

# Üzleti Intelligencia

## 8. Előadás: Objektum detekció

Kuknyó Dániel  
Budapesti Gazdasági Egyetem

2023/24  
1.félév

**1** Bevezetés

**2** Objektum detekció

**3** Tanítás

**4** További architektúrák

**1** Bevezetés

**2** Objektum detekció

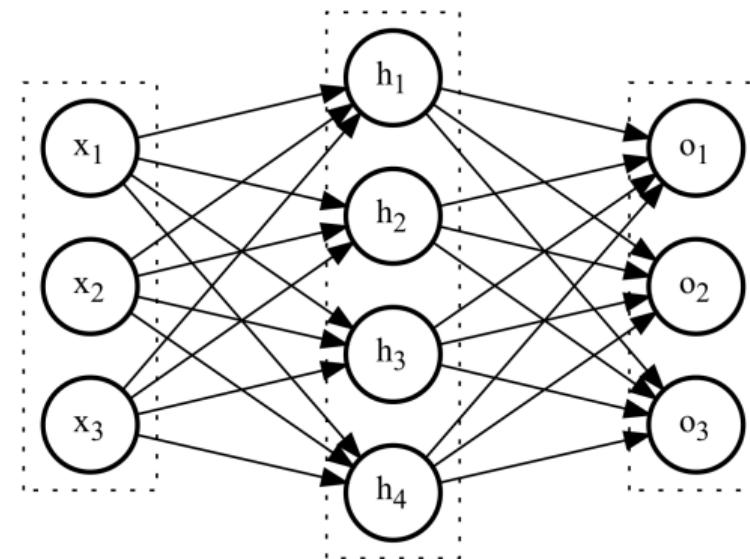
**3** Tanítás

**4** További architektúrák

# Osztályozás

Osztályozás esetén a neurális hálózat output rétegében minden lehetséges osztályhoz egy neuron tartozik. A neuronok azt a valószínűséget becsülik meg, hogy az adott mintaegyed mekkora valószínűsséggel esik a neuronhoz tartozó kategóriába.

Ez a softmax réteg, ahol az egyes neuronok aktivációs függvénye a softmax függvény. A softmax rétegen lévő neuronok outputjainak összege mindig 1.



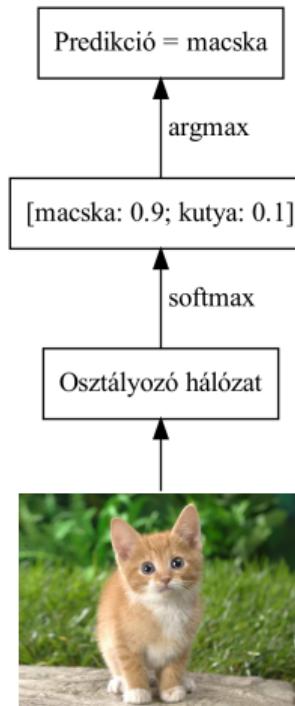
## Predikció osztályozás esetén

A legegyszerűbb módja az osztályozásnak, ha a hálózat outputként **összesen egy címkét** ad outputként, ami a predikciója az input képre (vagy adatra) vonatkozóan.

Ebben az esetben a hálózat softmax rétege azt becsüli meg, hogy **mekkora valószínűséggel tartozik a mintaegyed a tanító osztályok valamelyikébe**. Például:

[macska : 0.9; kutya : 0.1]

Ezután az output úgy áll elő, hogy a hálózat **kiválasztja a legnagyobb valószínűségű osztályt** az *argmax* operátorral, majd visszaadja a legnagyobb valószínűséghez tartozó címkét.



# Lokalizáció

Az objektum lokalizáció feladata az osztályozás egy speciális esete. A lokalizáció problémájában nem csak azt kell megbecsülnie a neurális hálózatnak, hogy milyen objektum található egy képen, hanem meg is kell adnia az **objektum pozíóját azzal, hogy megadja a kereteződobozának koordinátáit**.

Ebben az esetben a hálózat outputja nem csak egy címke, hanem a dobozt meghatározó négy koordináta is. Ezek a leggyakrabban:

- $x$ : A doboz bal felső sarkának  $x$  koordinátája.
- $y$ : A doboz bal felső sarkának  $y$  koordinátája.
- $w$ : A doboz szélessége (width).
- $h$ : A doboz magassága (height).

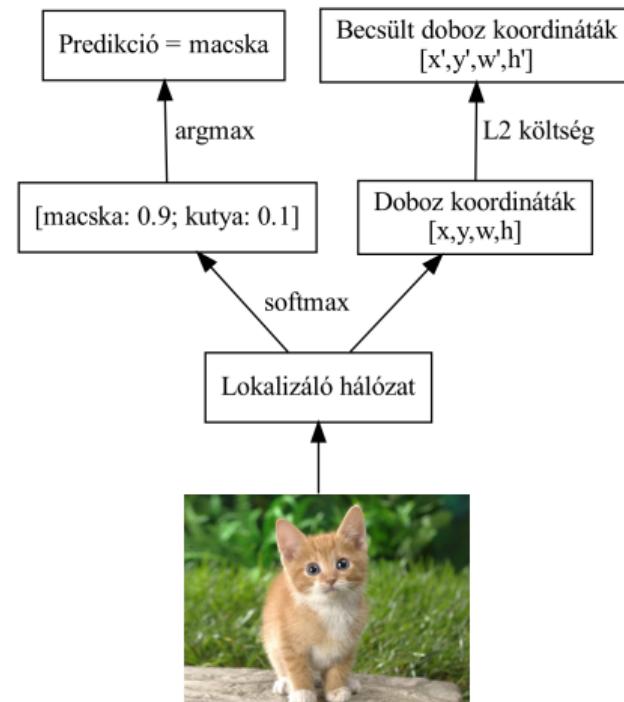


# Predikció lokalizáció esetén

Lokalizáció esetén a neurális hálózat két részre ágazik, mivel két különböző feladatot kell elvégeznie:

**Osztályozás:** ez megegyezik az osztályozó hálózat által végzett feladattal. A hálózat megbecsül egy valószínűségeszlást, majd ebből kiválasztja a legnagyobb valószínűséghez tartozó osztályt, amit visszaad mint predikció.

**Kereteződoboz predikciók:** a lokalizáció feladatához a neurális hálózatnak meg kell becsülnie az érdekelt régiók dobozainak koordinátáit. A neurális hálózat kiválasztja a becsült kereteződobozok közül azt, amelyik a **legnagyobb valószínűséggel tartalmazza a keresett objektumot**.



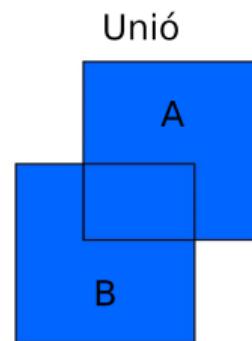
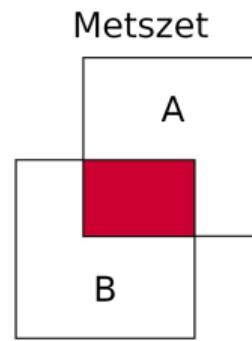
# Pontosság mérése kereteződobozokkal

Mivel az objektum detekció egy felügyelt tanulási algoritmus, **szüksége van címkékre**. Ebben az esetben a címkék a valós osztályt és a kereteződoboz  $x, y, w, h$  koordinátáit tartalmazzák.

A jóságfüggvény  $A$  valós és  $B$  becsült kereteződoboz koordinátáit hasonlítja össze az IoU (metszet/unió) mérőszámmal:

$$IoU = \frac{A \cap B}{A \cup B}$$

Tehát minél jobban illeszkedik a becsült doboz a valósra, annál közelebb van a mérőszám 1-hez. Ha a két doboznak nincs közös területe,  $IoU = 0$ .



1 Bevezetés

2 Objektum detekció

3 Tanítás

4 További architektúrák

# Objektum detekció

Az objektum detekció a lokalizáció általánosítása több objektumra egyetlen képen belül. Ebben az esetben a neurális hálózat feladata  **minden ismert objektum lokalizálása a képen.**

Ebben az esetben a hálózat outputja minden egyes ismert objektumra:

- Az objektum becsült címkéje.
- Az objektum kereteződobozának  $x, y, w, h$  koordinátái.

Mivel az érdekelt objektumok száma képenként eltérő lehet, az **objektum detekciós hálózatok outputjainak számossága is képenként különbözik!**



# Kezdeti detektor implementáció

Az első objektum detektor modellek implementációi meglehetősen naívak voltak. Az eljárás szerint a modell végigcsúsztat egy előre meghatározott méretű ablakot a képen, és minden egyes általa meghatározott képről eldönti, hogy háttér-e (*bgr*) vagy valamelyik keresett osztályba tartozik-e:

$$output = [bgr : 0.8, kutya : 0.1, macska : 0.1]$$

Ebben az esetben  $\hat{y} = \underset{p}{\operatorname{argmax}}(output) = bgr$ , és az algoritmus eldobja a kereteződoboz predikcióját.



# Kezdeti detektor implementáció

Az első objektum detektor modellek implementációi meglehetősen naívak voltak. Az eljárás szerint a modell végigcsúsztat egy előre meghatározott méretű ablakot a képen, és minden egyes általa meghatározott képről eldönti, hogy háttér-e (*bgr*) vagy valamelyik keresett osztályba tartozik-e:

$$output = [bgr : 0.0, \text{kutya} : 0.01, \text{macska} : 0.99]$$

Ebben az esetben  $\hat{y} = \underset{p}{\operatorname{argmax}}(output) = \text{macska}$ , és a becsült kereteződoboz az, ahol a legnagyobb a keresett osztály valószínűsége.



# Miért nem lehetséges ez az implementáció?

Hány lehetséges kereteződoboz van egy  $H \cdot W$  méretű képen?

Ha van egy  $h \cdot w$  méretű kereteződoboz a lehetséges  $x$  koordináták száma  $W - w + 1$ , és a lehetséges  $y$  koordináták száma  $H - h + 1$ . A lehetséges megoldások száma egyetlen kereteződoboz méretre:  $(W - w + 1) \cdot (H - h + 1)$ .

Az összes lehetséges kereteződoboz méretre pedig:

$$\sum_{h=1}^H \sum_{w=1}^W (W - w + 1) \cdot (H - h + 1) = \frac{H(H+1)}{2} \frac{W(W+1)}{2}$$

Ez azt jelenti, hogy egy  $800 \cdot 600$  méretű képre  $\sim 58$  millió lehetséges kereteződoboz létezik.

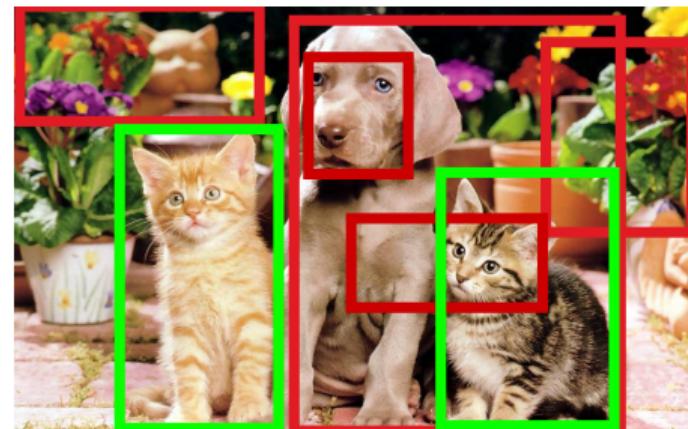


# Területi javaslatok

Az eljárás ami meggyorsította a folyamatot a területi javaslatok algoritmusa volt.

Ahelyett, hogy egy előre meghatározott méretű ablakot csúsztatna végig egy képen, a területi javaslatok algoritmusa megpróbálja a kép **pixeleteit klaszterezni hasonlóság szerint**, ezzel területeket formázva.

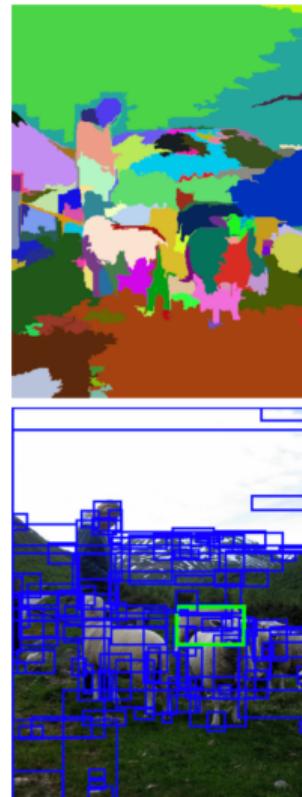
Később ezen területeknek a kereteződobozai lesznek átadva egy osztályozó neurális hálózatnak ami eldönti, hogy a keresett objektumok valamelyike mekkora valószínűsséggel található meg a területen.



# Területi javaslatok szelektív kereséssel

A szelektív keresés egy potenciális objektum területeket javasoló algoritmus, ami jóval hatékonyabban képes működni, mint a csúszóablakos algoritmus:

- ❶ **Kép szegmentálása:** Az input kép pixeleinek csoportosítása szín, textúra és egyéb vizuális tulajdonságok szerint.
- ❷ **Régiók hasonlósága:** A különböző régiók összeolvasztása valamilyen hasonlósági mérték szerint.
- ❸ **Hierarchikus csoportosítás:** Az összeolvasztott régiók hierarchikus klaszterezése.
- ❹ **Objektum javaslatok:** Objektum javaslatok halmazának elkészítése, amik területi javaslatként szolgálnak.
- ❺ **Objektum detekció:** A területi javaslatok átadódnak a neurális hálózatnak osztályozásra.



# Területi javaslatok szelektív kereséssel

A szelektív keresés egy potenciális objektum területeket javasoló algoritmus, ami jóval hatékonyabban képes működni, mint a csúszóablakos algoritmus:

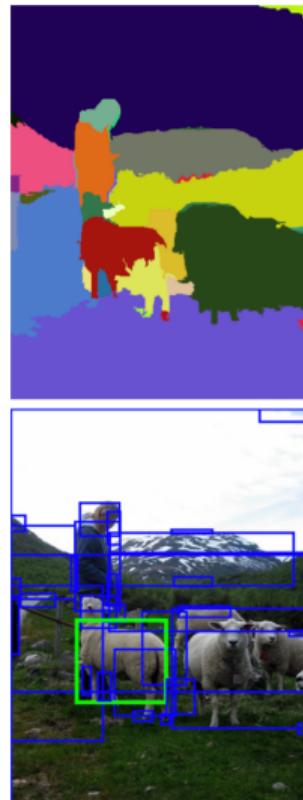
- ① **Kép szegmentálása:** Az input kép pixeleinek csoportosítása szín, textúra és egyéb vizuális tulajdonságok szerint.
- ② **Régiók hasonlósága:** A különböző régiók összeolvasztása valamilyen hasonlósági mérték szerint.
- ③ **Hierarchikus csoportosítás:** Az összeolvasztott régiók hierarchikus klaszterezése.
- ④ **Objektum javaslatok:** Objektum javaslatok halmazának elkészítése, amik területi javaslatként szolgálnak.
- ⑤ **Objektum detekció:** A területi javaslatok átadódnak a neurális hálózatnak osztályozásra.



# Területi javaslatok szelektív kereséssel

A szelektív keresés egy potenciális objektum területeket javasoló algoritmus, ami jóval hatékonyabban képes működni, mint a csúszóablakos algoritmus:

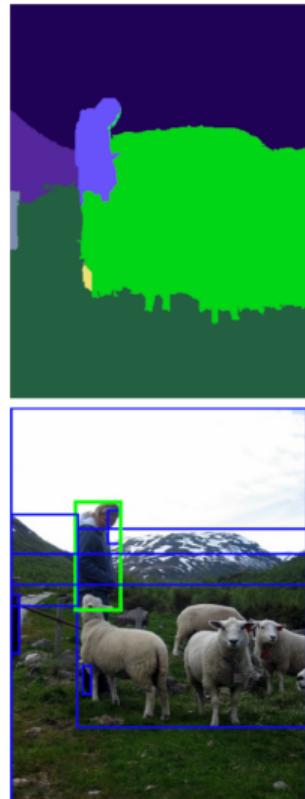
- 1 Kép szegmentálása:** Az input kép pixeleinek csoportosítása szín, textúra és egyéb vizuális tulajdonságok szerint.
- 2 Régiók hasonlósága:** A különböző régiók összeolvasztása valamilyen hasonlósági mérték szerint.
- 3 Hierarchikus csoportosítás:** Az összeolvasztott régiók hierarchikus klaszterezése.
- 4 Objektum javaslatok:** Objektum javaslatok halmazának elkészítése, amik területi javaslatként szolgálnak.
- 5 Objektum detekció:** A területi javaslatok átadódnak a neurális hálózatnak osztályozásra.



# Területi javaslatok szelektív kereséssel

A szelektív keresés egy potenciális objektum területeket javasoló algoritmus, ami jóval hatékonyabban képes működni, mint a csúszóablakos algoritmus:

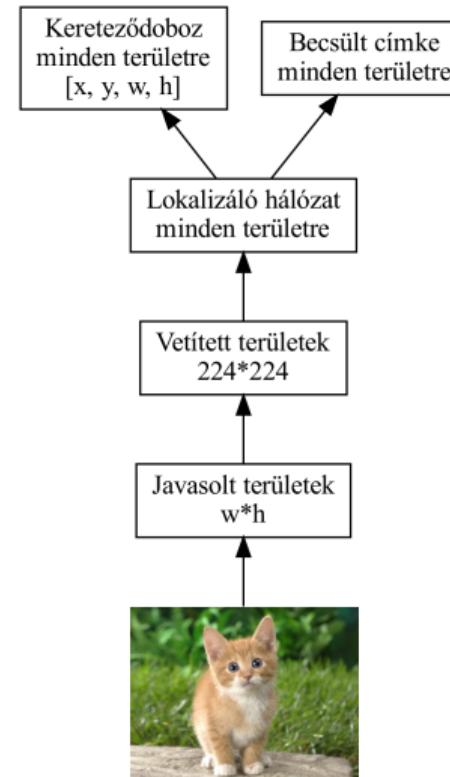
- 1 Kép szegmentálása:** Az input kép pixeleinek csoportosítása szín, textúra és egyéb vizuális tulajdonságok szerint.
- 2 Régiók hasonlósága:** A különböző régiók összeolvasztása valamilyen hasonlósági mérték szerint.
- 3 Hierarchikus csoportosítás:** Az összeolvasztott régiók hierarchikus klaszterezése.
- 4 Objektum javaslatok:** Objektum javaslatok halmazának elkészítése, amik területi javaslatként szolgálnak.
- 5 Objektum detekció:** A területi javaslatok átadódnak a neurális hálózatnak osztályozásra.



# R-CNN architektúra objektum detekcióra

Az objektum detekció folyamata területjavasolással:

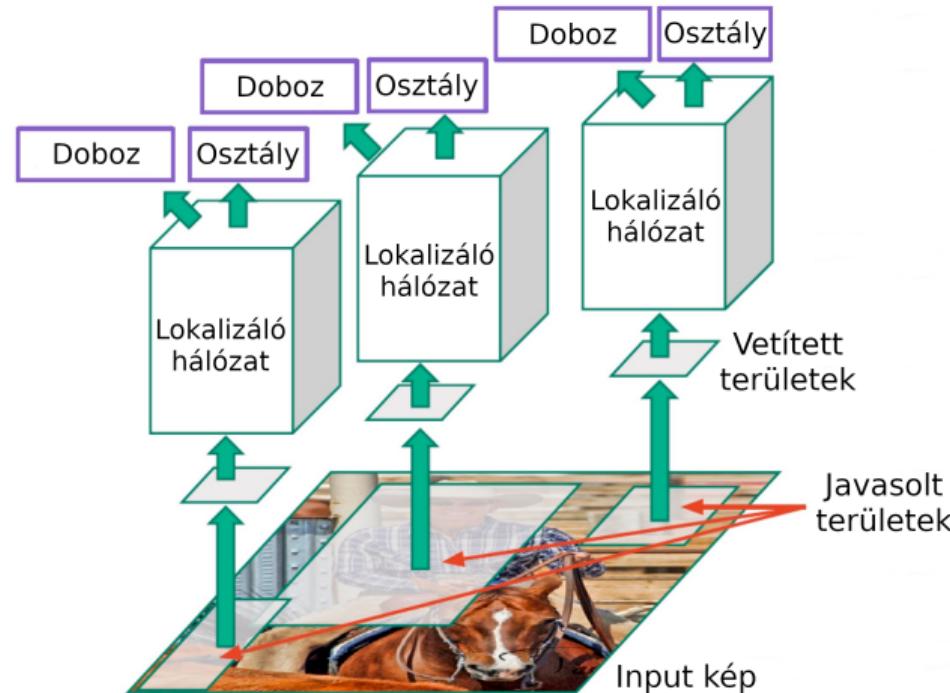
- ① A területjavasoló **kiválaszt érdekelt régiókat** a képről. Ezek lehetnek bármekkora méretűek.
- ② A javasolt régiók **fix méretűre vetítése**. Ez leggyakrabban  $224 \cdot 224$  a gyors osztályozás miatt.
- ③ A lokalizációs hálózat **megbecsüli az osztályokat** és a **kereteződobozok koordinátáit** a vetített terület alapján.



# R-CNN architektúra objektum detekcióra

Az objektum detekció folyamata területjavasolással:

- ① A területjavasoló **kiválaszt** érdekelt **régiókat** a képről. Ezek lehetnek bármekkora méretűek.
- ② A javasolt régiók **fix méretűre vetítése**. Ez leggyakrabban  $224 \cdot 224$  a gyors osztályozás miatt.
- ③ A lokalizációs hálózat **megbecsüli** az **osztályokat** és a **kereteződobozok koordinátáit** a vetített terület alapján.



1 Bevezetés

2 Objektum detekció

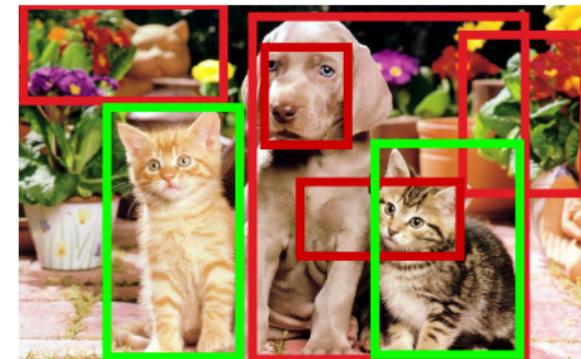
3 Tanítás

4 További architektúrák

# Objektum detektció kiértékelése

A pontosság és visszatérés mérhető objektum detekció esetén is. A **TP**, **FP** és **FN** osztályozás viszont csak specifikusan értelmezhető:

- **TP** (valós pozitív): a becsült osztály megegyezik a valóssal, és  $IoU > 50\%$ .
- **FP** (hamis pozitív): a becsült osztály nem egyezik meg a valóssal vagy  $IoU < 50\%$ .
- **FN** (hamis negatív): olyan valós kereteződoboz, amelyikhez nem tartozik predikció.
- **TN** (hamis negatív): nem értelmezett objektum detekció esetén.



# Az objektum detekció mérőszámai

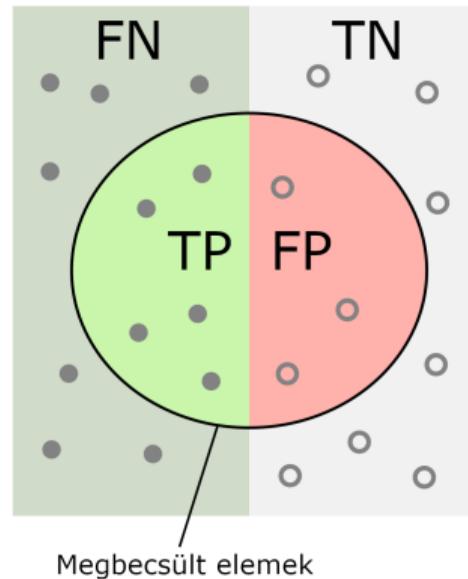
A detekció pontosságának két fontos mutatója:

- **Pontosság:** A megbecsült elemek közül hány volt valós pozitív?

$$P = \frac{TP}{TP + FP}$$

- **Visszahívás:** A valós pozitívak közül hány elem lett megbecsülve?

$$R = \frac{TP}{TP + FN}$$



$$\text{Pontosság} = \frac{\text{Green Area}}{\text{Red Area} + \text{Green Area}}$$
    
$$\text{Visszahívás} = \frac{\text{Green Area}}{\text{Green Area} + \text{Grey Area}}$$

# Átlagos pontosság

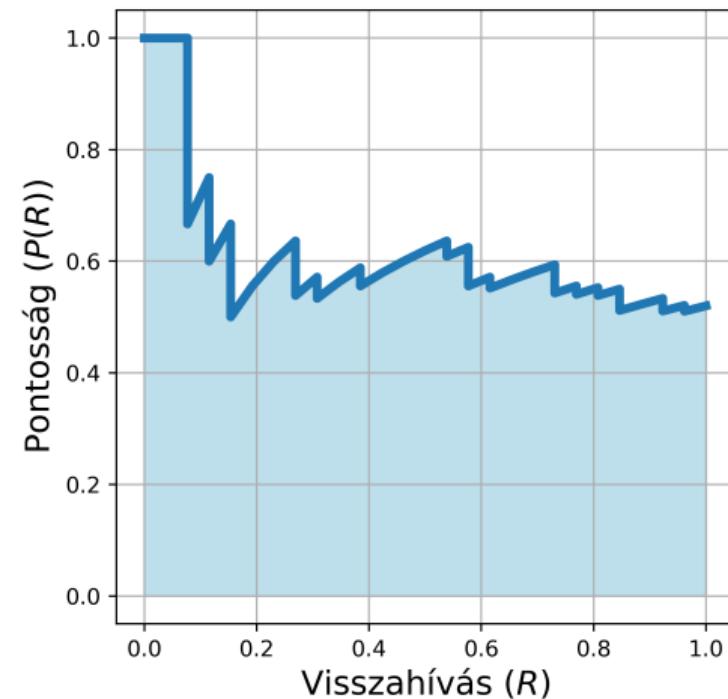
Az átlagos pontosság a pontosság-visszahívás görbe alatti terület.

## Átlagos pontosság (AP)

Az átlagos pontosság a pontossági értékek súlyozott átlaga minden pontossági küszöbértéken kiértékelve:

$$AP = \frac{1}{n} \sum_{R=0}^1 P(R)$$

Ahol  $n$  a mintavételi gyakoriság,  $P(\cdot)$  a pontosság és  $R$  a visszahívás értéke. Az  $R$  küszöbérték  $n$  db értéket vesz fel 0 és 1 között.



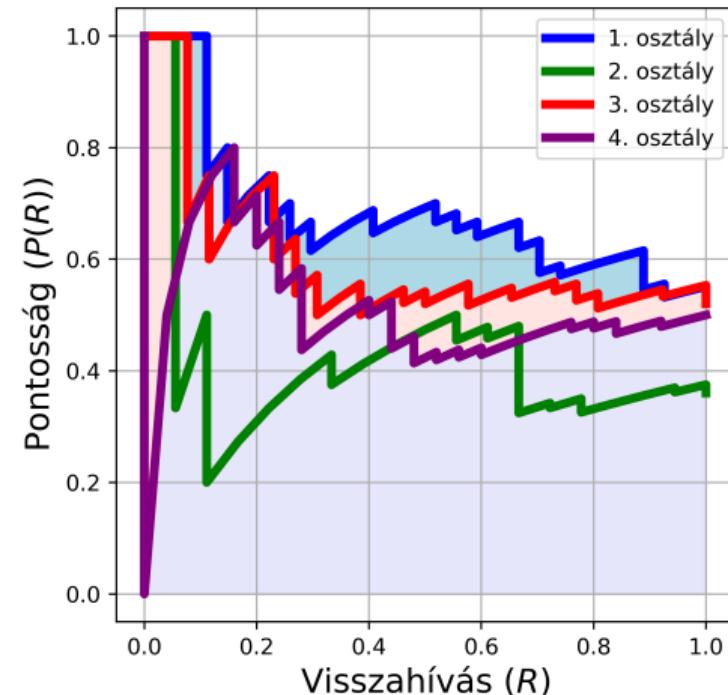
# Közepes átlagos pontosság

## Közepes átlagos pontosság (mAP)

A közepes átlagos pontosság az összes osztályra kiszámított átlagos pontosságok átlaga:

$$mAp = \frac{1}{N} \sum_{i=0}^{N-1} AP_i = \frac{1}{N} \sum_{i=0}^{N-1} \frac{1}{n} \sum_{R=0}^1 P_i(R)$$

Ahol  $N$  az osztályok számossága,  $n$  a küszöbértékeni mintavételek száma,  $AP_i$  az átlagos pontosság az  $i$ . osztályra.

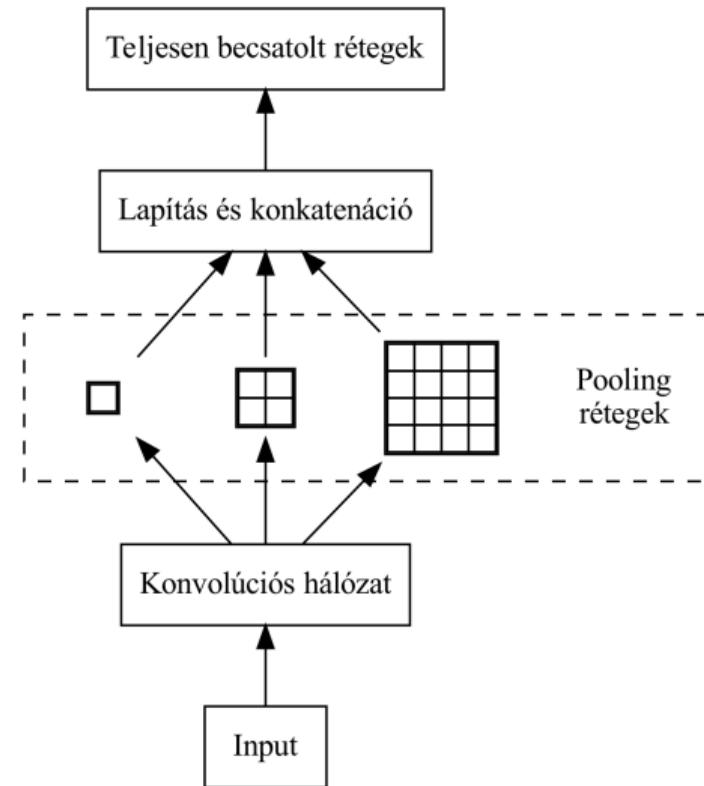


# Térbeli piramis lazítás

Térbeli piramis pooling (SPP) réteg az inputot több, különböző méretű max pooling réteggel dolgozza fel. Például ha érkezik egy konvolúciós rétegből valamilyen aktiváció feldolgozza egy  $1 \cdot 1$ ,  $2 \cdot 2$  és  $4 \cdot 4$  max pooling szűrővel is.

Ha a piramis pooling réteg után egy teljesen becsatolt réteg következik, a különböző pooling rétegek által adott outputot kilapítja, konkatenálja majd úgy adja tovább a következő rétegeknek.

Lapítás során az input mátrixból 1 dimenziós vektor lesz. Pl.  $4 \cdot 4 \rightarrow 16 \cdot 1$ .



1 Bevezetés

2 Objektum detekció

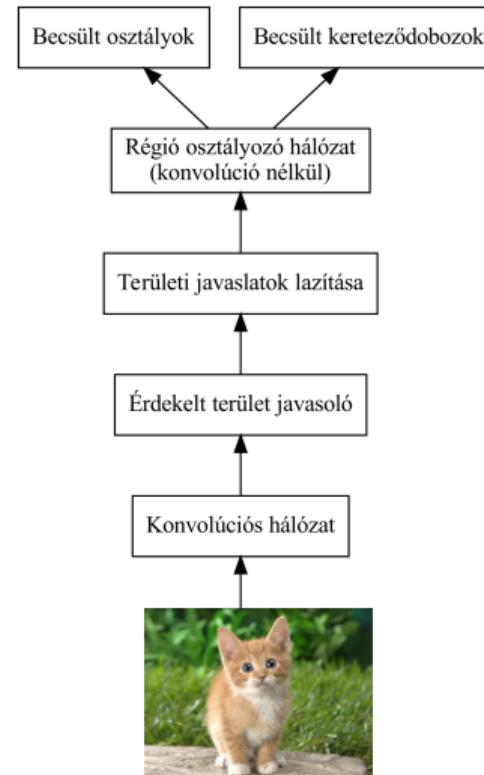
3 Tanítás

4 További architektúrák

# Fast R-CNN

Az eredeti R-CNN architektúra nagyon lassan futott, hiszen a sok műveletet igénylő konvolúciós szűrés minden javasolt érdekelt területre lefutott.

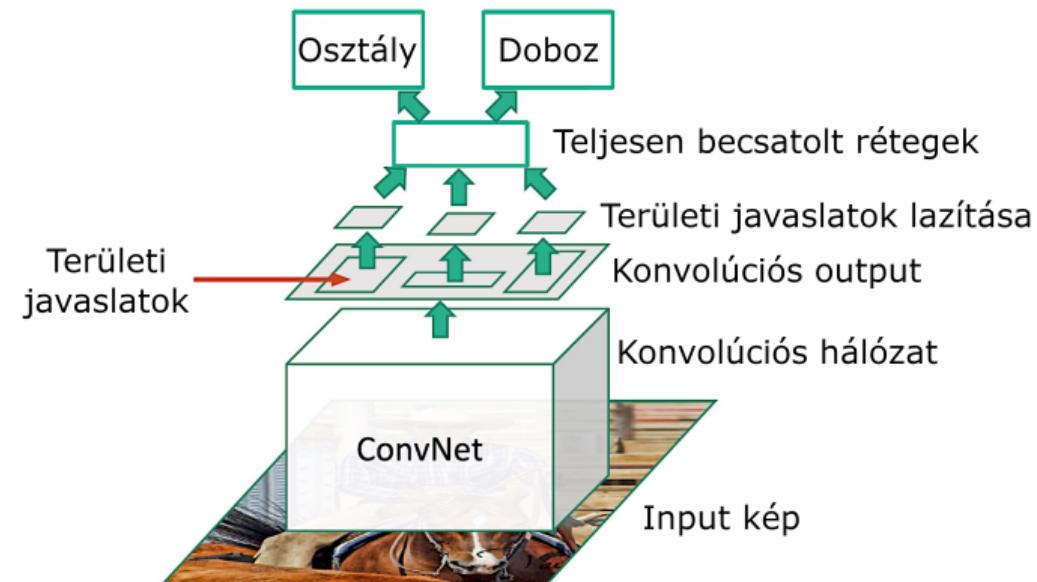
A Fast-RCNN az eredeti architektúrát úgy módosítja, hogy először a konvolúciós rétegeken áramoltatja át az input adatot, majd a konvolúciós rétegek outputján keresi meg az érdekelt területeket, amiket vetít és beosztályoz a régióalapú hálózattal.



# Fast R-CNN

Az eredeti R-CNN architektúra nagyon lassan futott, hiszen a sok műveletet igénylő konvolúciós szűrés minden javasolt érdekelt területre lefutott.

A Fast-RCNN az eredeti architektúrát úgy módosítja, hogy először a konvolúciós rétegeken áramoltatja át az input adatot, majd a konvolúciós rétegek outputján keresi meg az érdekelt területeket, amiket vetít és beosztályoz a régióalapú hálózattal.

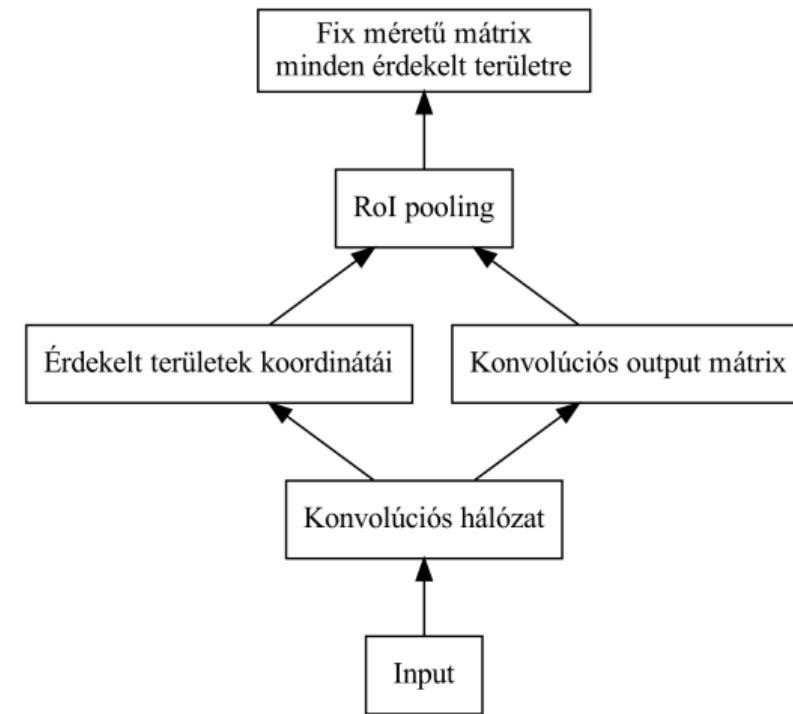


# Területi javaslatok lazítása (RoI pooling)

A területenkénti konvolúciós hálózatk **fix méretű inputot** várnak a korábbi rétegektől, viszont a területi javaslatok eltérő méretűek.

Ebből adódóan szükség van a területi javaslatok átméretezésére. A RoI pooling réteg feladata **fix méretű mátrixokká transzformálni** a bemeneti mátrixokat.

Például egy RoI pooling réteg kimenete bármilyen méretű mátrixra lehet  $224 \cdot 224$ .



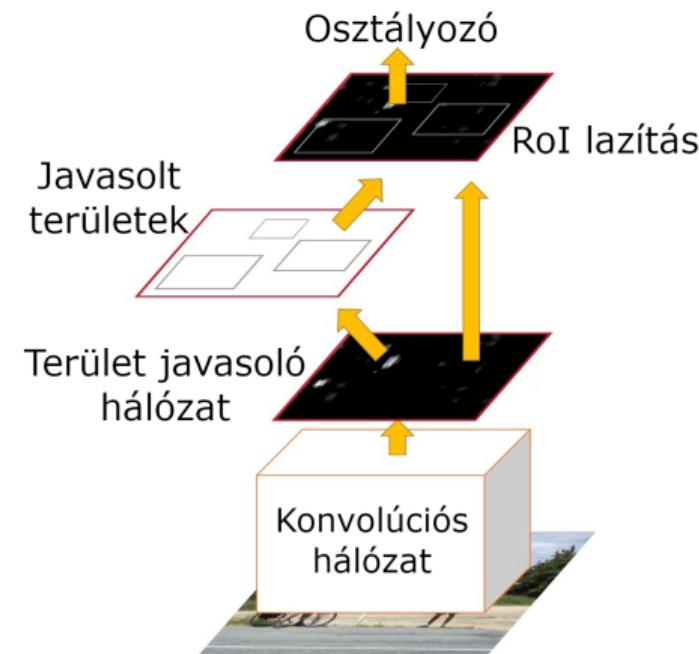
# Faster R-CNN

A Faster R-CNN architektúra a **területi javaslatokat egy neurális hálózat segítségével állítja elő (RPN)**.

Ebben az esetben a területjavasoló hálózatnak már **tanítható súlyai** vannak. A teljes hálózatot 4 különböző költséggel tanítódik.

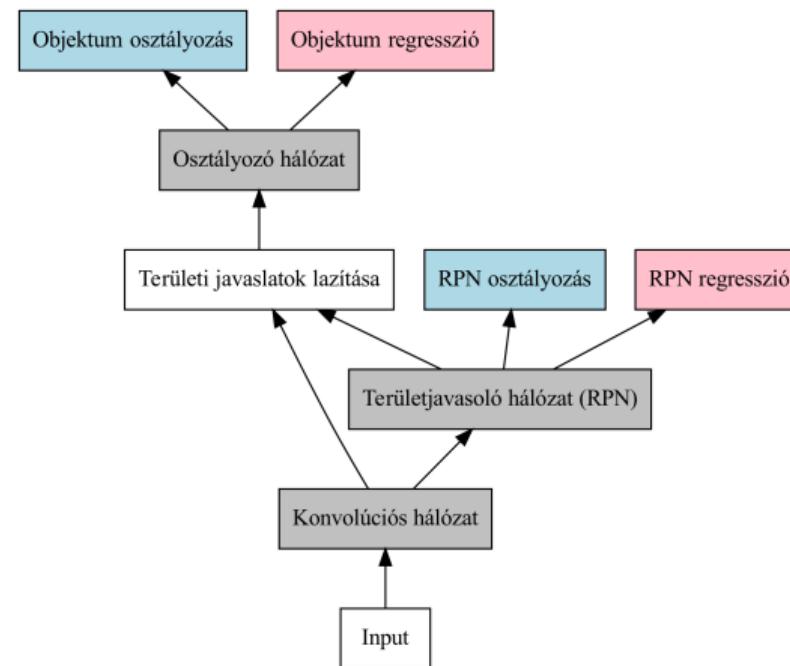
A Faster R-CNN egy kétlépéses objektum detektor:

- 1 Első lépés:** Képenként egyszeri futtatással lefut a gerinc hálózat és a területjavasoló hálózat.
- 2 Második lépés:** Régiónként egyszeri futtatással



# Faster R-CNN

- ① **RPN osztályozás:** A kereteződoboz objektum / nem objektum?
- ② **RPN regresszió:** A javasolt doboz és a kereteződoboz közötti transzformáció megbecslése.
- ③ **Objektum osztályozás:** A javaslatok osztályának megbecslése.
- ④ **Objektum regresszió:** A javasolt doboz és objektum doboz közötti transzformáció megbecslése.

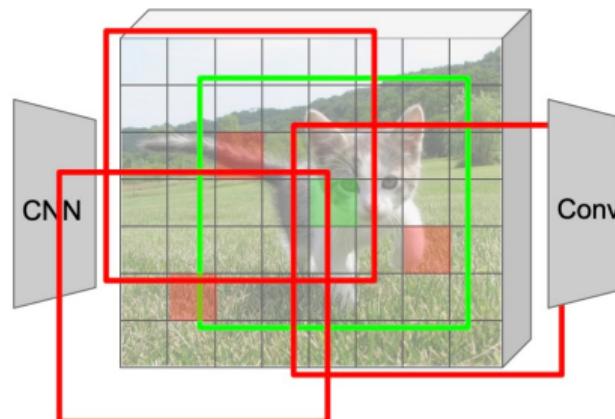


# Területjavasoló hálózat (RPN) - Osztályozás

A területjavasoló hálózat minden képpont körül egy rögzített méretű dobozt hoz létre, amiről eldönti, hogy **objektum-e** (bináris osztályozás).



Input kép



Képi jellemzők

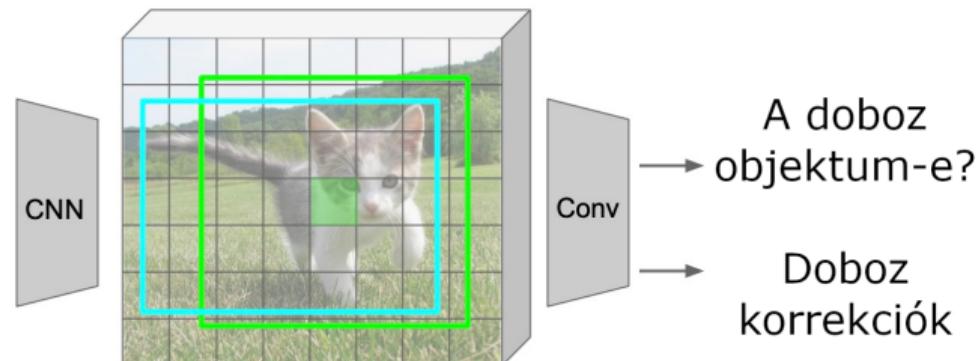
Minden dobozról eldöntení, hogy objektum-e

# Területjavasoló hálózat (RPN) - Osztályozás

Minden, az előző lépésben pozitívan osztályozott kereteződobozra a területjavasoló hálózat egy **korrekciót generál**, ami **módosíthat az eredeti kereteződoboz koordinátáin**, hogy jobban illeszkedjen a dobozban megtalálható objektumra.



Input kép



Képi jellemzők

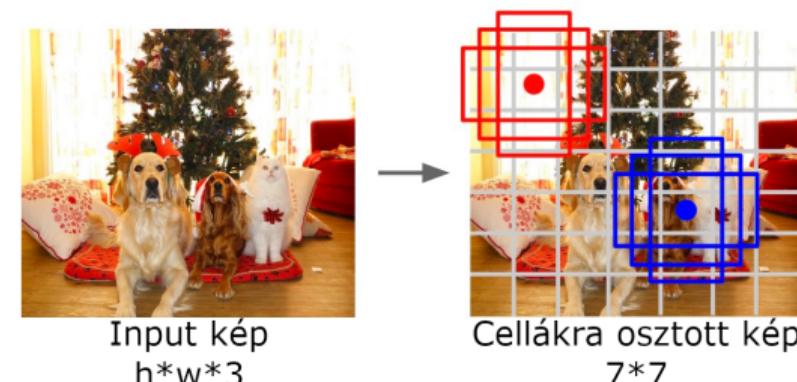
A doboz  
objektum-e?  
→ Doboz  
korrekciók

# Egylépéses detekció - YOLO architektúra

A YOLO (You Only Look Once) architektúra egy népszerű objektum detektor konfiguráció.

Ebben az esetben a hálózat az input képet **felosztja egy  $n \cdot n$  db cellából álló hálóra**. minden cellához hozzárendel  $B$  db **alapdobozt**, amelynek középpontja a cella közepe.

Minden cellára a hálóban megbecsül a  $B$  db alapdobozból 5 értéket:  $[dx, dy, dh, dw, p]$  ahol az első 4 érték az eltolást jelenti  $x, y, w, h$  dimenziók mentén,  $p$  pedig a softmax aktiváció minden osztályra, beleértve a háttérét is.



A YOLO hálózat outputja ebben az esetben  $7 \cdot 7 \cdot (5 \cdot B + C)$ , ahol  $C$  a lehetséges osztályok száma.