





Gyires-Tóth Bálint

Deep Learning a gyakorlatban Python és LUA alapon Konvolúciós hálózatok





Jogi nyilatkozat

Jelen előadás diái a „*Deep Learning a gyakorlatban Python és LUA alapon*” című tantárgyhoz készültek és letölthetők a http://smartlab.tmit.bme.hu honlapról.

A diák nem helyettesítik az előadáson való részvételt, csupán emlékeztetőül szolgálnak.

Az előadás diái a szerzői jog védelme alatt állnak. Az előadás diáinak vagy bármilyen részének újra felhasználása, terjesztése, megjelenítése csak a szerző írásbeli beleegyezése esetén megengedett. Ez alól kivétel, mely diákon külső forrás külön fel van tüntetve.

Kis házi feladat

1. Javítás alatt.
2. Leadás ma éjfélig.

Nagy házi feladat további teendők

http://smartlab.tmit.bme.hu/oktatas-deep-learning-nagy-hazi

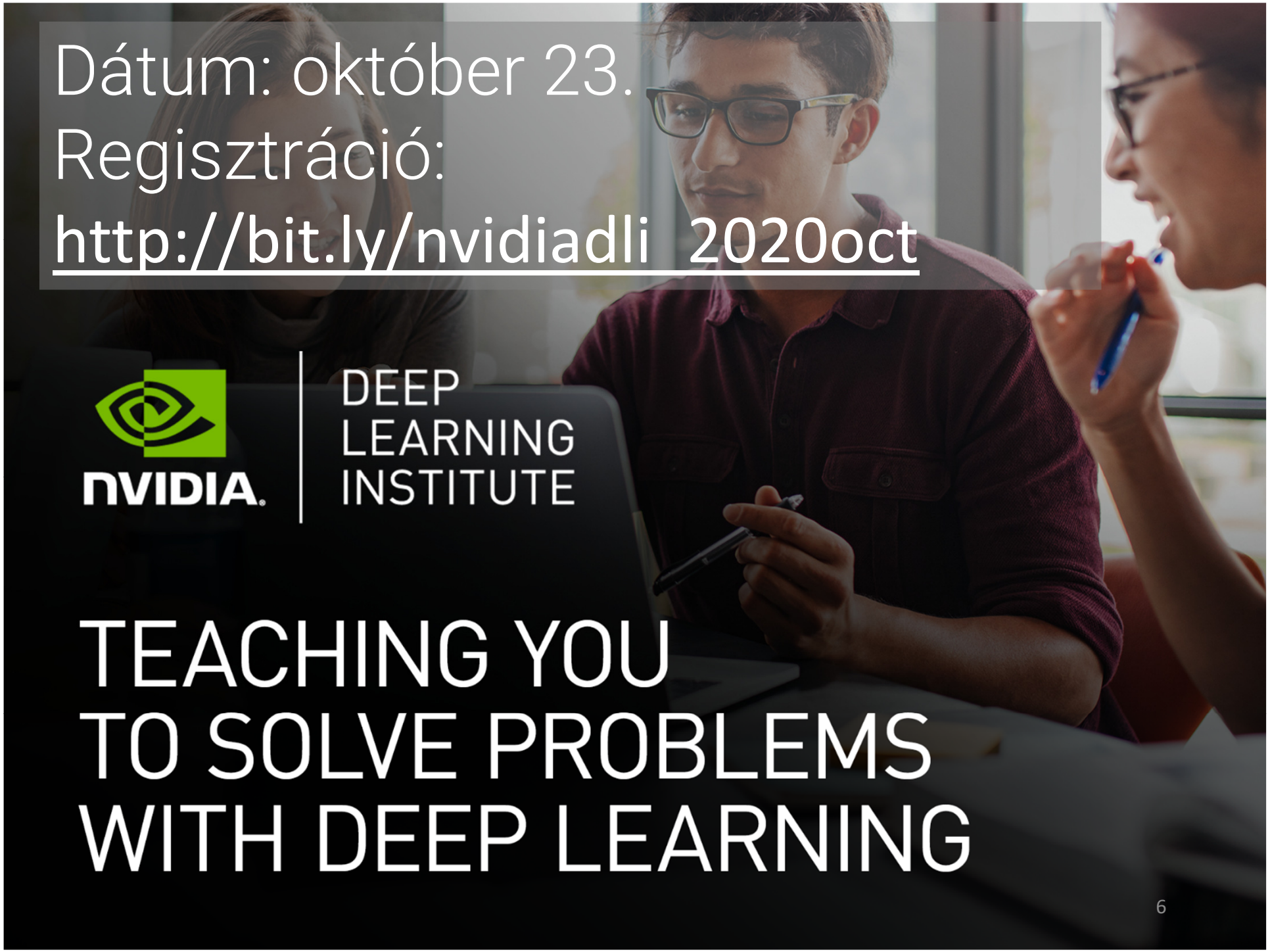
Milestone 1: Október 25., 23:59

Adatok beszerzése, adatfeltárás, vizualizáció (ha szükséges) és előkészítés tanításhoz.
Eredmény: tanító, validációs és teszt adatbázisok

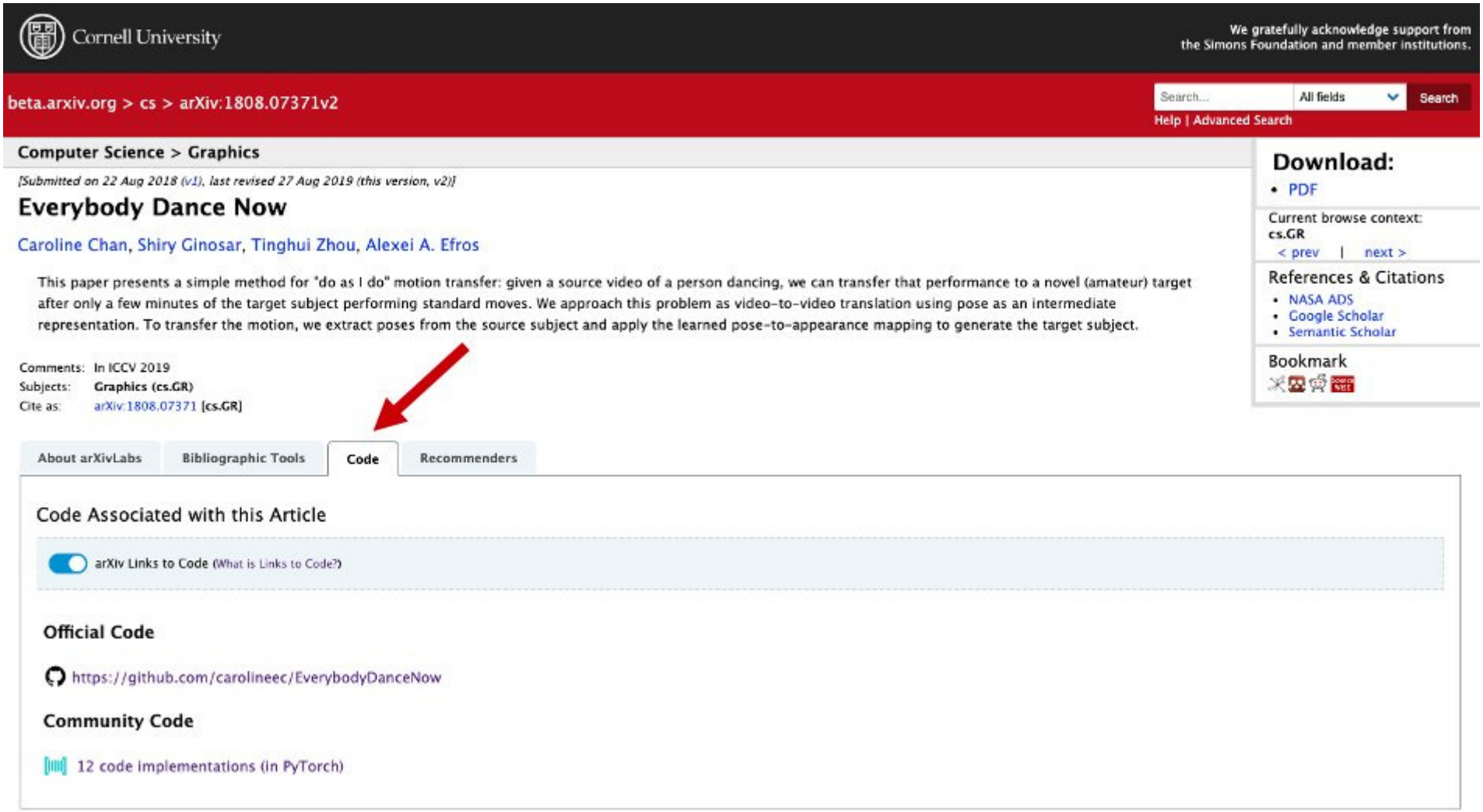
Megajánlott jegyért a nyelv: ANGOL

Deep Learning Híradó

Hírek az elmúlt 168 órából

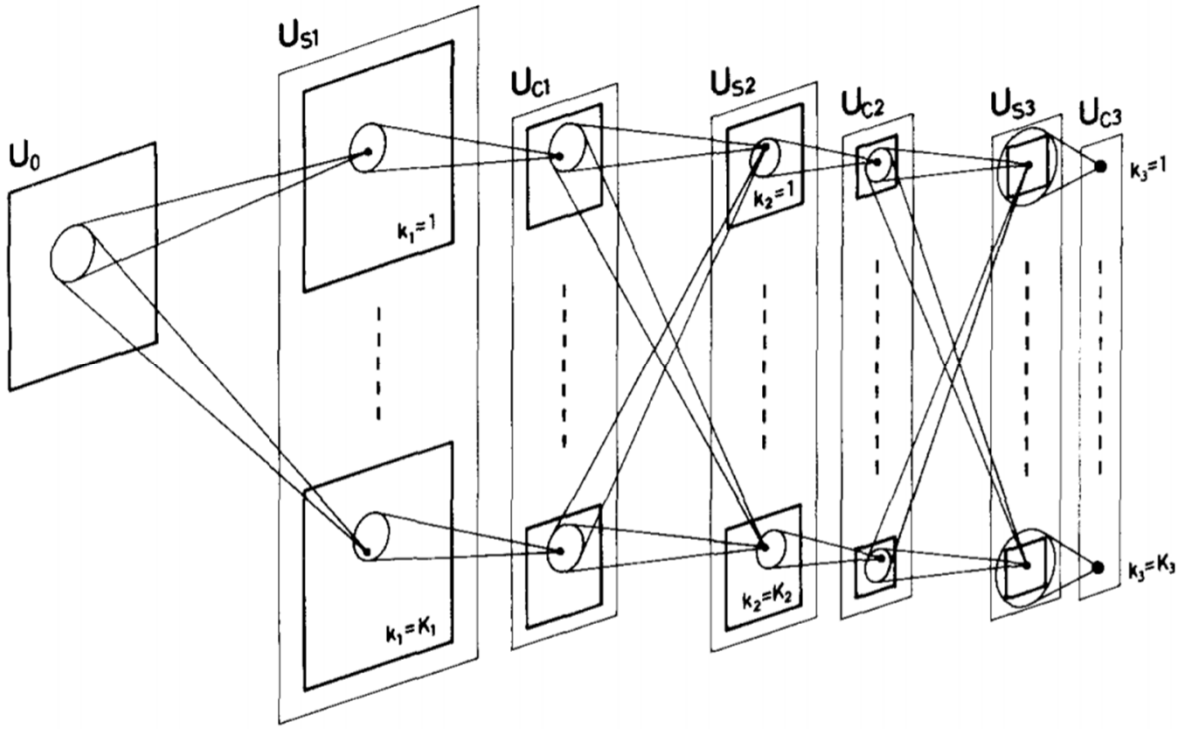

Dátum: október 23.
Regisztráció:
http://bit.ly/nvidiadli_2020oct
NVIDIA
DEEP LEARNING INSTITUTE
TEACHING YOU TO SOLVE PROBLEMS WITH DEEP LEARNING

ArXiv.org + Papers with Code



Főbb mérföldkövek

Neocognitron: Fukushima, Kunihiko, and Sei Miyake. "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition." In Competition and cooperation in neural nets, pp. 267-285. Springer, Berlin, Heidelberg, 1982.



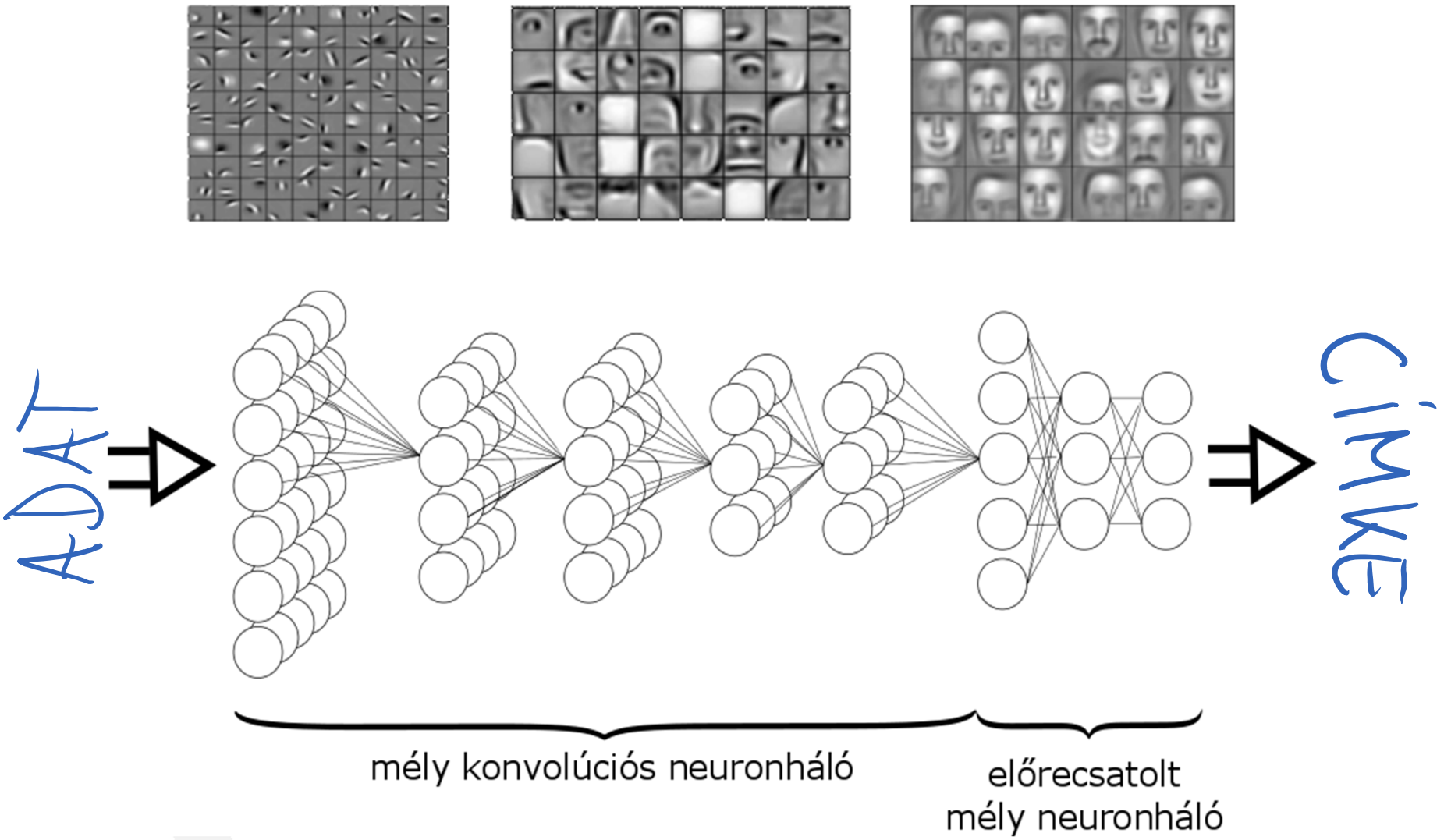
Főbb mérföldkövek

- LeNet: LeCun, Y., & Bengio, Y. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, *3361*(10), 1995.
- LeNet5: LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.
- AlexNet: Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).
- … és elindult a CNN lavina!

De mi is az a CNN?

- Convolutional Neural Network (CNN)
  - Konvolúciós és Pooling rétegek és aktivációs függvények struktúrált összessége.
  - Közöttük számos további réteg lehet, pl. DropOut, BatchNorm, stb.
  - A hálózat utolsó rétegei jellemzően előrecsatolt rétegek, melyek a megtanult jellemzők osztályokba sorolását végzik.
- Gyors és hatékony megoldás *jellemző tanulásra* (*feature learning*). Térbeli függőség a minták között.
- Jellemző tanulás és modellezés egy lépésben.
- Hang, kép, szöveg, videó és általános felhasználási terület (pl. AlphaGo, szenzoradatok).

CNN háló általános felépítése



MNIST: Yann LeCun kézírás adatbázis


```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,51,159,253,159,50,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,48,2
38,252,252,252,237,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,54,227,253,252,239,
233,252,57,6,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,60,224,252,253,252,202,84,252,
253,122,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,163,252,252,252,253,252,252,96,189,253
,167,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,51,238,253,253,190,114,253,228,47,79,255,16
8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,48,238,252,252,179,12,75,121,21,0,0,253,243,50,0
,0,0,0,0,0,0,0,0,0,0,0,0,38,165,253,233,208,84,0,0,0,0,0,0,253,252,165,0,0,0,0,0
,0,0,0,0,0,0,0,7,178,252,240,71,19,28,0,0,0,0,0,0,253,252,195,0,0,0,0,0,0,0,0,0,
0,0,0,57,252,252,63,0,0,0,0,0,0,0,0,0,253,252,195,0,0,0,0,0,0,0,0,0,0,0,0,198,25
3,190,0,0,0,0,0,0,0,0,0,0,255,253,196,0,0,0,0,0,0,0,0,0,0,76,246,252,112,0,0,0
,0,0,0,0,0,0,0,253,252,148,0,0,0,0,0,0,0,0,0,0,85,252,230,25,0,0,0,0,0,0,0,0,7
,135,253,186,12,0,0,0,0,0,0,0,0,0,0,85,252,223,0,0,0,0,0,0,0,7,131,252,225,7
1,0,0,0,0,0,0,0,0,0,0,0,0,85,252,145,0,0,0,0,0,0,0,48,165,252,173,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,86,253,225,0,0,0,0,0,0,114,238,253,162,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,85,252,249,146,48,29,85,178,225,253,223,167,56,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
85,252,252,252,229,215,252,252,252,196,130,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,28,
199,252,252,253,252,252,233,145,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25,128,2
52,253,252,141,37,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

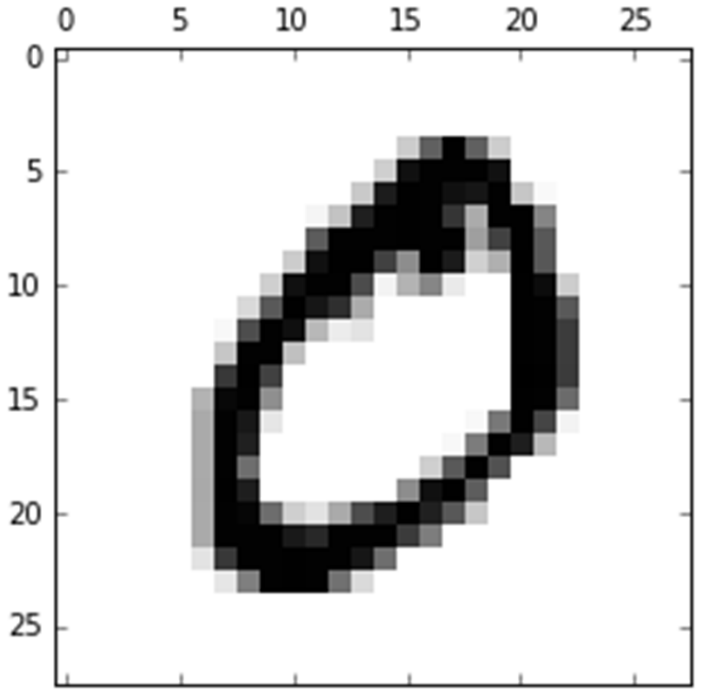
MNIST

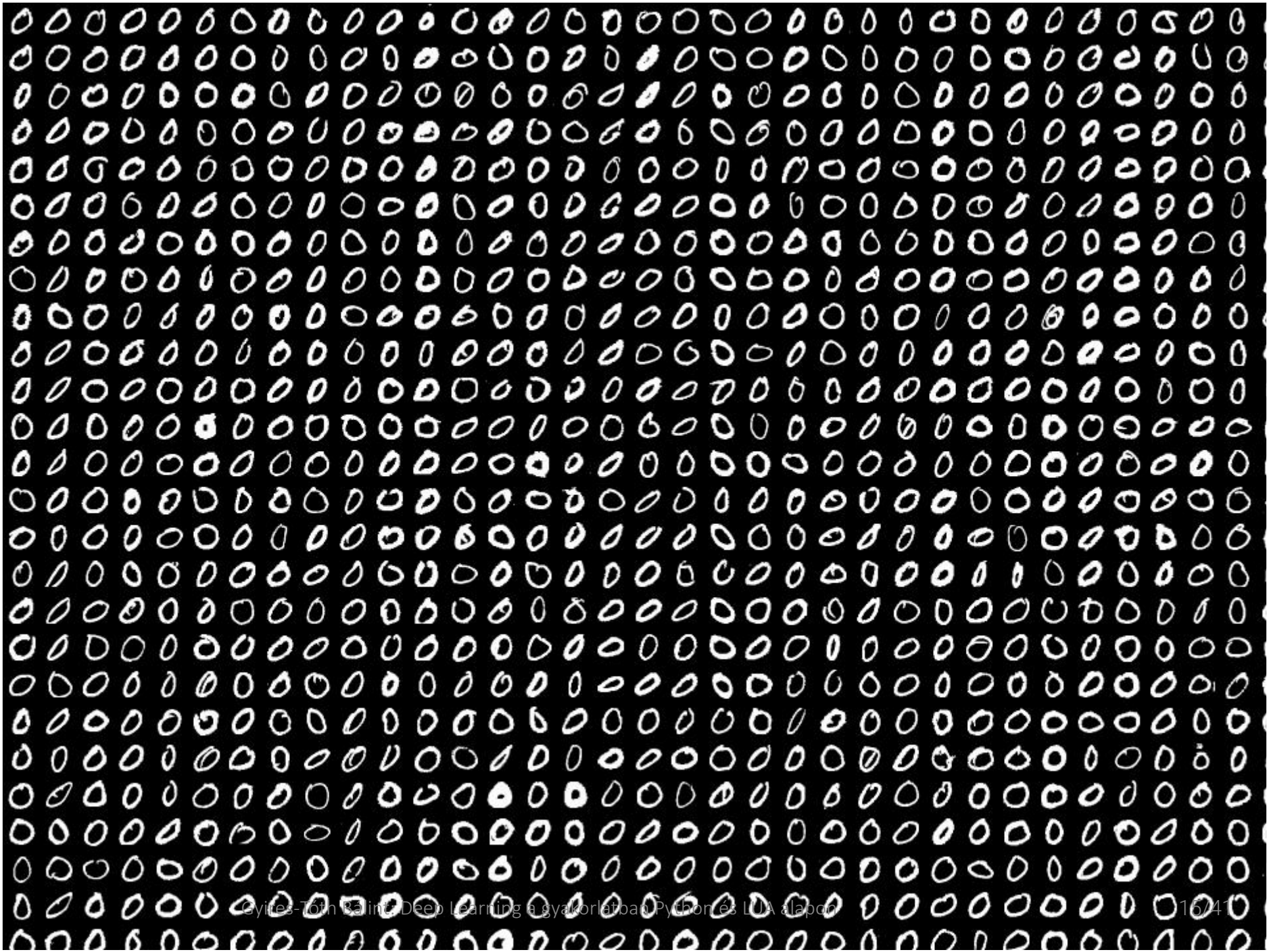
```
[[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  51, 159, 253, 159,  50,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  48, 238, 252, 252, 252, 237,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  54, 227, 253, 252, 239, 233, 252,  57,   6,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  10,  60, 224, 252, 253, 252, 202,  84, 252, 253, 122,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 163, 252, 252, 252, 253, 252, 252,  96, 189, 253, 167,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  51, 238, 253, 253, 190, 114, 253, 228,  47,  79, 255, 168,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,  48, 238, 252, 252, 179,  12,  75, 121,  21,   0,   0, 253, 243,  50,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,  38, 165, 253, 233, 208,  84,   0,   0,   0,   0,   0,   0, 253, 252, 165,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   7, 178, 252, 240,  71,  19,  28,   0,   0,   0,   0,   0,   0, 253, 252, 195,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,  57, 252, 252,  63,   0,   0,   0,   0,   0,   0,   0,   0,   0, 253, 252, 195,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0, 198, 253, 190,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 255, 253, 196,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,  76, 246, 252, 112,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 253, 252, 148,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,  85, 252, 230,  25,   0,   0,   0,   0,   0,   0,   0,   0,   7, 135, 253, 186,  12,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,  85, 252, 223,   0,   0,   0,   0,   0,   0,   0,   0,   7, 131, 252, 225,  71,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,  85, 252, 145,   0,   0,   0,   0,   0,   0,   0,  48, 165, 252, 173,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,  86, 253, 225,   0,   0,   0,   0,   0,   0, 114, 238, 253, 162,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,  85, 252, 249, 146,  48,  29,  85, 178, 225, 253, 223, 167,  56,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,  85, 252, 252, 252, 229, 215, 252, 252, 252, 196, 130,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,  28, 199, 252, 252, 253, 252, 252, 233, 145,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,  25, 128, 252, 253, 252, 141,  37,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
 [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0]]
```

MNIST

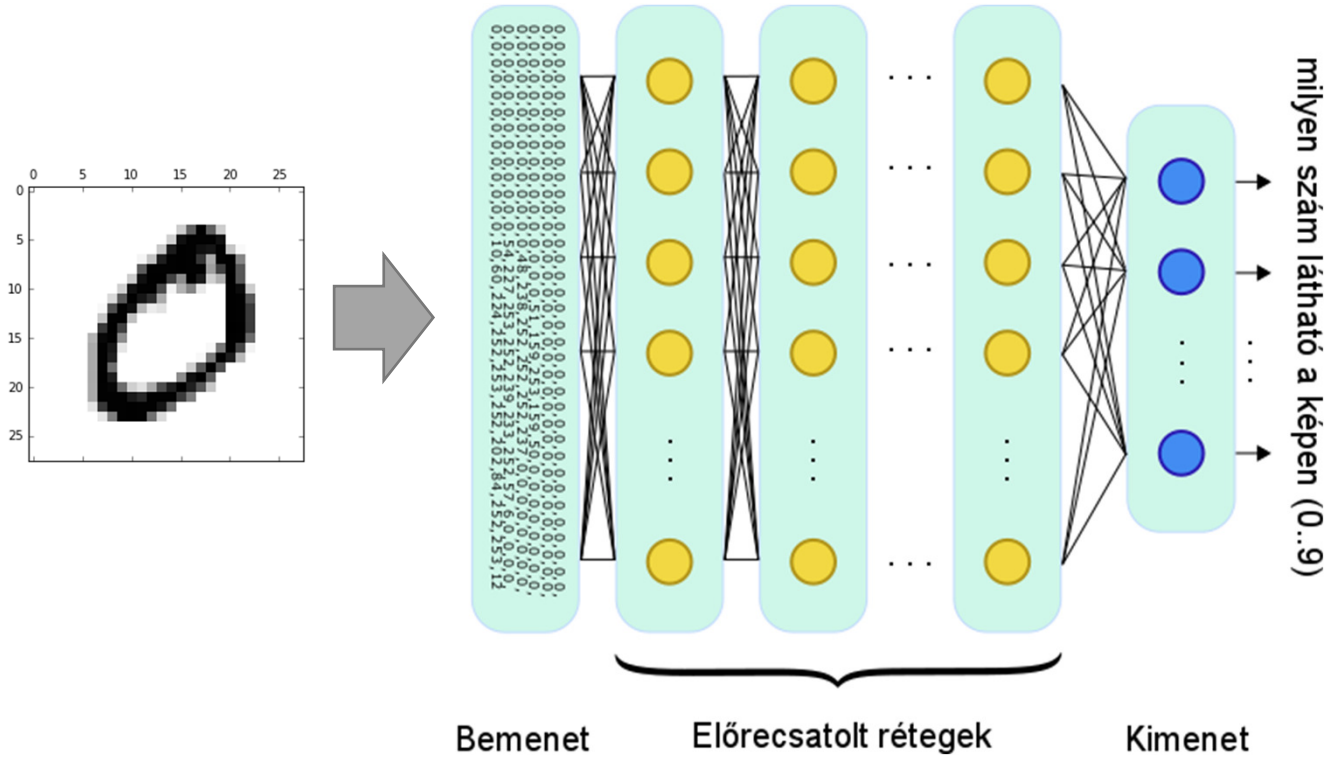
```
from keras.datasets import mnist
(X_train, y_train), (X_test, y_test) = \
                    mnist.load_data()

%matplotlib inline
import matplotlib.pylab as plt
image=X_train[1]
plt.matshow(image, \
            cmap = matplotlib.cm.binary)
```





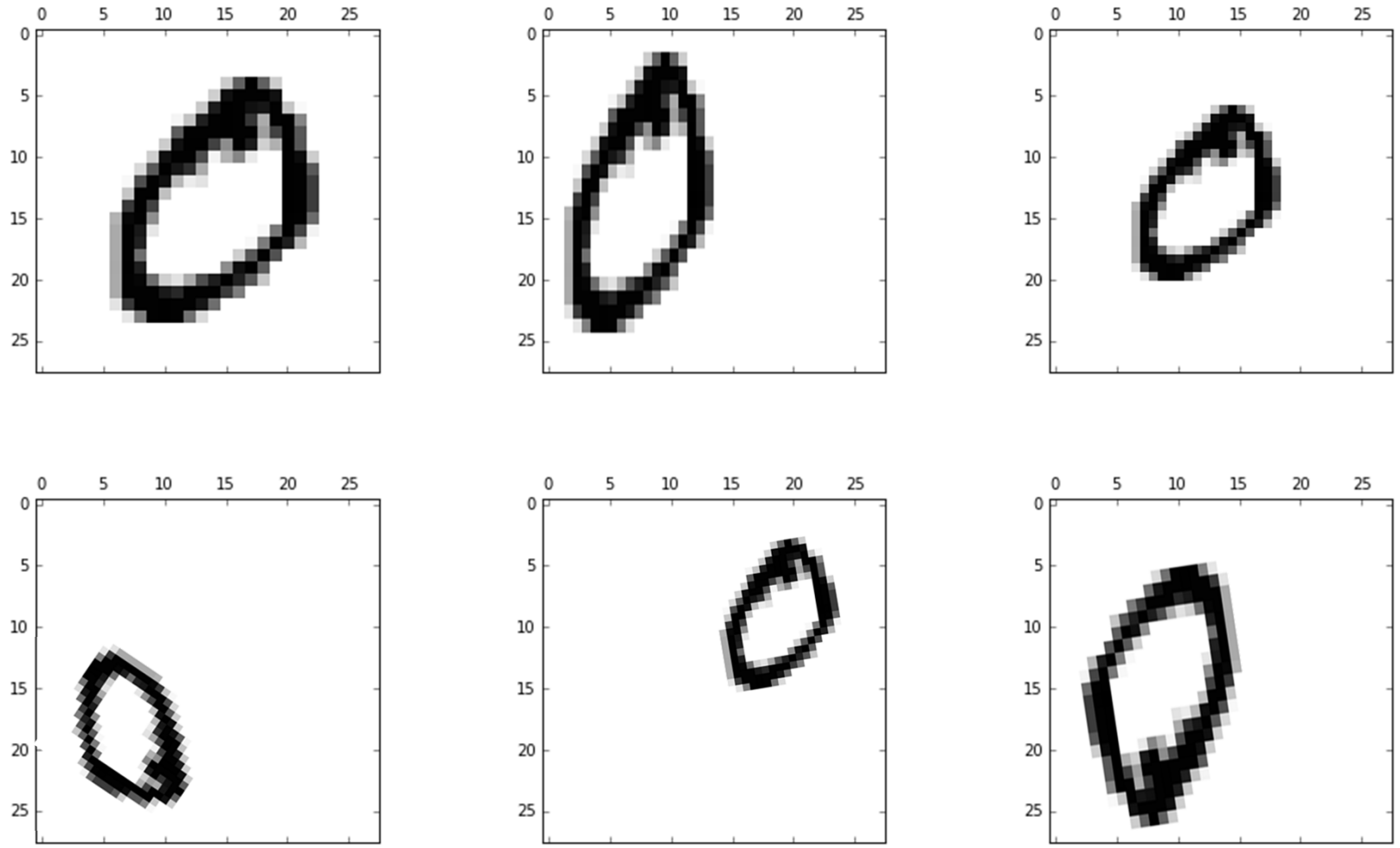
Előrecsatolt hálós megkötelítés



Legfőbb probléma

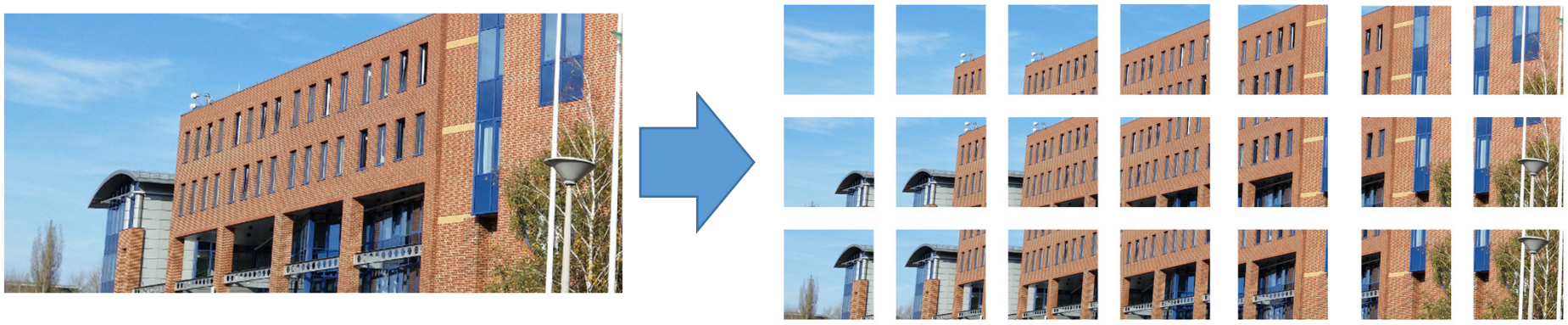
- Túl nagy paraméter tér
  - Pl. 200x200x3=120000 bemenet, további összeköttetések…)
- Nem mintát, hanem adatot ismer fel

Nem mintát, hanem adatot ismer fel



Konvolúciús neurális hálózat 1.

- Az adatot kisebb darabokra lebontva tanítjuk a hálózatot különböző szinteken.
- Ötlet 1: az adathalmaznak csak egy, térben egymáshoz közeli része releváns számunkra.
- Szemléltetés: képek, de ma már szinte minden tématerületen használják!

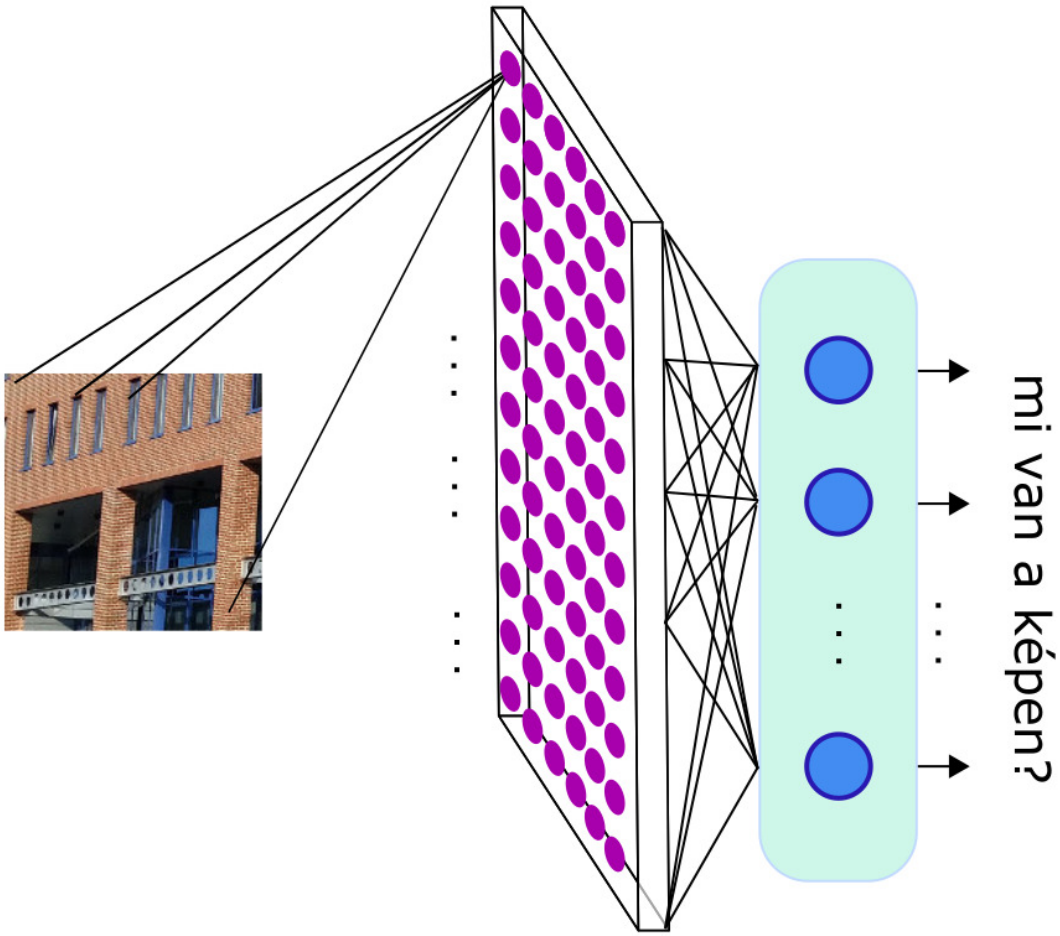


Építőelemek: konvolúciús réteg 1.

Az eredeti kép darabkáival tanítjuk a hálót.

Pl. 20x20 px darabka, akkor 400 súly + 1 bias.

Most az előrecsatolt háló neuronjait 2D-be rendeztük.



Építőelemek: konvolúciús réteg 2.

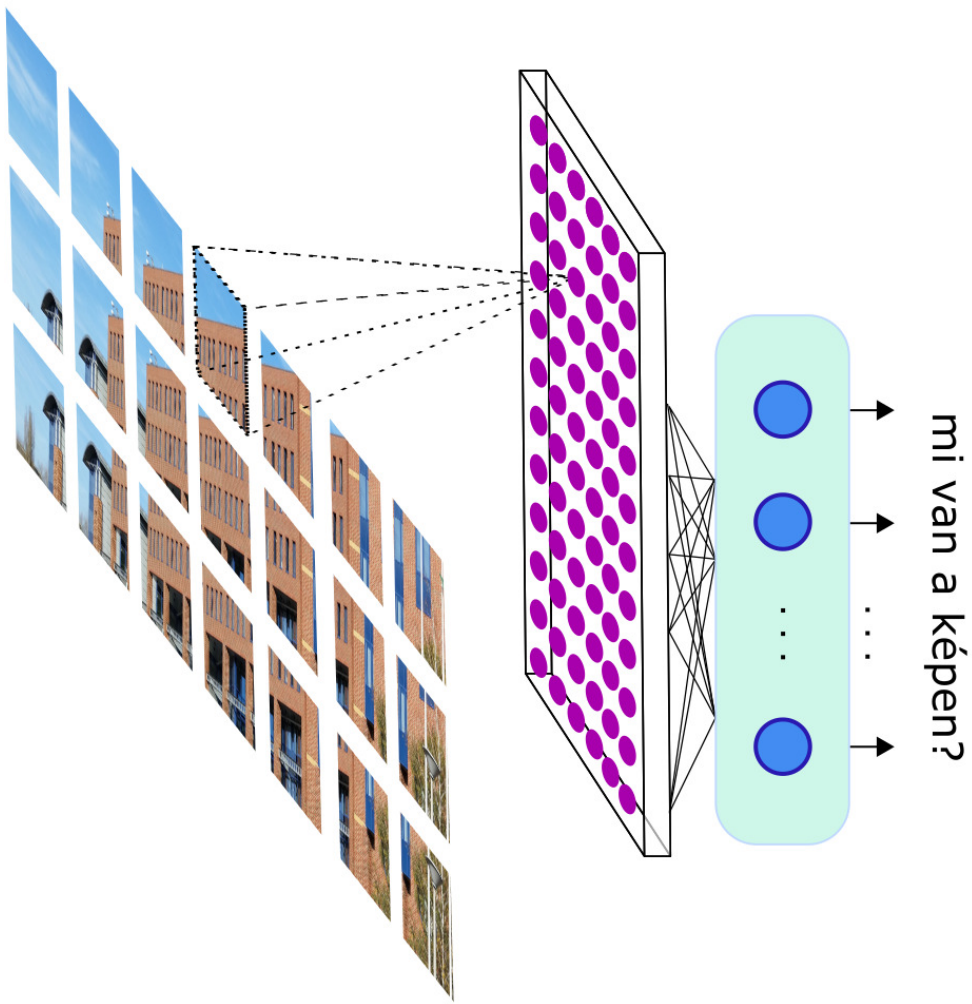
Megosztott súlyok (shared weights):

Az eredeti kép minden darabkájához ugyanazokat a súlyokat használjuk fel.

Ha egy jellemző a kép egyik pontján fontos, máshol is az lesz!

Miért hívjuk CNN-nek?

Bemenetet és megosztott súlyok (szűrők) konvolúciója.



CNN vizualizáció

http://scs.ryerson.ca/~aharley/vis/conv/

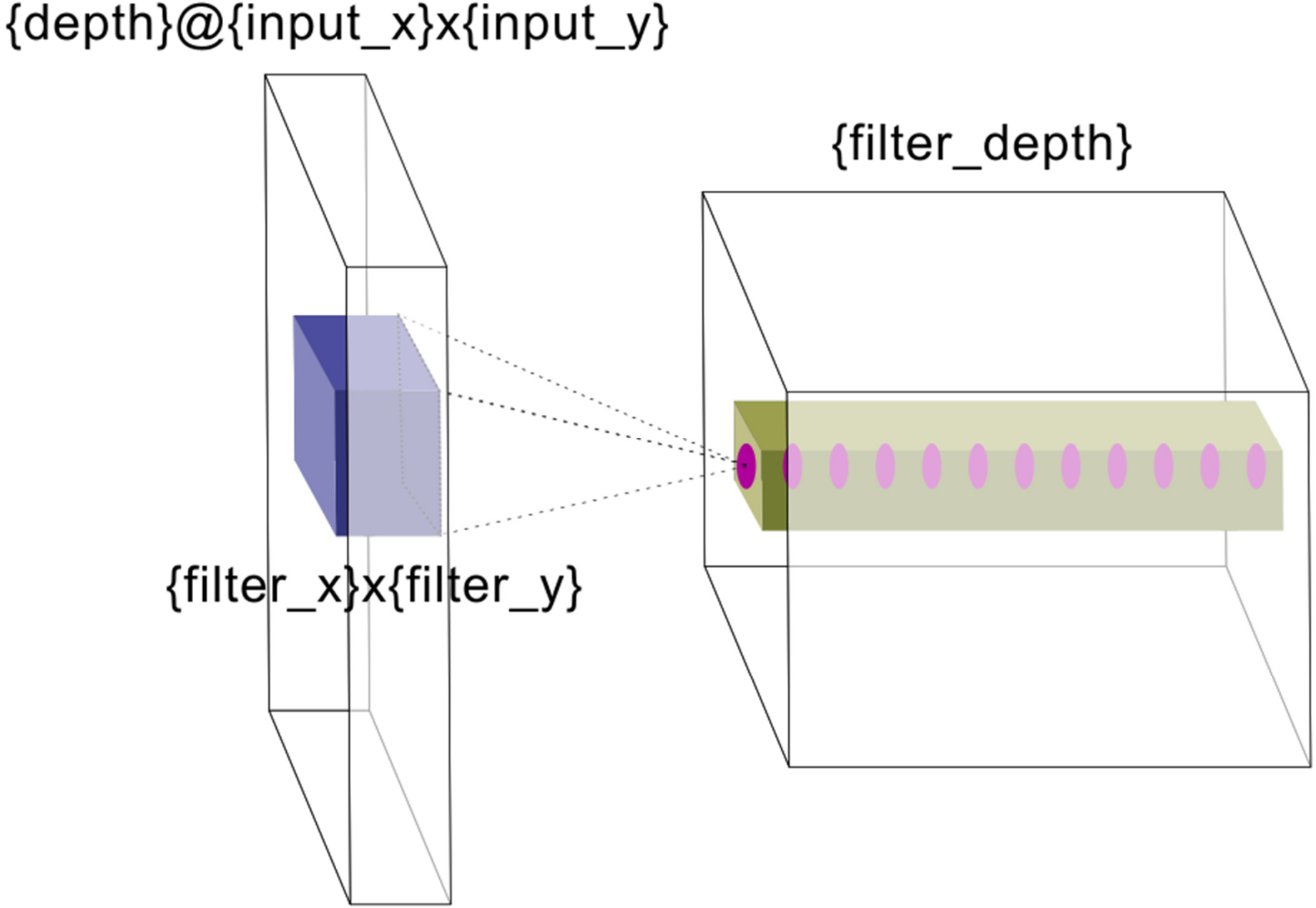
Hiperparaméterek (conv2d)

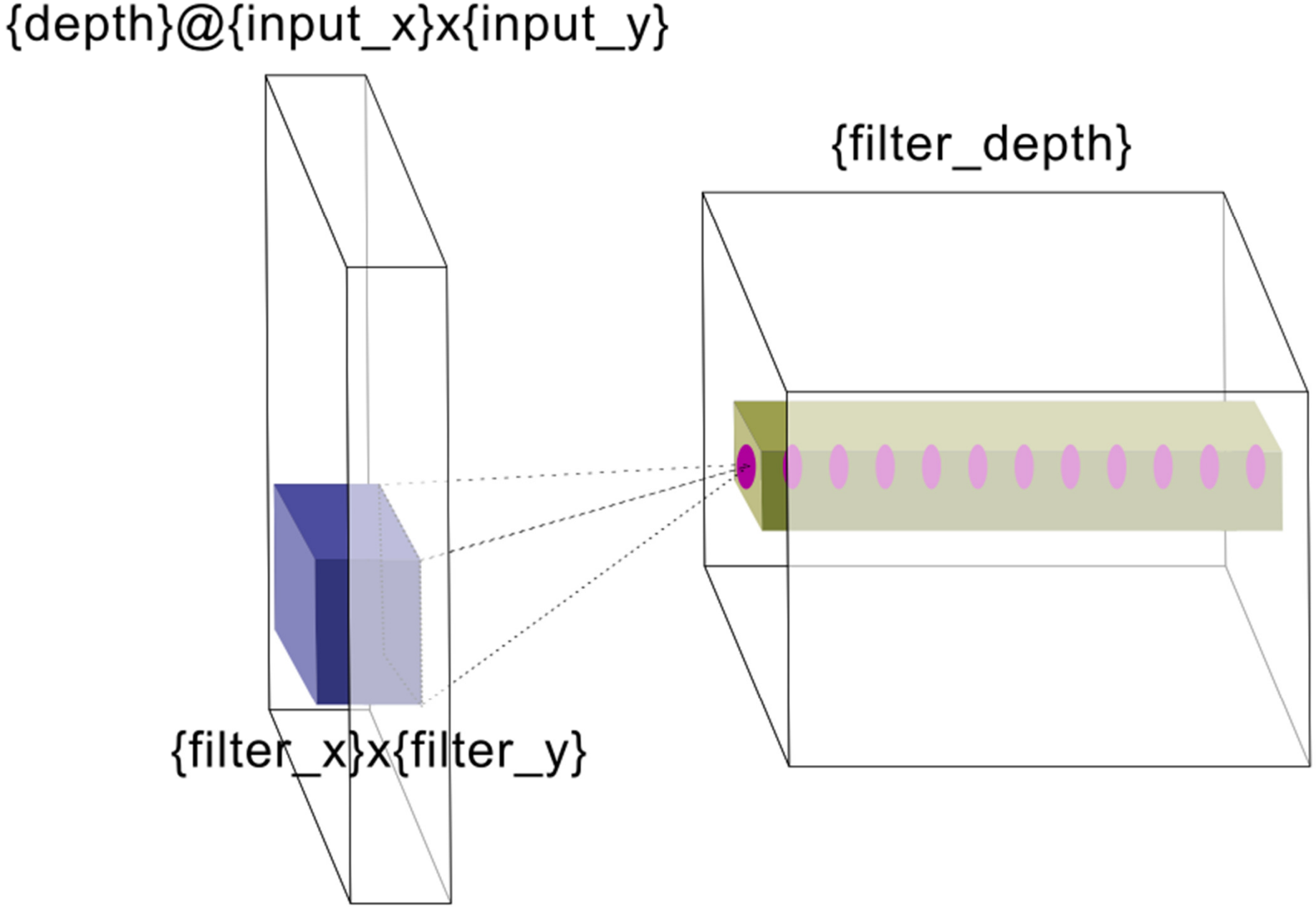
1. {depth}@{input_x}x{input_y},
2. {filter_x}x{filter_y}@*{filter_depth}* *(tipikus méret: 3x3 vagy 5x5)*
3. {stride_x}x{stride_y} *(tipikus méret: 1x1 vagy 2x2)*
4. *{zeropadding_x}x{zeropadding_y}*
   Hogy bemenettel azonos méretet kapjunk (s:1x1): *zeropadding_*=(filter_*-1)/2*

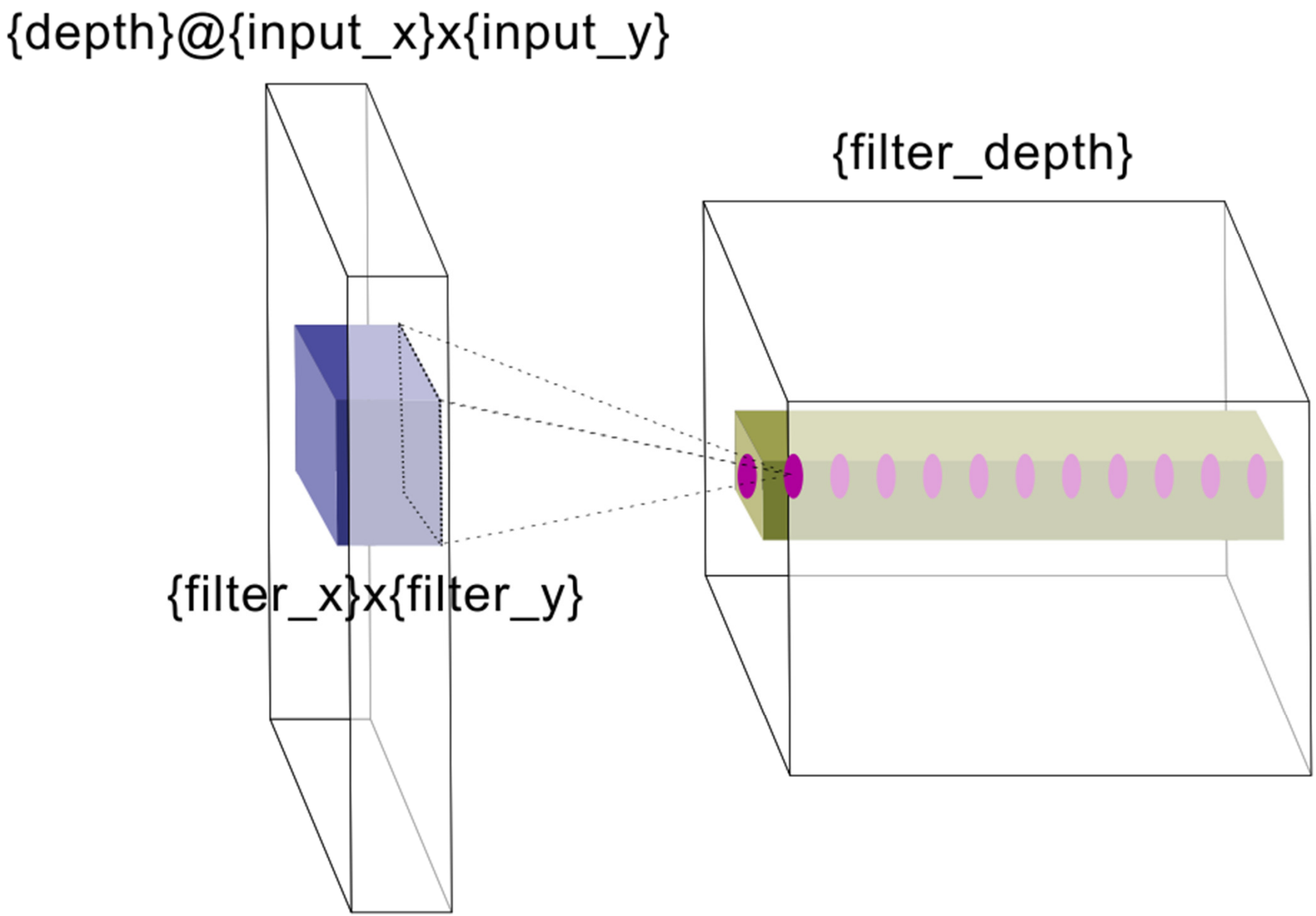
- Kimenet:

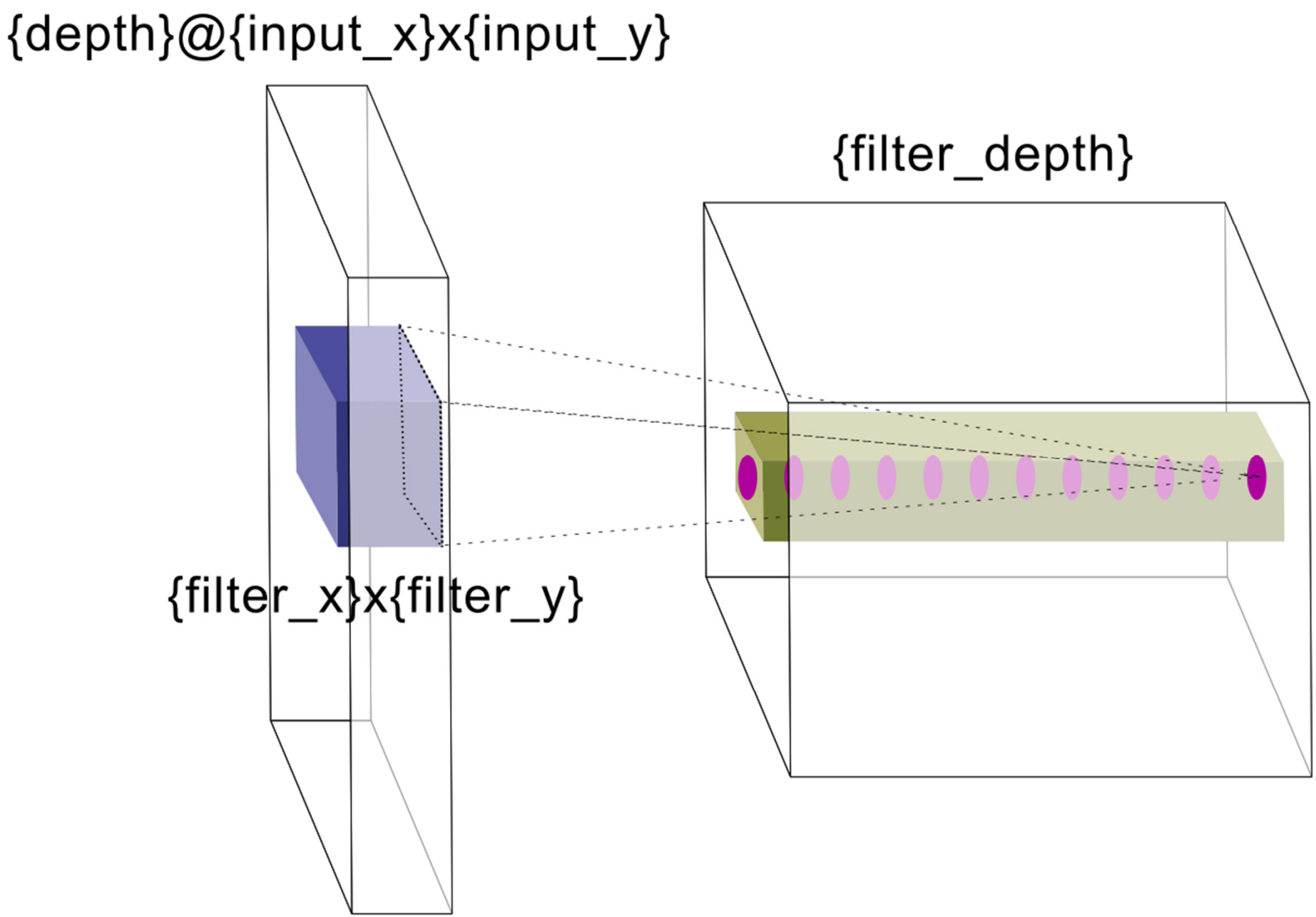
  {filter_depth}@
  {(input_x−filter_x+2*zeropadding_x)/stride_x+1}x
  {(input_y−filter_y+2*zeropadding_y)/stride_y+1}

- *Röviden példa: 3@32x32,f:5x5@12,s:1x1,z: 0x0*
- Backpropagation: kiszámoljuk az összes súly gradiensét, és szeletenként összeadjuk őket.


{depth}@{input_x}x{input_y}
{filter_depth}
{filter_x}x{filter_y}


{depth}@{input_x}x{input_y}
{filter_depth}
{filter_x}x{filter_y}


{depth}@{input_x}x{input_y}
{filter_depth}
{filter_x}x{filter_y}


{depth}@{input_x}x{input_y}
{filter_depth}
{filter_x}x{filter_y}

Példa: *3@32x32,f:5x5@12,s:1x1,z:0x0*

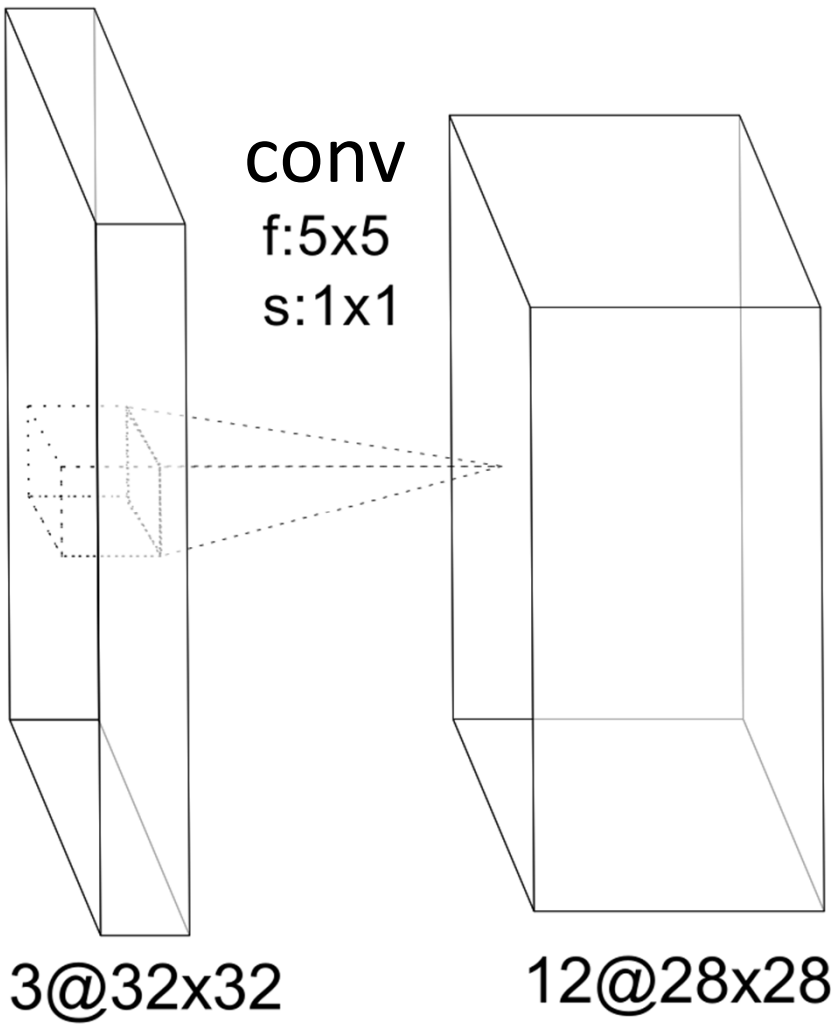
Kimenet: 12@28x28

Súlyok száma első réteg és egy második rétegben lévő sík között:

5*5*3 + 1 (bias) = 76

Súlyok száma összesen:

(5*5*3 +1)*12 = 912

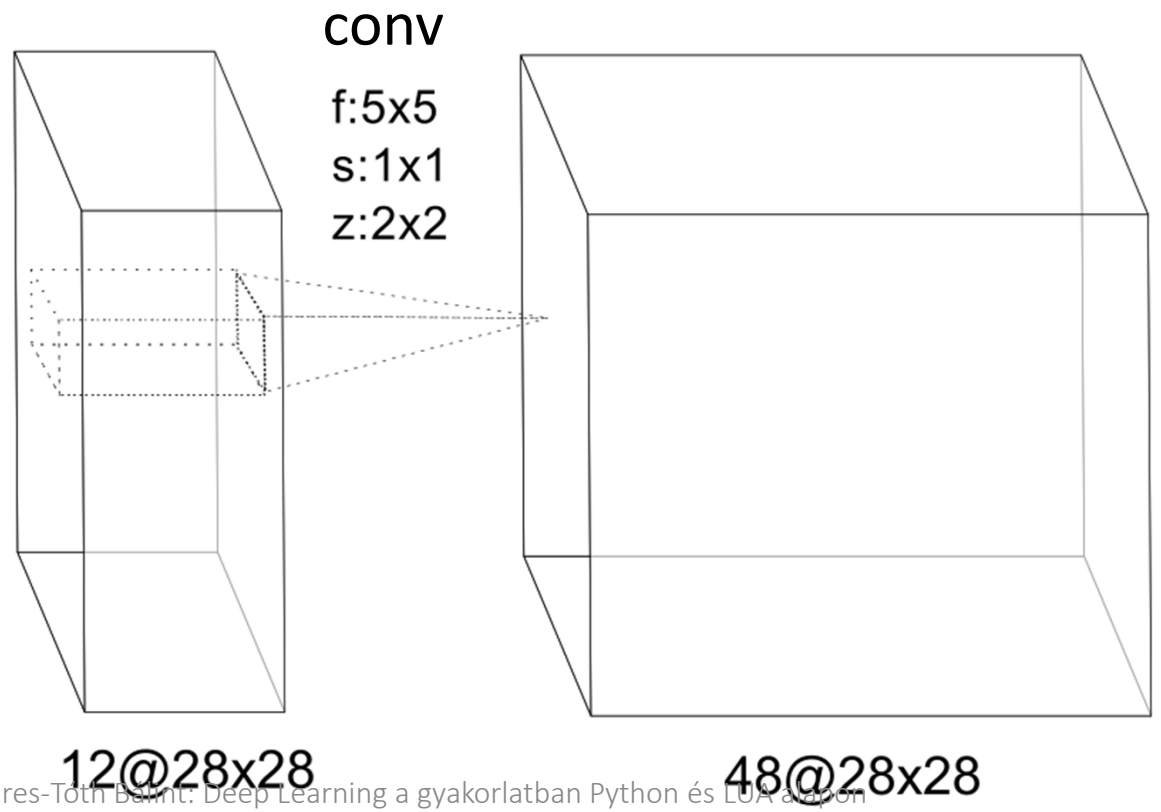


Példa: *12@28x28,f:5x5@48,s:1x1,z:2x2*

Kimenet: 48@28x28

Súlyok száma egy sík esetén: 5*5*12 + 1 (bias) = 301

Súlyok száma: (5*5*12+1)*48 = 14448

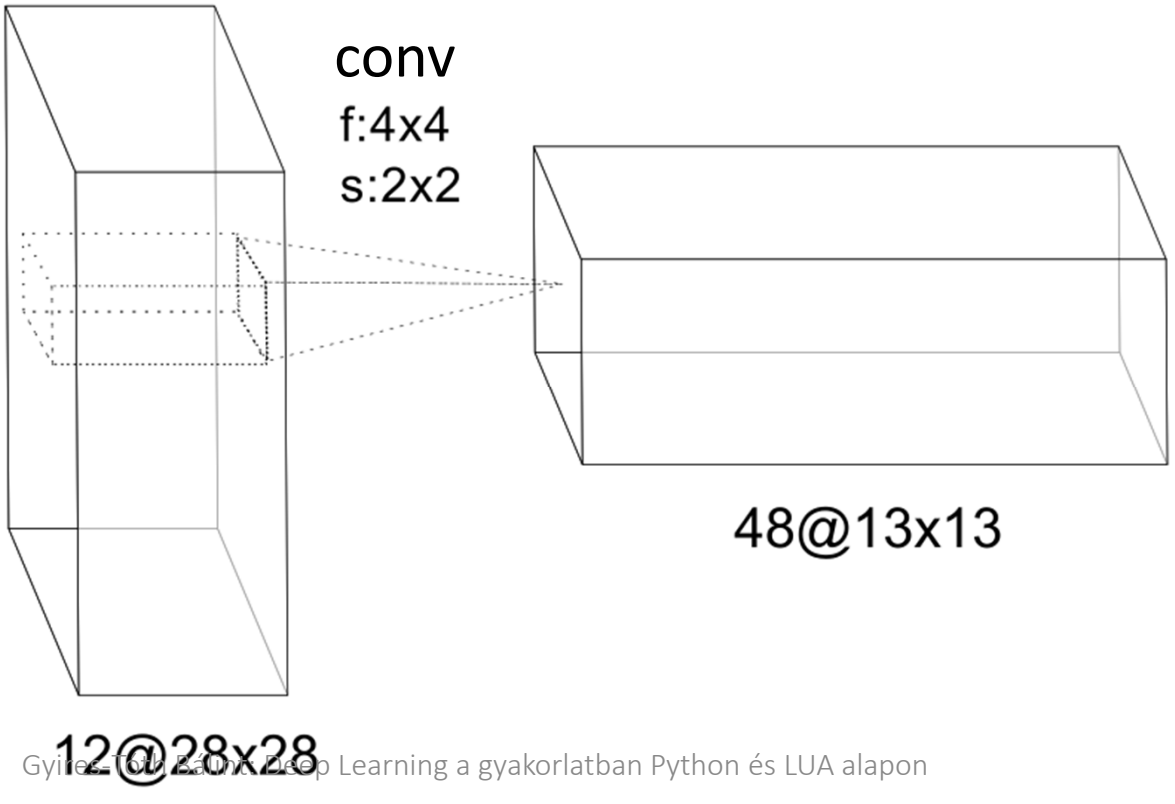


Példa: *12@28x28,f:4x4@48,s:2x2,z:0x0*

Kimenet: 48@13x13

Súlyok száma egy sík esetén: 4*4*12+1=193

Súlyok száma: (4*4*12+1)*48=9264



Építőelemek: max-pool réteg

- Dimenzió redukció + túltanulás ellen.
- Maximumot veszi szeletenként.
- Mélység azonos marad.
- Újabban egyre kevesebbet használnak (pl. VAE, GAN).
- Backprop: csak a legnagyobb érték mentén. (ezt tárolni kell)

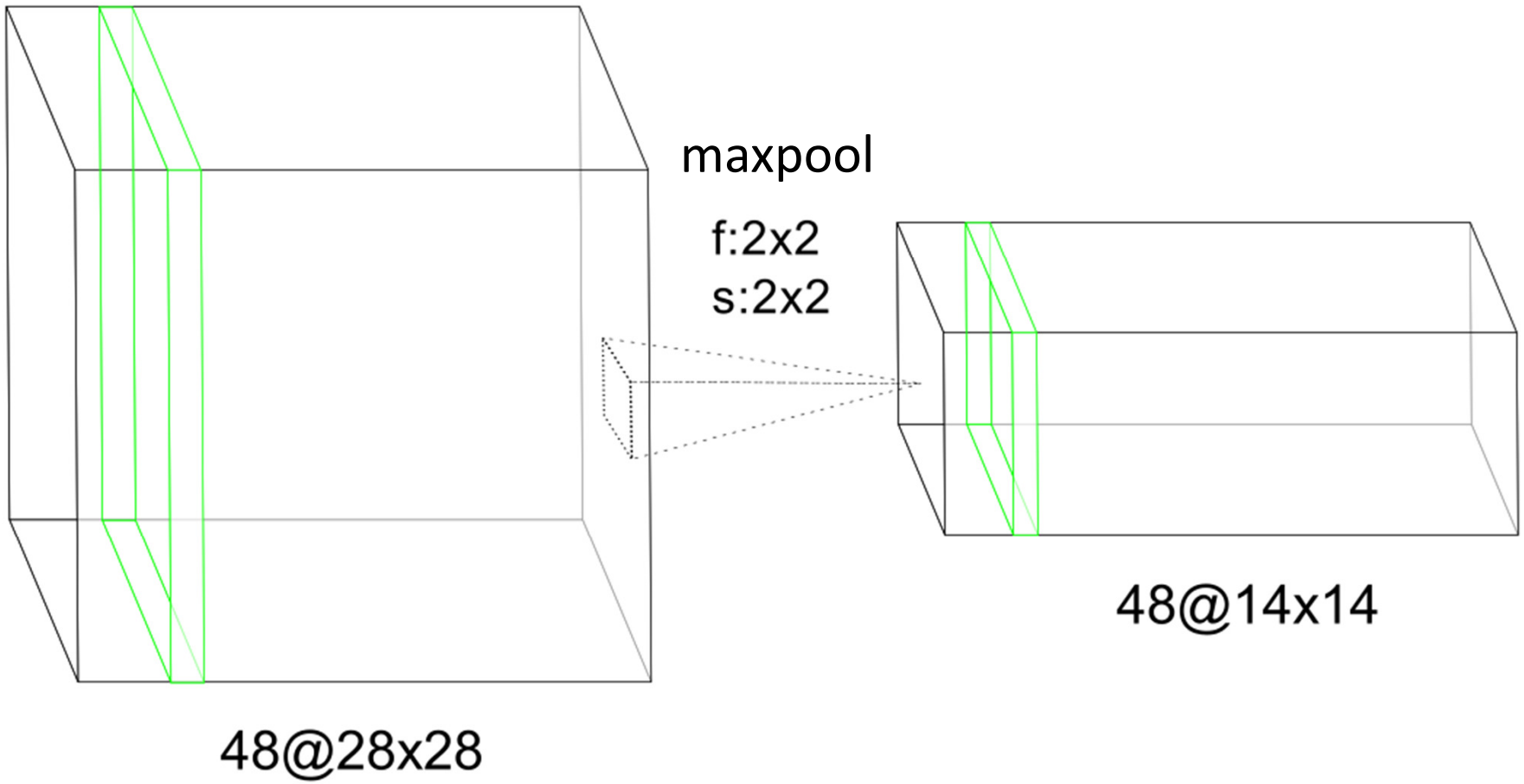
Hiperparaméterek:

1. {filter_x}x{filter_y},
2. {stride_x}x{stride_y}

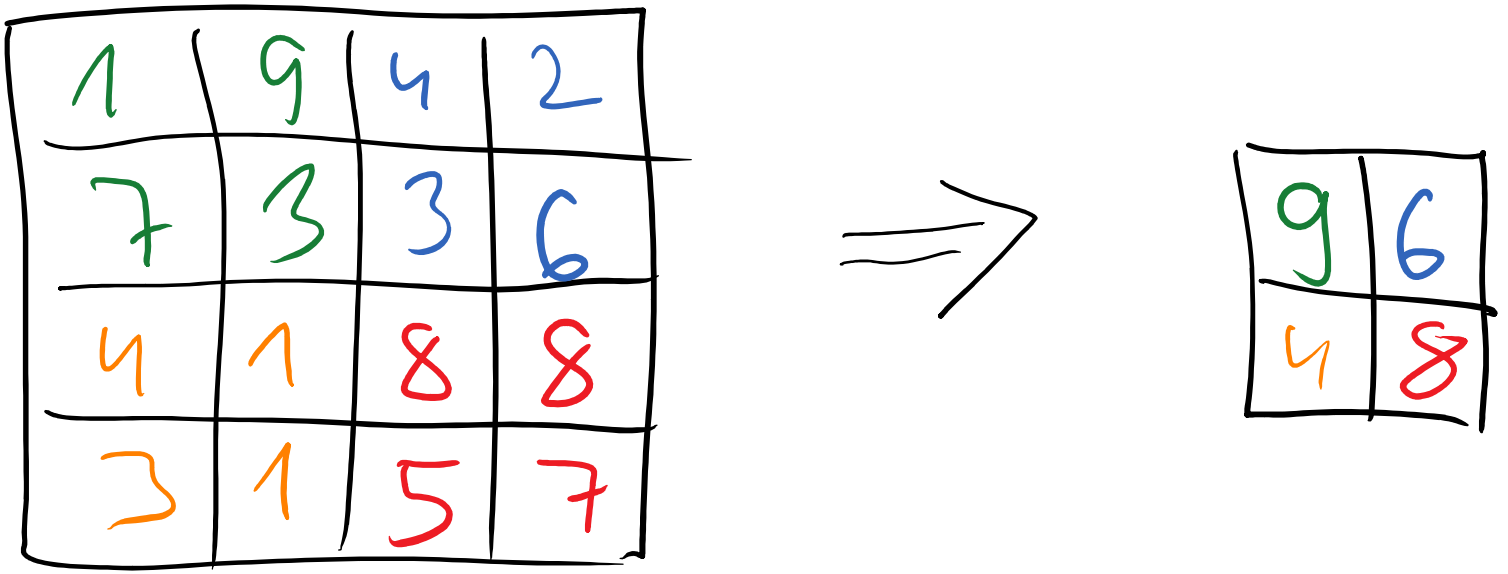
- Kimenet:
  ({input_*}-{filter_*})/{stride_*}+1

Max-pool példa: *48@28x28*,f:2x2,s:2x2

- Kimenet: 48@14x14



Max-pool példa

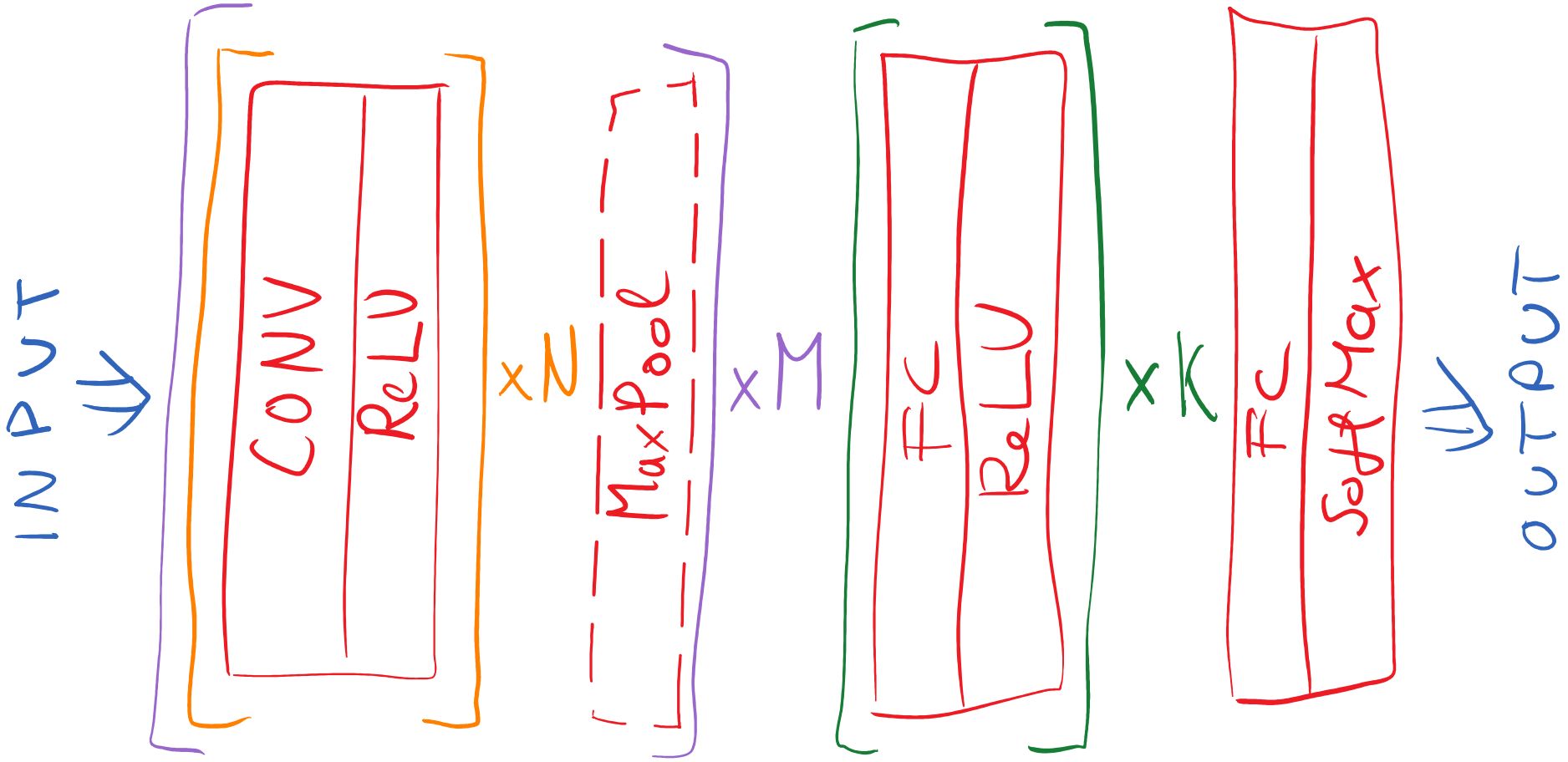


$f: 2 \times 2, s: 2 \times 2$

Egyszerű konvolúciós háló

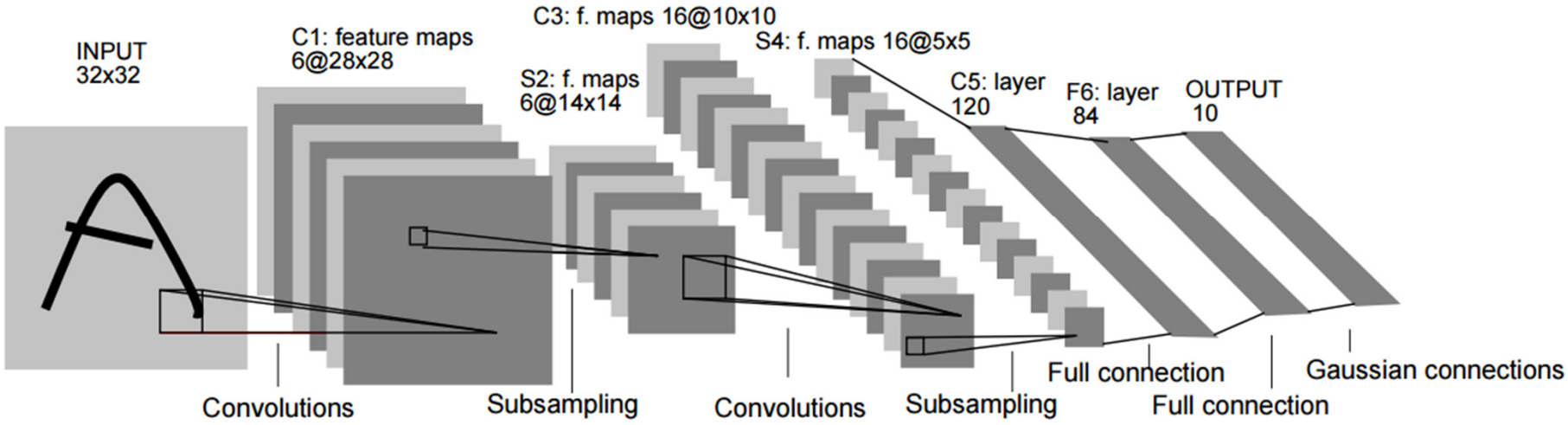
1) Bemenet: 3@32x32
2) Conv: 12@32x32
3) ReLU: 12@32x32
4) MaxPool: 12@16x16
5) FC: 10@1x1

Általános konv. háló architektúra



LeNet5

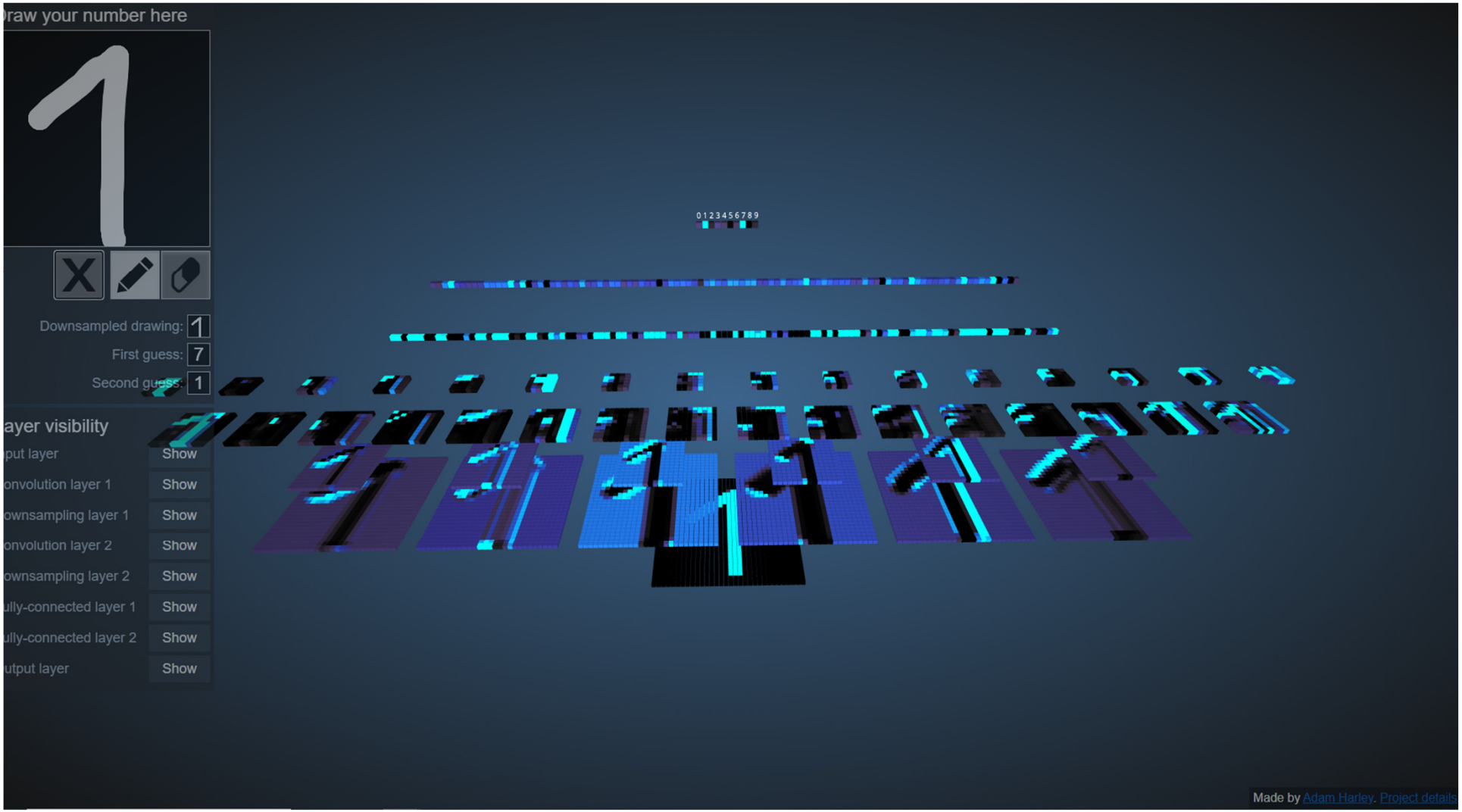
LeNet5: LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.



LeNet5 – Keras

```
model.add(Convolution2D(6, 5, 5, activation
    = 'tanh', input_shape=in_shape))
model.add(MaxPooling2D(pool_size=(2, 2),
    strides=(2, 2)))
model.add(Convolution2D(16, 5, 5,
    activation = 'tanh'))
model.add(MaxPooling2D(pool_size=(2, 2) ,
    strides=(2, 2)))

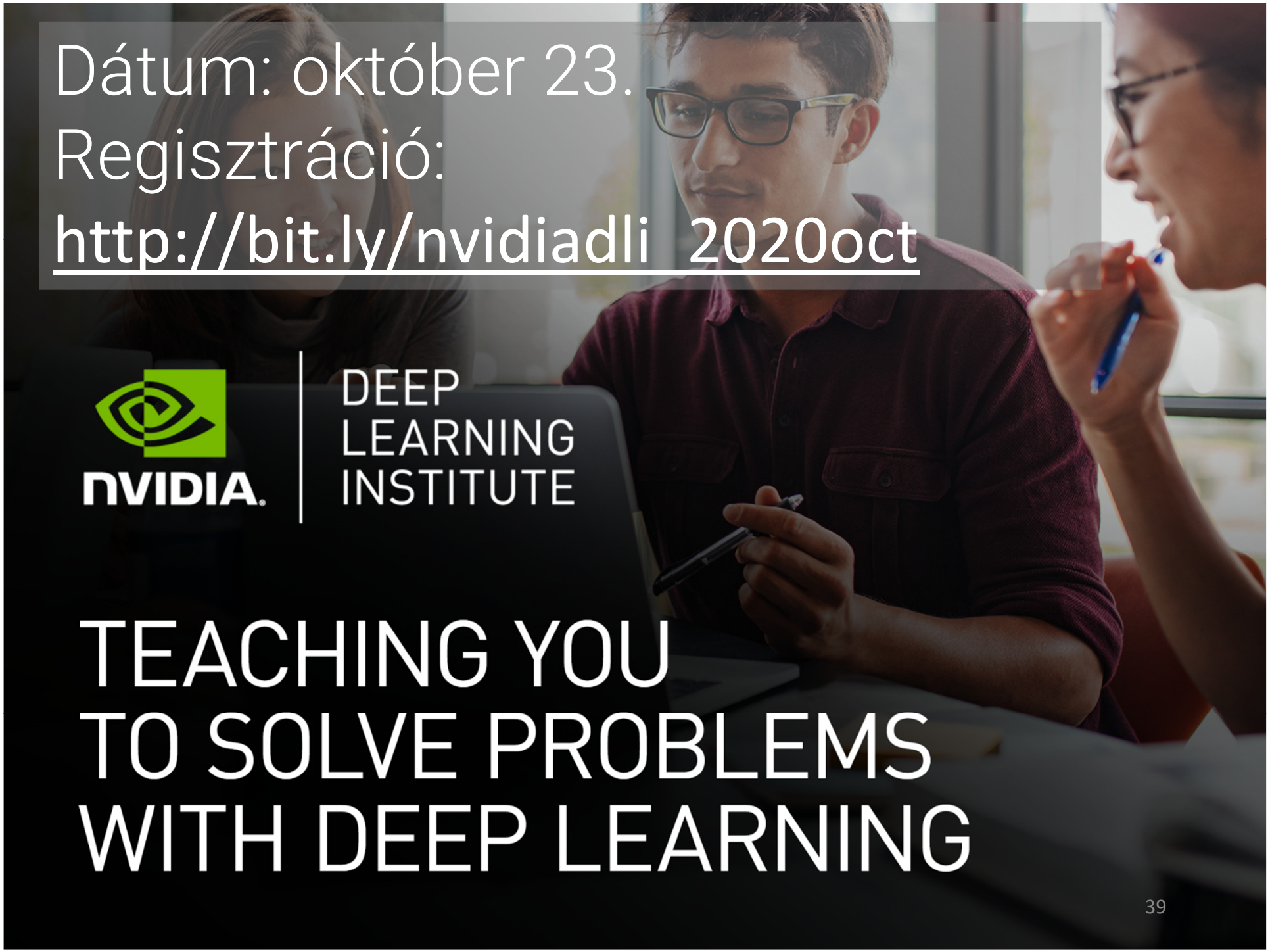
model.add(Flatten())
model.add(Dense(120, activation = 'tanh'))
model.add(Dense(84, activation = 'tanh'))
model.add(Dense(nb_classes, activation =
    'linear'))
```



https://www.cs.ryerson.ca/~aharley/vis/

3. Kis házi feladat

- Készíts előrecsatolt mély neuronhálót/hálókat, ami megjósolja a várható hőmérsékletet egy napra, egy hétre és egy hónapra előre Budapesten.
- Lehetséges források:
  - http://idojarasbudapest.hu/archivalt-idojaras
  - https://www.metnet.hu/?m=napi-adatok
  - http://www.amsz.hu/ws/index.php?view=archiv
  - http://wieser.hu/wxtempdetail.php
  - Bármi, amit találsz!
- Dokumentáld (kommentek, readme.md)
- Legyen egyszerűen futtatható.
- A tanítóadatokat és a 3 jósolt értéket is rögzítsd külön:
  - október 28, november 3, november 24
- Tetszőleges deep learning keretrendszert használhatsz.
- ANGOLUL!
- Határidő: 2020. október 27. (kedd) 23:59

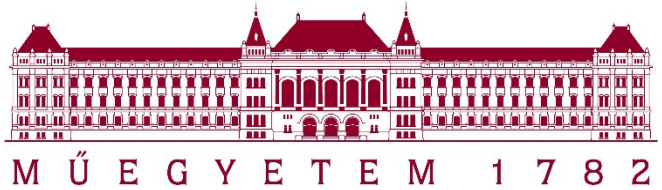

Dátum: október 23.
Regisztráció:
http://bit.ly/nvidiadli_2020oct
NVIDIA
DEEP LEARNING INSTITUTE
TEACHING YOU TO SOLVE PROBLEMS WITH DEEP LEARNING



Névnek a NEPTUN kódodat add meg!

https://kahoot.it/







Köszönöm a figyelmet!

toth.b@tmit.bme.hu

SmartLab
Intelligent Interactions

http://smartlab.tmit.bme.hu

NVIDIA
GPU
EDUCATION
CENTER