Félig árnyékos karakterek megtalálása, felismerése

Kis Ádám

Eötvös Loránd Tudományegyetem
Alkalmazott matematikus MSc.

Önálló projekt II, 2018

Feladatleírás

A probléma félig árnyékos rendszámtáblák leolvasása képről. Ezt két lépésből áll:

- rendszámtábla megtalálása a képen
- rendszámtábla elolvasása szövegfelismeréssel

Én az első részre koncentráltam. Továbbra is mélytanulásos módszereket próbáltam.

Eredeti példahalmaz:

- 500 kép
  - fekete fehér és színes
  - különböző helyekről
  - különböző fajta rendszámtáblák

Példahalmaz

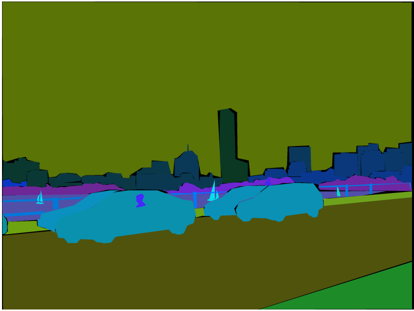
Példahalmaz

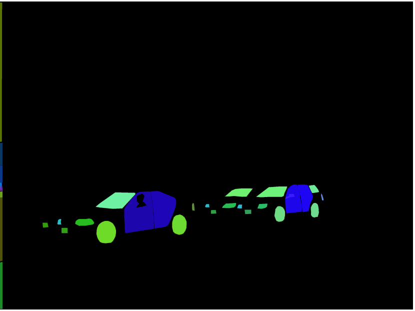
Példahalmaz

Probléma: kis példahalmaz - gépi tanulási módszerek általában sokat igényelnek
Több példa kell. Ingyenesen - és könnyen - elérhető felcímkézett adathalmaz kevés:

- Brazis rendszámtábla adatbázis: http://www.ssig.dcc.ufmg.br/
  - E-mail-ben kell elkérni
  - Még nem válaszoltak
- Ade20k - MIT: http://groups.csail.mit.edu/vision/datasets/ADE20K/
  - szegmentációs adatbázis
  - nem csak rendszámtáblák

Ade20k





Ade20k

- Nem kifejezetten erre a feladatra van
  - Kis rendszámtáblák
  - Nincs mind felcímkézve
- Azonos felbontás: 1600x1200
  - Túl nagy egy hálónak, de átméretezve alig látszik a rendszámtábla - ez hátrány, de előny is, rákényszerítheti a hálót, hogy más szempontok alapján találja meg
    - autó pozíciója
    - út alakja

Súly inícializáció

Hogyan inícializáljuk a háló súlyait? Én két módszert implementáltam le:

- Glorot (vagy Xavier)
- ortogonális

Glorot a legáltalánosabban használt.

Glorot inícializáció

- Eredetileg szigmoid aktivációs függvényre találták ki
- Adaptálható Leaky ReLU-ra is

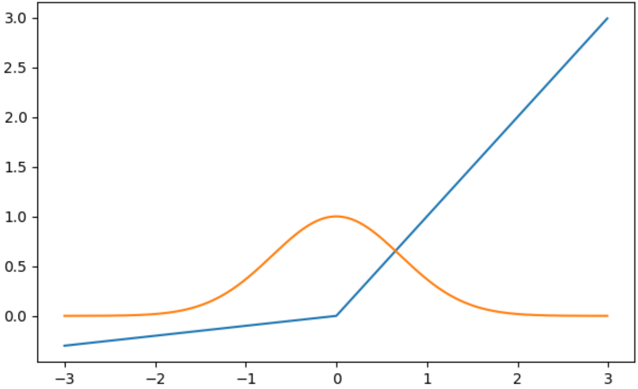
Az elv a háló súlyainak varianciáját normalizálni

- szigmoid esetén a szaturáció elkerülésére - most már inkább csak LSTM hálókban
- ReLU esetén az eltűnő gradiens ellen
    - Az ideális ha egy réteg kimenő éleihez tartozó súlyok összegének varianciája 1
        - Eredetileg a bemenő élek és kimenő élek összegének átlaga volt normalizálva
        - Függetlenséget feltételez a súlyok között

$$w_i \sim \mathcal{N}(0, \frac{2}{1.1n})$$

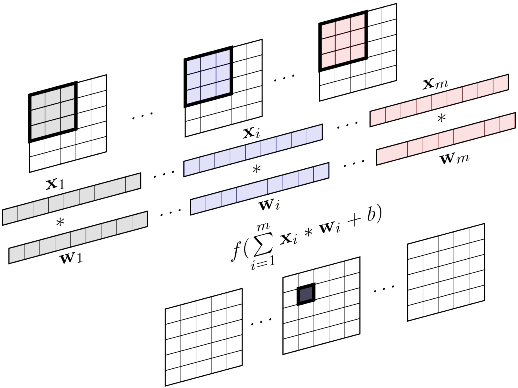
Várható érték

A ReLU (és Leaky ReLU) 0-ban szeparál



Konvolúció mint skaláris szorzat

A konvolúció (keresztkorreláció) vektorok skaláris szorzata



Ortogonális inícializáció

Lineárisan összefüggő konvolúciós kernelek nem jók
Ha x_1 és x_2 két neuron kimenetei és $x_2 = \lambda x_1$

$$w_1 x_1 + w_2 x_2 = (w_1 + \lambda w_2) x_1$$

Normál eloszlású változók csak megközelítőleg ortogonálisak.
Megoldás:

1. generáljunk egy normál eloszlású mátrixot (ez általában téglalap)
2. SVD-vel tegyük merőlegessé (amennyire lehet)
3. ez után varianciát korrigálni kell

$$A \leftarrow \frac{A}{\sqrt{\mathrm{Var}[A]}} \sqrt{\frac{2}{1.1n}}$$

Előre inícializálás

Alsó rétegeket szokták előre inícializálni

- Hasonló problémán betanított háló alapján

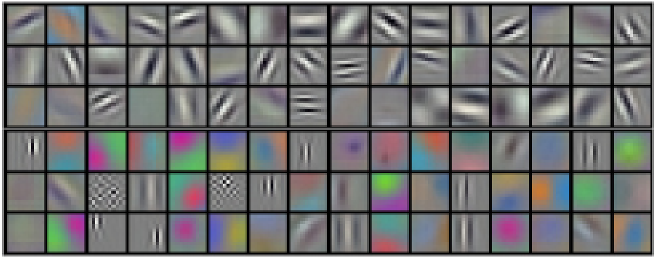
Én valamivel más megoldást próbáltam ki (csak első rétegre):

- élkeresők
- pontkeresők
- Gábor-wavelet
- És persze pár ortogonális kernel

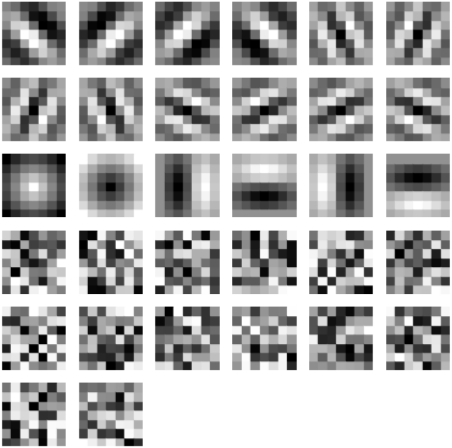
Motiváció:

- Klasszikus képfeldolgozó módszereknél már beváltak
  - De: itt tanítható
- AlexNet első rétege

AlexNet első rétege



Előre inícializált első réteg



Eredmény

Nem igazán vált be

- Drága kiszámolni: legalsó réteg bemenete a legnagyobb
- Ezt a számítási kapacitást praktikusabb több rétegre költeni
- Fast YOLO is inkább 3x3-as első réteget használ (normál YOLO 7x7-eset)

YOLO architektúra

Motiváció:

- Több rendszám / kép
- Cutting edge

Architektúra

- Klasszikus AlexNet 1x1-es konvolúciókkal kiegészítve
- Igazi újdonság a veszteségfüggvény

YOLO veszteségfüggvény

$$
\begin{aligned}
\text{loss} = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{\text{obj}} (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \\
+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{\text{obj}} ((\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2) \\
+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{\text{obj}} (c_i - \hat{c}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{\text{noobj}} (c_i - \hat{c}_i)^2 \\
+ \sum_{i=0}^{S^2} 1_{i,j}^{\text{obj}} \sum_{j=0}^{B} (p_i(c) - \hat{p}_i(c))^2
\end{aligned}
$$

YOLO veszteségfüggvény

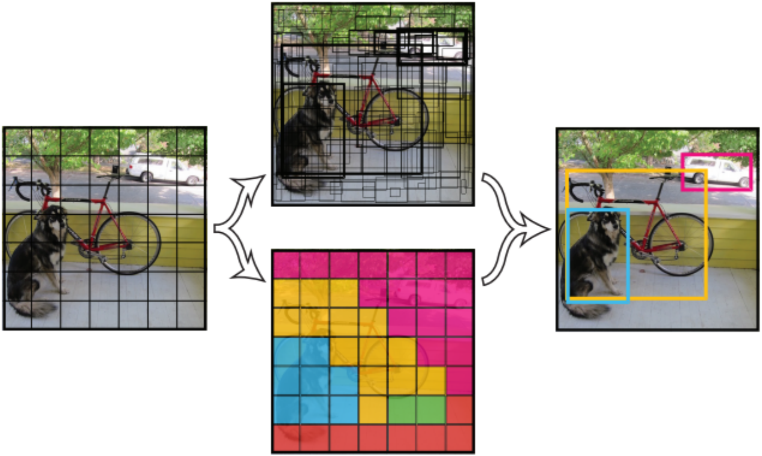
5 részből áll:

- Középpont
- szélesség / magasság
- "van objektum"
- "nincs objektum"
- Osztály

A "van objektum" és "nincs objektum" kivételével mind annak a cellának a felelőssége amelyikben az objektum közepe van.

YOLO veszteségfüggvény

Cellánkénti kimenet: $[x, y, w, h, p, c_1, c_2, \ldots]$



YOLO veszteségfüggvény

- Középpont

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{\text{obj}} (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2$$

- szélesség / magasság

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{\text{obj}} ((\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2)$$

- "van objektum"

$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{\text{obj}} (c_i - \hat{c}_i)^2$$

YOLO veszteségfüggvény

- "nincs objektum"

$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{\text{obj}} (c_i - \hat{c}_i)^2$$

- Osztály

$$\sum_{i=0}^{S^2} 1_{i,j}^{\text{obj}} \sum_{j=0}^{B} (p_i(c) - \hat{p}_i(c))^2$$

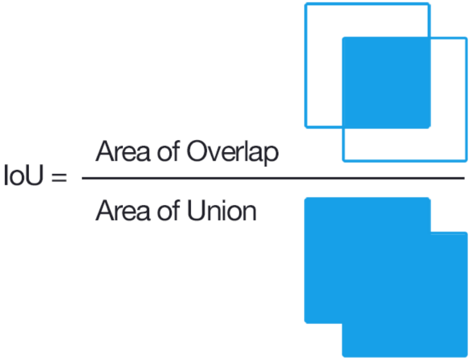
Non-max suppression

Egy tárgyat többször is megtalál. Megoldás: non-max suppression.

- Lenagyobb bizonyosságú kimenetet vesszük
- Ha valamelyik másik kimenet IoU-ja nagyobb, mint α, akkor eldobjuk
- Kivesszük a következő legnagyobb bizonyosságú kimenetet
- Addig folytatjuk amíg elfogynak

IOU

Intersection over union



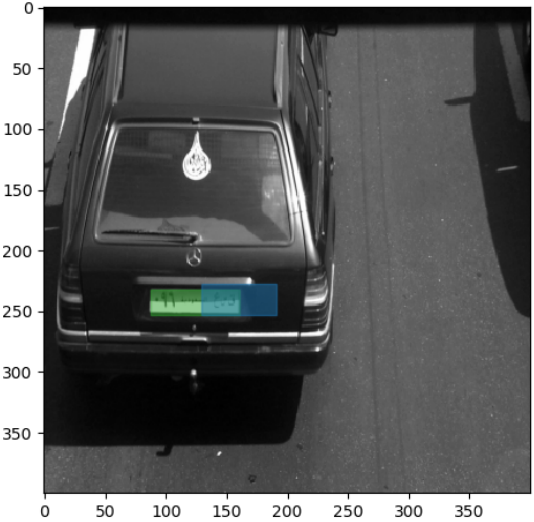
Eredmény

A háló betanulása nem fejeződött be

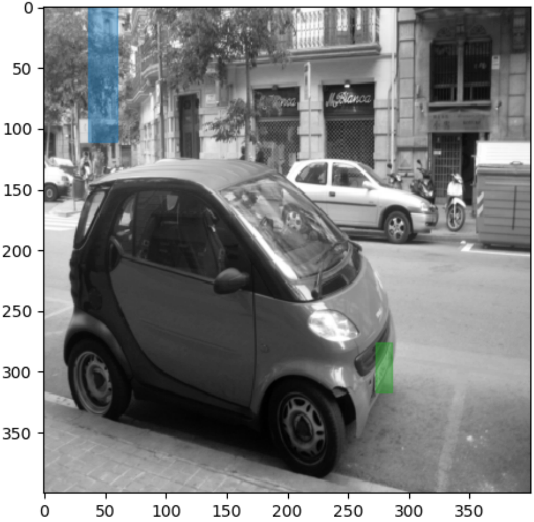
- YOLO veszteségfüggvény miatt négyszer lassabb betanulás
  - Ezt optimalizációval levittem háromszorosra
- Új adathalmaz
  - Fel kellett venni a felbontást
  - Gyakorlatilag két különböző feladatra tanult

Ezek miatt a betanítás és a hiperparaméter optimalizáció is legalább háromszor annyi ideig tart.

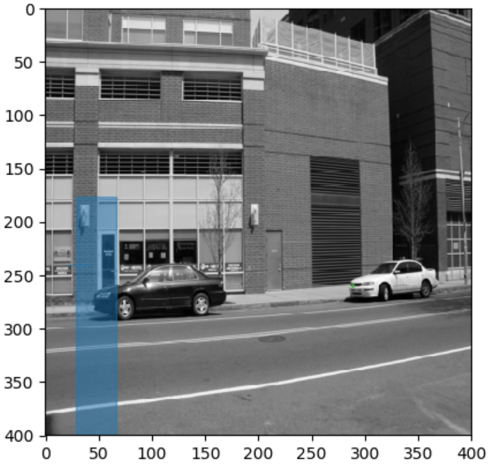
Eredmény



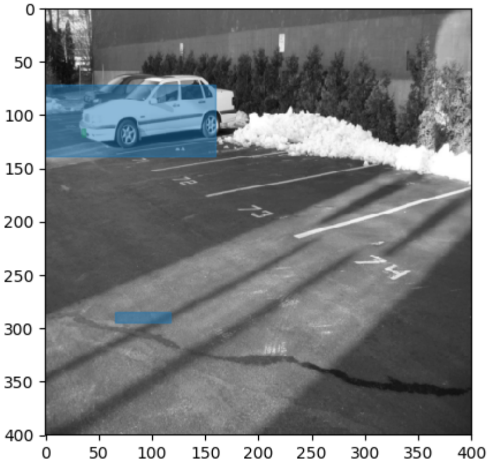
Eredmény



Eredmény



Eredmény



További irány

- Legegyszerűbb: csúszóablakos megoldás (esetleg képpiramissal)
- Előfeldolgozás: pl élkeresővel
- Pásztázó vonal: pl. 3x1-es konvolúció
- Előtanítás hasonló példahalmazon
- Sample augmentation
- R-CNN