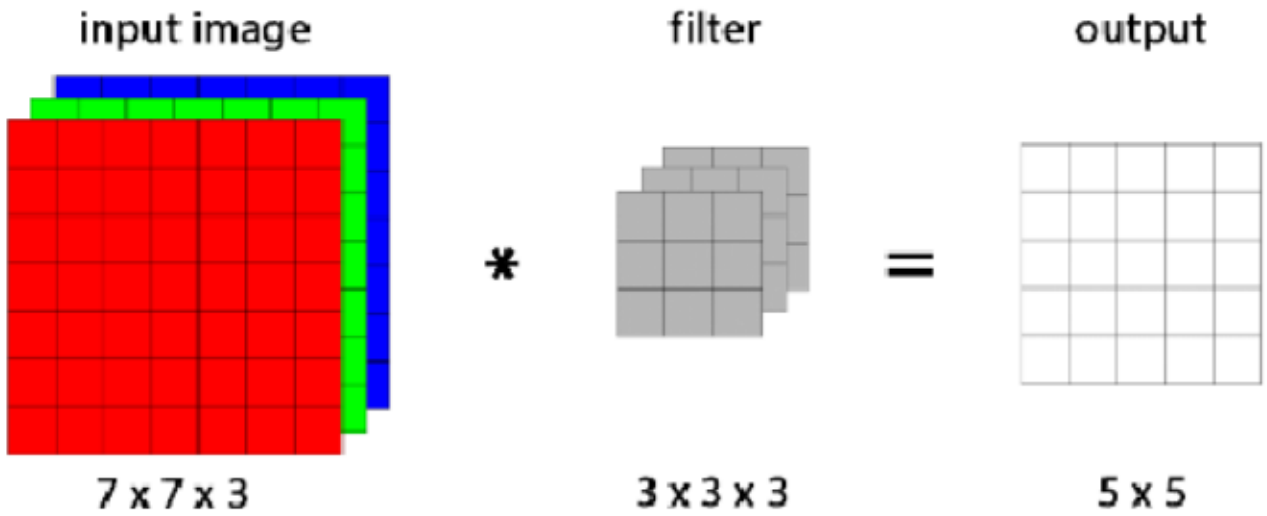
- Výsledný rozměr pro obrázek s rozměrem $n \times n$, filtr $k \times k$ a krok s je:

$$\lfloor((n-k+1)/s)\rfloor \times \lfloor((n-k+1)/s)\rfloor$$



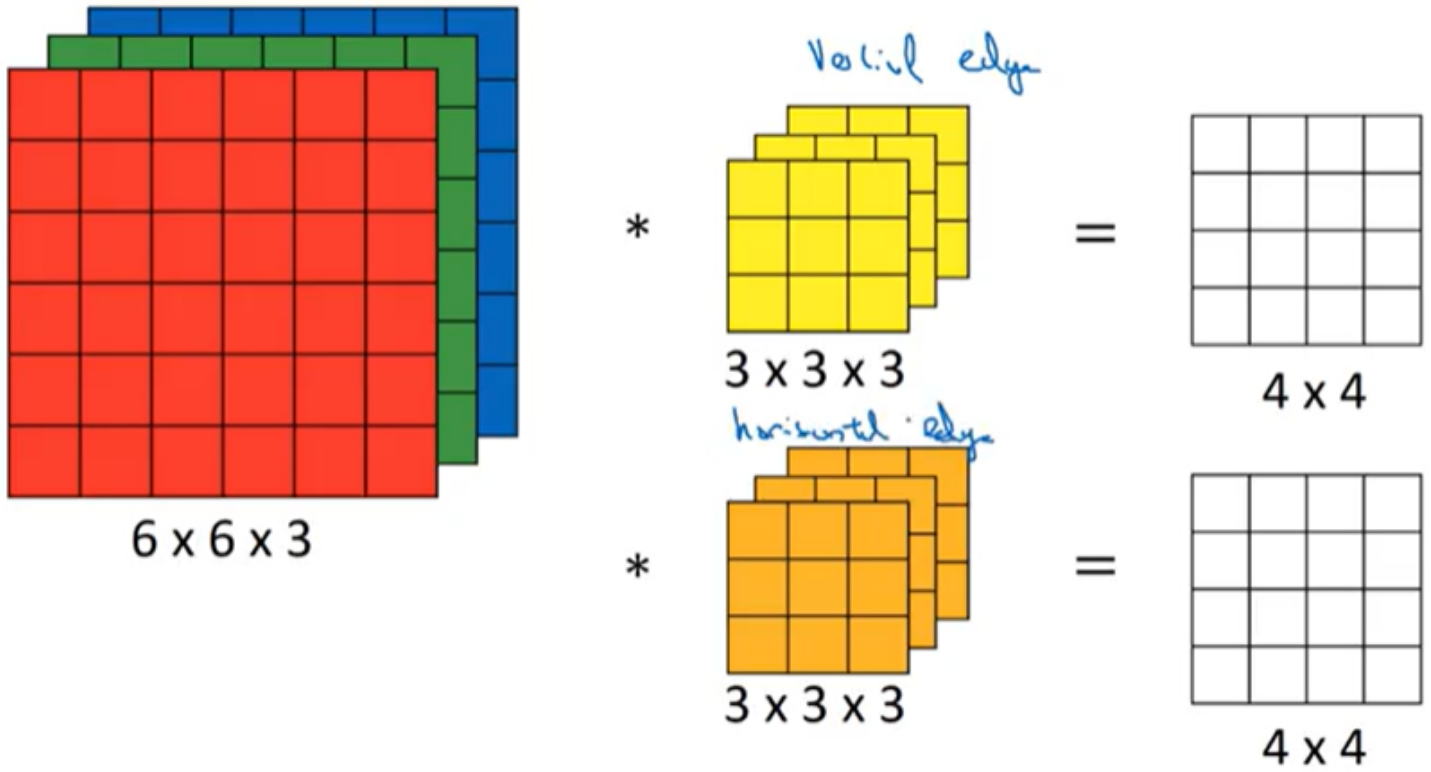
Konvoluce přes více barevných kanálů

- Počet kanálů (barev) na vstupu určuje počet kanálů filtru.
- Např. pro RGB obrázek 3 kanály, filtr 3x3x3.
- Bereme hodnoty jako by se filtr choval jako krychle a bereme součet přes pozice a kanály, takže celkem sčítáme 27 hodnot.



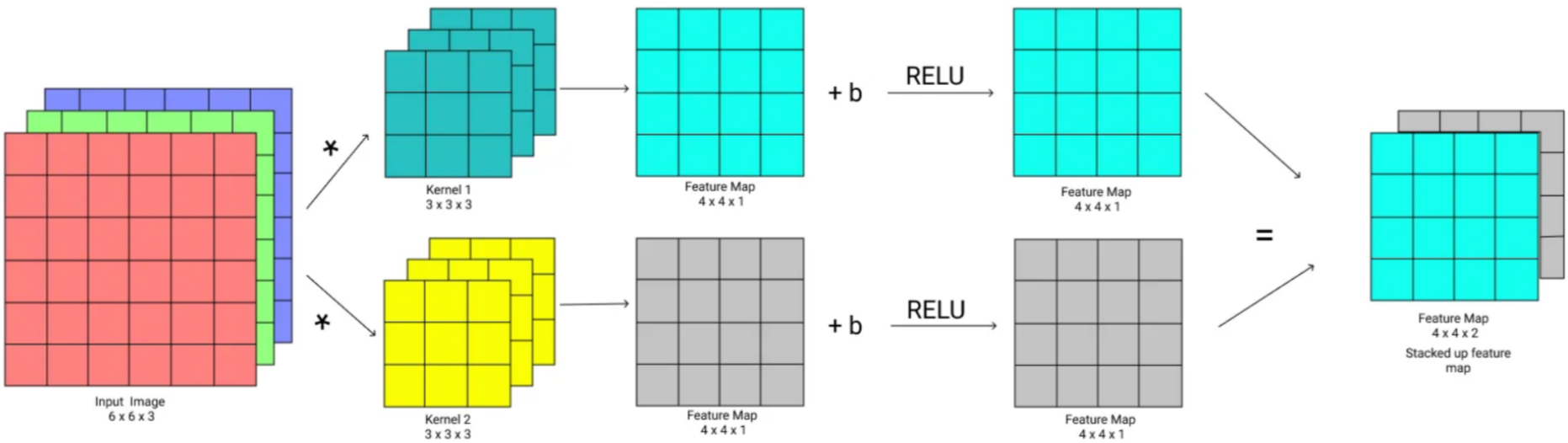
Více násobné filtry

- Většinou se používá více filtrů, které detekují různé vzory.
- Např. pro RGB obrázek 6x6x3 kanály, filtr 3x3x3 a 16 filtrů.
- Výstupní rozměr bude 4x4x16.



Ukázka konvoluční vrstvy

- Ke každému prvku výstupní matice přičteme hodnotu biasu a následně na vstup aplikujeme aktivační funkci (obvykle ReLU).



Počet parametrů konvoluční vrstvy

- Počet parametrů konvoluční vrstvy je dán počtem filtrů, velikostí filtru a počtem kanálů.
- Pro n filtrů, velikost filtru $k \times k$ a c kanálů je počet parametrů:

$$n \cdot (k^2 \cdot c + 1)$$

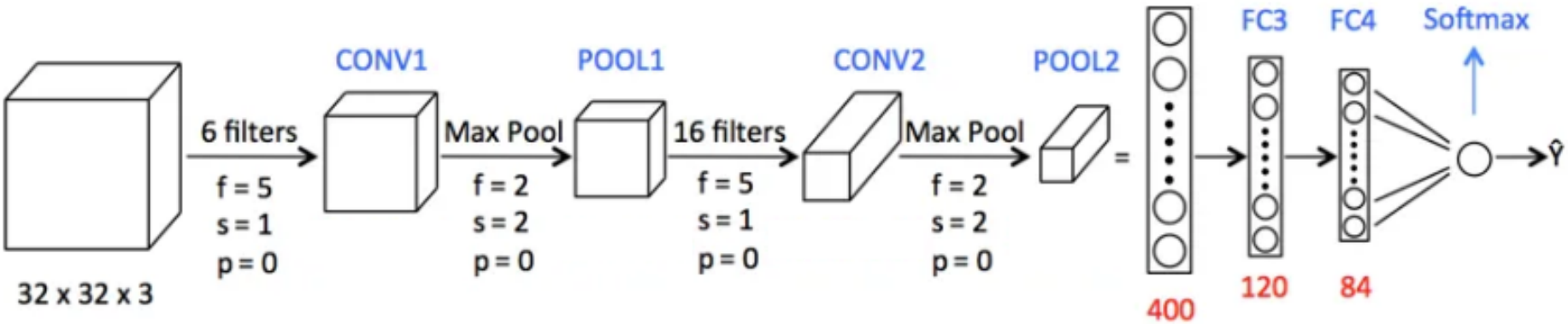
- Poslední člen je bias pro každý filtr.
- Pro filtr 3x3 a 10 filtrů na RGB obrázku je počet parametrů:

$$10 \cdot (3^2 \cdot 3 + 1) = 10 \cdot (27 + 1) = 280$$

- Všimněme si je nezávislý na velikosti obrázku.

Typy vrstev v konvolučních neuronových sítích

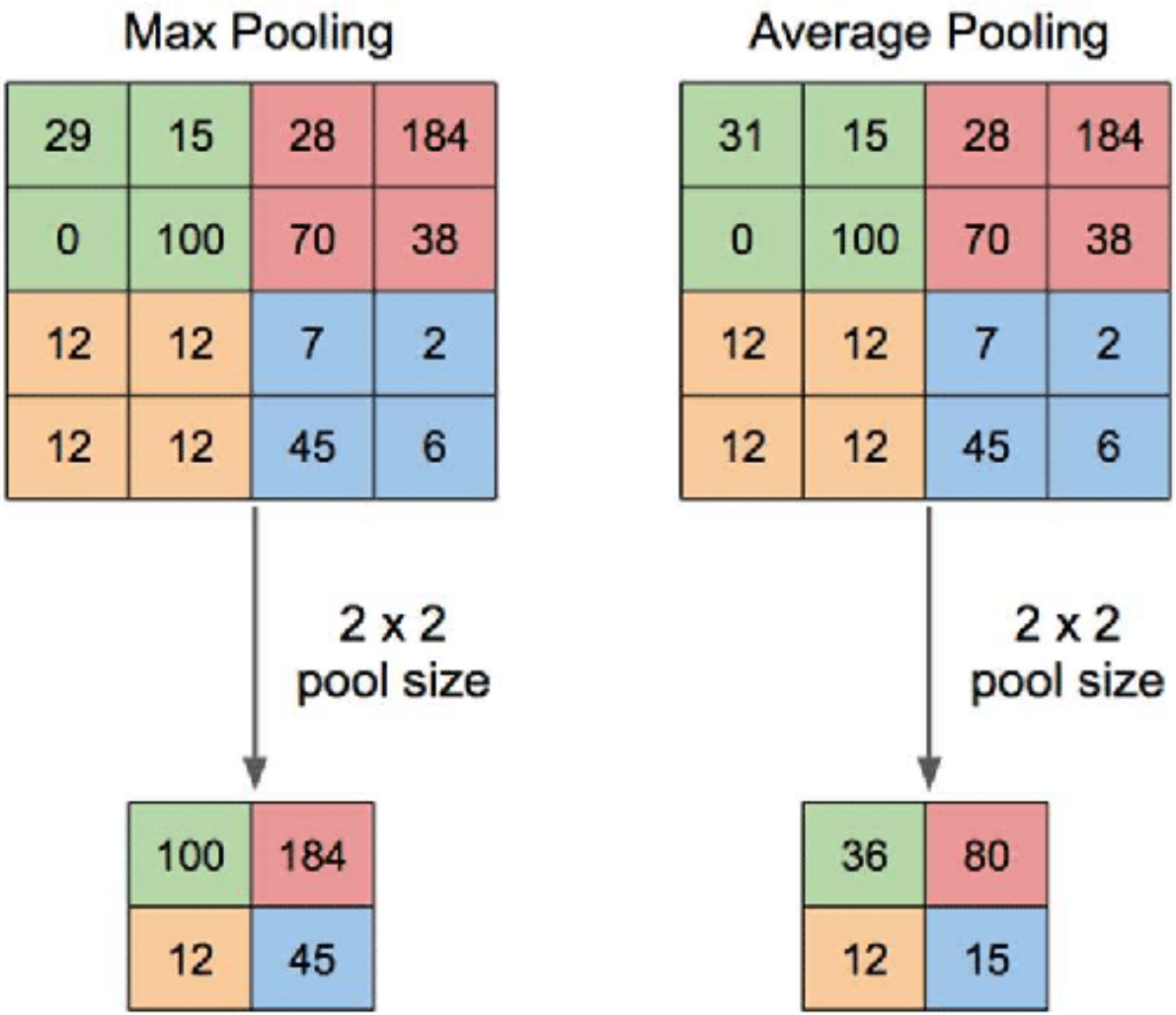
- Konvoluční vrstvy (Conv2D)
- Pooling vrstvy (MaxPooling, AveragePooling)
- Dense vrstvy (plně propojené)



Pooling

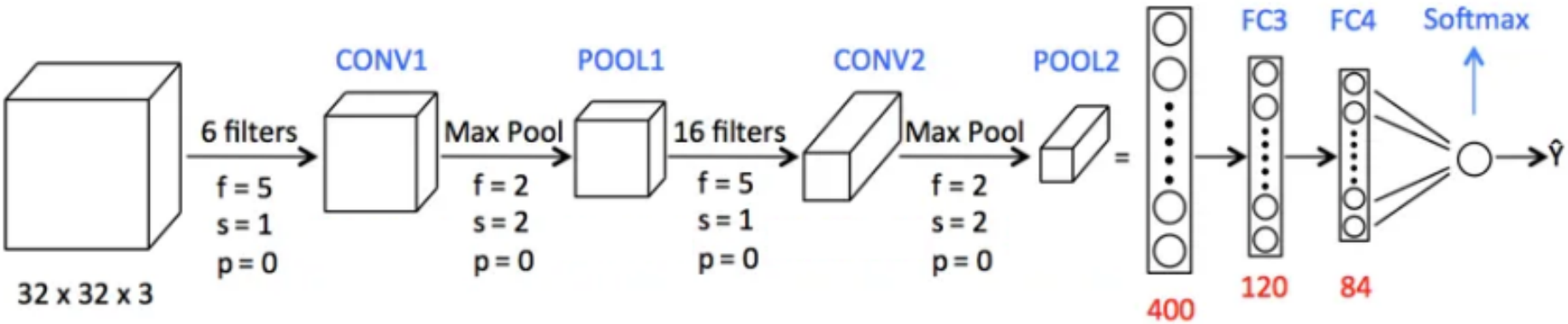
- Pooling vrstvy zmenšují rozměr výstupu konvoluční vrstvy.
- Většinou se používá MaxPooling, který bere maximum z oblasti.
- AveragePooling bere průměr z oblasti.
- Pooling vrstvy snižují počet parametrů a zvyšují robustnost vůči šumu.
- Nemají žádné parametry, pouze velikost okna a krok.

Pooling - ukázka



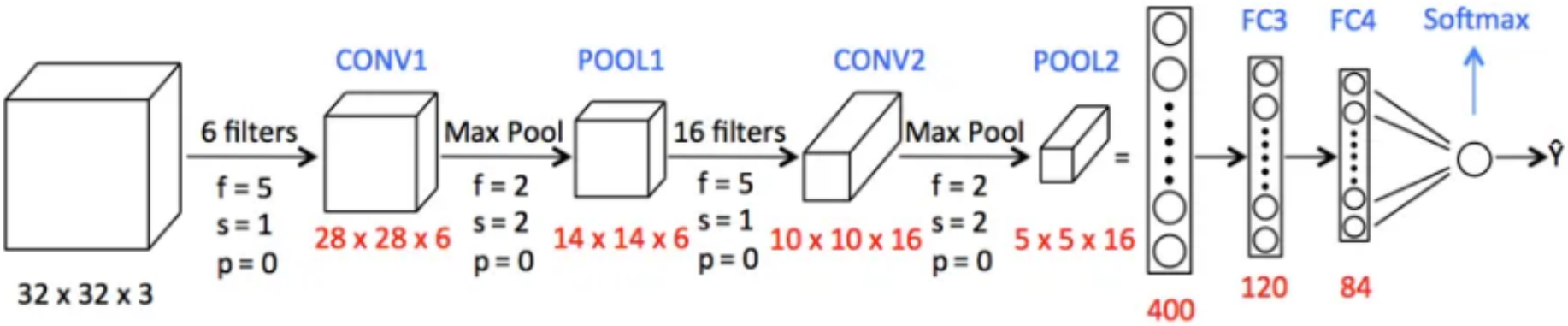
Úkol

- Jaké budou rozměry jednotlivých vrstev po aplikování konvoluční vrstvy a pooling vrstvy?



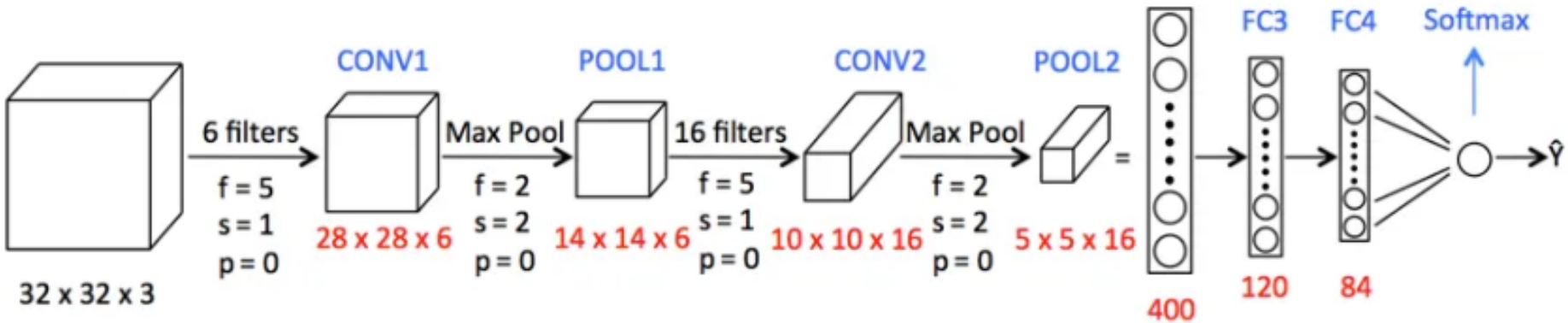
Úkol - řešení

- Jaké budou rozměry jednotlivých vrstev po aplikování konvoluční vrstvy a pooling vrstvy?



LeNet-5

- Jedna z prvních konvolučních neuronových sítí
- Používá se pro rozpoznávání číslic na obrázcích
- Obsahuje 2 konvoluční vrstvy, 2 pooling vrstvy a 2 plně propojené vrstvy
- Využívá ReLU jako aktivační funkci.
- Autorem je Yann LeCun (v současnosti Meta Chief AI Researcher)



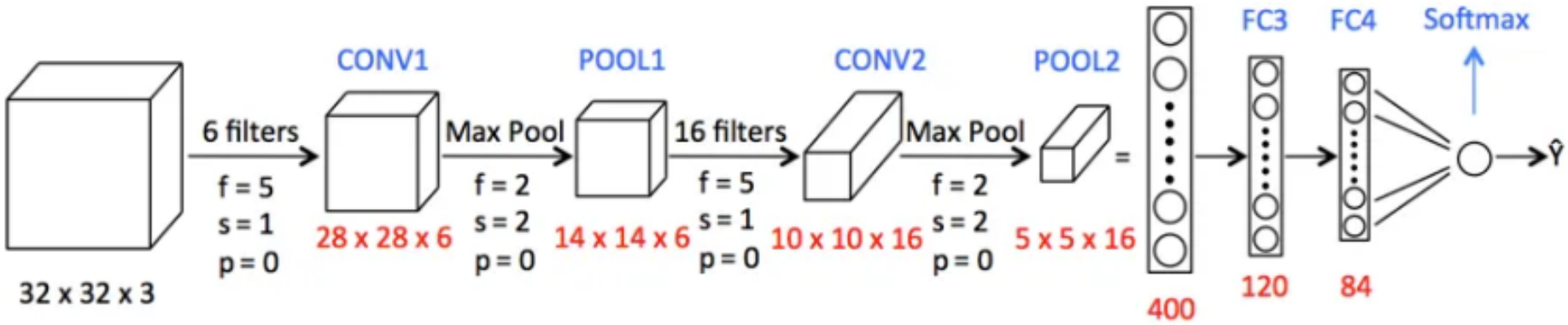
Proč konvoluce

- Sdílení parametrů - filtr se aplikuje na celou plochu obrázku a může být užitečný na různých místech.
- Menší počet parametrů - oproti plně propojeným vrstvám.
  - Např. pro obrázek 32x32x3 a 6 filtrů 5x5 je počet parametrů: 156, výsledný rozměr obrázku by byl 28x28x6.
  - Pokud bychom stejného chování chtěli docílit plně propojenou vrstvou, potřebovali bychom 32x32x3x28x28x6 = 14,5 milionů parametrů.

Neuronová síť?

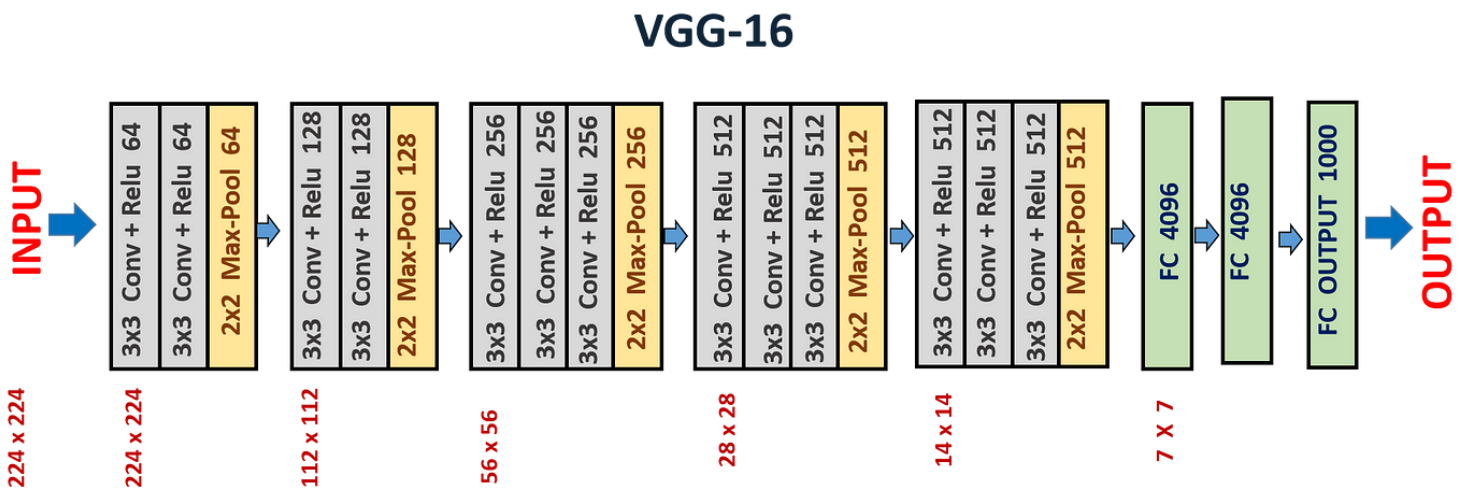
- Využijeme gradientní sestup pro aktualizaci váhy filtrů.
- Optimalizujeme ztrátovou funkci:

$$J = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{y}_i)$$



Architektura CNN

- Typická struktura: `Conv` → `ReLU` → `Pool` opakovaně
- Nakonec fully-connected vrstvy (dense)
- Příklady známých architektur: LeNet, VGG, ResNet



Aplikace CNN v biologii

- Klasifikace mikroskopických snímků (např. zdravá vs. nemocná buňka)
- Detekce a počítání buněk nebo kolonií
- Automatická anotace biologických vzorků
- Příklady: DeepCell, CellPose, Bioimage.io

Úloha

- Modifikujte příklad na Colabu, tak aby odpovídal modelu Lecun-5.

Shrnutí bloku 2

- Co je CNN a jak funguje
- Jak ji využít v biologických datech
- Praktická ukázka tréninku modelu