Konvolutivne Neuronske Mreže
Convolutional Neural Networks, CNN

Predavač: Aleksandar Kovačević

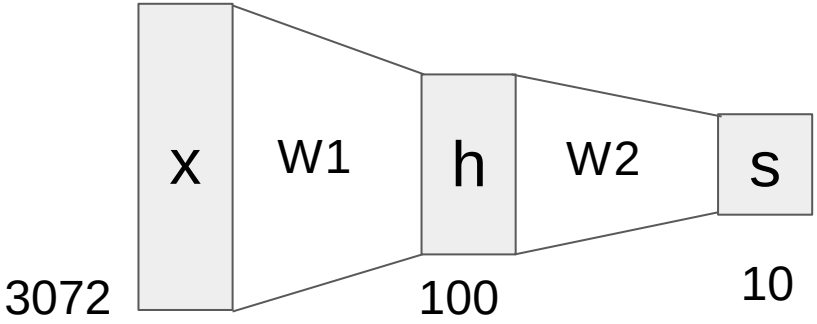
Slajdovi preuzeti sa CS 231n, Stanford

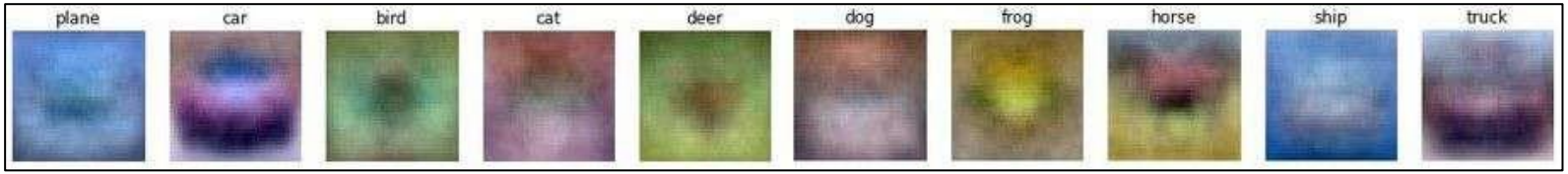
http://cs231n.stanford.edu/

Do sada: Neuronske mreže

Linearna skor funkcija: $f = Wx$

2-slojna Neuronska Mreža $f = W_2 \max(0, W_1 x)$





Danas: Konvolutivne Neuronske Mreže

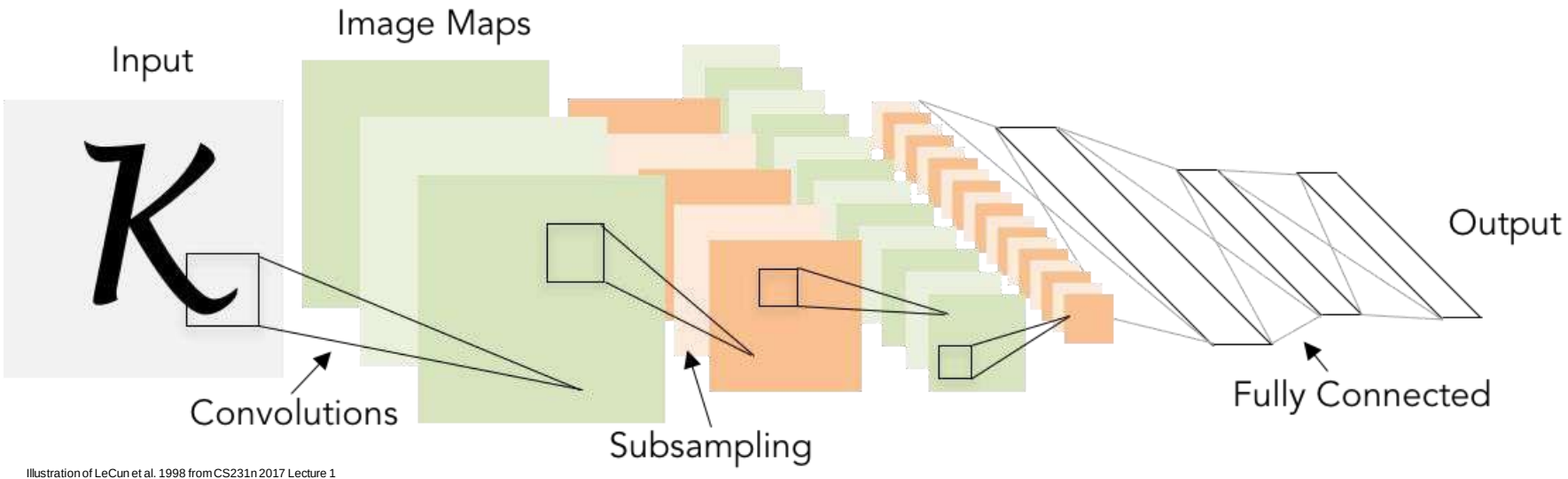


Illustration of LeCun et al. 1998 from CS231n 2017 Lecture 1

Malo istorije...

Mašina **Mark I Perceptron** je prva implementacija perceptron algoritma.

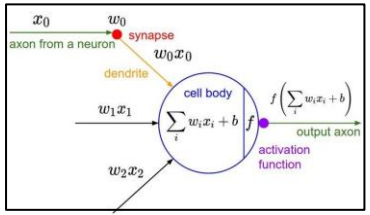
Mašina je bila povezana na kameru koja je koristila 20×20 kadmijum sulfid fotoćelije da proizvede sliku od 400 piksela.

mašina je mogla da prepoznaje slova abecede

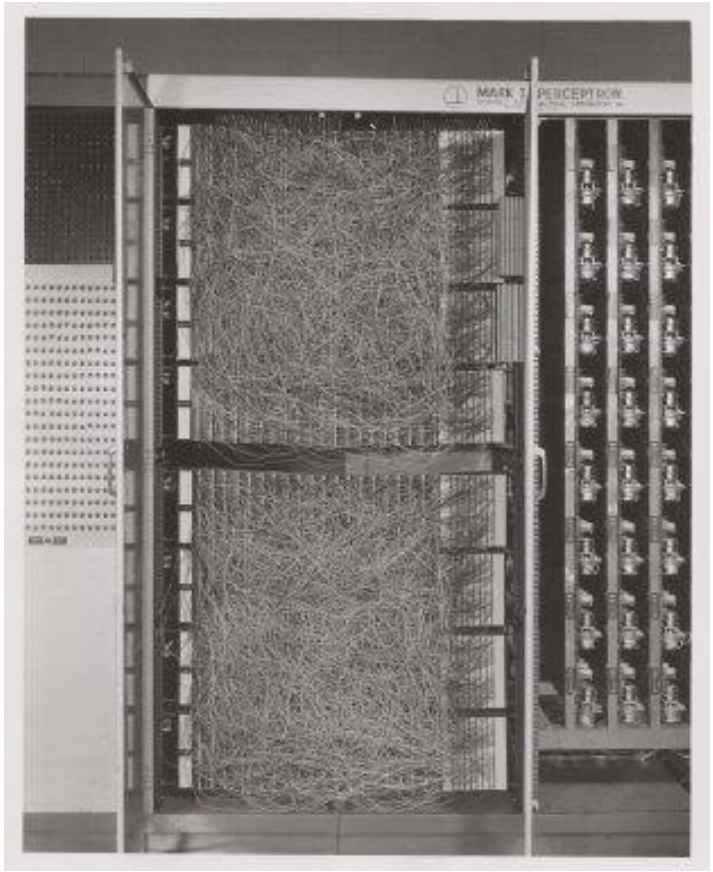
$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

pravilo za promenu težina:

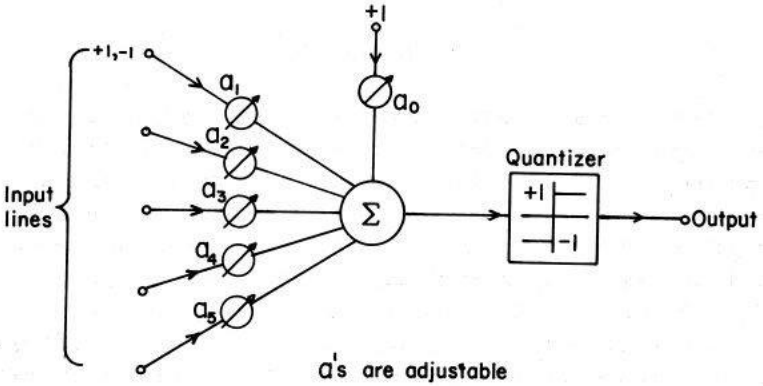
$$w_i(t+1) = w_i(t) + \alpha(d_j - y_j(t))x_{j,i},$$

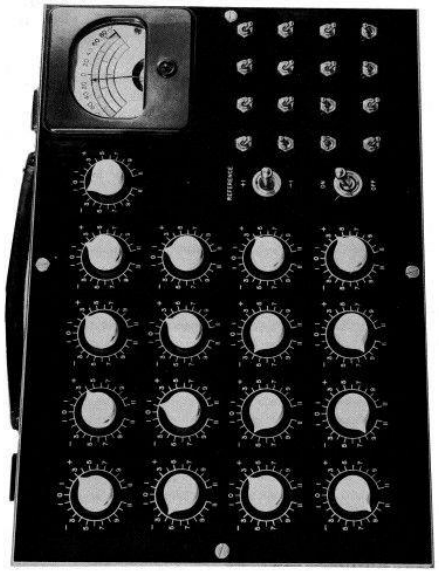


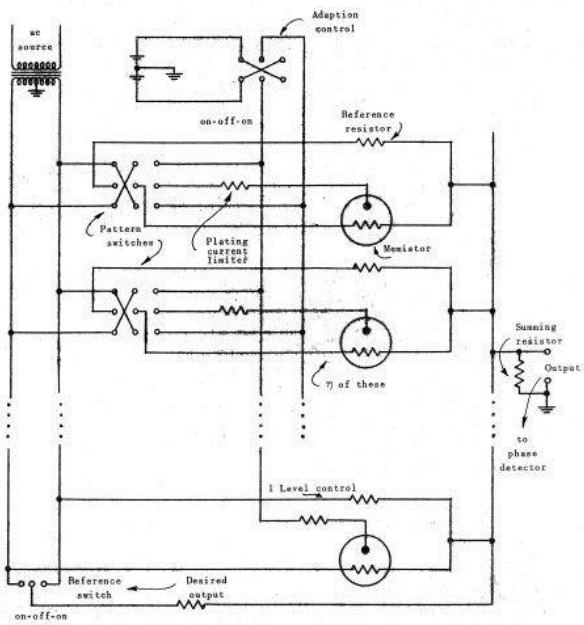
Frank Rosenblatt, ~1957: Perceptron



Malo istorije...



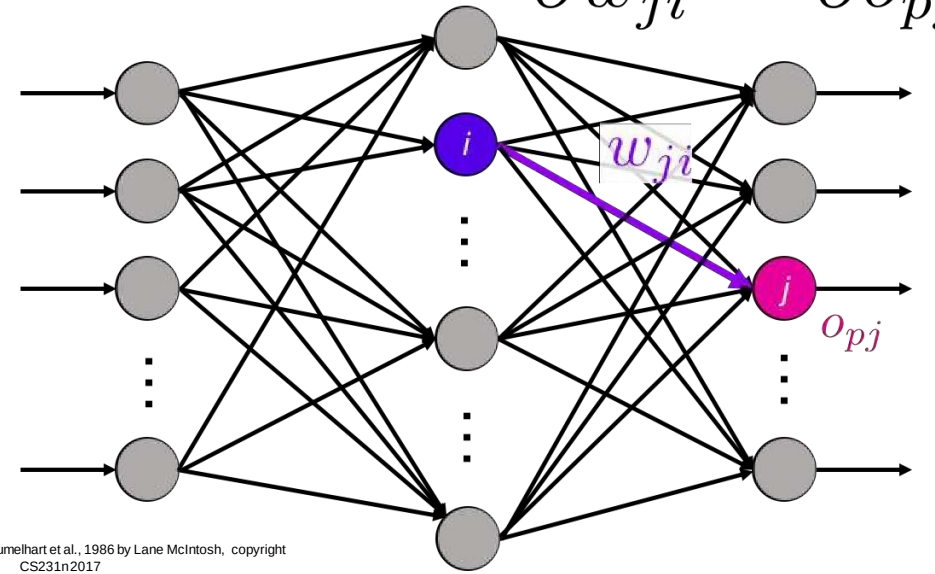




Widrow and Hoff, ~1960: Adaline/Madaline

Malo istorije...

$$\frac{\partial E_p}{\partial w_{ji}} = \frac{\partial E_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial w_{ji}}$$



input pattern

output pattern *p*

error E_p

backprop matematika

Illustration of Rumelhart et al., 1986 by Lane McIntosh, copyright CS231n2017

Rumelhart et al., 1986: Prvi put da je back-propagation postao popularan

Malo istorije...

[Hinton and Salakhutdinov 2006]

Počinje interesovanje za Deep Learning

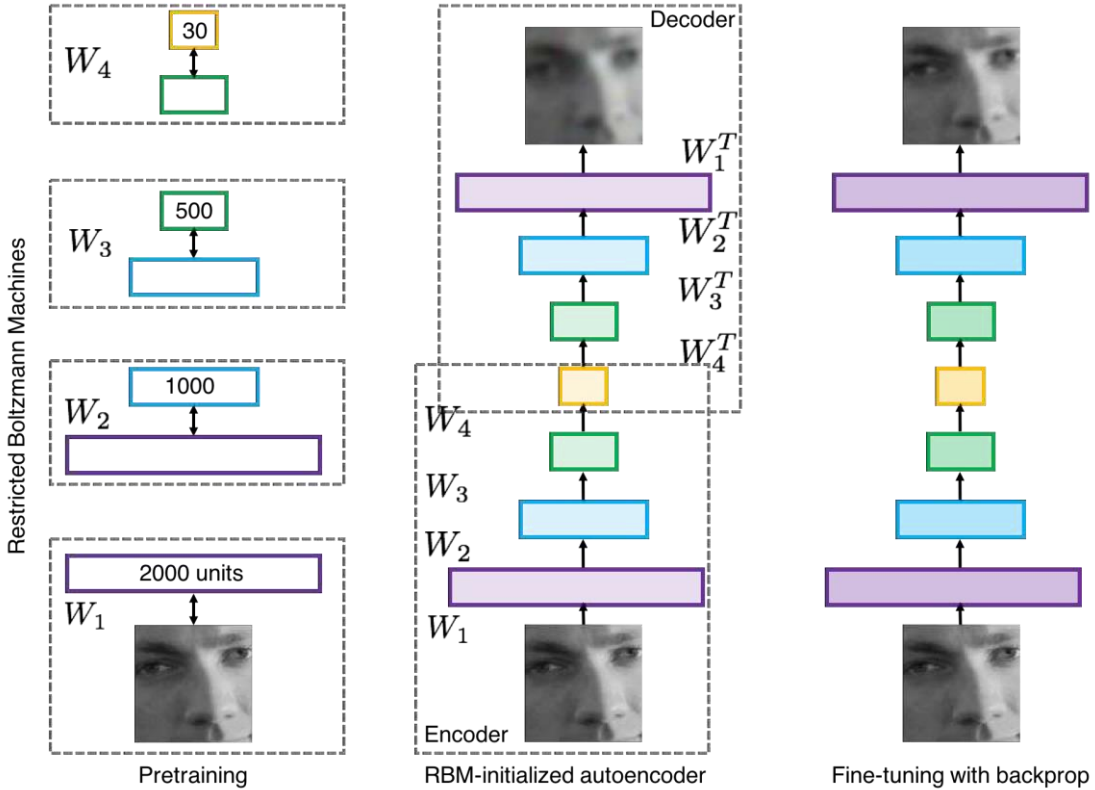


Illustration of Hinton and Salakhutdinov 2006 by Lane McIntosh, copyright CS231n 2017

Prvi dobri rezultati

Acoustic Modeling using Deep Belief Networks
Abdel-rahman Mohamed, George Dahl, Geoffrey Hinton, 2010
Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition
George Dahl, Dong Yu, Li Deng, Alex Acero, 2012

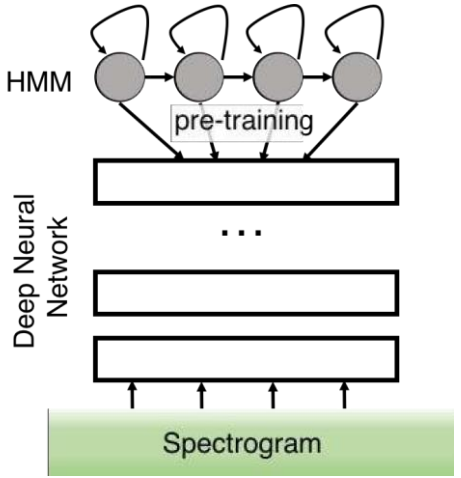
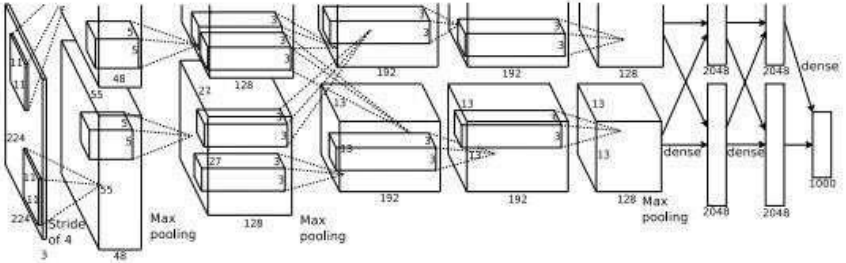
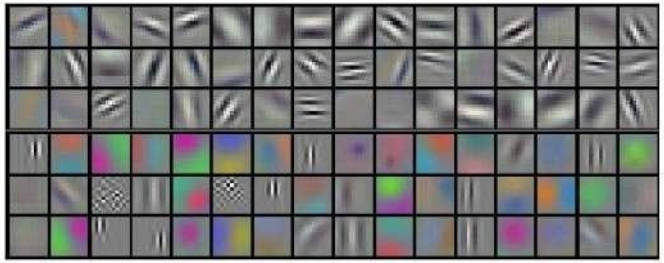


Illustration of Dahl et al. 2012 by Lane McIntosh, copyright CS231n 2017

Imagenet classification with deep convolutional neural networks
Alex Krizhevsky, Ilya Sutskever, Geoffrey E Hinton, 2012

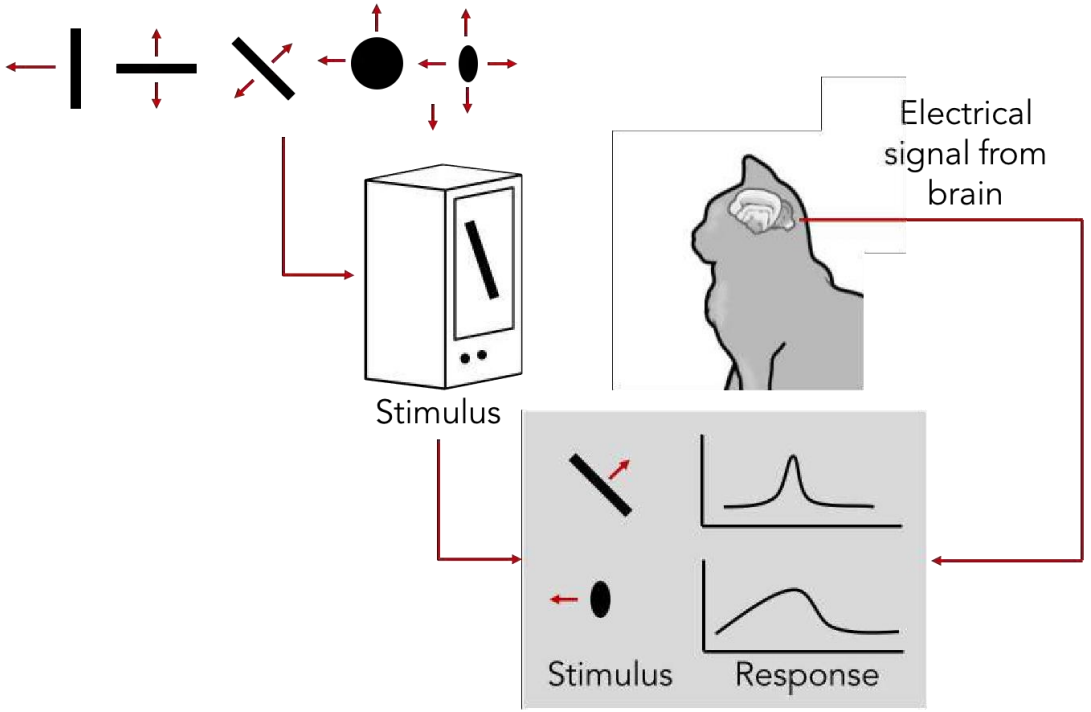




Figures copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

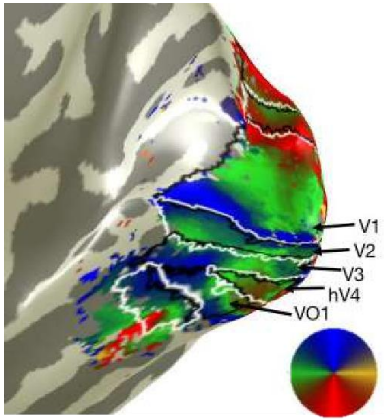
Malo Istorije:

Hubel & Wiesel,
1959
RECEPTIVE FIELDS OF SINGLE NEURONES IN
THE CAT'S STRIATE CORTEX
1962
RECEPTIVE FIELDS, BINOCULAR INTERACTION
AND FUNCTIONAL ARCHITECTURE IN
THE CAT'S VISUAL CORTEX
1968...

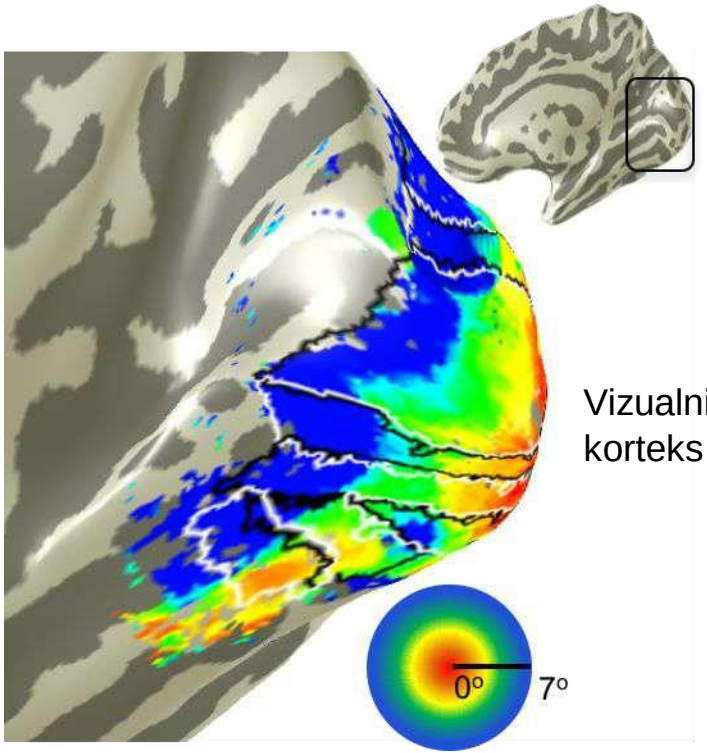


Malo istorije...

Topografska mapa korteksa:
ćelije koje su blizu u korteksu procesiraju regione koji su blizu u vizualnom polju



Ljudski mozak



Vizualni korteks

Retinotopy images courtesy of Jesse Gomez in the Stanford Vision & Perception Neuroscience Lab.

Hijerahjijska Organizacija

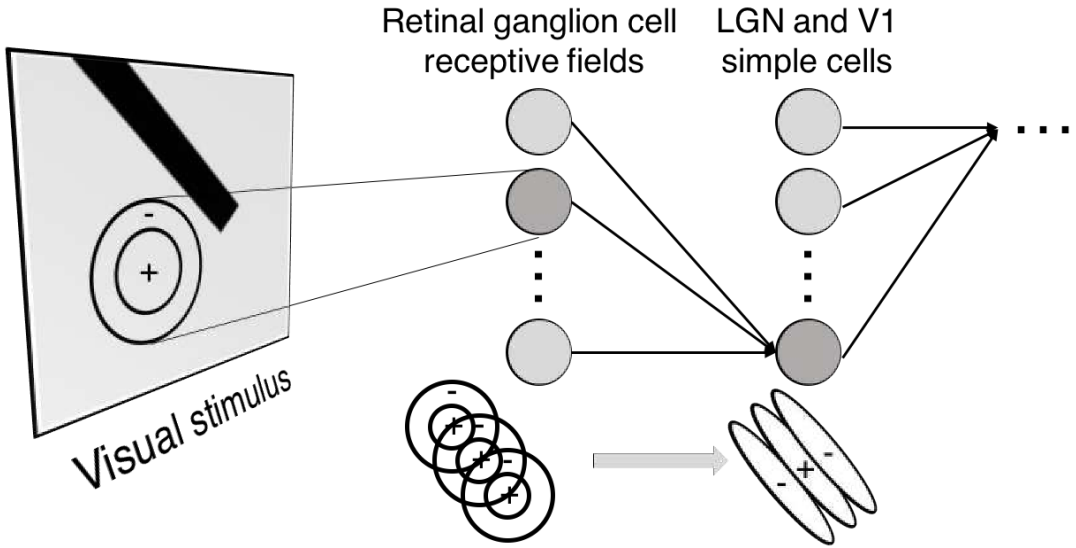
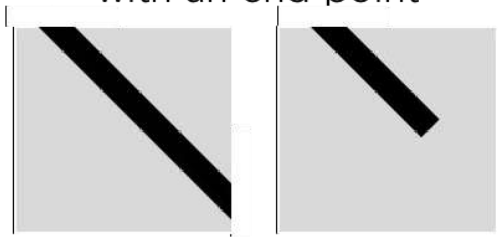


Illustration of hierarchical organization in early visual pathways
by Lane McIntosh, copyright CS231n2017

Simple cells:
Response to light orientation

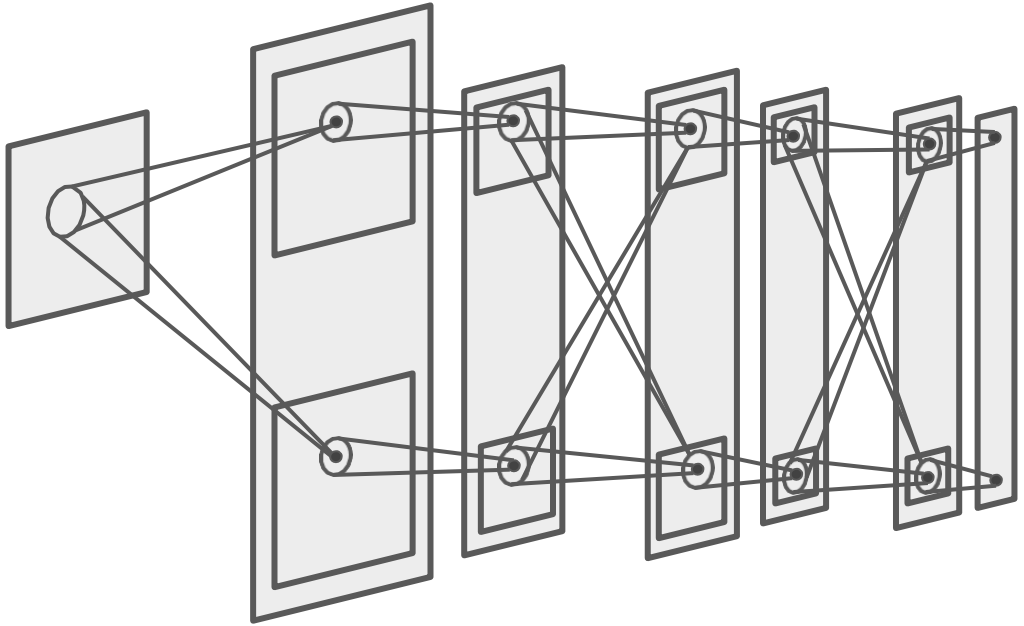
Complex cells:
Response to light orientation and movement

Hypercomplex cells:
response to movement with an end point



Malo istorije:

Neocognitron
[Fukushima 1980]

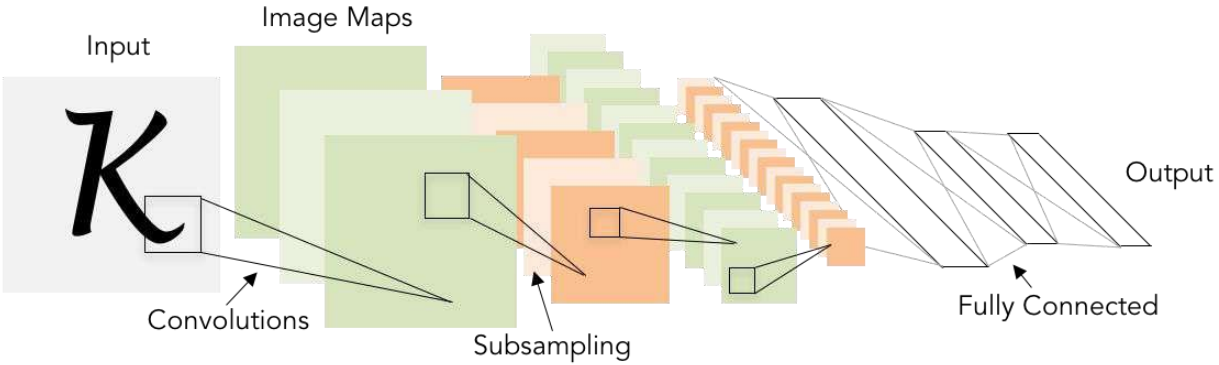


“sendvič” arhitektura (SCSCSC…)
jednostavne ćelije: parametri koji se uče
kompleksne ćelije: agregacija vrednosti jednostavnih ćelija

Malo istorije:

Gradient-based learning applied to document recognition

[LeCun, Bottou, Bengio, Haffner 1998]



LeNet-5

Malo istorije:

ImageNet Classification with Deep Convolutional Neural Networks *[Krizhevsky, Sutskever, Hinton, 2012]*

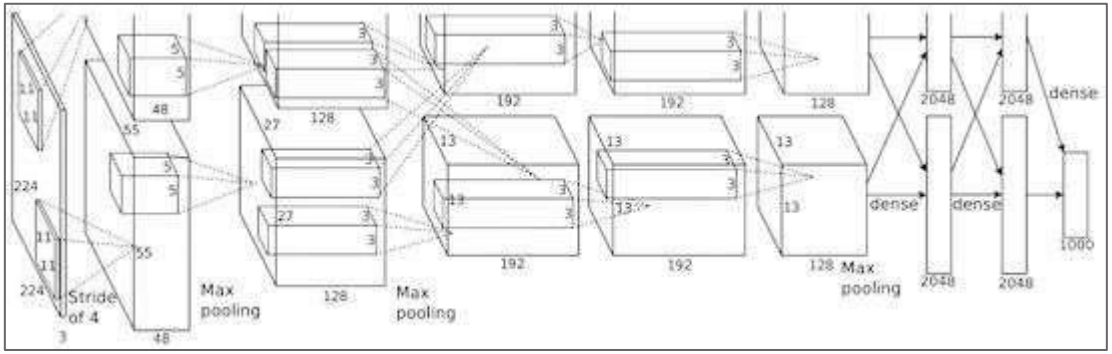


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

"AlexNet"

Danas: Konvolutivne Mreže su svuda

Klasifikacija

Pronalaženje

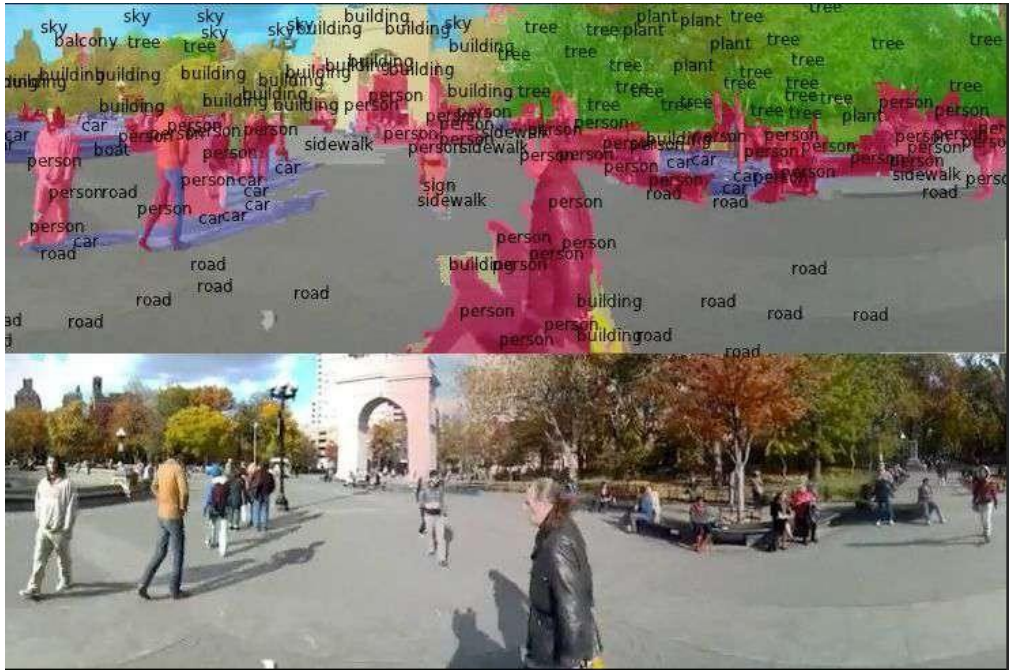
Danas: Konvolutivne Mreže su svuda

Detekcija

Figures copyright Shaoqing Ren, Kaiming He, Ross Girschick, Jian Sun, 2015. Reproduced with permission.

[Faster R-CNN: Ren, He, Girshick, Sun 2015]

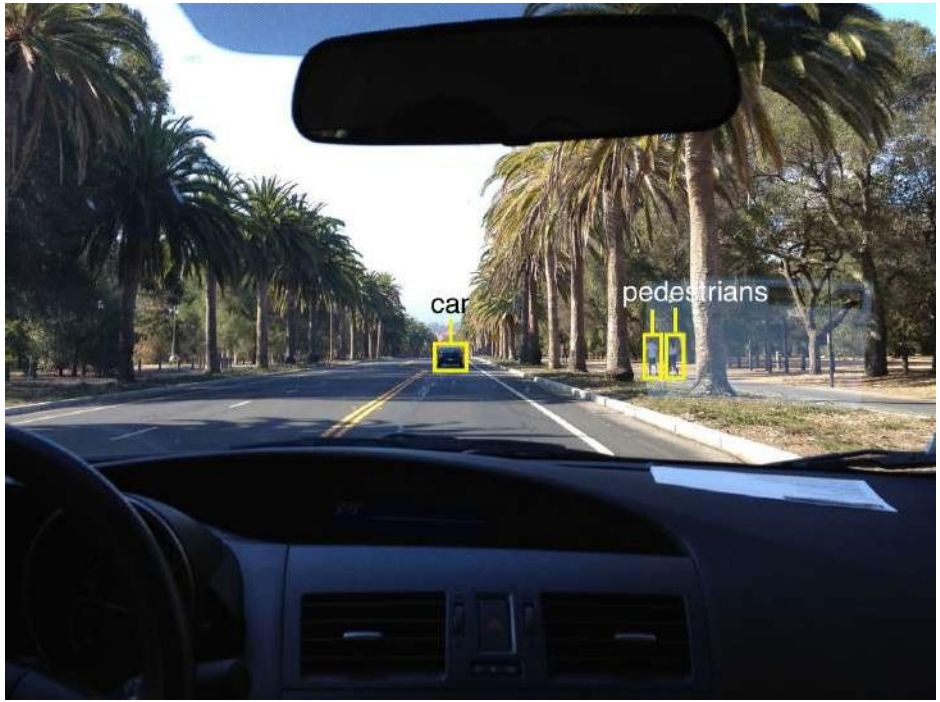
Segmentacija



Figures copyright Clement Farabet, 2012. Reproduced with permission.

[Farabet et al., 2012]

Danas: Konvolutivne Mreže su svuda



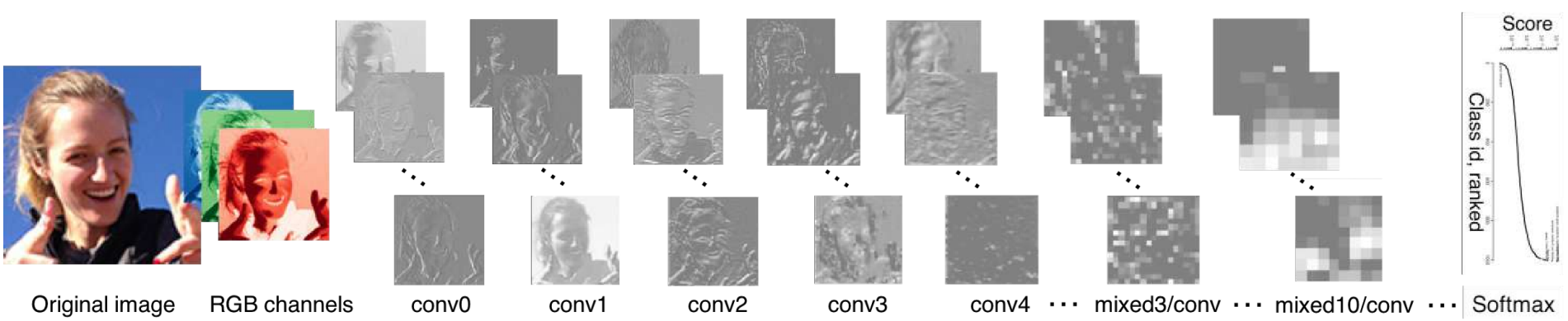
self-driving cars

Photo by Lane McIntosh. Copyright CS231n 2017.

NVIDIA Tesla line

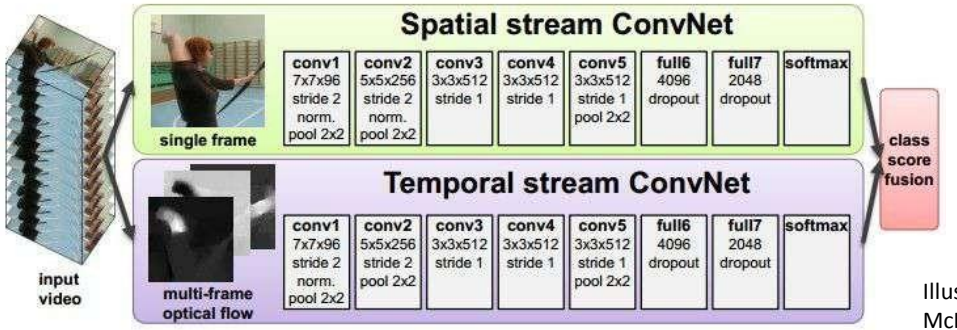
Za *embedded sisteme* tipično se koriste NVIDIA Tegra kartice, sa integrisanim GPU i procesoru zasnovanom na ARM arhitekturi.

Danas: Konvolutivne Mreže su svuda



[Taigman et al. 2014]

Activations of inception-v3 architecture [Szegedy et al. 2015] to image of Emma McIntosh, used with permission. Figure and architecture not from Taigman et al. 2014.



[Simonyan et al. 2014]

Figures copyright Simonyan et al., 2014. Reproduced with permission.

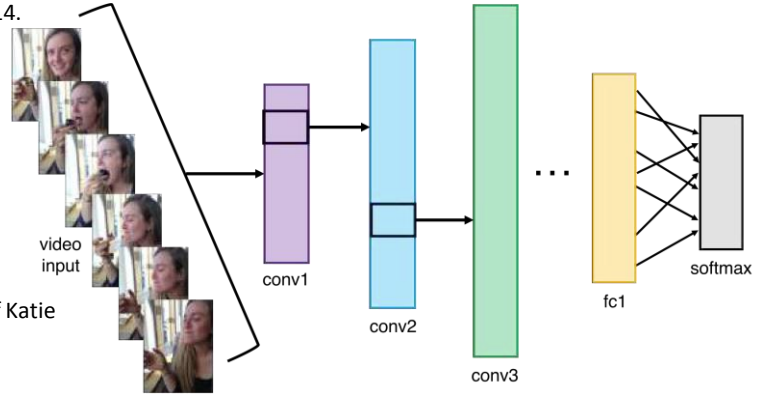
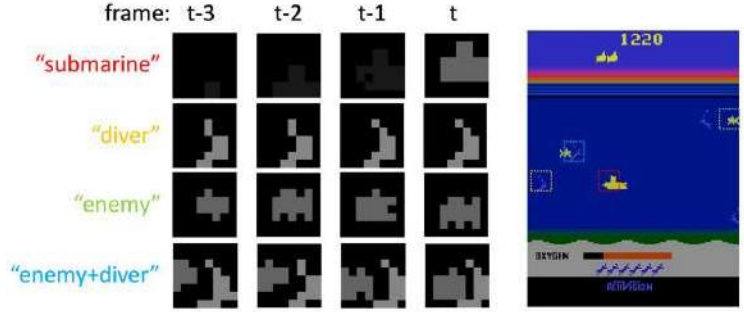


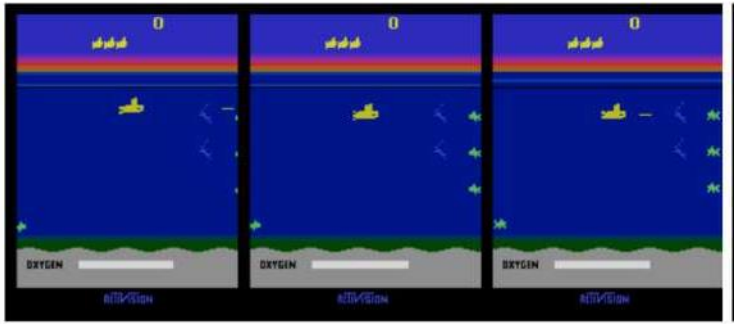
Illustration by Lane McIntosh, photos of Katie Cumnock used with permission.

Danas: Konvolutivne Mreže su svuda

[Toshev, Szegedy 2014]

Images are examples of pose estimation, not actually from Toshev & Szegedy 2014. Copyright Lane McIntosh.

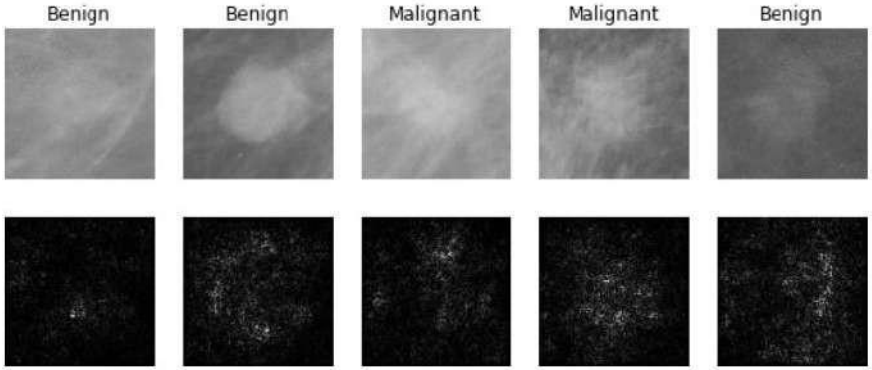




[Guo et al. 2014]

Figures copyright Xiaoxiao Guo, Satinder Singh, Honglak Lee, Richard Lewis, and Xiaoshi Wang, 2014. Reproduced with permission.

Danas: Konvolutivne Mreže su svuda



[Levy et al. 2016]

Figure copyright Levy et al. 2016.
Reproduced with permission.

[Dieleman et al. 2014]

From left to right: public domain by NASA, usage permitted by ESA/Hubble, public domain by NASA, and public domain.

[Sermanet et al. 2011]
[Ciresan et al.]

Photos by Lane McIntosh.
Copyright CS231n 2017.

This image by Christin Khan is in the public domain and originally came from the U.S. NOAA.

Whale recognition, Kaggle Challenge

Photo and figure by Lane McIntosh; not actual example from Mnih and Hinton, 2010 paper.

Mnih and Hinton, 2010

Bez grešaka

A white teddy bear sitting in the grass

Manje greške

A man in a baseball uniform throwing a ball

Ima neke veze sa slikom

A woman is holding a cat in her hand

Image Captioning

[Vinyals et al., 2015]
[Karpathy and Fei-Fei, 2015]

A man riding a wave on top of a surfboard

A cat sitting on a suitcase on the floor

A woman standing on a beach holding a surfboard

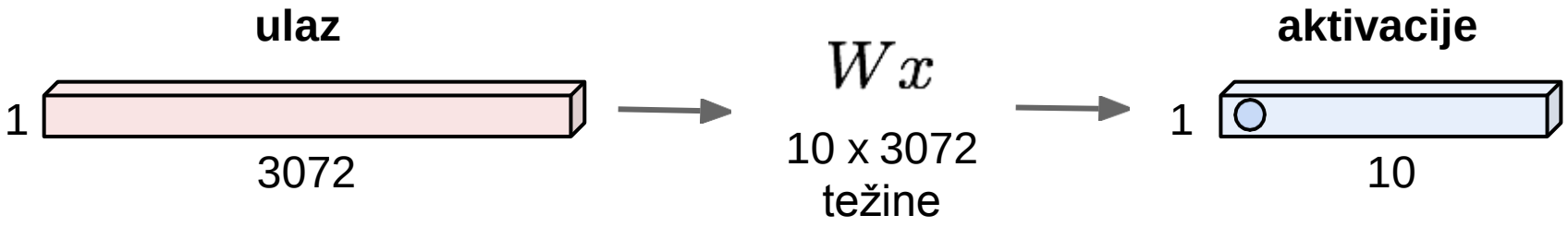
Captions generated by Justin Johnson using Neuraltalk2

Konvolutivne Neuronske Mreže

(Prvo bez analogije sa ljudskim mozgom)

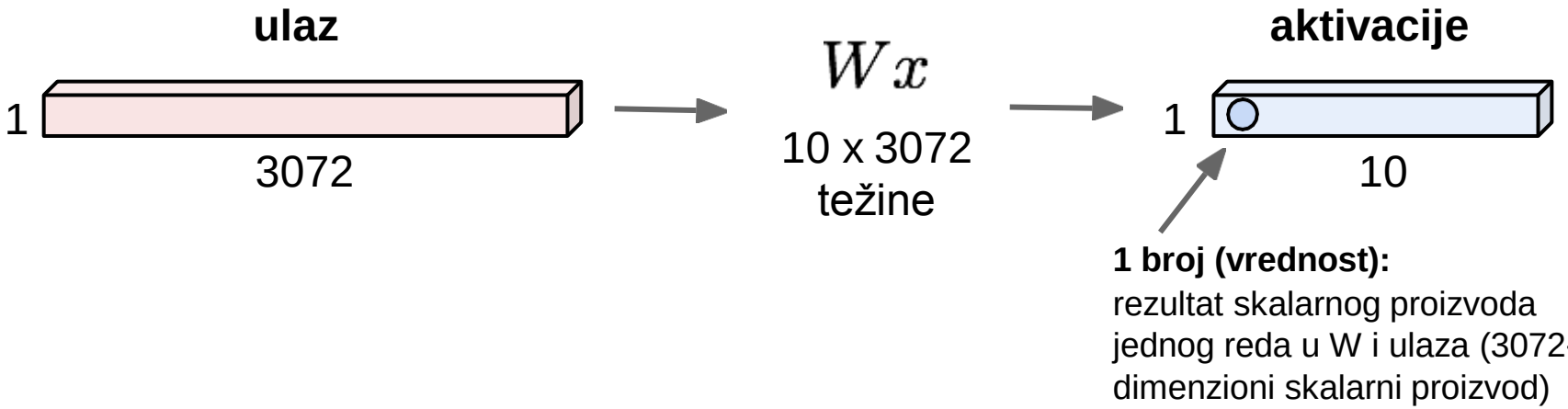
Potpuno Povezan Sloj

32x32x3 slika -> razvučemo u niz 3072 x 1



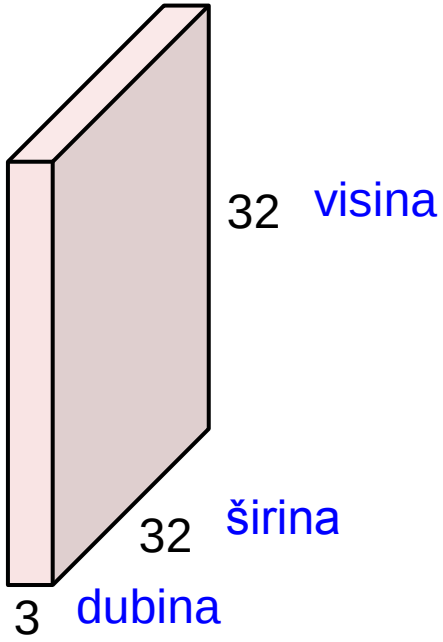
Potpuno Povezan Sloj

32x32x3 slika -> razvučemo u niz 3072 x 1



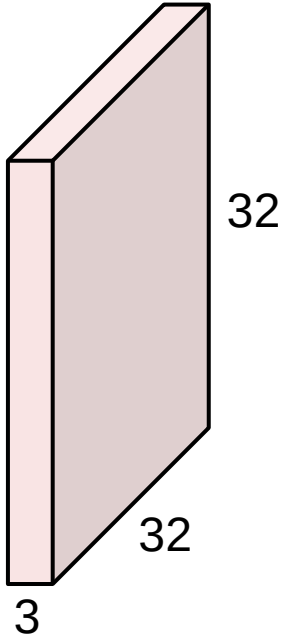
Konvolutivn Sloj

32x32x3 slika -> očuvavamo prostornu strukturu (slika ostaje 2d, a ne 1d niz)

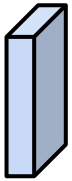


Konvolutivni Sloj

32x32x3 slika



5x5x3
filter

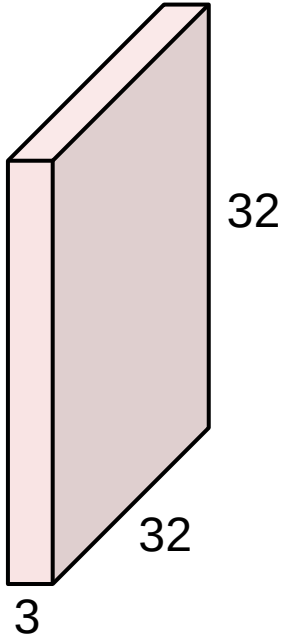


Vršimo Konvoluciju filtera sa slikom tj. “vučemo” filter prostorno po slici i računamo skalarni proizvod

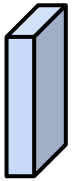
Konvolutivni Sloj

Filteri su uvek iste dubine kao i slika

32x32x3 slika

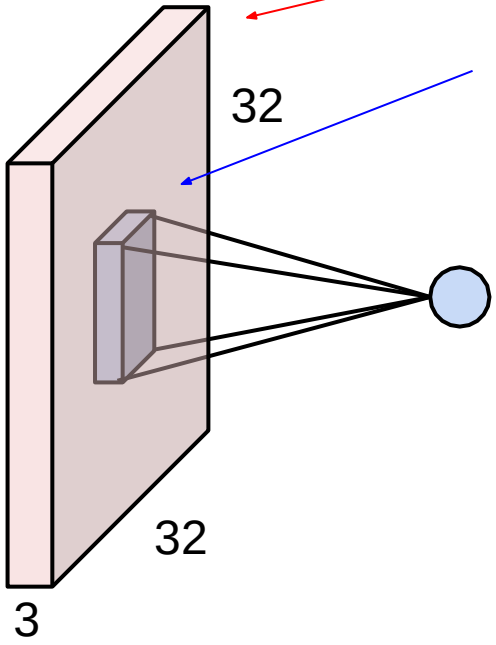


5x5x3 filter



Vršimo Konvoluciju filtera sa slikom tj. “vučemo” filter prostorno po slici i računamo skalarni proizvod

Konvolutivni Sloj



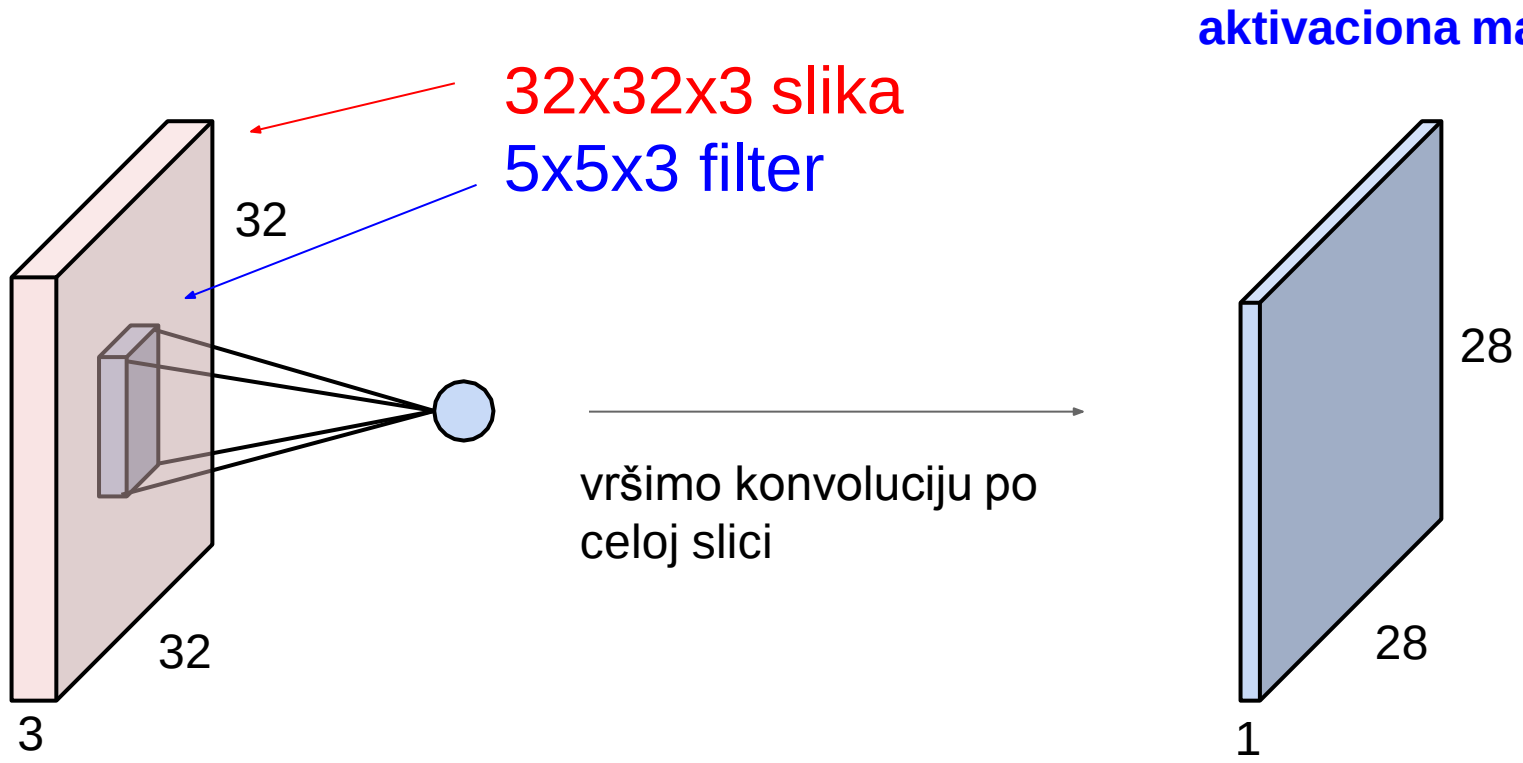
32x32x3 slika

5x5x3 filter w

1 broj (vrednost):
rezultat skalarnog proizvoda između filtera i malog 5x5x3 dela slike
(tj. 5*5*3 = 75-dim. skalarni proizvod + bijas)

$$w^T x + b$$

Konvolutivni Sloj

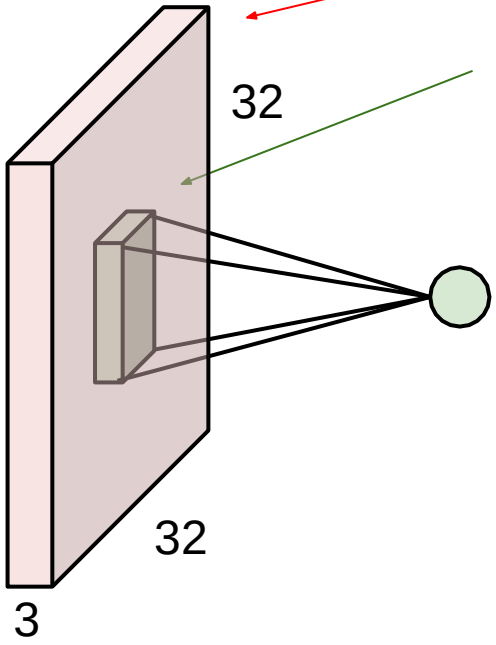


Konvolutivni Sloj

recimo da imamo, zeleni filter

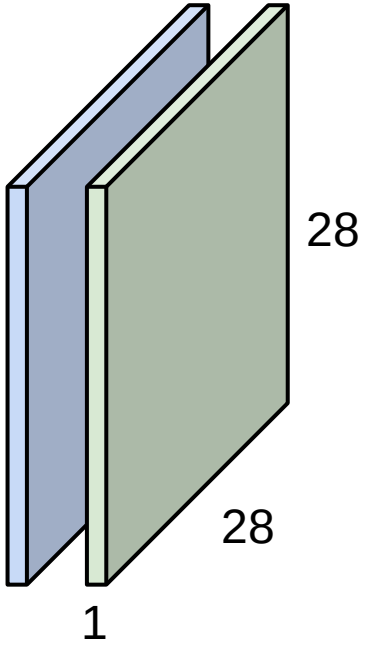
32x32x3 slika

5x5x3 filter

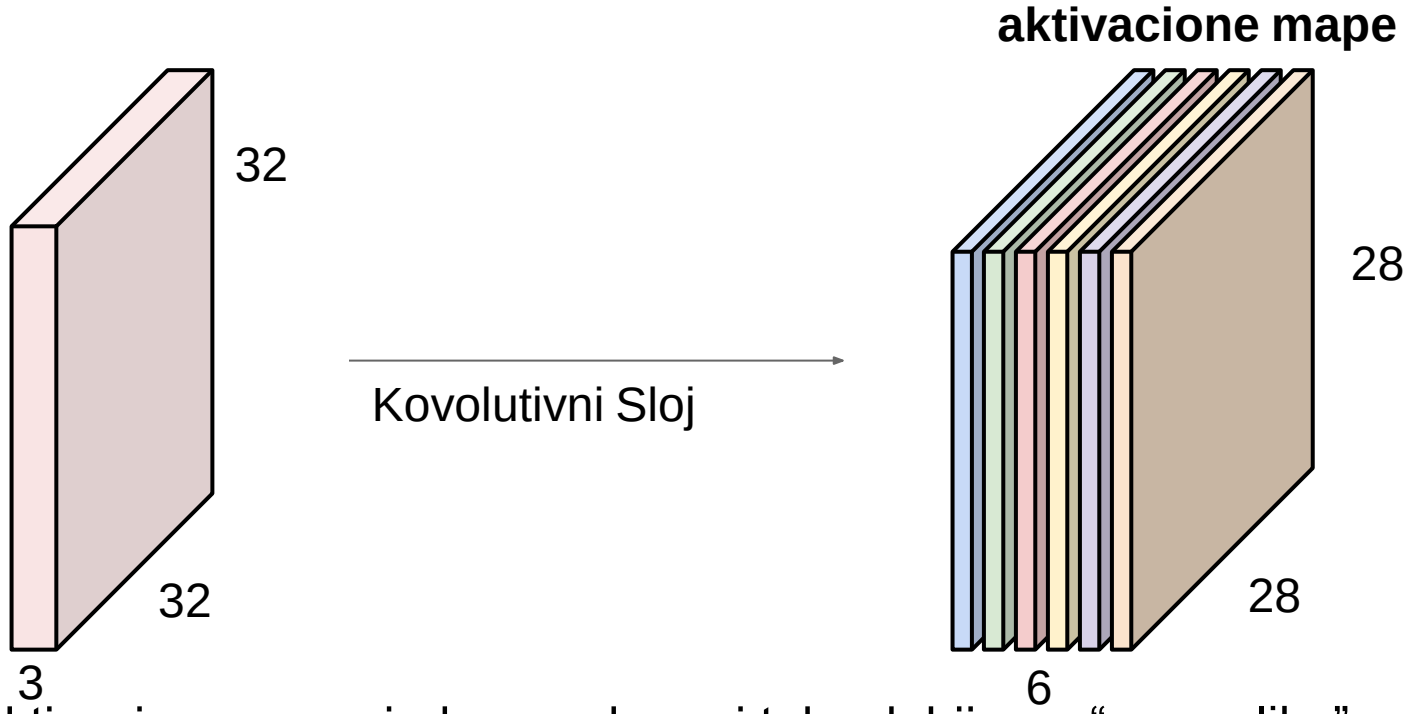


vršimo konvoluciju po celoj slici

aktivacione mape



Na primer, ako imamo 6 5x5 filtera, dobićemo 6 odvojenih aktivacionih mapa:

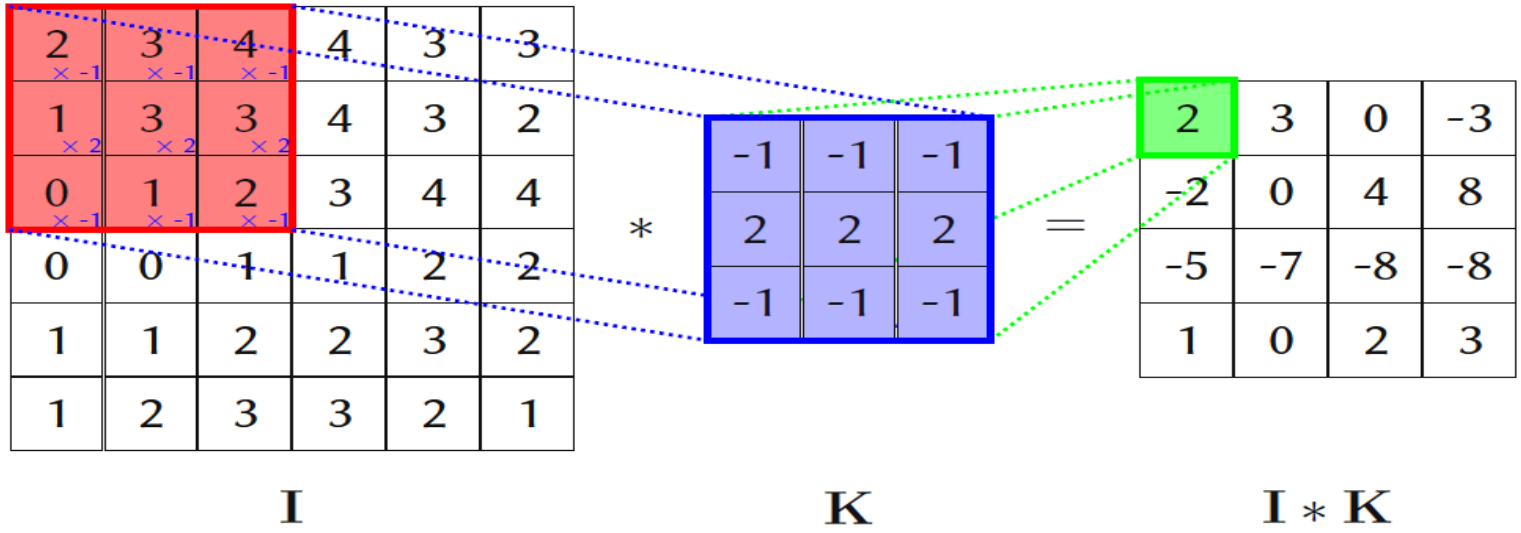


Slažemo aktivacione mape jednu na drugu i tako dobijamo “novu sliku” veličine 28x28x6!

Primer konvolucije

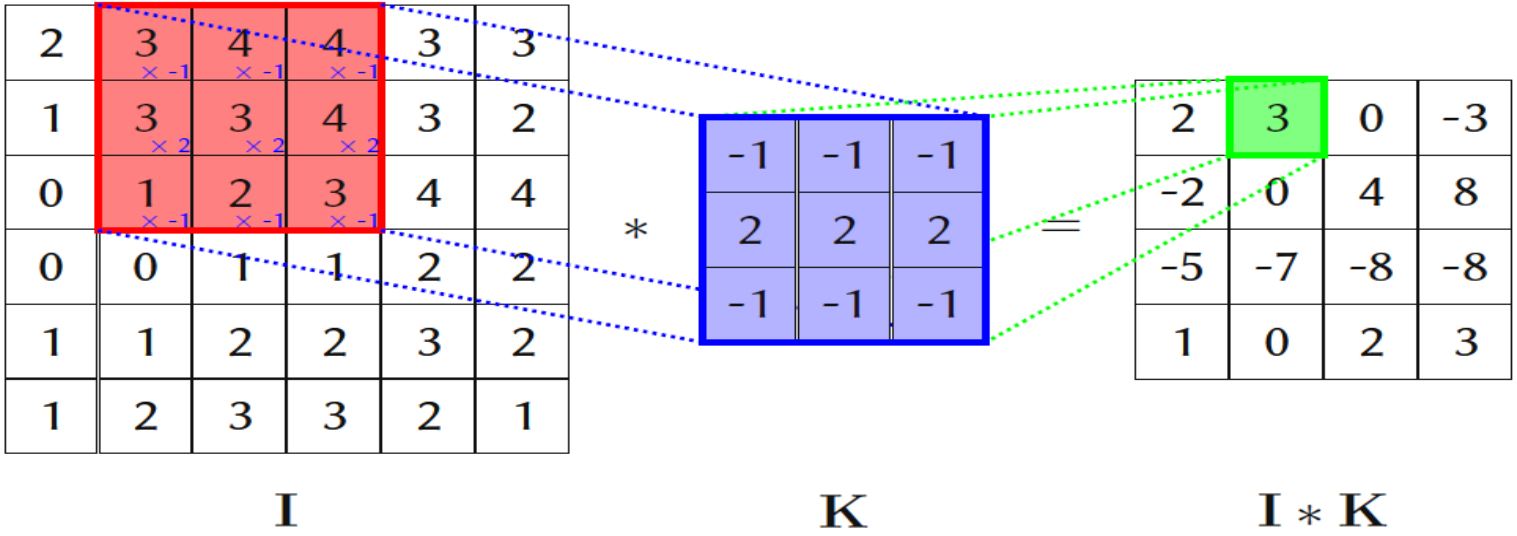
Narednih nekoliko slajdova zasnovano je na
http://www.csnedelja.mg.edu.rs/static/resources/v3.0/sre1_neuralne_pv.pdf

Primer konvolucije

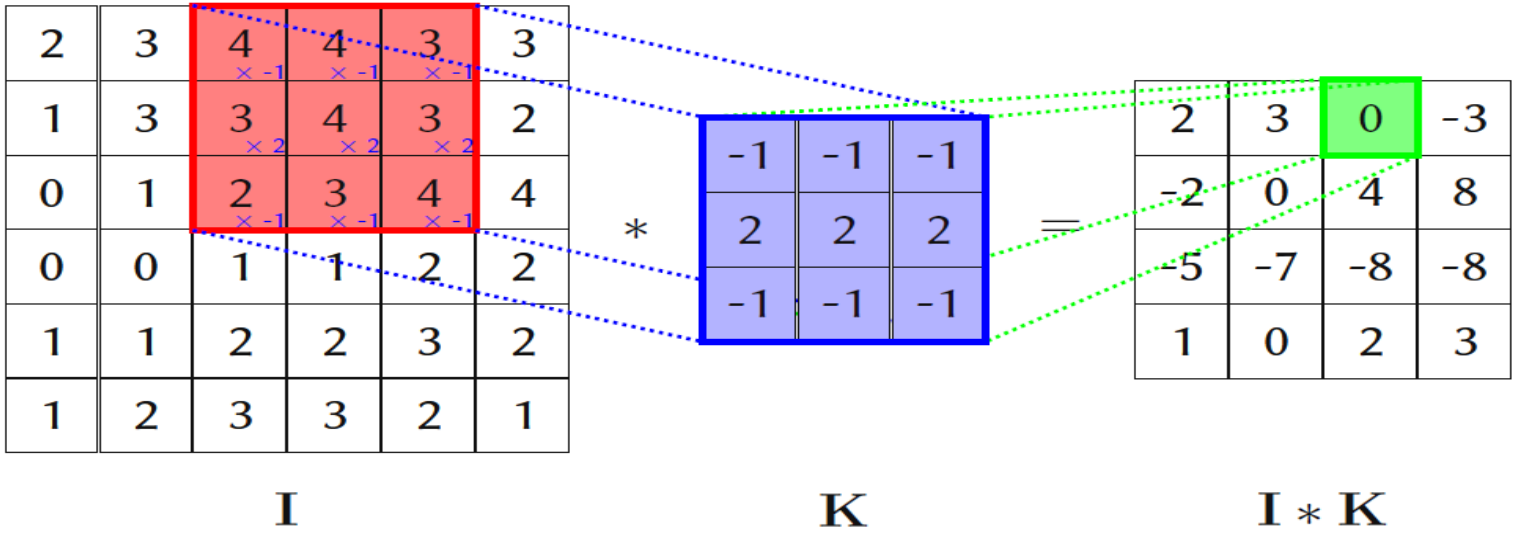


Napomena: ovaj konretan primer sadrži bijas, ali je dovoljan za razumevanje način na koji funkcionišu filteri kod konvolucija.

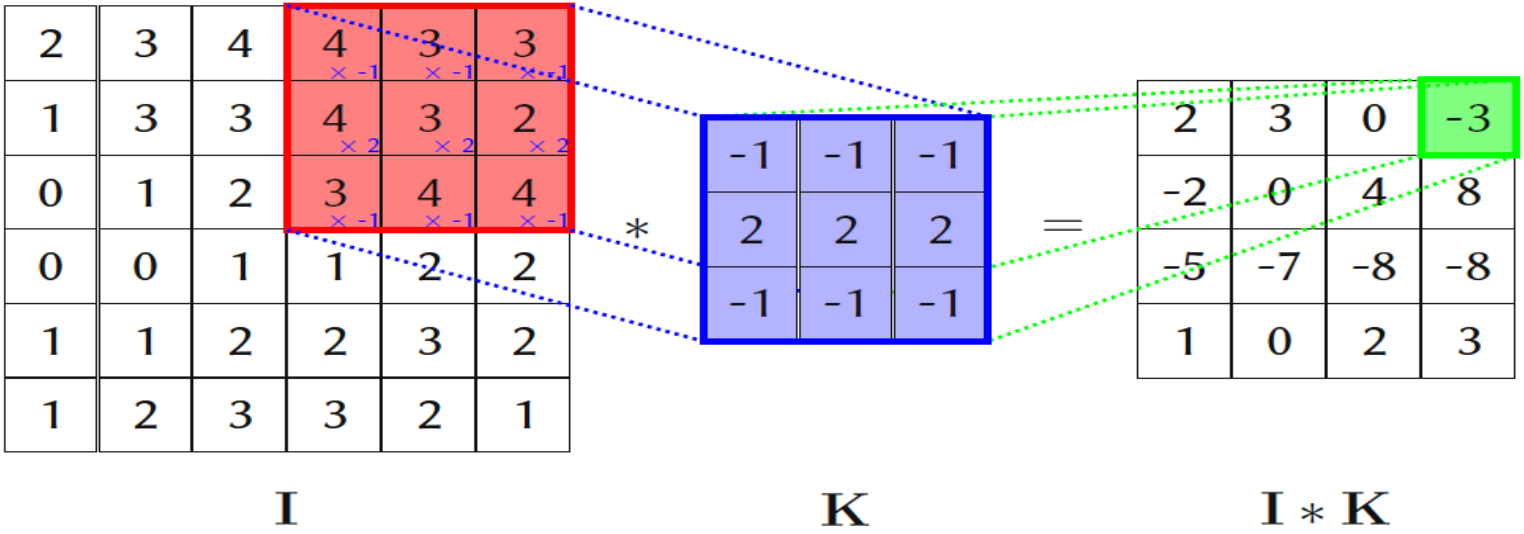
Primer konvolucije



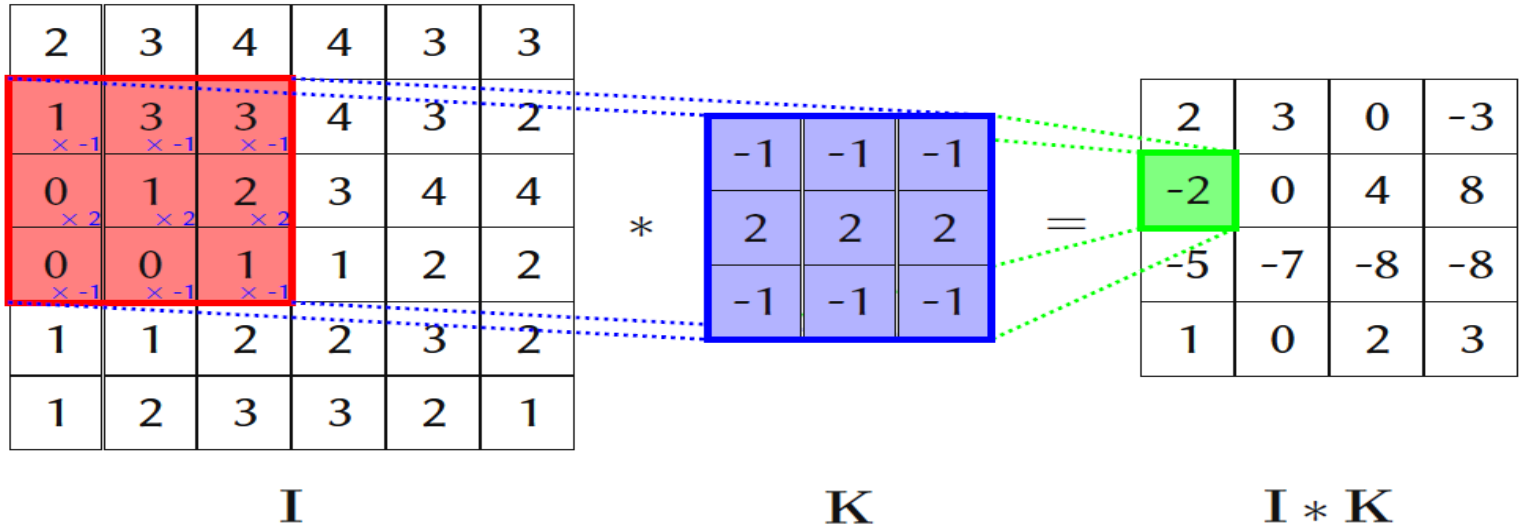
Primer konvolucije



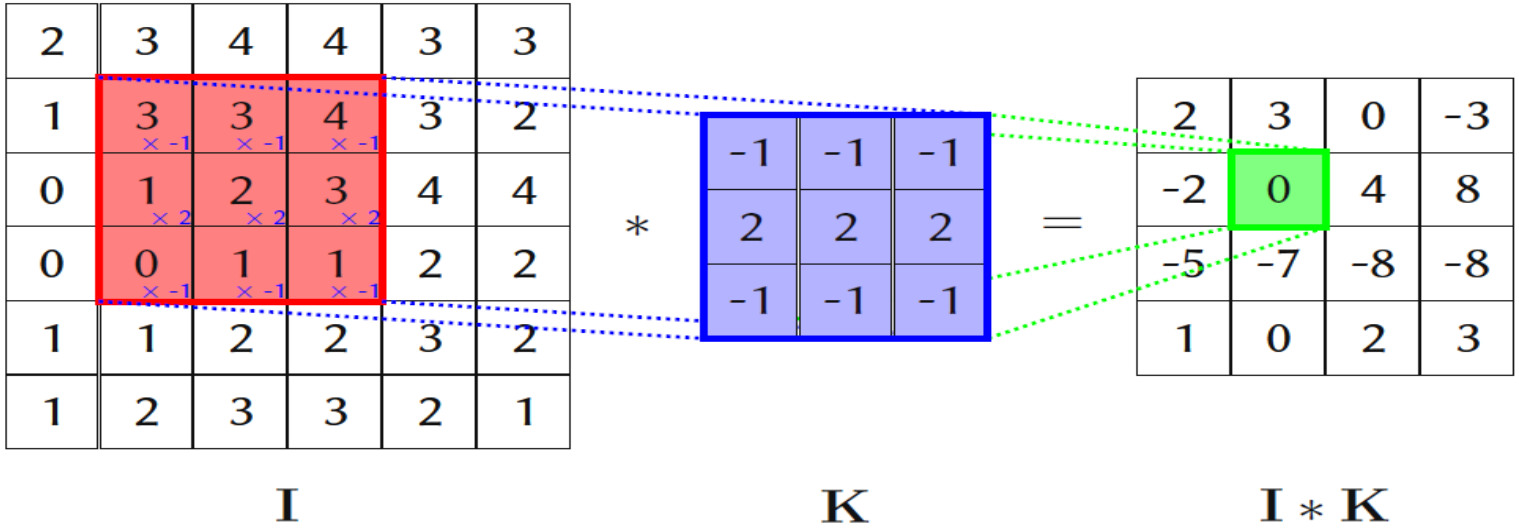
Primer konvolucije



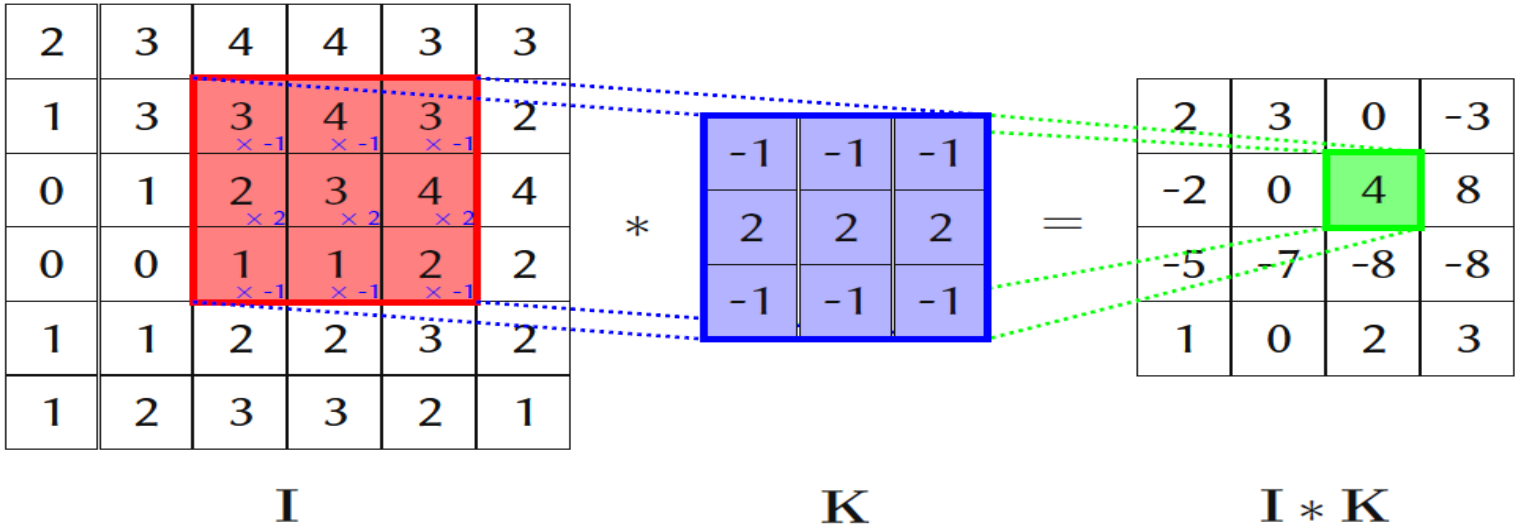
Primer konvolucije



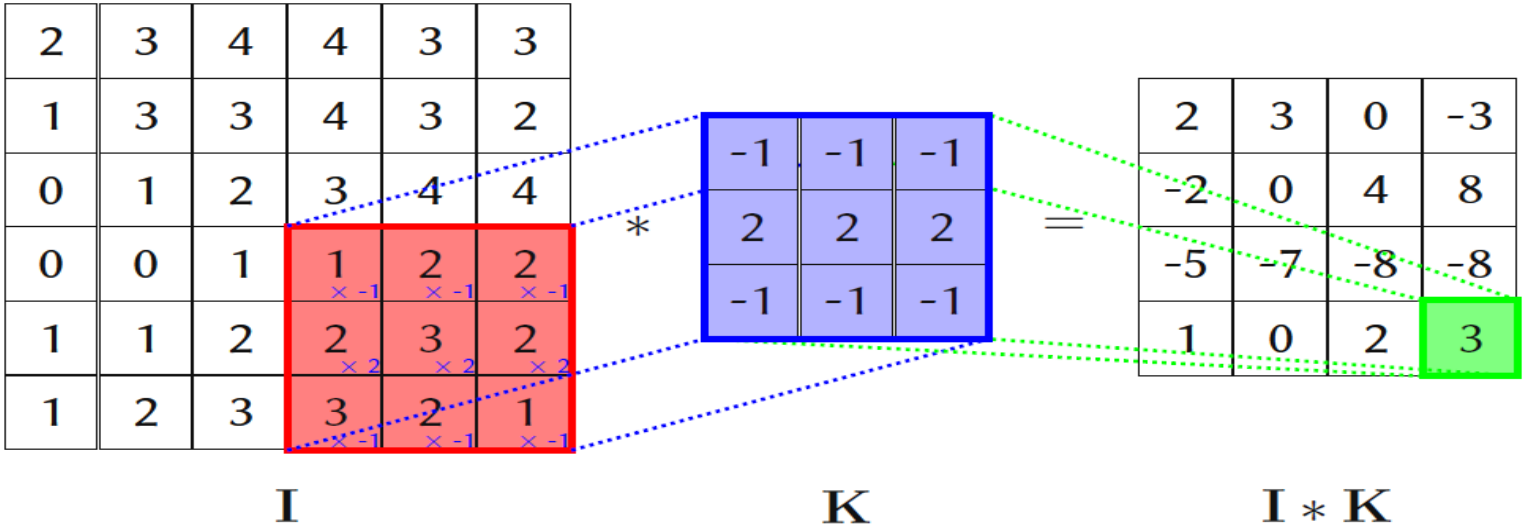
Primer konvolucije



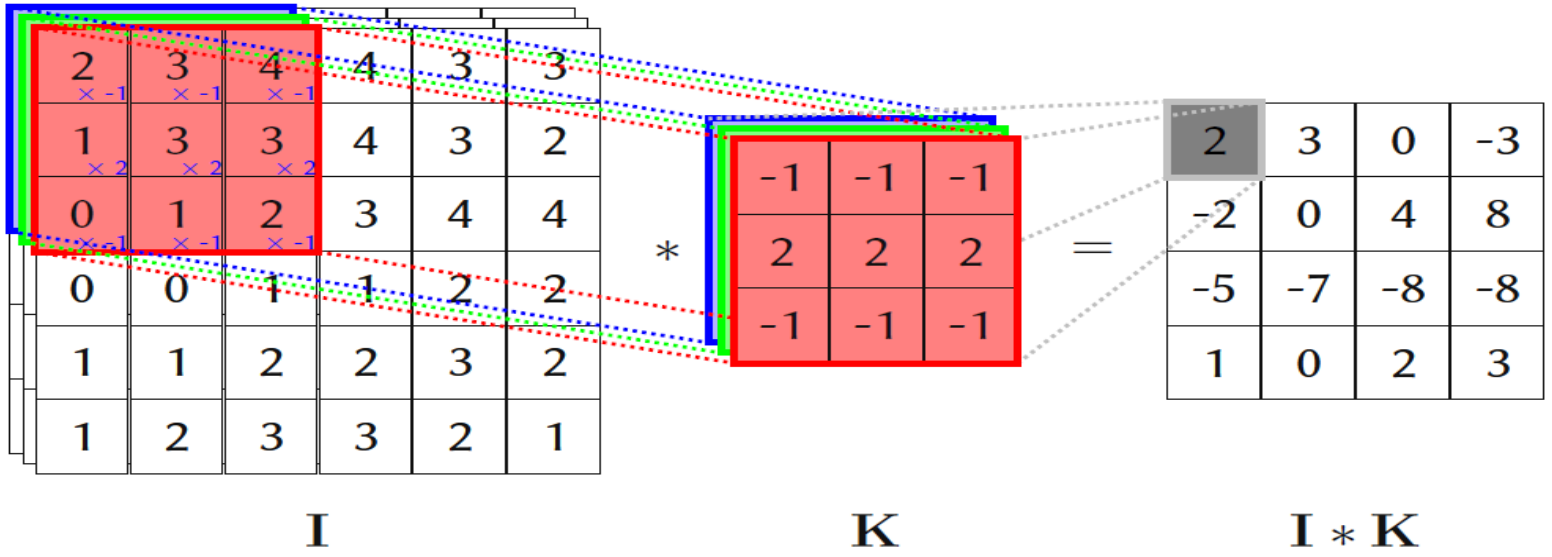
Primer konvolucije

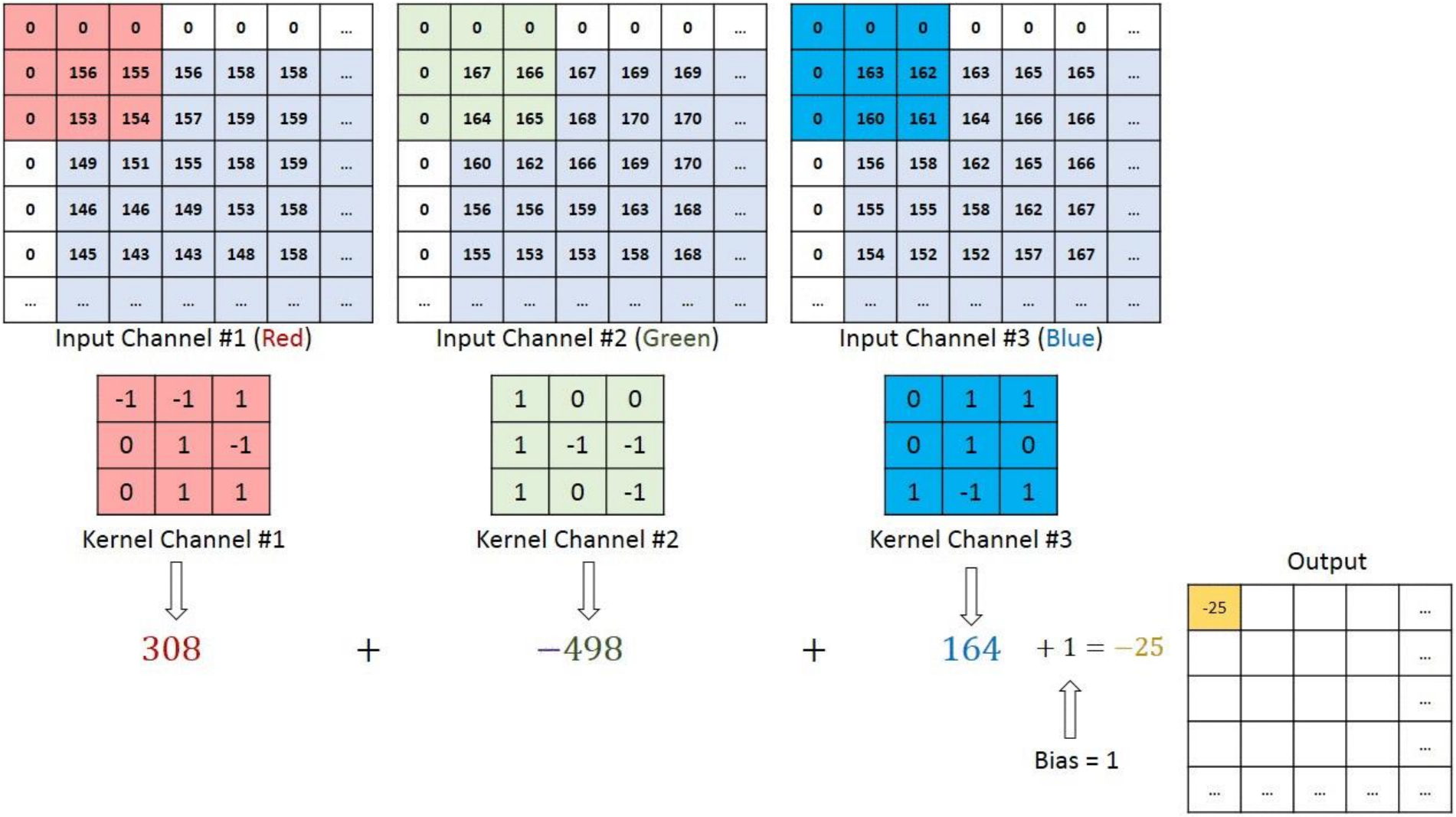


Primer konvolucije



Konvolucija sa *bojom/kanalima*




0 0 0 0 0 0 ...
0 156 155 156 158 158 ...
0 153 154 157 159 159 ...
0 149 151 155 158 159 ...
0 146 146 149 153 158 ...
0 145 143 143 148 158 ...
... ...
Input Channel #1 (Red)
0 0 0 0 0 0 ...
0 167 166 167 169 169 ...
0 164 165 168 170 170 ...
0 160 162 166 169 170 ...
0 156 156 159 163 168 ...
0 155 153 153 158 168 ...
... ...
Input Channel #2 (Green)
0 0 0 0 0 0 ...
0 163 162 163 165 165 ...
0 160 161 164 166 166 ...
0 156 158 162 165 166 ...
0 155 155 158 162 167 ...
0 154 152 152 157 167 ...
... ...
Input Channel #3 (Blue)
-1 -1 1
0 1 -1
0 1 1
Kernel Channel #1
1 0 0
1 -1 -1
1 0 -1
Kernel Channel #2
0 1 1
0 1 0
1 -1 1
Kernel Channel #3
308
+
−498
+
164
+ 1 = −25
Bias = 1
Output
-25 ... ...
... ...
... ...
... ...
... ...

Konvolucija

Šta je ustvari konvolucija? To je filter koji nam služi za ekstrakciju karakteristika slike.

Konvolucija

Na primer na ovoj slici primenom kovolucije dobijamo ivice.

Konvolucija

Zajednica ljudi koji se bave Kompjuterskom Vizijom je ranije morala ručno da razvija konvolucije

Konvolucija

Konvolutivne mreže same uče konvolucije! To je poenta Deep Learning!

Primer konvolucije sa za detekciju izvica

Input image

Convolution Kernel

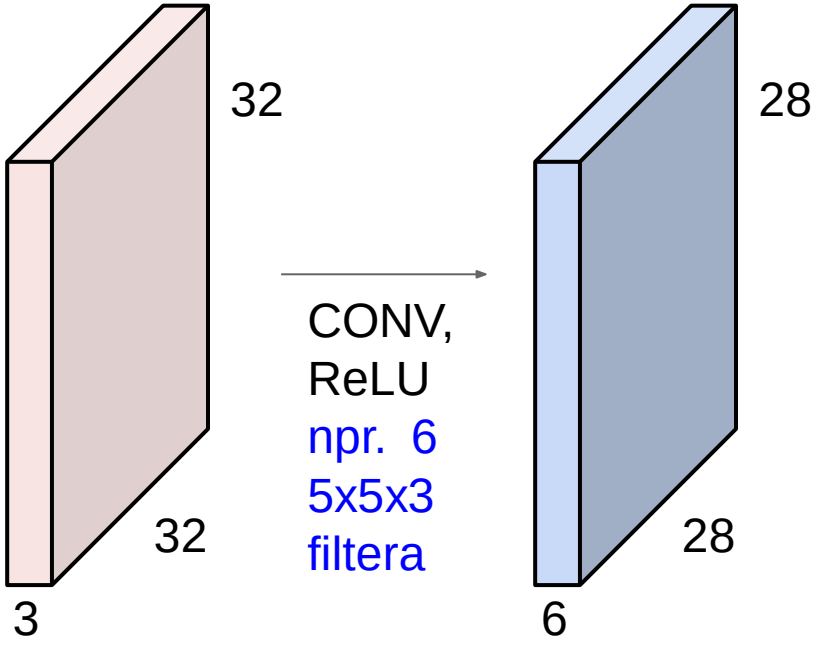
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Feature map

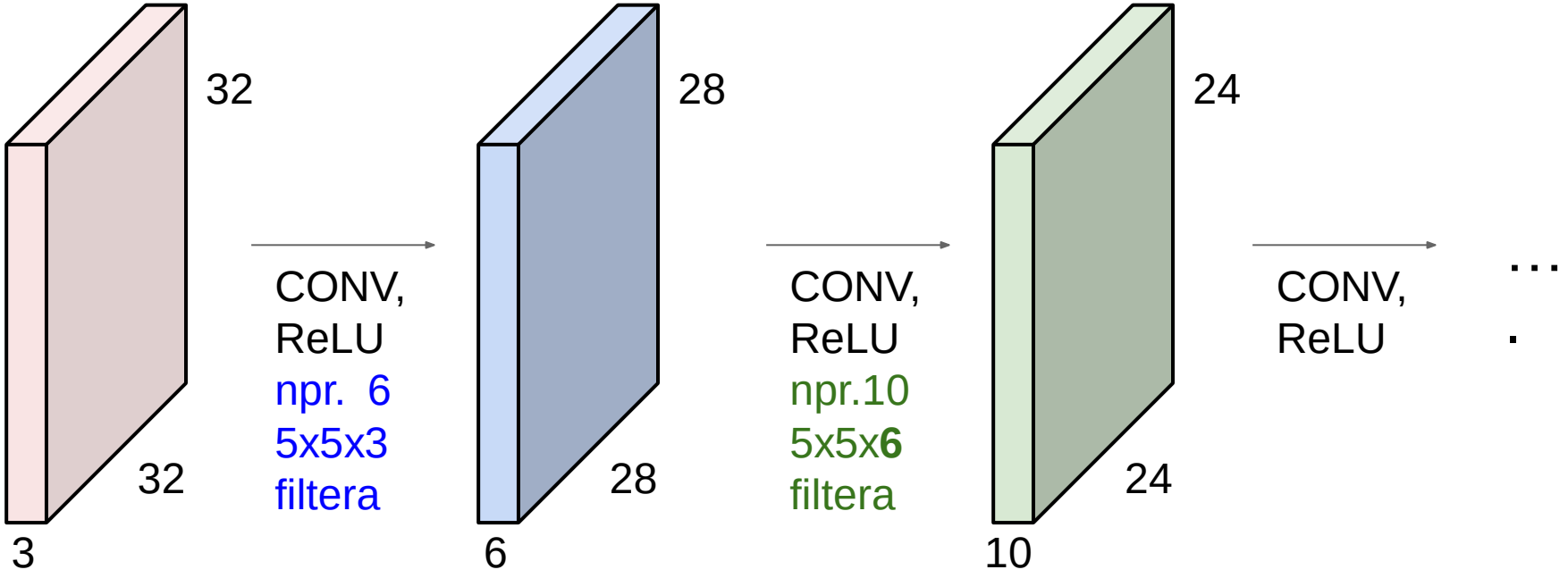
Vrednosti matrice je ručno kreirao ekspert.
Dok ih kovolutivna mreža sama uči!

slika preuzeta sa https://pgaleone.eu/neural-networks/2016/11/24/convolutional-autoencoders/

Konvolutivna mreža je niz Konvolutivnih Slojeva, između kojih su aktivacione fukcije

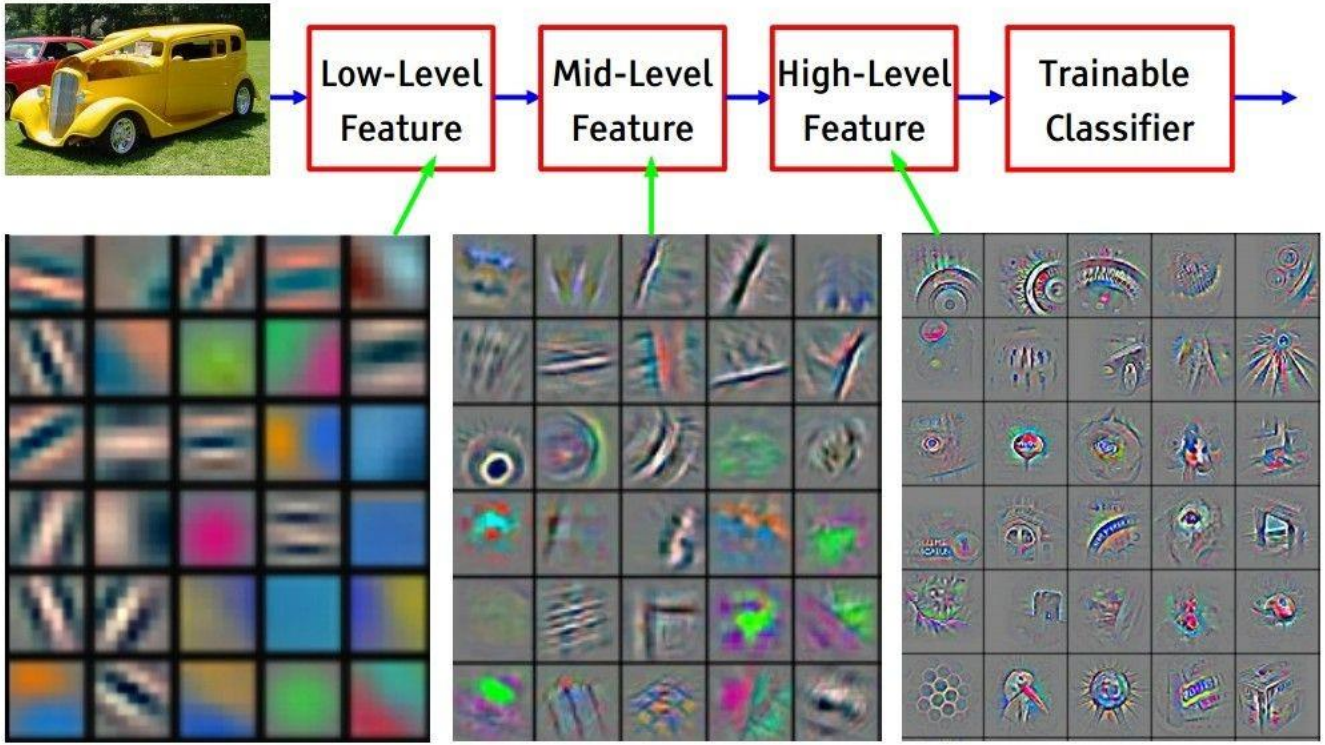


Konvolutivna mreža je niz Konvolutivnih Slojeva, između kojih su aktivacione fukcije



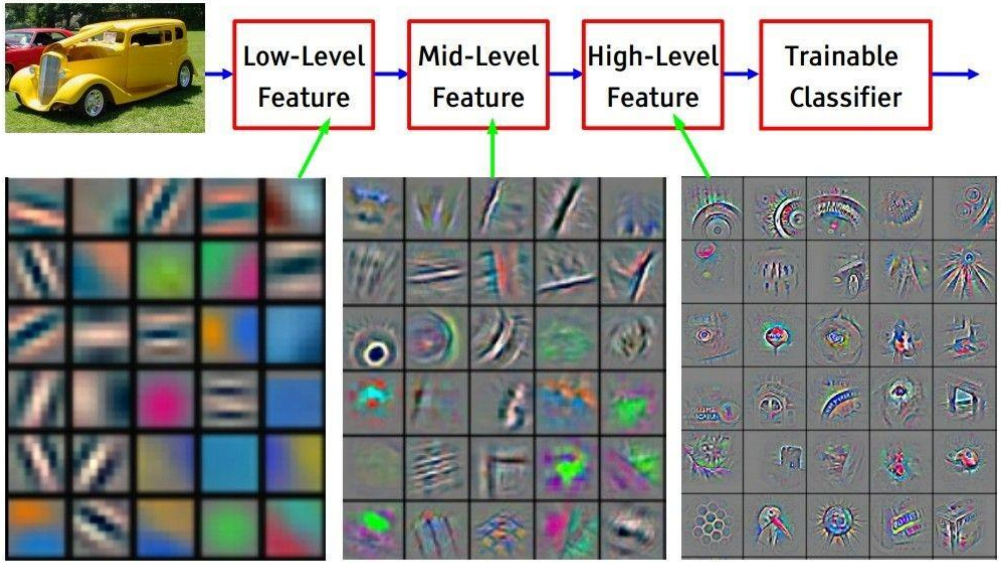
Kako izgledaju slojevi

[From recent Yann LeCun slides]



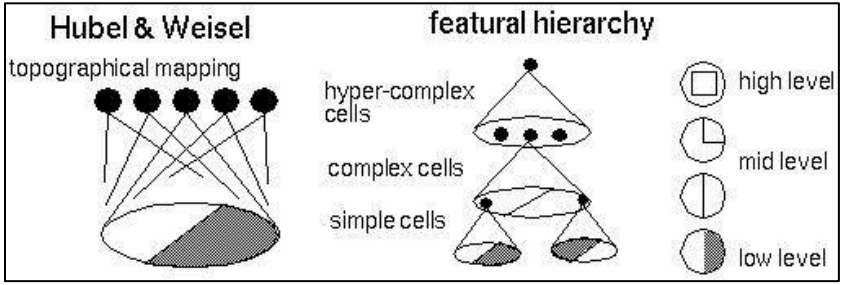
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

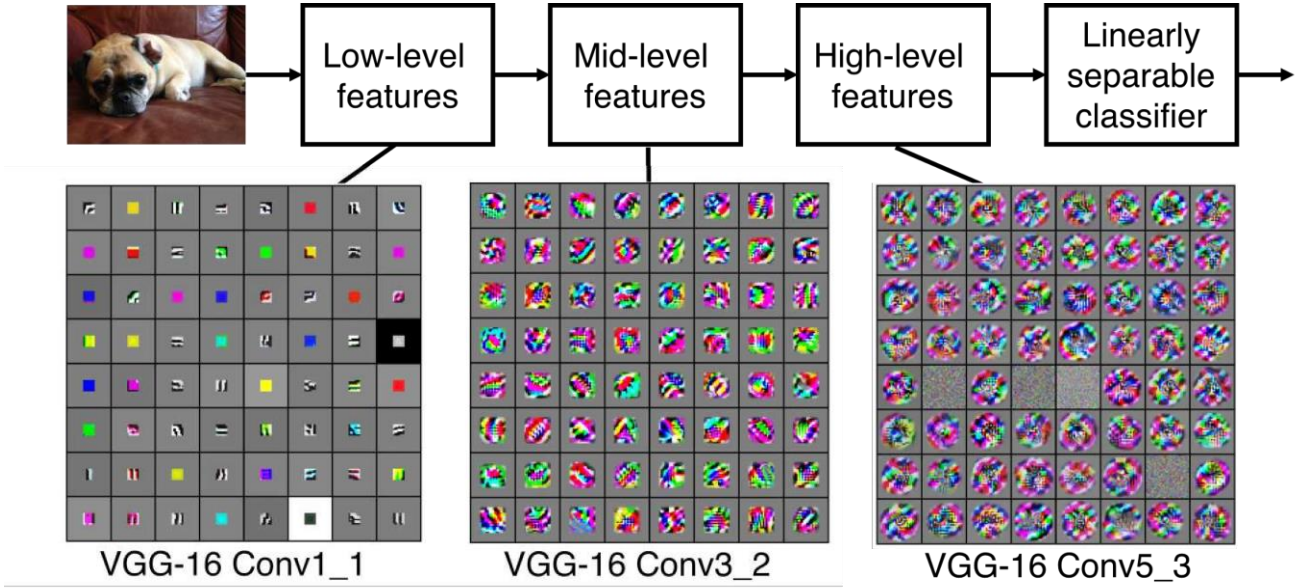
Kako izgledaju slojevi

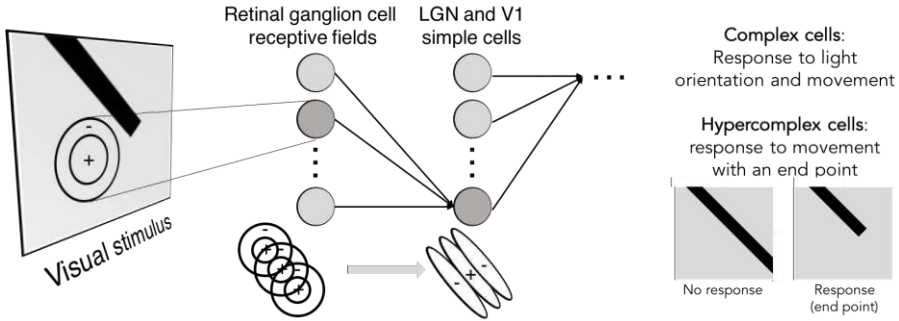


[From recent Yann LeCun slides]

Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

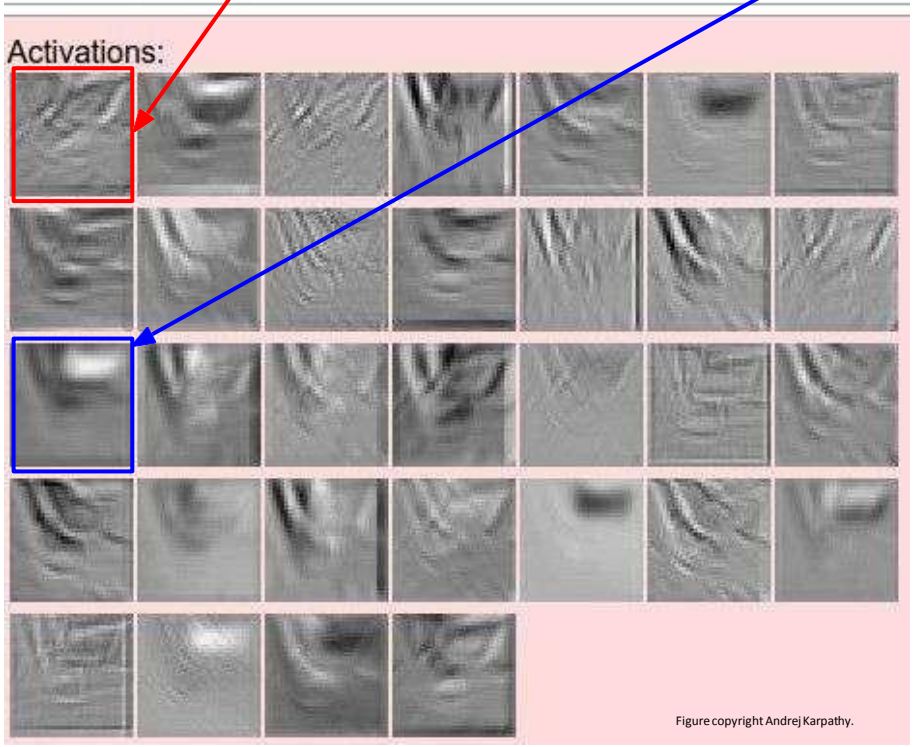



Low-level features
Mid-level features
High-level features
Linearly separable classifier
VGG-16 Conv1_1
VGG-16 Conv3_2
VGG-16 Conv5_3

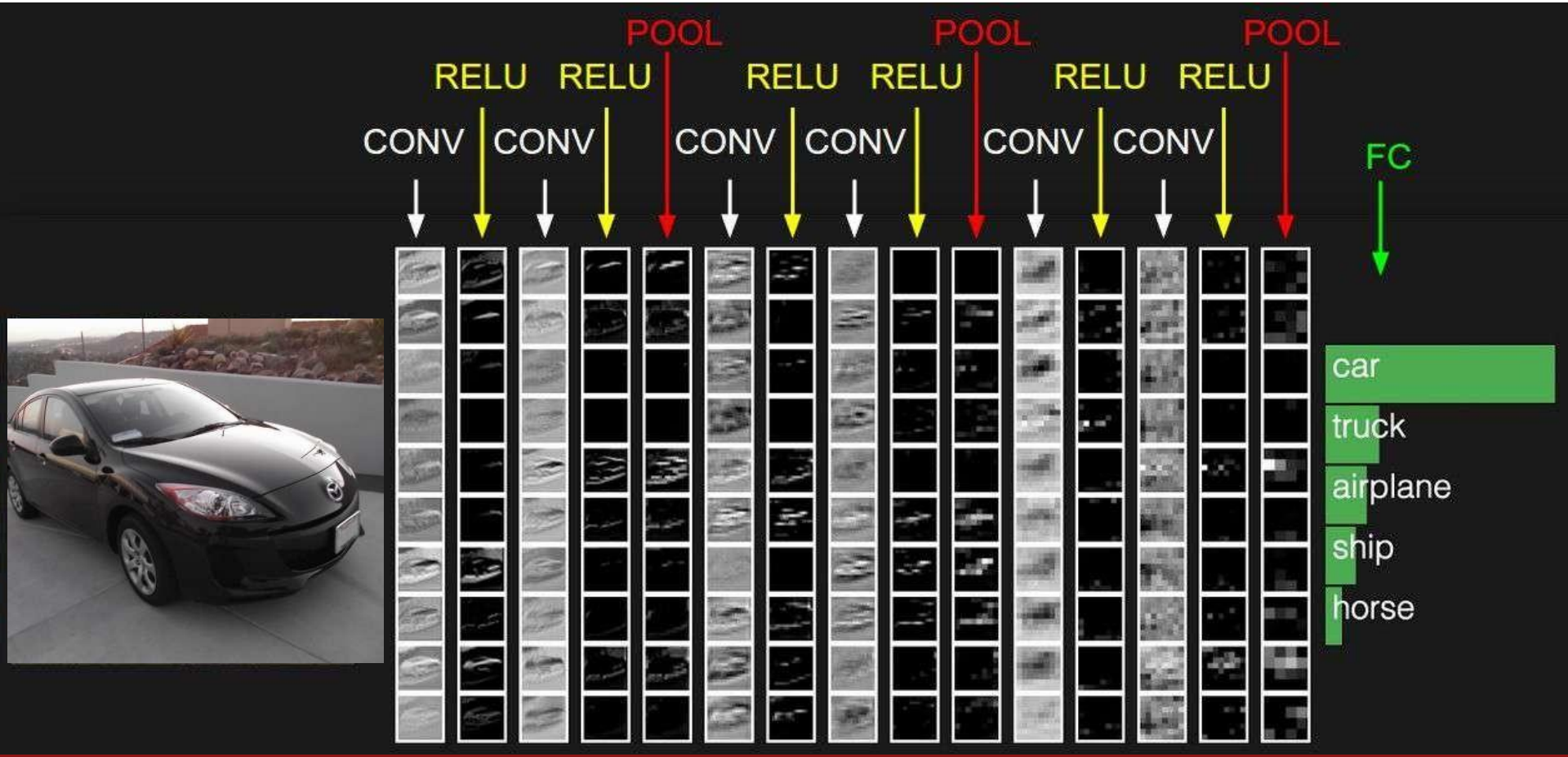

Retinal ganglion cell receptive fields
LGN and V1 simple cells
Visual stimulus
Complex cells:
Response to light orientation and movement
Hypercomplex cells:
response to movement with an end point
No response
Response (end point)

jedan filter =>
jedna aktivaciona mapa

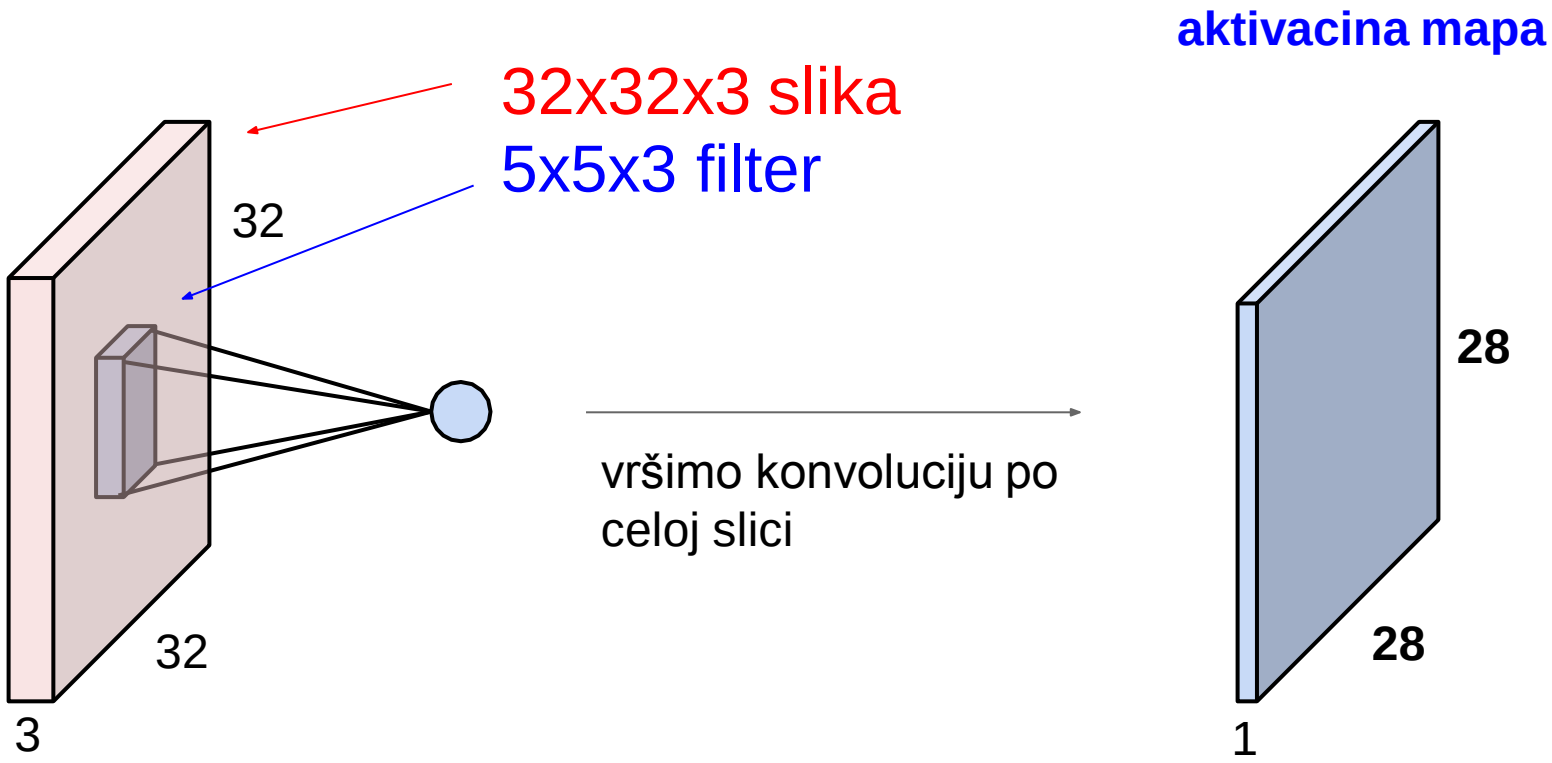
primer 5x5 filteri
(32 ukupno)



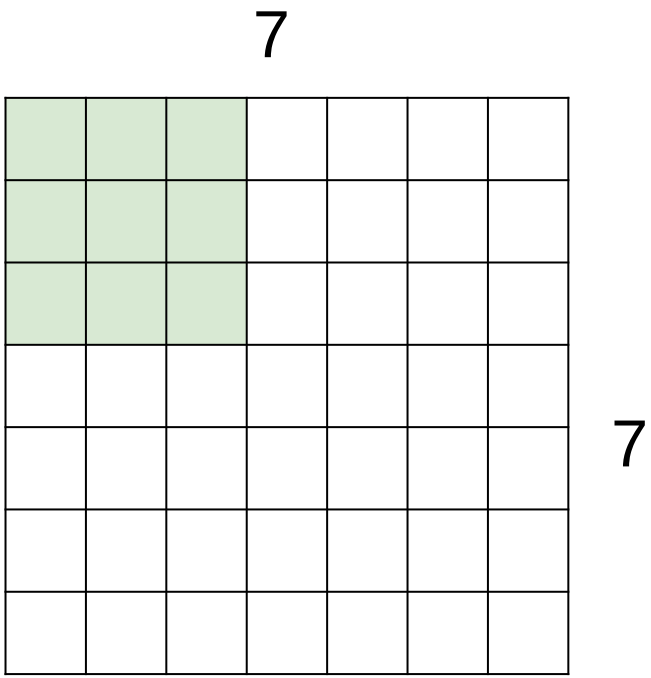
Kako tipično izgleda Konvolutivna Mreža



Detaljnije o dimenzijama:

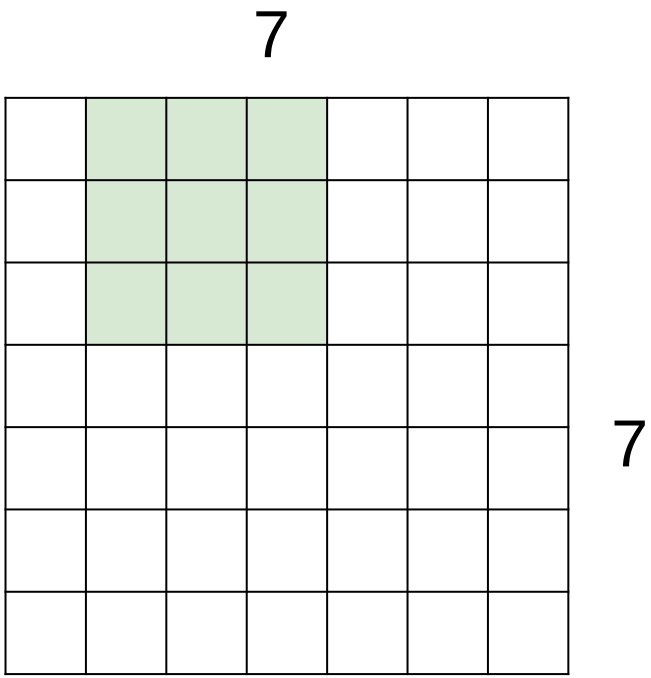


Detaljnije o dimenzijama:



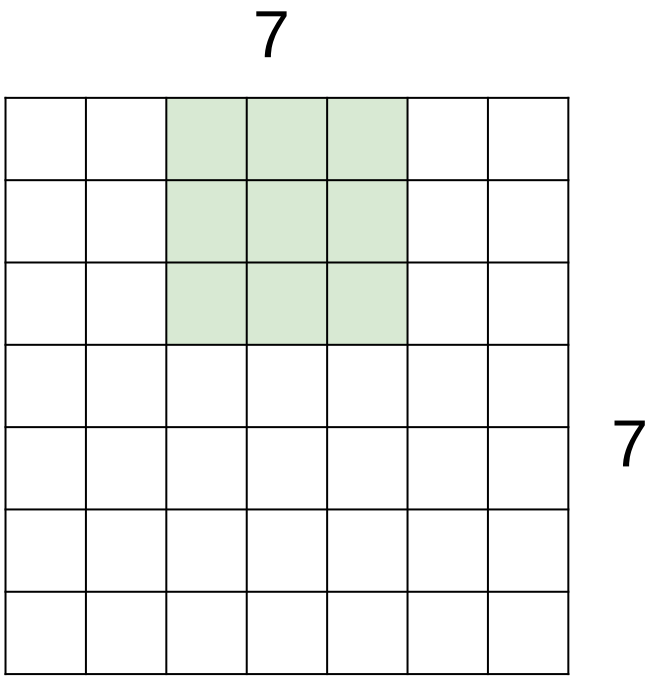
7x7 ulaz
3x3 filter

Detaljnije o dimenzijama:



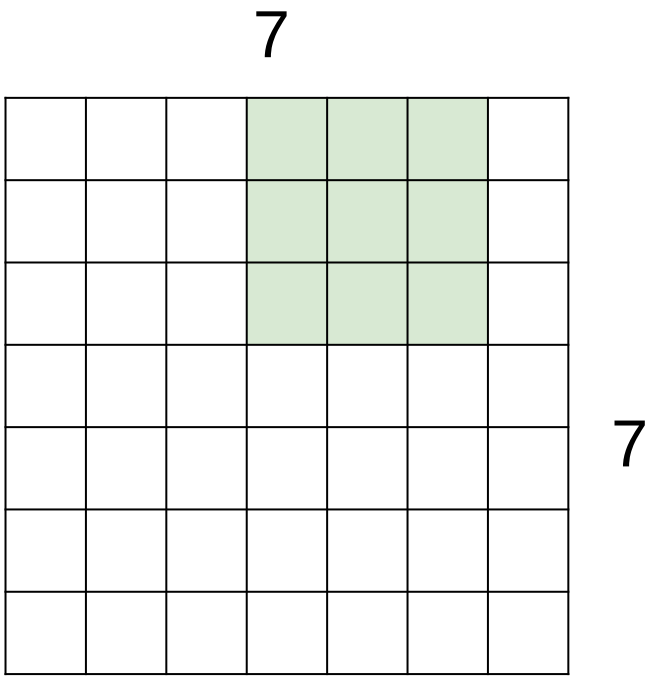
7x7 ulaz
3x3 filter

Detaljnije o dimenzijama:



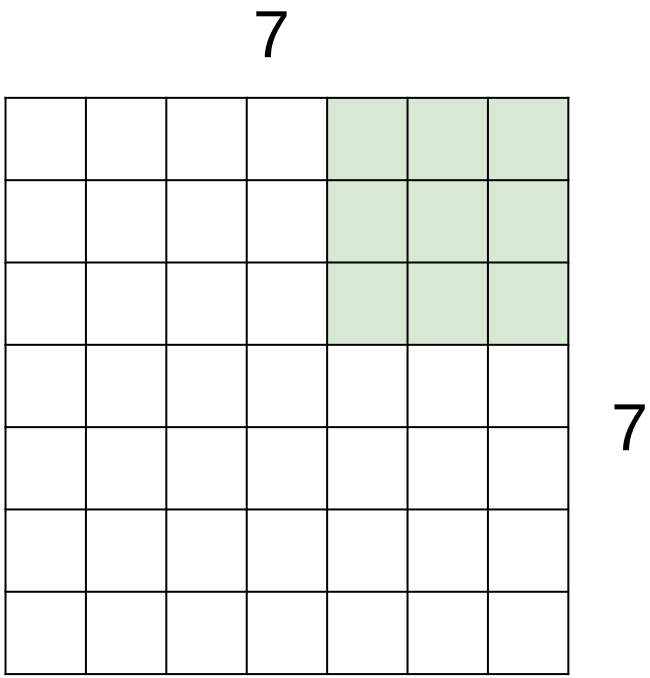
7x7 ulaz
3x3 filter

Detaljnije o dimenzijama:



7x7 ulaz
3x3 filter

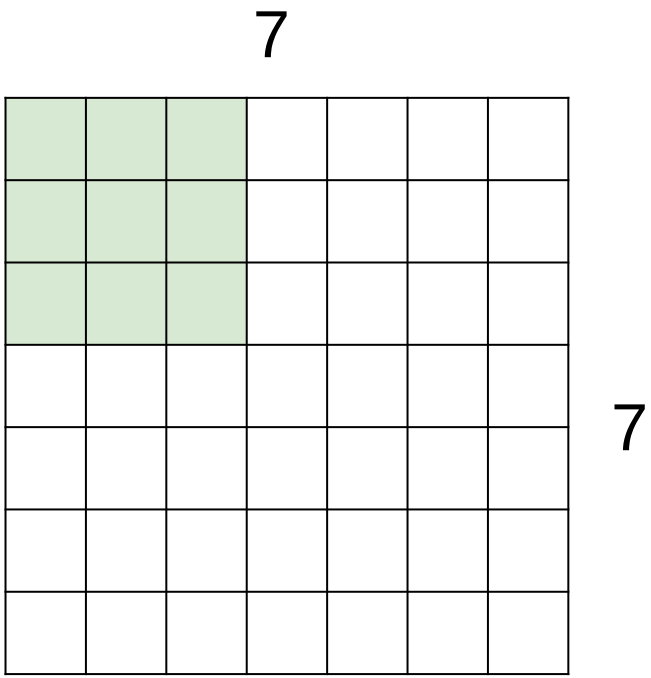
Detaljnije o dimenzijama:



7x7 ulaz
3x3 filter

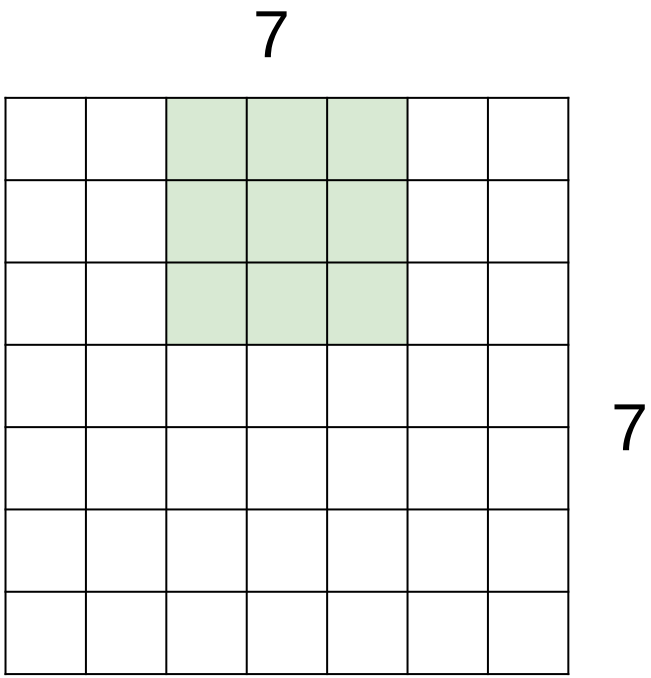
=> 5x5 izlaz

Detaljnije o dimenzijama:



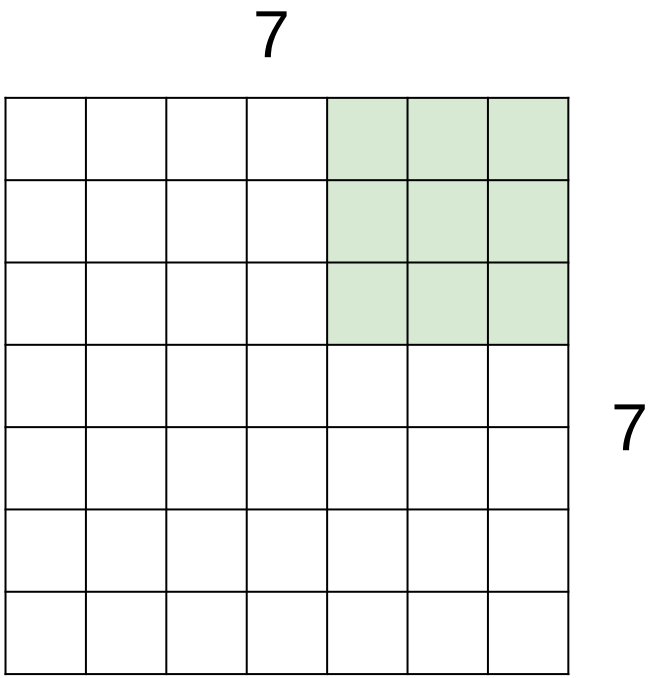
7x7 ulza
3x3 filter
primenjen sa **korakom (*stride*) 2**

Detaljnije o dimenzijama:



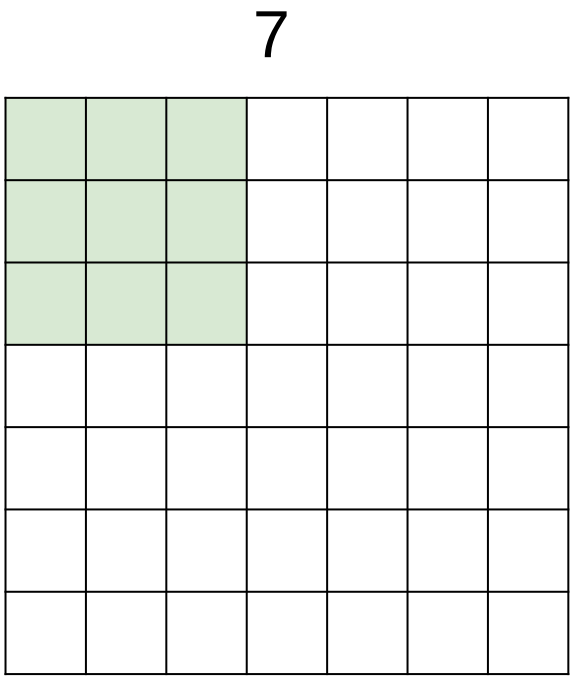
7x7 ulza
3x3 filter
primenjen sa **korakom (*stride*) 2**

Detaljnije o dimenzijama:



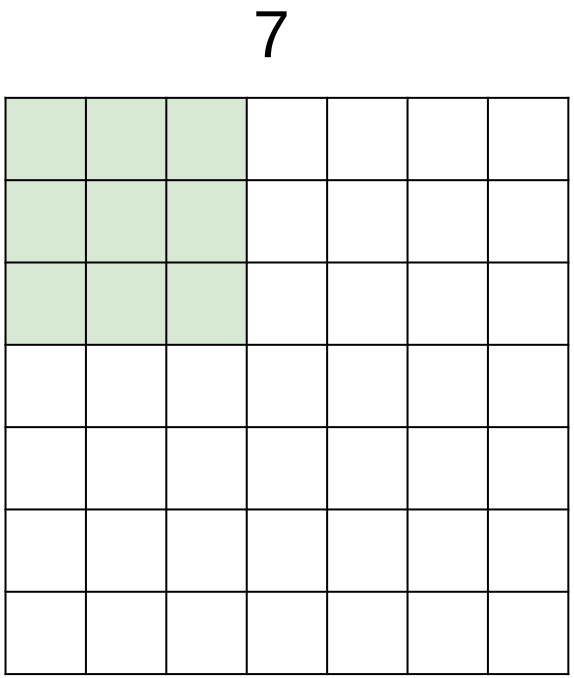
7x7 ulza
3x3 filter
primenjen sa **korakom (*stride*) 2**
=> 3x3 izlaz!

Detaljnije o dimenzijama:



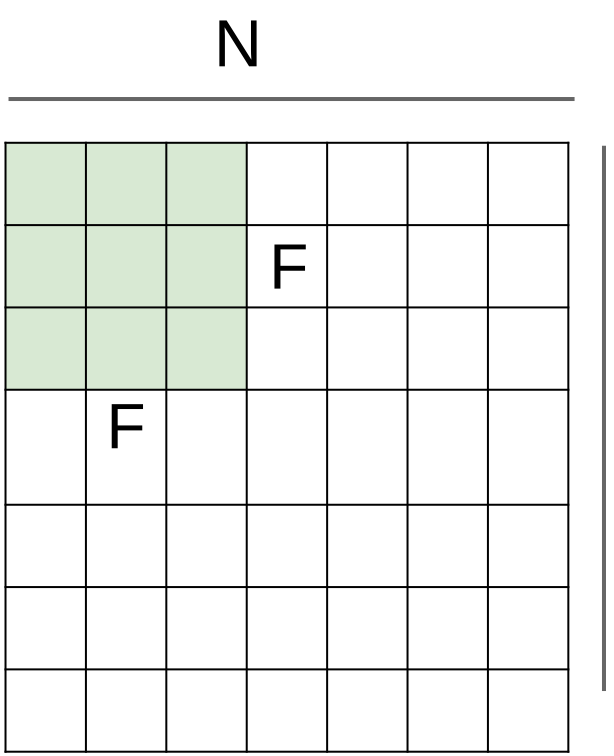
7x7 ulaz
3x3 filter
primenjen sa **korakom (*stride*) 3**
Da li je primena moguća?

Detaljnije o dimenzijama:



7x7 ulaz
3x3 filter
primenjen sa **korakom (*stride*) 3**
Da li je primena moguća?

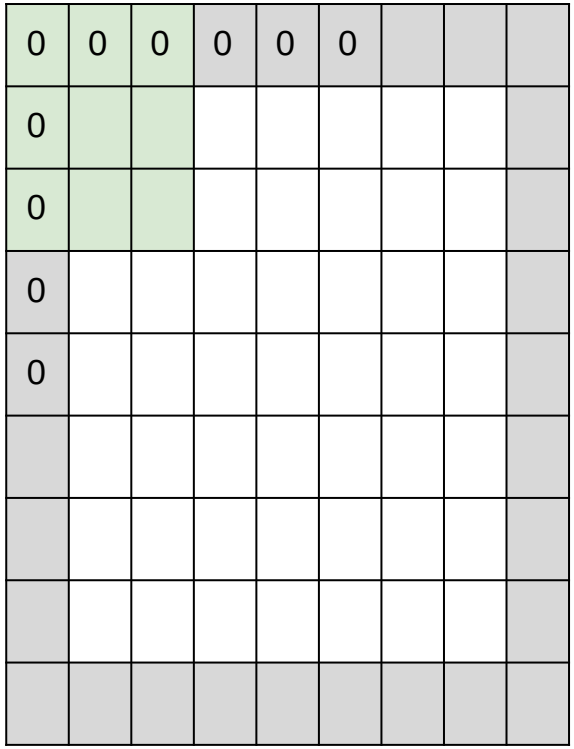
Ne, filter se ne uklapa!
ne možemo da primenimo 3x3 filter na 7x7 ulaz sa korakom 3.



Dimenzionalnost izlaza:
((N - F) / korak) + 1

npr. N = 7, F = 3:
korak 1 => (7 - 3)/1 + 1 = 5
korak 2 => (7 - 3)/2 + 1 = 3
korak 3 => (7 - 3)/3 + 1 = 2.33
– ne dobijamo ceo broj

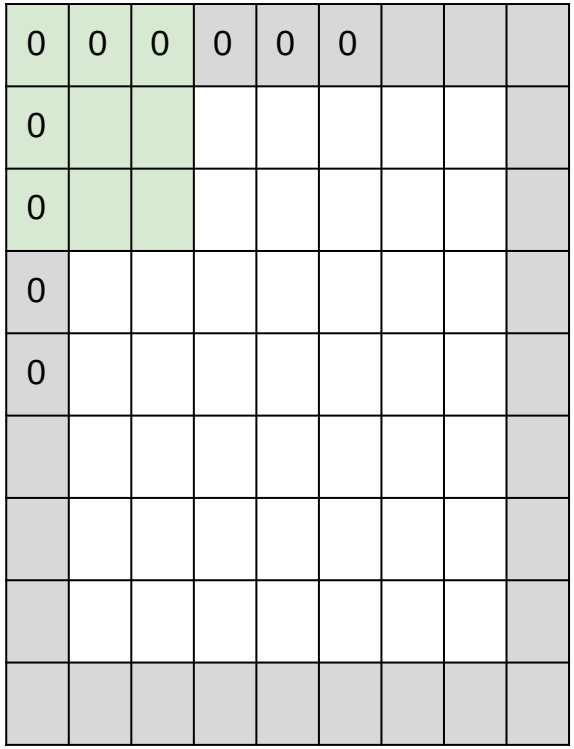
U praksi: Dodajemo nule u okvir slike



npr. ulaz 7x7
3x3 filter, primenjen sa **korakom 1**
dodajemo okvir od 1 piksela => koje je su dimenzije izlaza?

(podsetite se:)
(N - F) / korak + 1

U praksi: Dodajemo nule u okvir slike

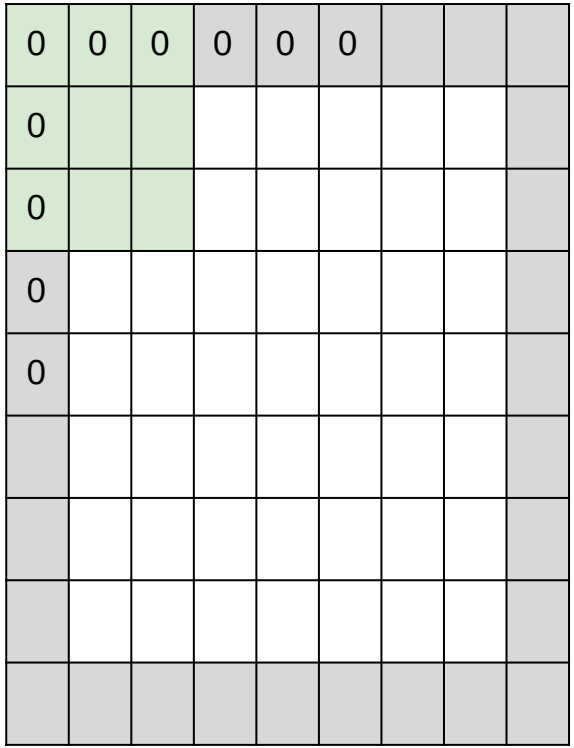


npr. ulaz 7x7
3x3 filter, primenjen sa **korakom 1**
dodajemo okvir od 1 piksela => koje je su dimenzije izlaza?

Izlaz je 7X7! (7+2-3)/1 + 1=7

podsetite se:
(N - F) / korak + 1

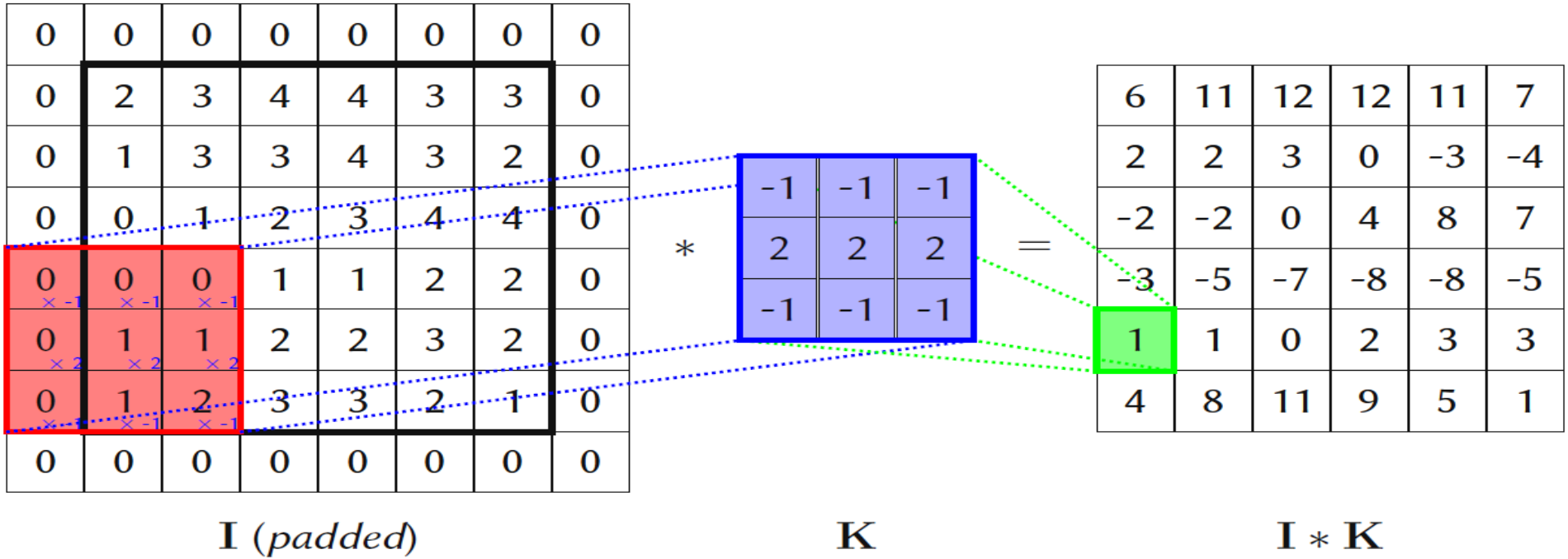
U praksi: Dodajemo nule u okvir slike



npr. ulaz 7x7
3x3 filter, primenjen sa **korakom 1**
dodajemo okvir od 1 piksela => koje je su dimenzije izlaza?

Izlaz je 7X7! (7+2-3)/1 + 1=7
tipično se koriste CONV slojevi sa korakom 1, filteri veličine FxF, dodavanje 0 u okvir dimenzije (F-1)/2.
npr.
F = 3 => dodajemo okvir dim. 1
F = 5 => dodajemo okvir dim. 2
F = 7 => dodajemo okvir dim. 3

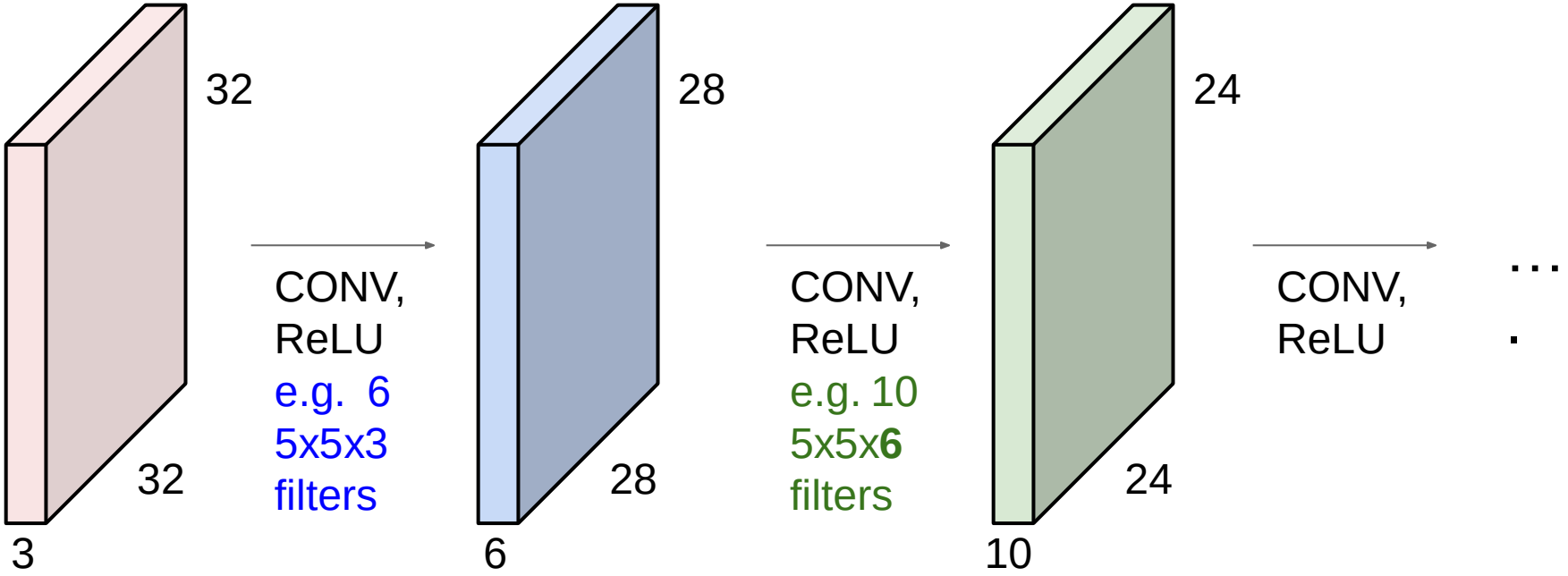
Primer konvolucije (sa *padding*-om)



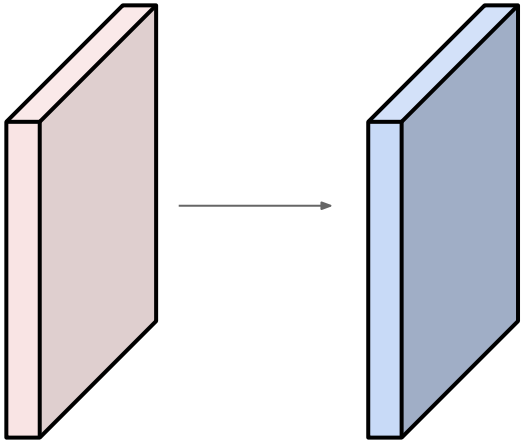
I (*padded*) **K** **I** ∗ **K**

Prisetimo se…

Npr. ulaz 32x32 na kome se u više slojeva primenjuje filter 5x5 se smanjuje po veličini! (32 -> 28 -> 24 ...). Prebrzo smanjivanje nije dobro, gubimo na informacijama, dobijamo loše rezultate. Zato koristimo padding.



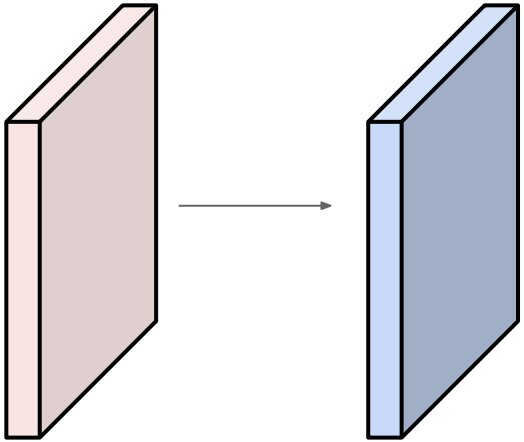
Primer:



Ulaz: **32x32x3**
10 5x5x3 filtera sa korakom 1,
i dodavanjem okvira 0 dim. 2

Koja je veličina izlaza: ?

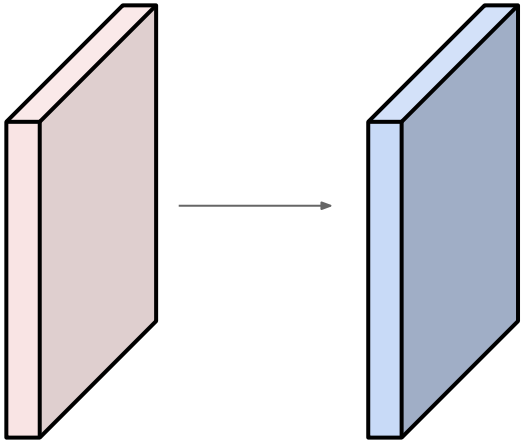
Primer:



Ulaz: **32x32x3**
10 5x5x3 filtera sa korakom 1,
i dodavanjem okvira 0 dim. 2

Veličina izlaza:
(32+2*2-5)/1+1 = 32 što daje,
32x32x10

Primer:



Ulaz: **32x32x3**
10 5x5x3 filter sa korakom 1,
i dodavanjem okvira 0 dim. 2

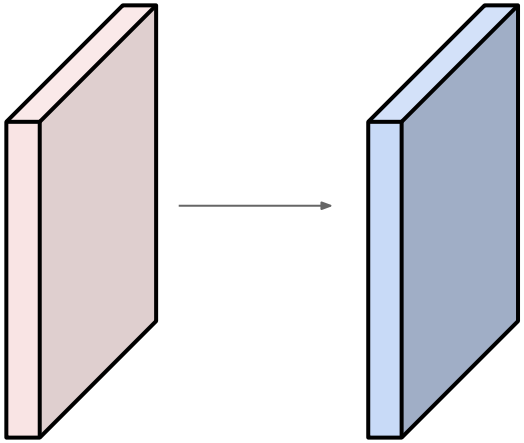
Koji je broj parametara (težina i bijasa)
za obučavanje u ovom sloju?

Primer:

Ulaz: **32x32x3**

10 5x5x3 filter sa korakom 1,

i dodavanjem okvira 0 dim. 2



Koji je broj parametara (težina i bijasa) za obučavanje u ovom sloju?

svaki filter ima 5*5*3 + 1 = 76 parametara (+1 za bijas)

=> 76*10 = **760**

Summary. To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- In the output volume, the d-th depth slice (of size $W_2 \times H_2$) is the result of performing a valid convolution of the d-th filter over the input volume with a stride of S, and then offset by d-th bias.

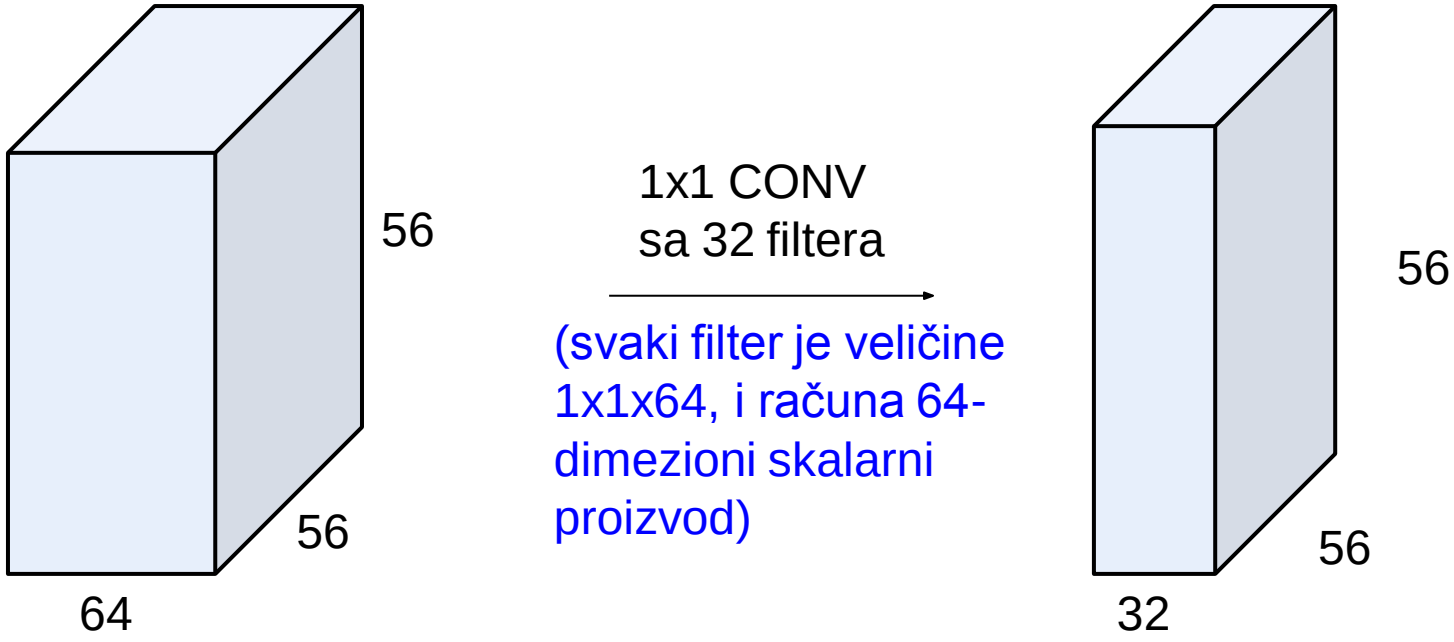
Tipična podešavanja:

K = (stepeni 2, npr. 32, 64, 128, 512)
- F = 3, S = 1, P = 1
- F = 5, S = 1, P = 2
- F = 5, S = 2, P = ? (šta god odgovara)
- F = 1, S = 1, P = 0

Summary. To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- In the output volume, the d-th depth slice (of size $W_2 \times H_2$) is the result of performing a valid convolution of the d-th filter over the input volume with a stride of S, and then offset by d-th bias.

(napomena, 1x1 konvolutivni slojevi imaju smisla)



Primer: CONV sloj u okruženju *Torch*

Summary. To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.

SpatialConvolution

```
module = nn.SpatialConvolution(nInputPlane, nOutputPlane, kW, kH, [dW], [dH], [padW], [padH])
```

Applies a 2D convolution over an input image composed of several input planes. The `input` tensor in `forward(input)` is expected to be a 3D tensor (`nInputPlane x height x width`).

The parameters are the following:

- `nInputPlane` : The number of expected input planes in the image given into `forward()` .
- `nOutputPlane` : The number of output planes the convolution layer will produce.
- `kW` : The kernel width of the convolution
- `kH` : The kernel height of the convolution
- `dW` : The step of the convolution in the width dimension. Default is `1` .
- `dH` : The step of the convolution in the height dimension. Default is `1` .
- `padW` : The additional zeros added per width to the input planes. Default is `0` , a good number is `(kW-1)/2` .
- `padH` : The additional zeros added per height to the input planes. Default is `padW` , a good number is `(kH-1)/2` .

Note that depending of the size of your kernel, several (of the last) columns or rows of the input image might be lost. It is up to the user to add proper padding in images.

If the input image is a 3D tensor `nInputPlane x height x width` , the output image size will be `nOutputPlane x oheight x owidth` where

```
owidth  = floor((width  + 2*padW - kW) / dW + 1)
oheight = floor((height + 2*padH - kH) / dH + 1)
```

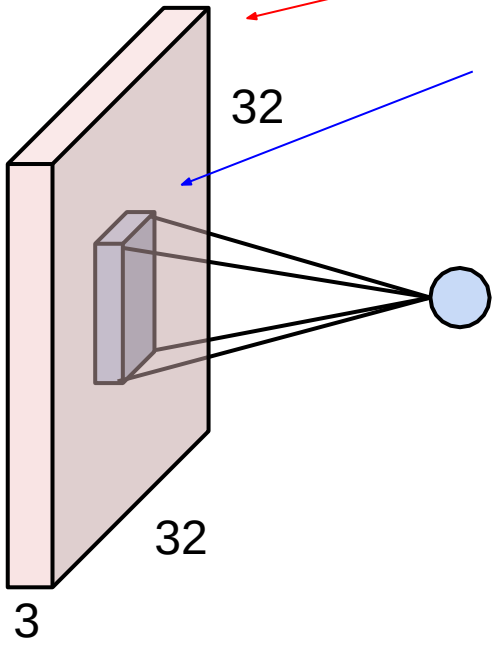
Primer: CONV sloj u okruženju *Caffe*

```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data"
  top: "conv1"
  # learning rate and decay multipliers for the filters
  param { lr_mult: 1 decay_mult: 1 }
  # learning rate and decay multipliers for the biases
  param { lr_mult: 2 decay_mult: 0 }
  convolution_param {
    num_output: 96     # learn 96 filters
    kernel_size: 11    # each filter is 11x11
    stride: 4          # step 4 pixels between each filter application
    weight_filler {
      type: "gaussian" # initialize the filters from a Gaussian
      std: 0.01        # distribution with stdev 0.01 (default mean: 0)
    }
    bias_filler {
      type: "constant" # initialize the biases to zero (0)
      value: 0
    }
  }
}
```

Summary. To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.

Mozak/neuron pogled na CONV Sloj

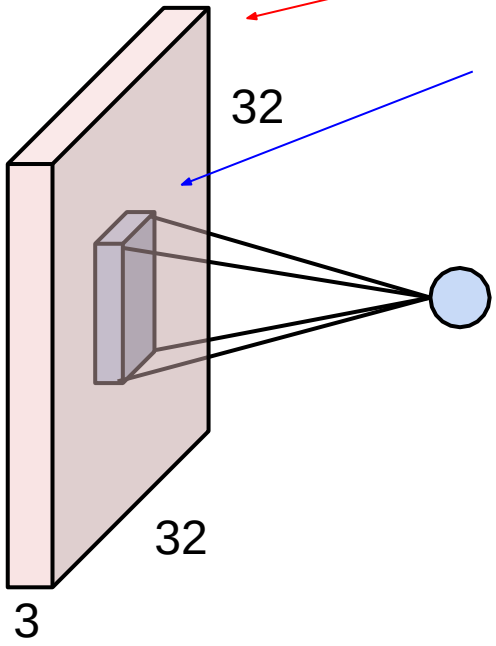


32x32x3 slika

5x5x3 filter

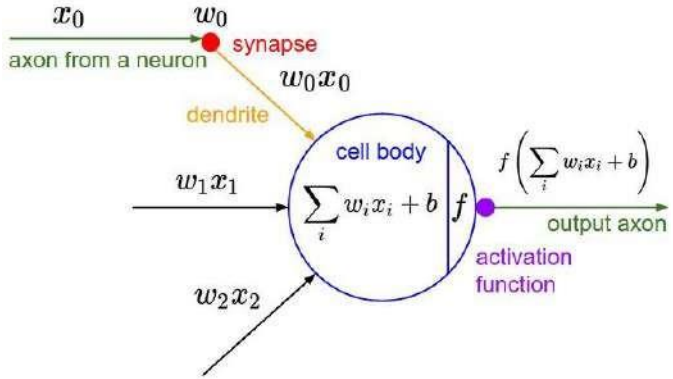
1 broj (vrednost):
rezultat skalarnog proizvoda filtera i tog dela slike
(tj. 5*5*3 = 75-dimenzioni skalarni proizvod)

Mozak/neuron pogled na CONV Sloj



32x32x3 slika

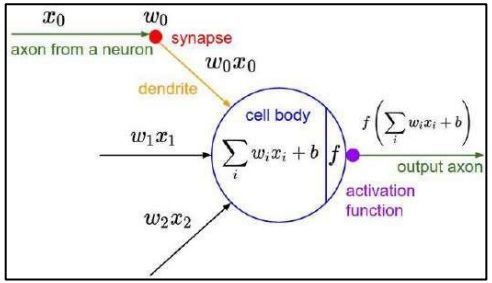
5x5x3 filter

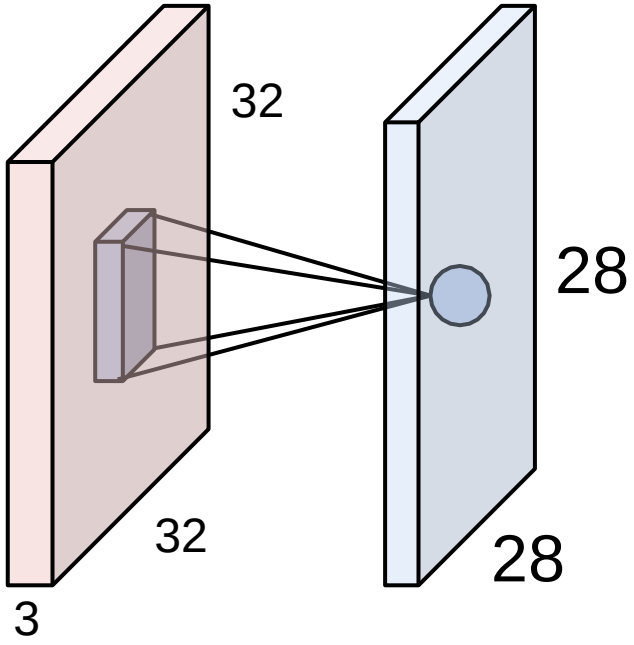


to je samo neuron koji je povezan sa samo jednim delom slike...

1 broj (vrednost):
rezultat skalarnog proizvoda filtera i tog dela slike
(tj. 5*5*3 = 75-dimenzioni skalarni proizvod)

Mozak/neuron pogled na CONV Sloj



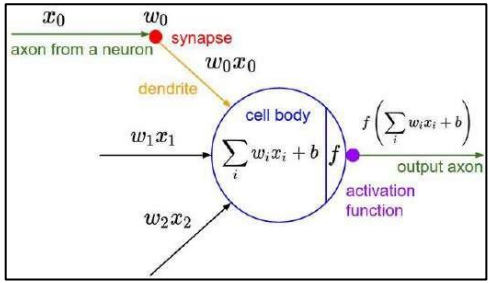


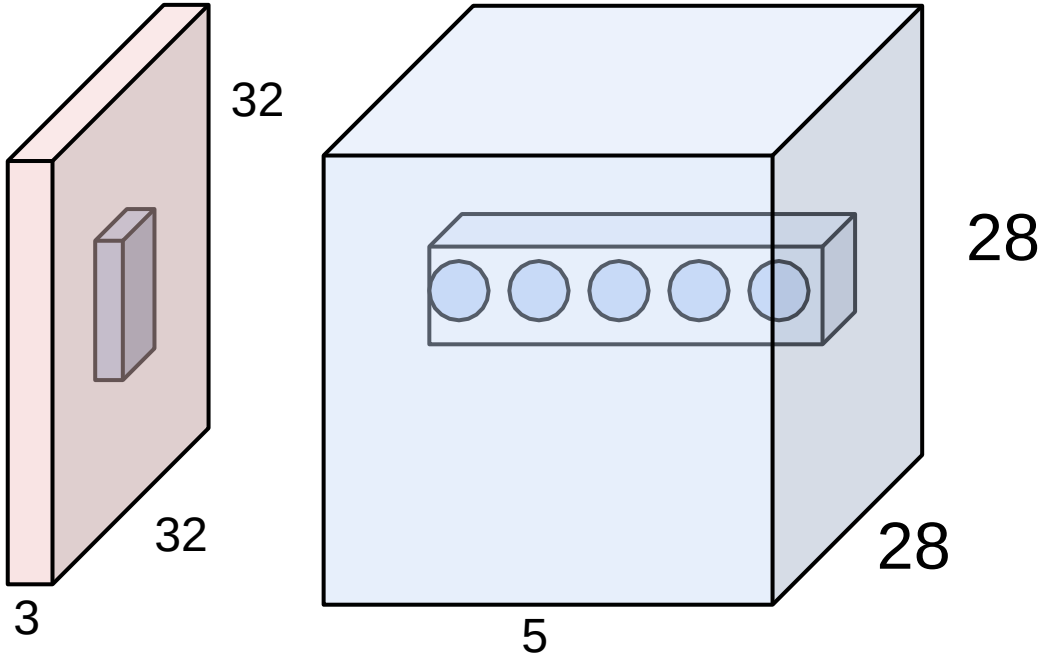
Aktivaciona mapa dimezionalnosti 28x28 je sloj izlaza neurona:

1. Svaki je povezan sa malim delom ulaza
2. Svi dele iste parametre

“5x5 filter” -> “vidno polje dimenzionalnosti 5x5”

Mozak/neuron pogled na CONV Sloj





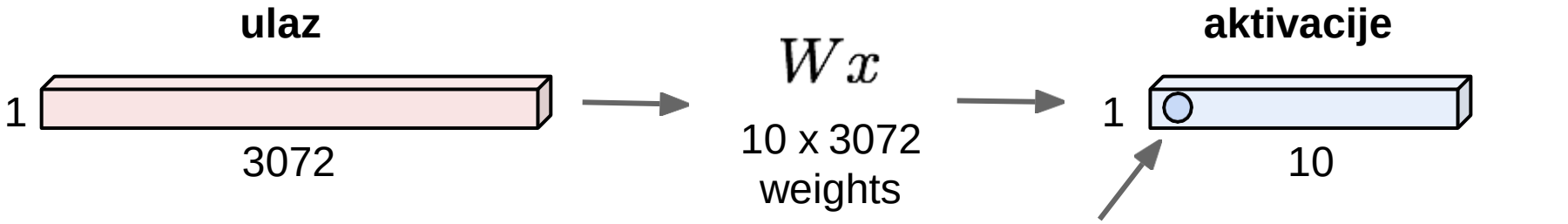
Npr. 5 filtera,
CONV sloj se sastoji od neurona složenih u mrežu (28x28x5)

Imamo 5 različitih neurona koji su povezani sa istim delom ulaza

Podsetnik: Potpuno Povezani Sloj

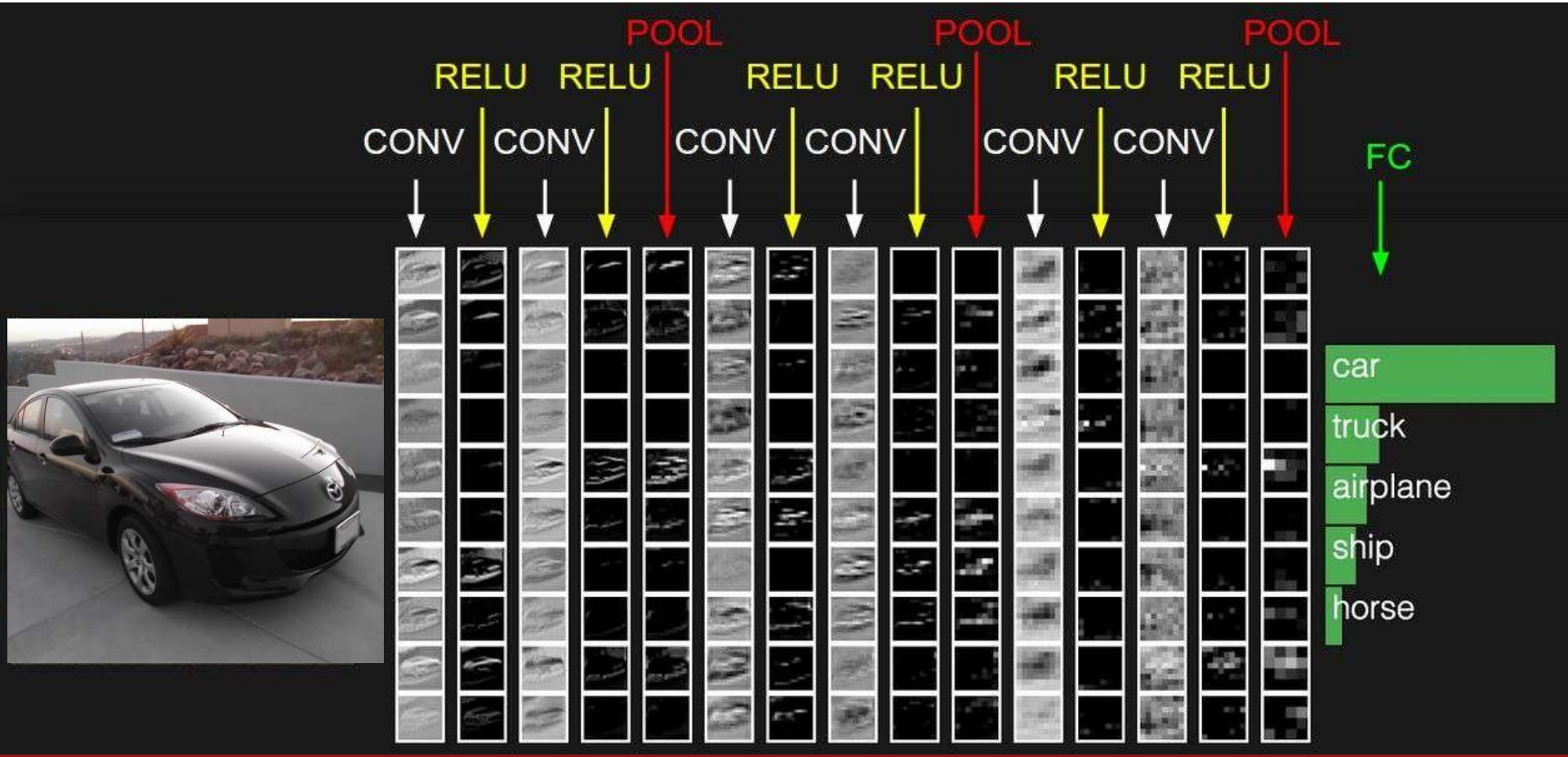
32x32x3 slika -> razvijamo u 3072 x 1

Svaki neuron je povezan sa celom slikom.



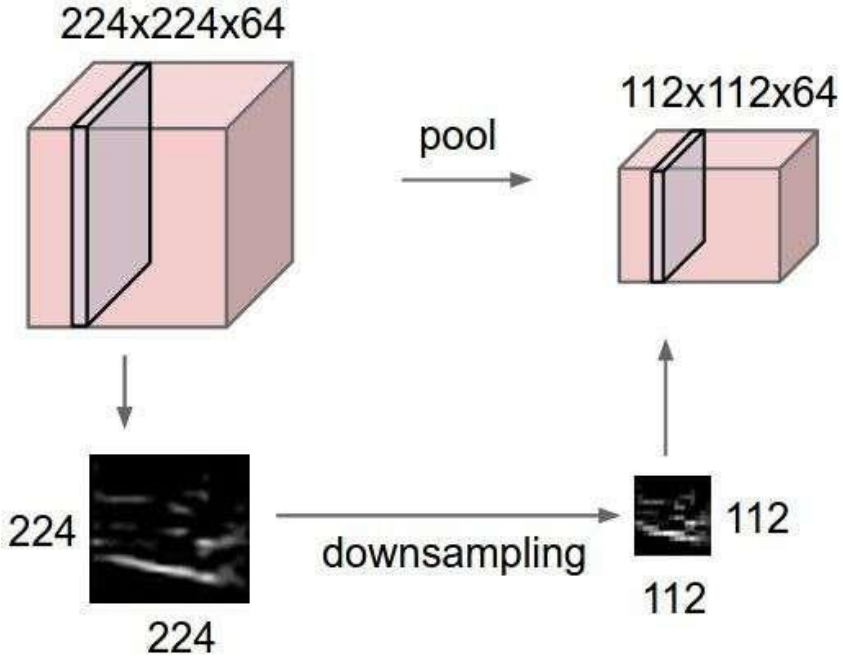
1 broj (vrednost):
rezultat skalarnog proizvoda jednog reda u W i celog ulaza (3072-dimenzioni skalarni proizvod)

Nismo još objasnili sledeće slojeve: POOL/FC



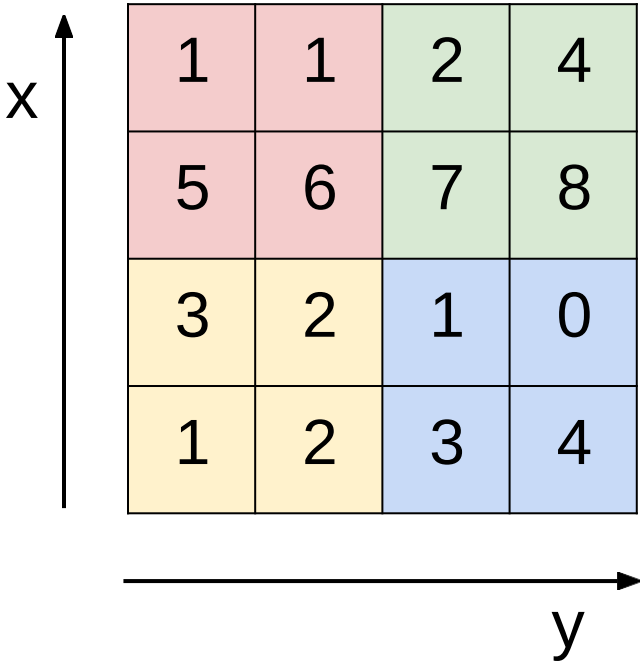
Pooling Sloj

- smanjujemo dimenzionalnost slojeva, sa ciljem da možemo lakše da obučimo mrežu (manje parametra), ali da pri tom sačuvamo što više informacija
- primenjuje se nezavisno na svaku aktivacionu mapu:



MAX POOLING

Jedna aktivaciona mapa dubine 1



max pool sa 2x2
fliterima i korakom 2

6	8
3	4

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires three hyperparameters:
  - their spatial extent F,
  - the stride S,
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F)/S + 1$
  - $H_2 = (H_1 - F)/S + 1$
  - $D_2 = D_1$
- Introduces zero parameters since it computes a fixed function of the input
- Note that it is not common to use zero-padding for Pooling layers

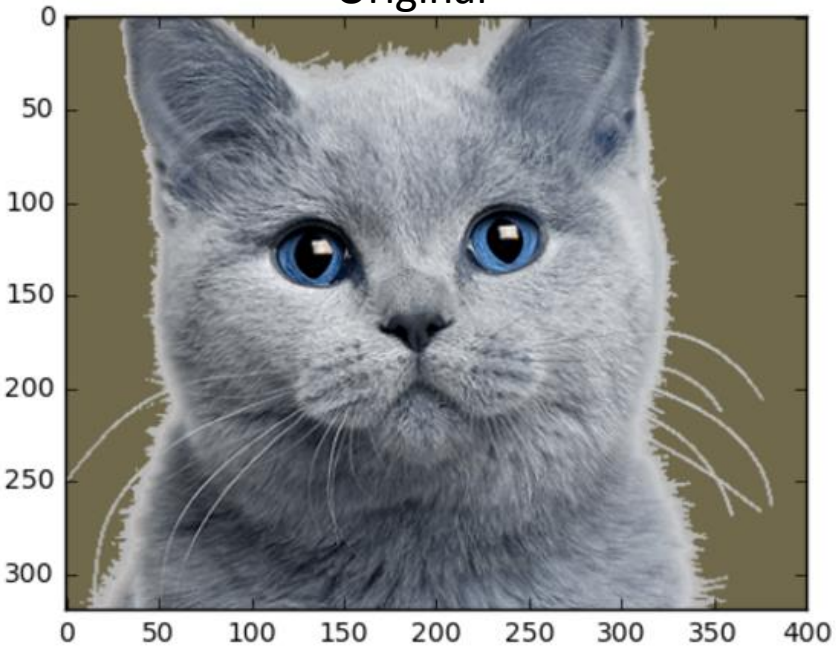
- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires three hyperparameters:
  - their spatial extent F,
  - the stride S,
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F)/S + 1$
  - $H_2 = (H_1 - F)/S + 1$
  - $D_2 = D_1$
- Introduces zero parameters since it computes a fixed function of the input
- Note that it is not common to use zero-padding for Pooling layers

Tipična podešavanja:

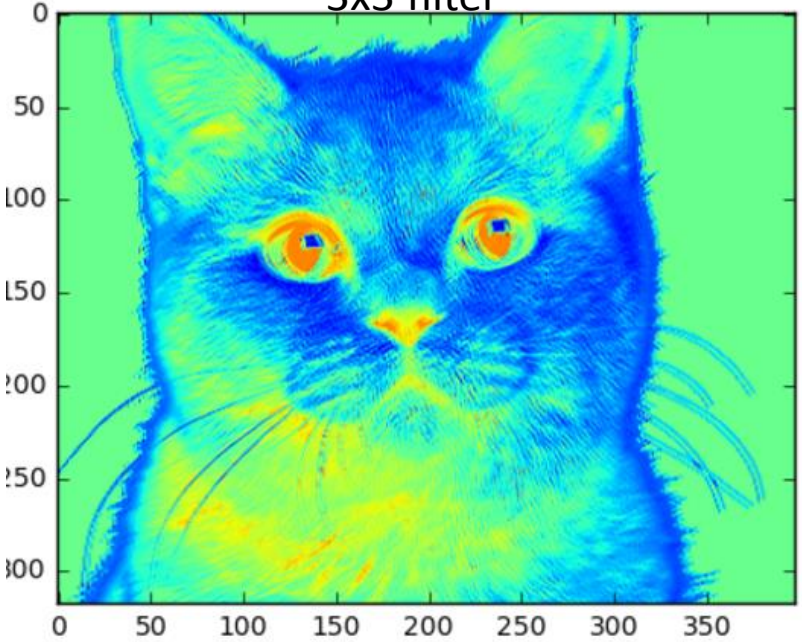
F = 2, S = 2
F = 3, S = 2

Primer jednog konvolucionog filtera 1

Original



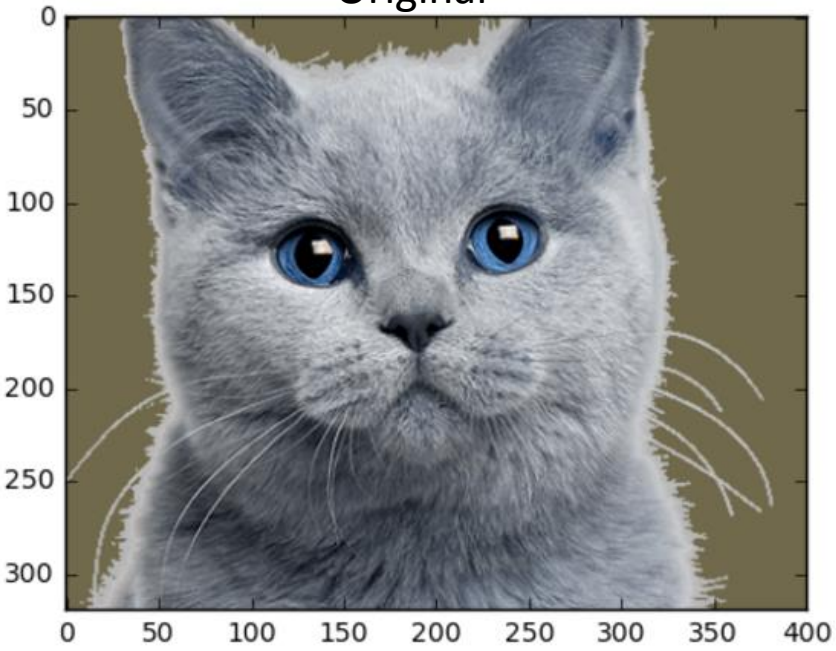
3x3 filter



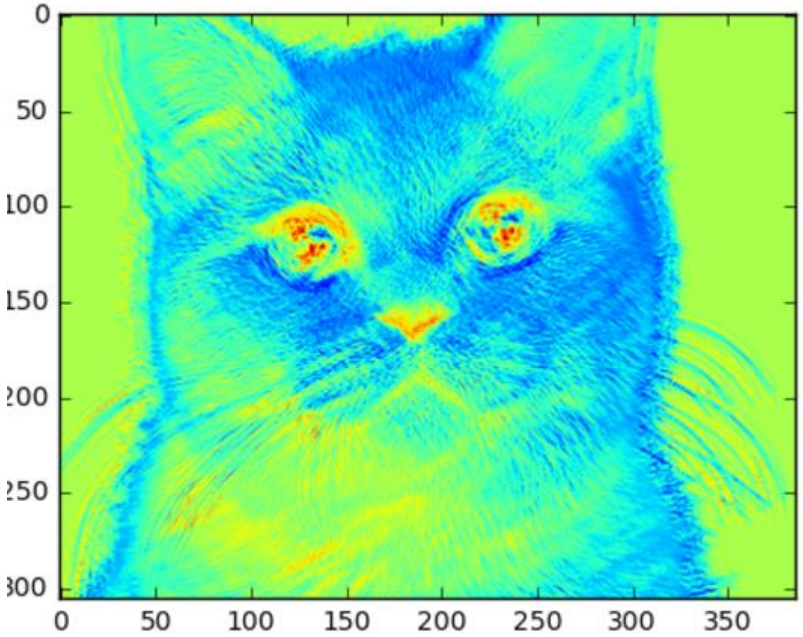
Izvor: https://hackernoon.com/visualizing-parts-of-convolutional-neural-networks-using-keras-and-cats-5cc01b214e59

Primer jednog konvolucionog filtera 2

Original



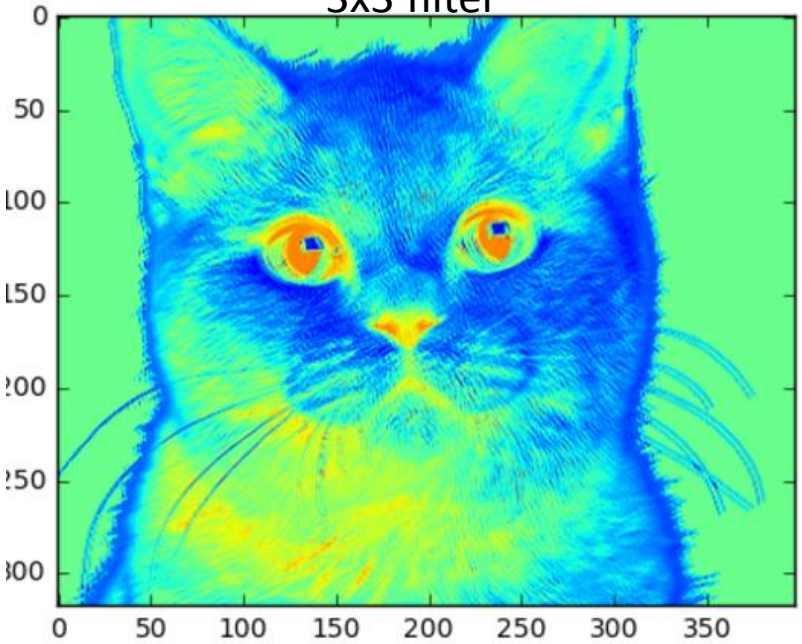
15x15 filter



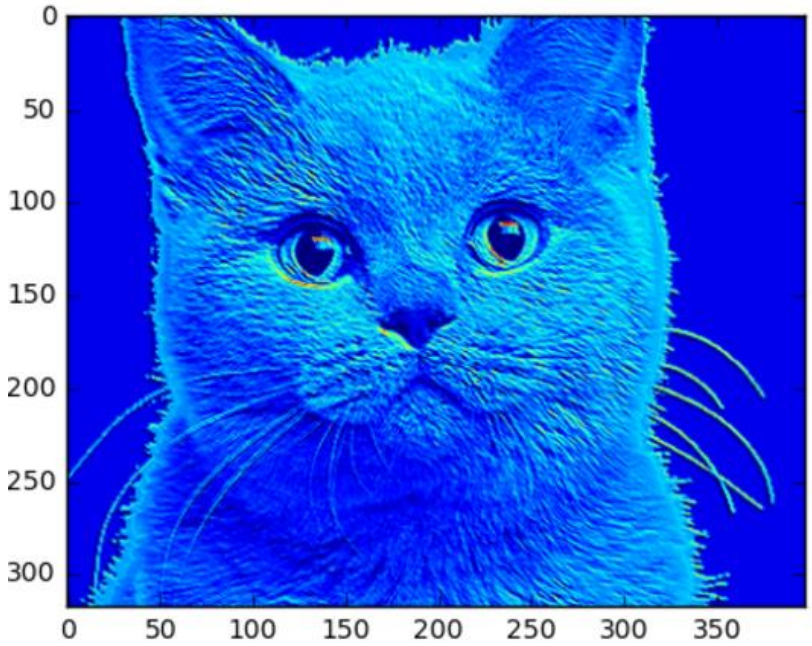
Izvor: https://hackernoon.com/visualizing-parts-of-convolutional-neural-networks-using-keras-and-cats-5cc01b214e59

Primer jednog konvolucionog filtera 3, ReLU posle 3x3 filtera

3x3 filter



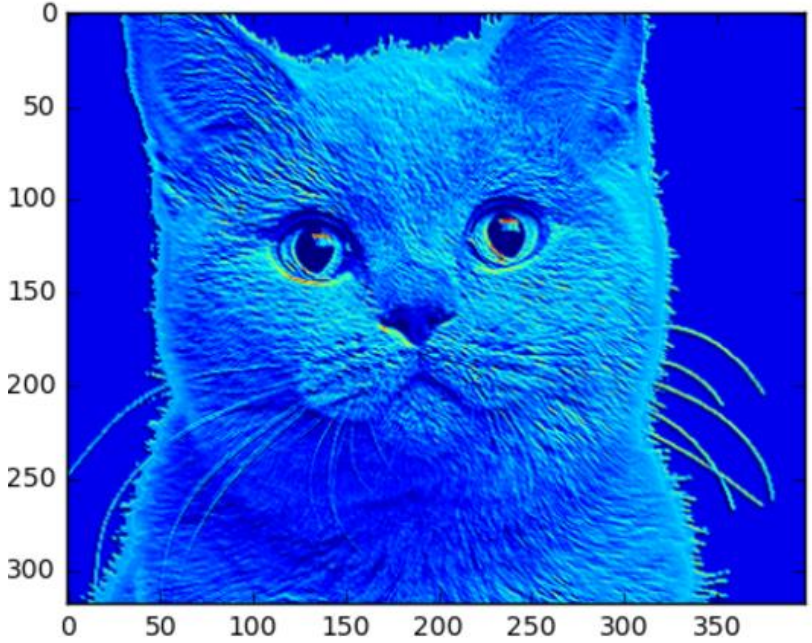
ReLU



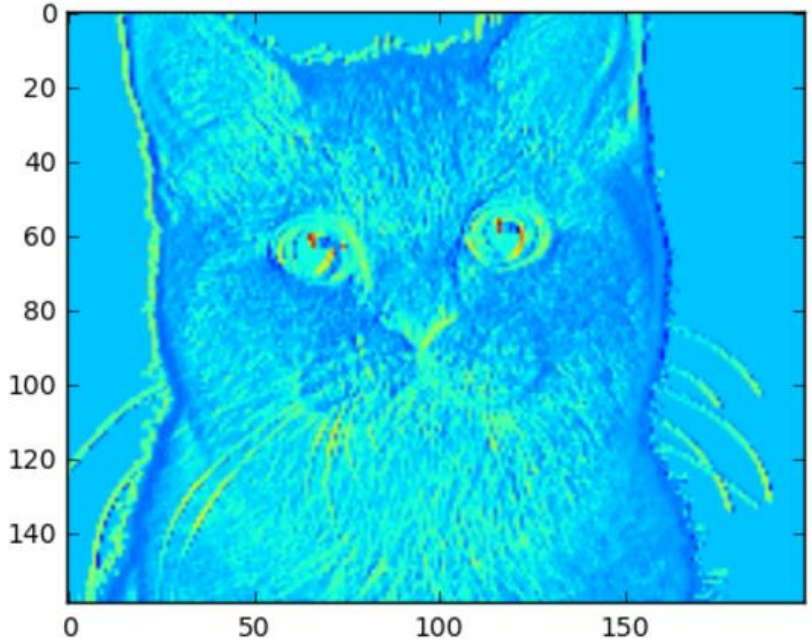
Izvor: https://hackernoon.com/visualizing-parts-of-convolutional-neural-networks-using-keras-and-cats-5cc01b214e59

Primer jednog konvolucionog filtera 4, Max Pooling 2x2 posle ReLU

ReLU

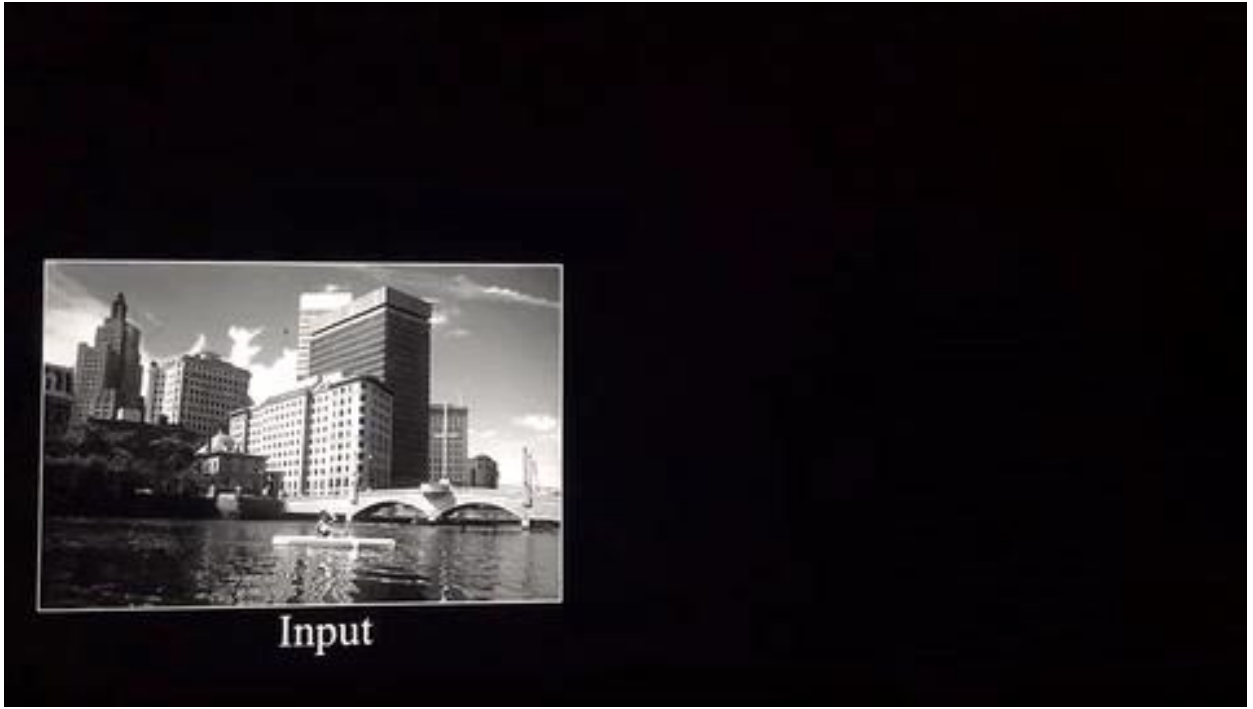


Max Pooling 2x2



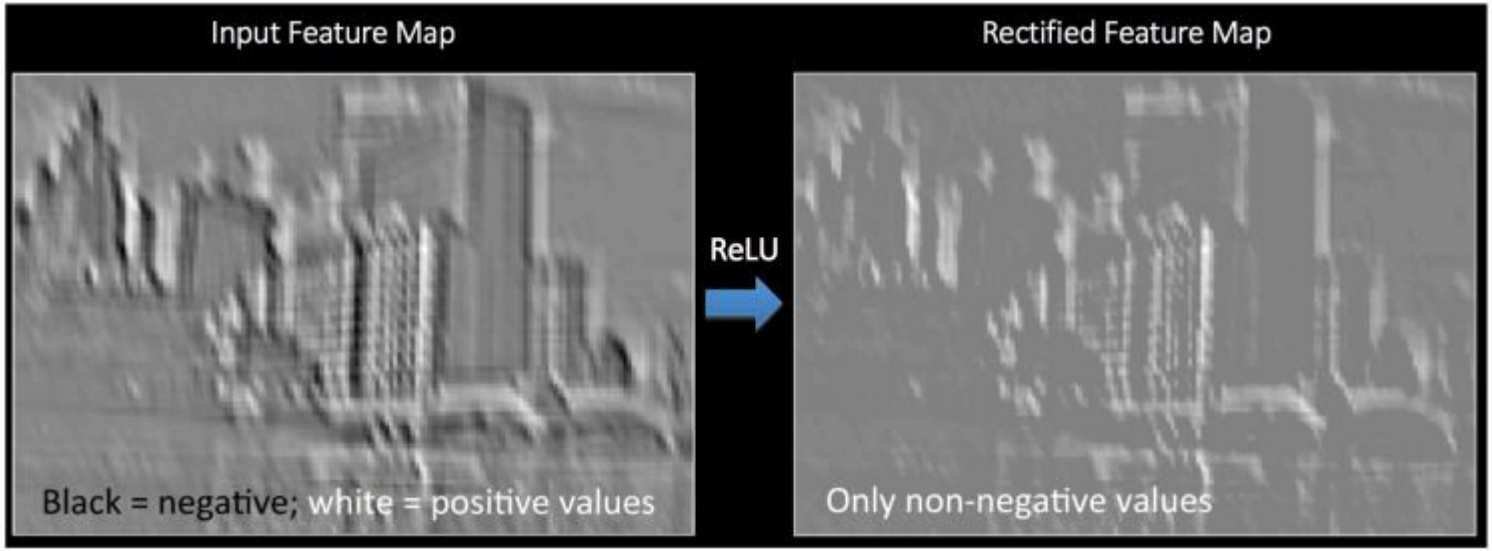
Izvor: https://hackernoon.com/visualizing-parts-of-convolutional-neural-networks-using-keras-and-cats-5cc01b214e59

Primer još jednog konvolucionog filtera 1



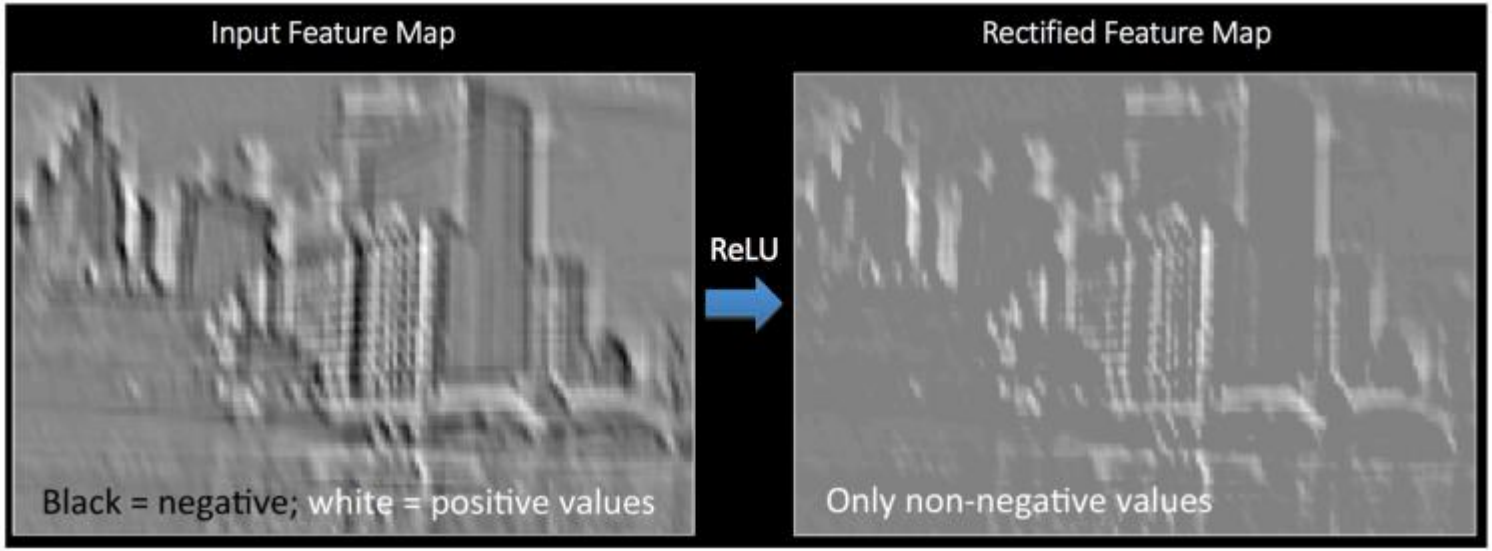
Izvor: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

Primer još jednog konvolucionog filtera 2, ReLU



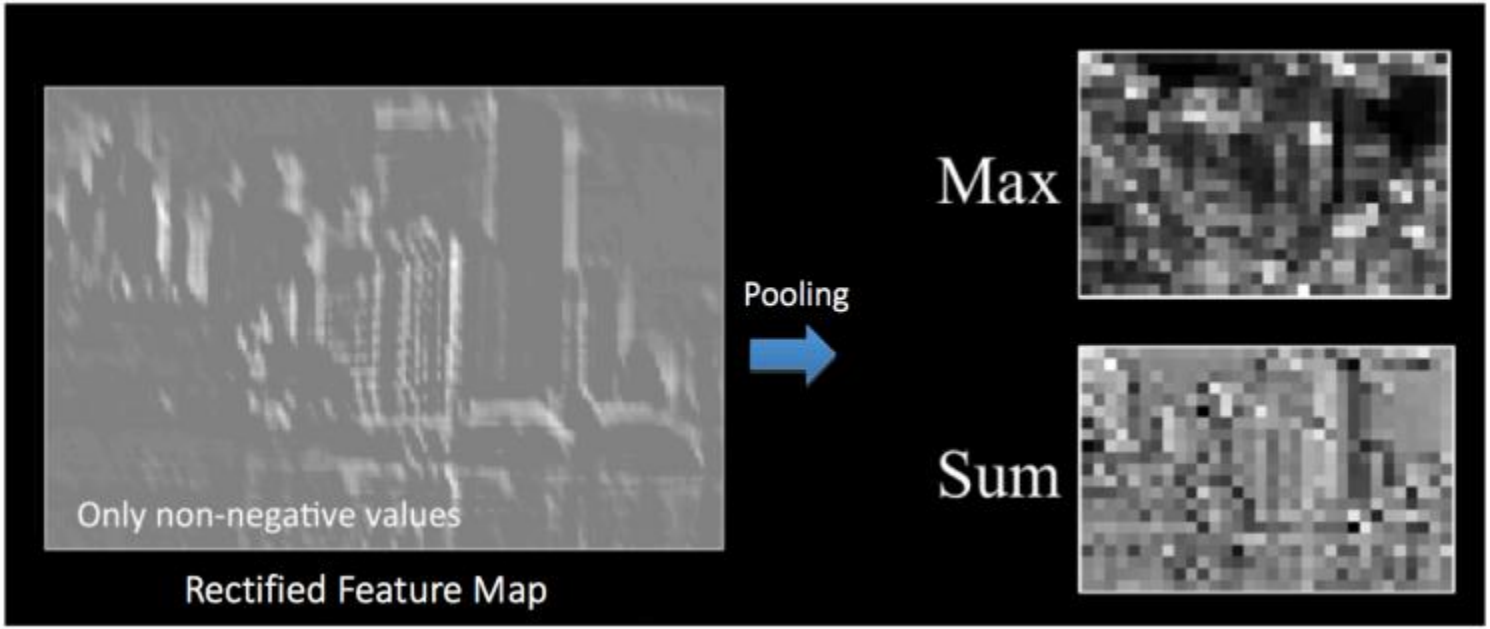
Izvor: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

Primer još jednog konvolucionog filtera 2, ReLU



Izvor: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

Primer još jednog konvolucionog filtera 3, Pooling

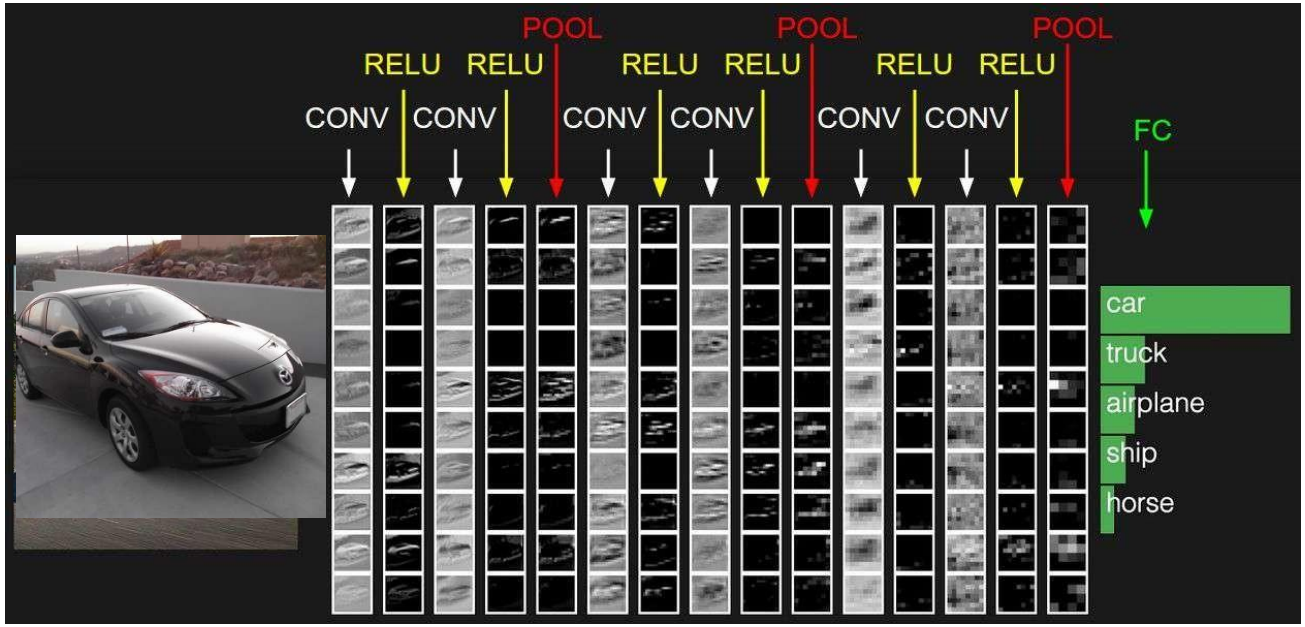


Izvor: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

Potpuno Povezani Sloj

Fully Connected Layer (*FC layer*)

- Sadrži neurone koji su potpuno povezani sa svojim ulazom, kao u tipičnoj neuronskoj mreži

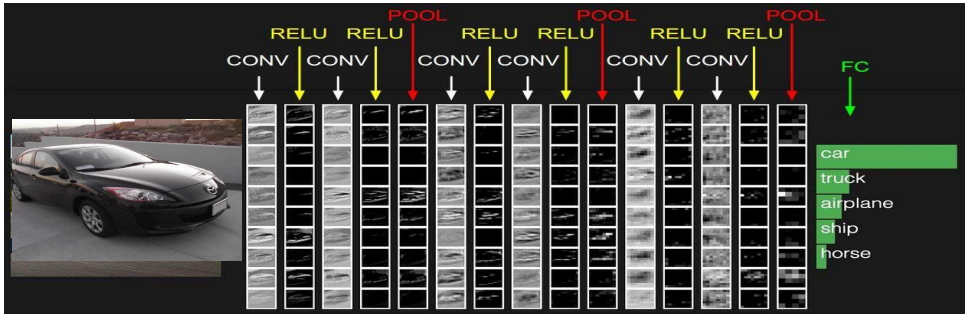


Potpuno Povezani Sloj

Fully Connected Layer (*FC layer*)

- Sadrži neurone koji su potpuno povezani sa svojim ulazom, kao u tipičnoj neuronskoj mreži
  Konvolutivne slojeve možemo da shvatimo kao ekstrakciju osobina iz slike koje pomoću kojih onda obučavamo potpuno povezanu neuronsku mrežu na kraju.

  Zašto nisu svi slojevi potpuno povezani?

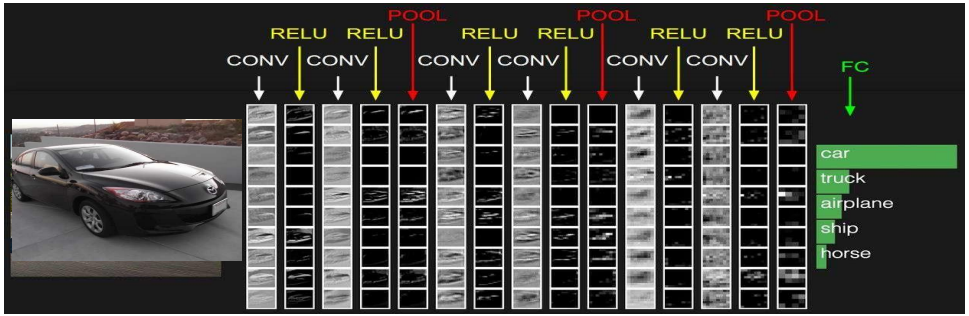


Potpuno Povezani Sloj

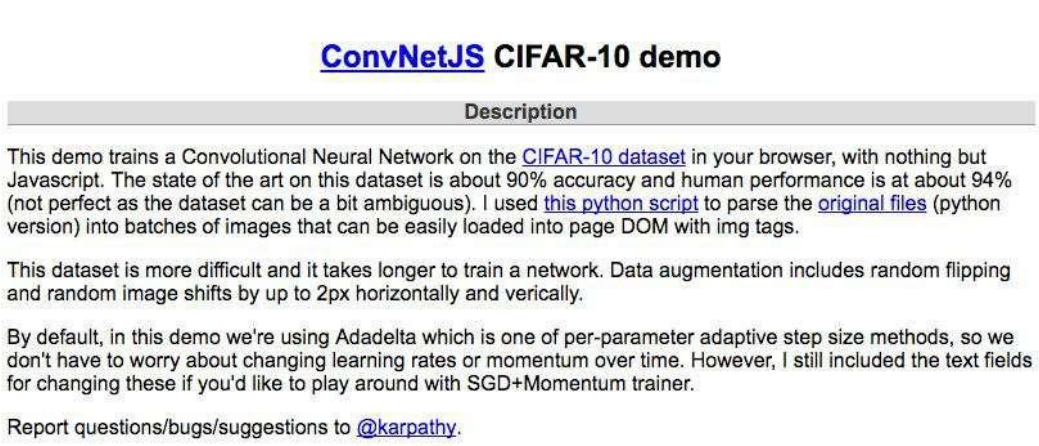
Fully Connected Layer (*FC layer*)

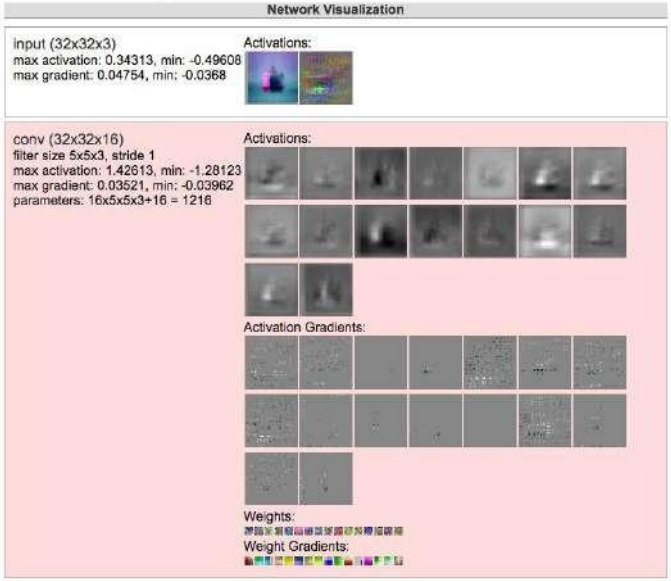
- Sadrži neurone koji su potpuno povezani sa svojim ulazom, kao u tipičnoj neuronskoj mreži
  Konvolutivne slojeve možemo da shvatimo kao ekstrakciju osobina iz slike koje pomoću kojih onda obučavamo potpuno povezanu neuronsku mrežu na kraju.

  Zašto nisu svi slojevi potpuno povezani?
  Takava mreža bi imala previše parametra da bi realno mogla da se obuči!



[ConvNetJS demo: obučavanje na CIFAR-10]





http://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html

Rezime

- Kovolutivne mreže slažu CONV,POOL i FC slojeve
- Trenutno je trend ka manjim filterima, a dubljim mrežama
- Eksperimentiše se i sa uklanjanjem POOL/FC slojeva (ostaju samo CONV)
- Tipična arhitektura izgleda ovako:

  [(CONV-RELU)*N-POOL?]*M-(FC-RELU)*K,SOFTMAX

  gde je N~5, M je veliko, 0 <= K <= 2.
  - ali neke od trenutno najboljih mreža ResNet/GoogLeNet menjaju ovu arhitekturu