

Vedere Artificială (Computer Vision)

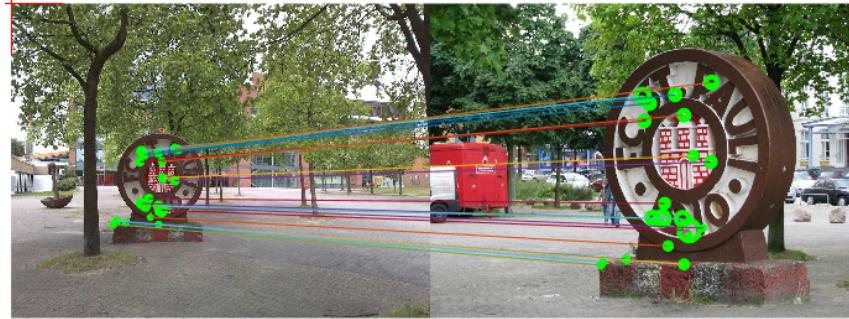
Bogdan Alexe

bogdan.alexe@fmi.unibuc.ro

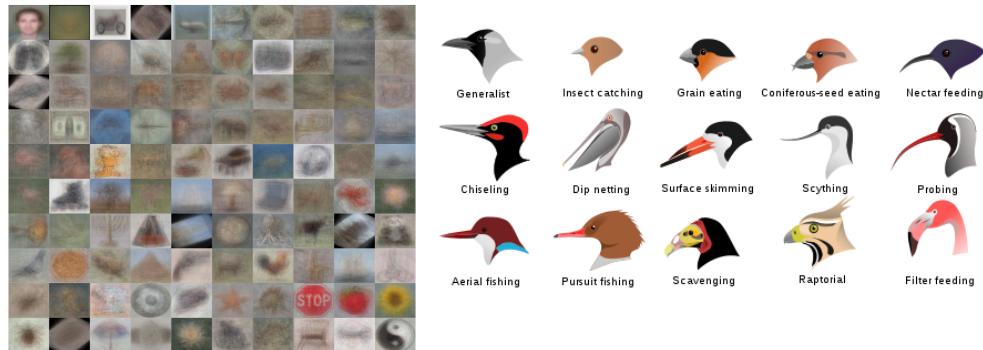
anul 2, master Informatică, semestrul I, 2019-2020

Recapitulare – cursul trecut

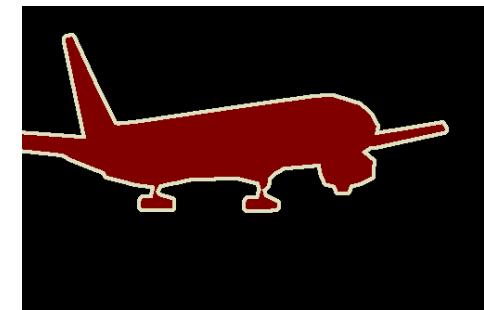
- Prezentare tema 2



- Clasificare în Vederea Artificială



- Detectarea de obiecte în imagini

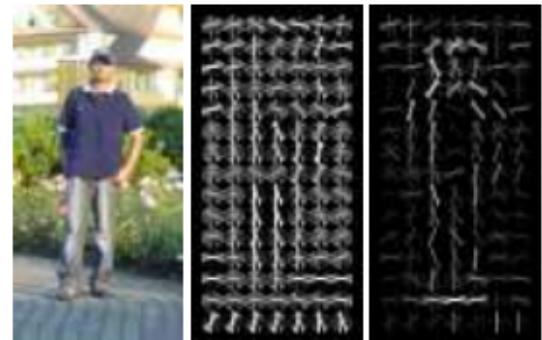


Cursul de azi

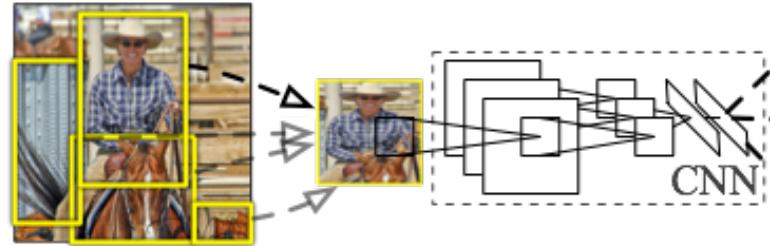
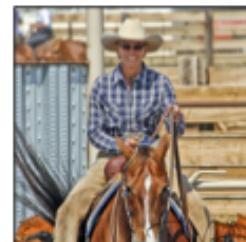
- Glisarea unei ferestre



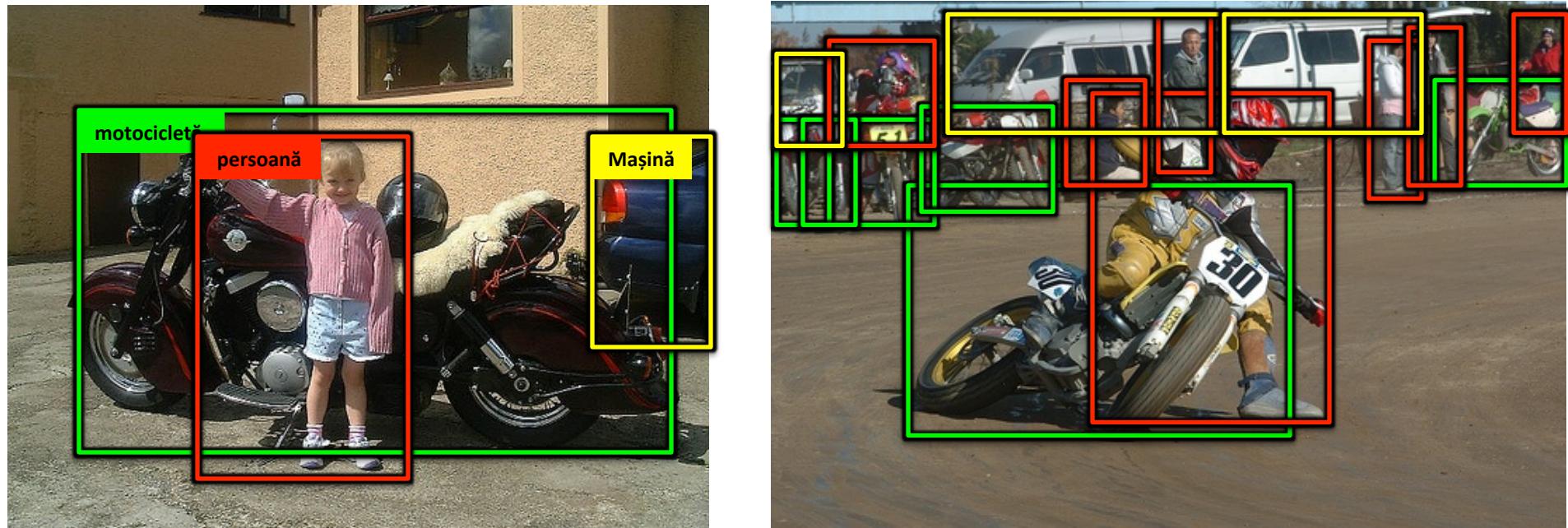
- HOG pentru detectarea oamenilor, detectare facială



- Detectarea de obiecte în imagini cu R-CNN



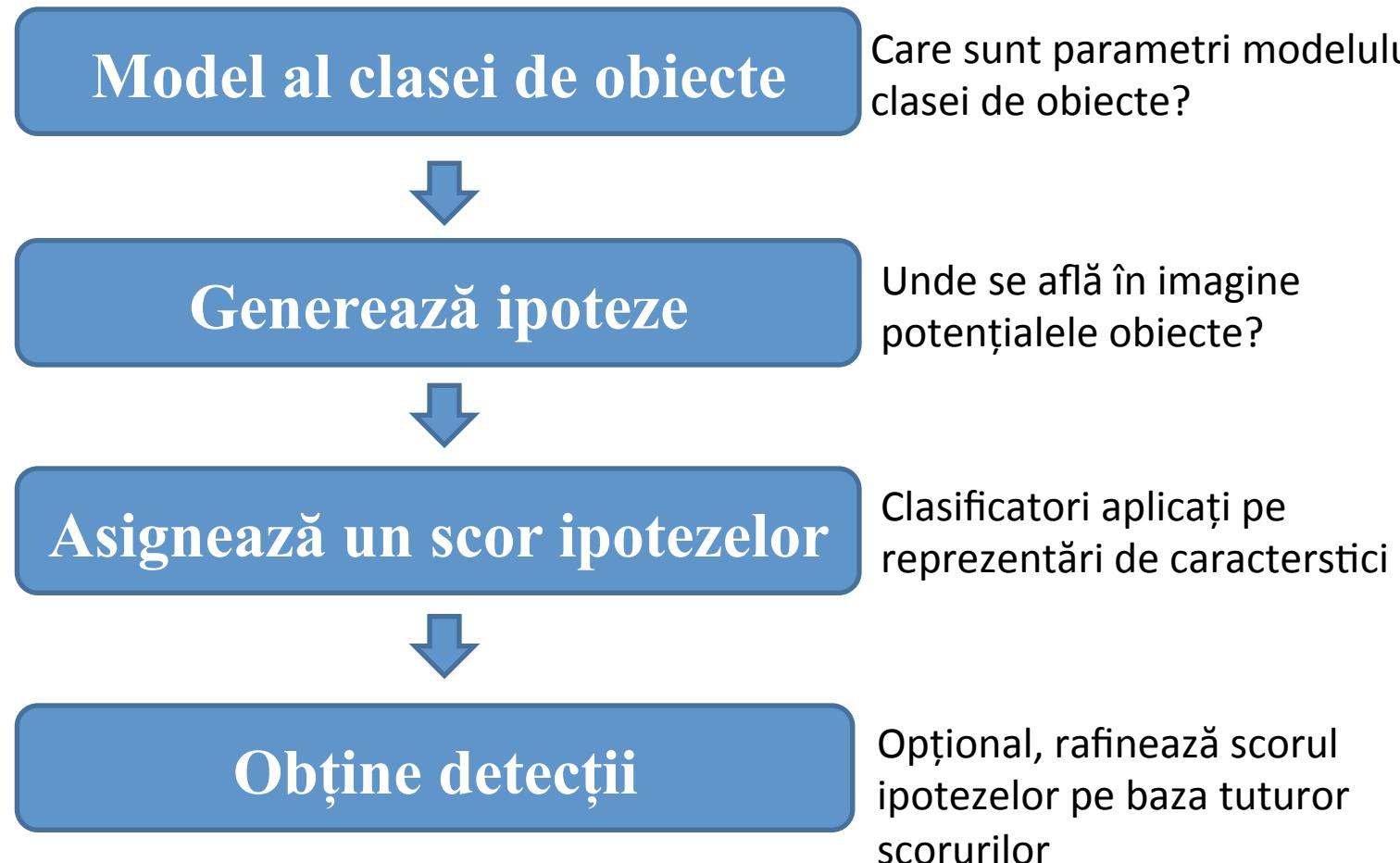
Detectarea de obiecte



Problemă – detectează toate instanțele unei clase specifice de obiecte

Dificultăți – variația intra-clasă în înfățișare, mărime, postură, unghiul camerei, iluminarea scenei, mascare, background aglomerat

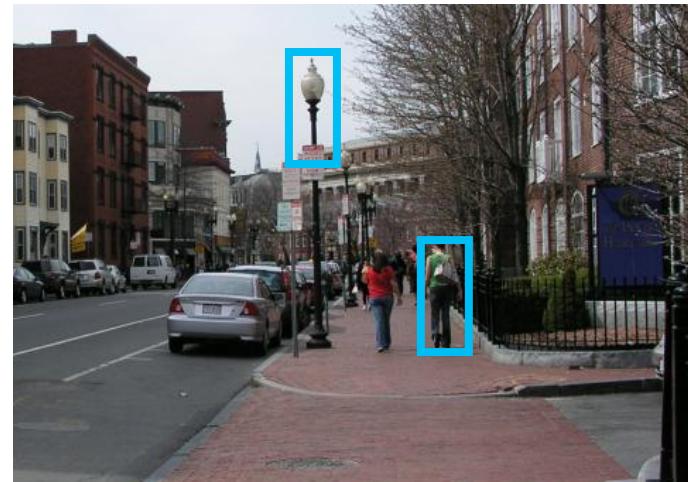
Detectarea de obiecte – rețetă generală



Etapele de bază în detectarea de obiecte

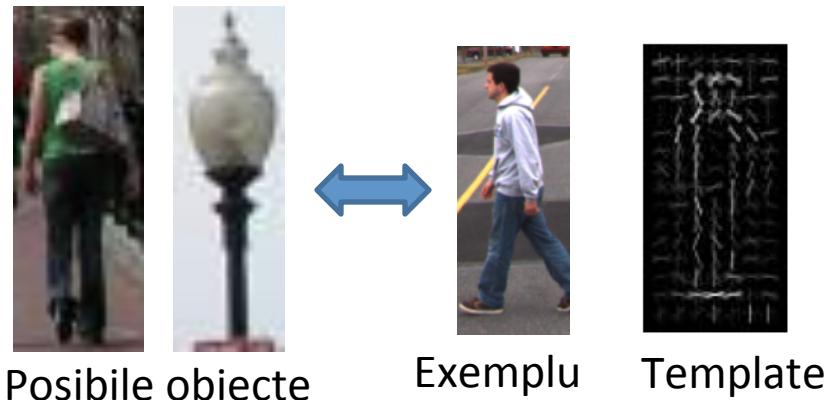
1. Generează ipoteze

- regiune vs ferestre
- poziție, mărime, orientare
- complexitate crescută



2. Comparare

- calculează similaritatea față de un exemplu sau față de un template
- modelarea diferențelor în înfățișare



Glisarea unei ferestre



Fiecare fereastră este clasificată separat



Asociază un scor ipotezelor



$$+3 \text{ } +2 \text{ } -2 \text{ } -1 \text{ } -2.5 = -0.5 > 7.5 \quad ?$$

Non-obiect



$$+4 \text{ } +1 \text{ } +0.5 \text{ } +3 \text{ } +0.5 = 10.5 > 7.5 \quad ?$$

Obiect

Dificultăți

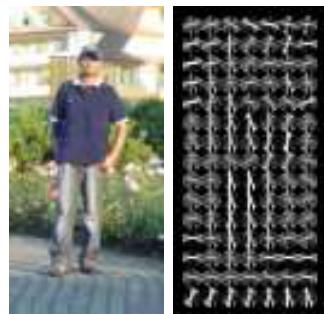
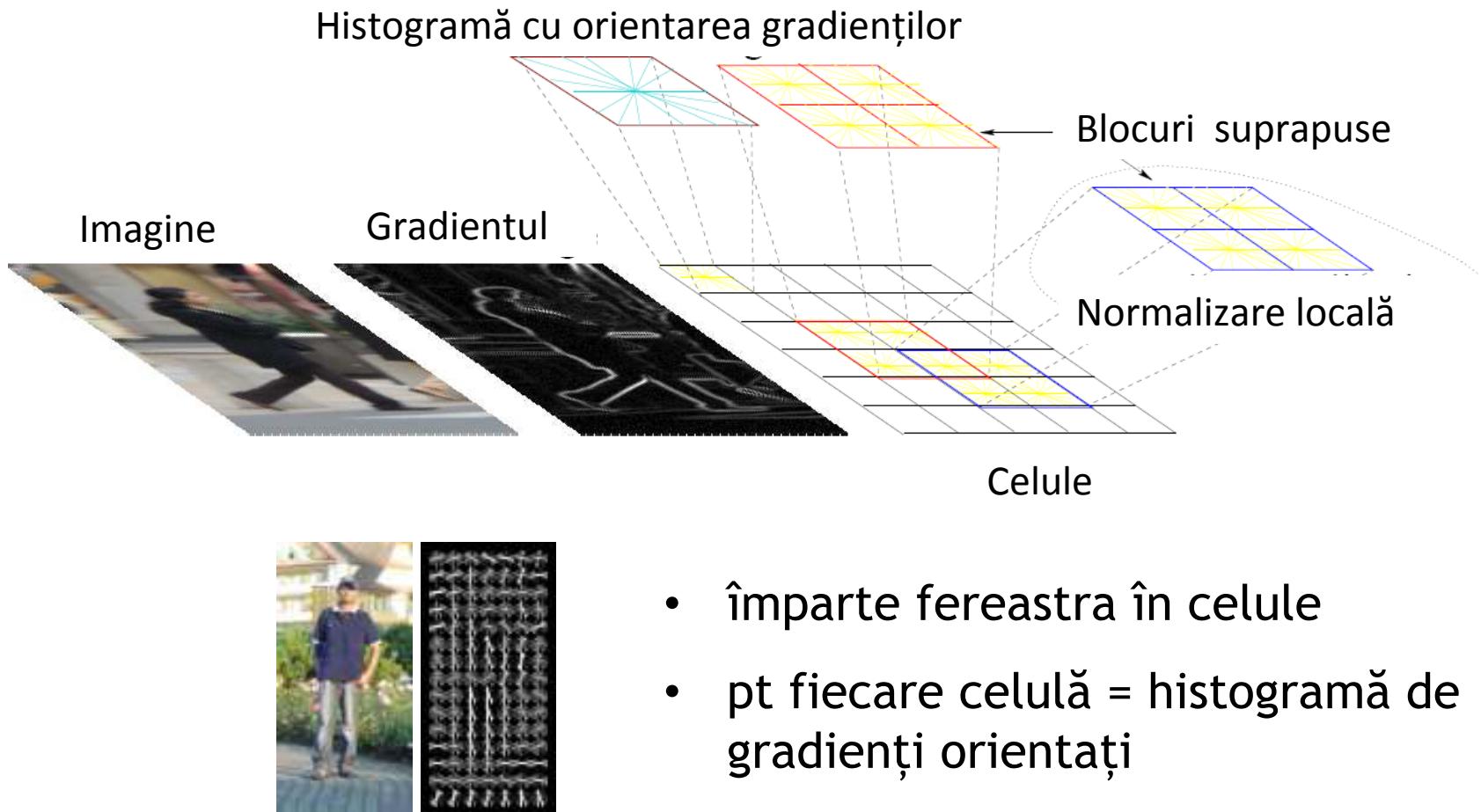
- Căutare eficientă a obiectelor în imagine:
 - glisarea unei ferestre necesită evaluarea a milioane de ferestre într-o imagine (poziție, mărime, orientare)
- Caracteristici optime?
 - cum modelăm variabilitatea claselor de obiecte în înfațisare
- Robustete la unghiuri diferite ale camerei
 - modele diferite pentru unghiuri diferite
- Detalii de implementare
 - dimensiunea unei ferestre
 - aspect ratio = proporție = lățime/înălțime
 - translație/mărime
 - suprimarea non-maximelor

HOG pentru detectarea oamenilor



1. Extragă o fereastră de dimensiuni 128x64 pixeli la fiecare punct și scală din imagine.
2. Calculează HOG (histogramă de gradienți orientați) pentru fiecare fereastră.
3. Asociază un scor acestei ferestre pe baza clasificatorului liniar învățat
4. Realizează suprimarea non-maximelor pentru eliminarea detecțiilor ce se suprapun și au un scor mai mic.

Reprezentări bazate pe gradienți: histograme de gradienți orientați (HOG)



Navneet Dalal and Bill Triggs,
Histograms of Oriented Gradients for Human Detection, CVPR05

<https://www.youtube.com/watch?v=7S5qXET179I>

<http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf>

Slide adaptat după K. Grauman

HOG pentru detectarea oamenilor



HOG pentru detectarea oamenilor

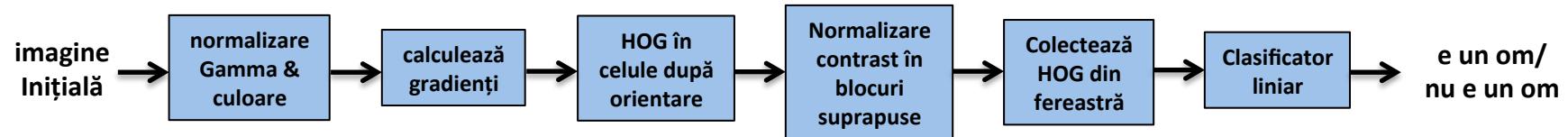


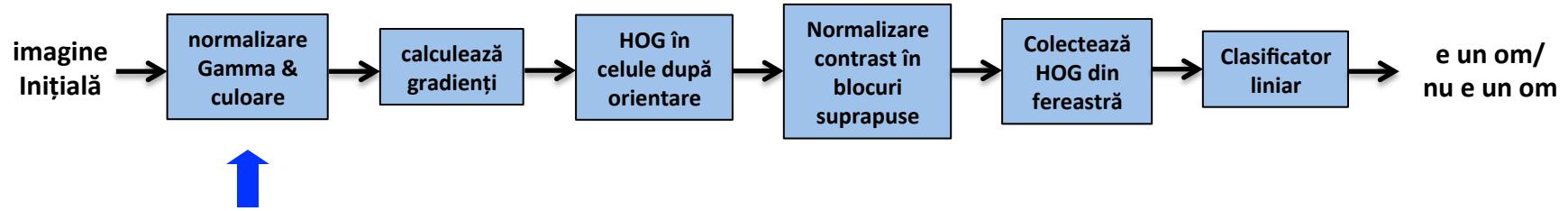
- redimensionează fereastra la 128×64 pixeli
- împarte imaginea în celule, fiecare celulă are 8×8 pixeli
- calculează gradientul imaginii (pentru fiecare pixel avem orientare și magnitudine)
- pentru fiecare celulă calculează o histogramă de gradienți orientați
- grupează celule în blocuri
- HOG = descriptor obținut prin concatenarea histogramelor a 105 blocuri

Navneet Dalal and Bill Triggs,
Histograms of Oriented Gradients for Human Detection, CVPR05

<https://www.youtube.com/watch?v=7S5qXET179I>
<http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf>

Slide adaptat după K. Grauman





- Testat cu

— RGB

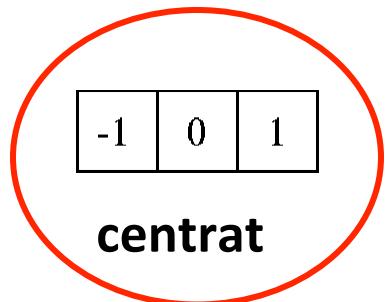
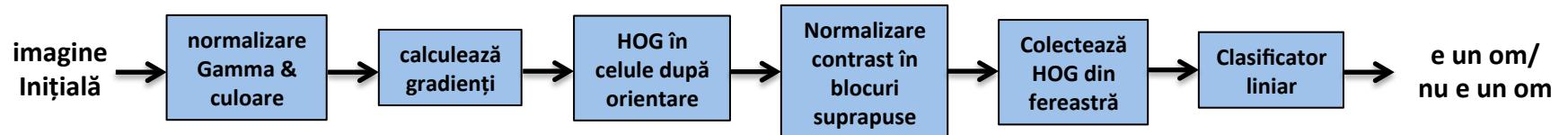
– tonuri de gri

} performa

A young boy stands in front of a house, wearing a purple long-sleeved shirt, blue jeans, and brown boots. He has short dark hair and is looking towards the camera.

A black and white photograph of a man standing outdoors. He is wearing a dark, long-sleeved shirt or jacket over a light-colored shirt, and dark shorts. He is holding a long, thin object, possibly a cane or a walking stick, in his right hand. The background shows some foliage and what might be a building or a bridge structure above him.

- Normalizare Gamma (transformarea valorilor intensităților/colorilor)
 - fără normalizare
 - rădăcină pătrată
 - logaritm

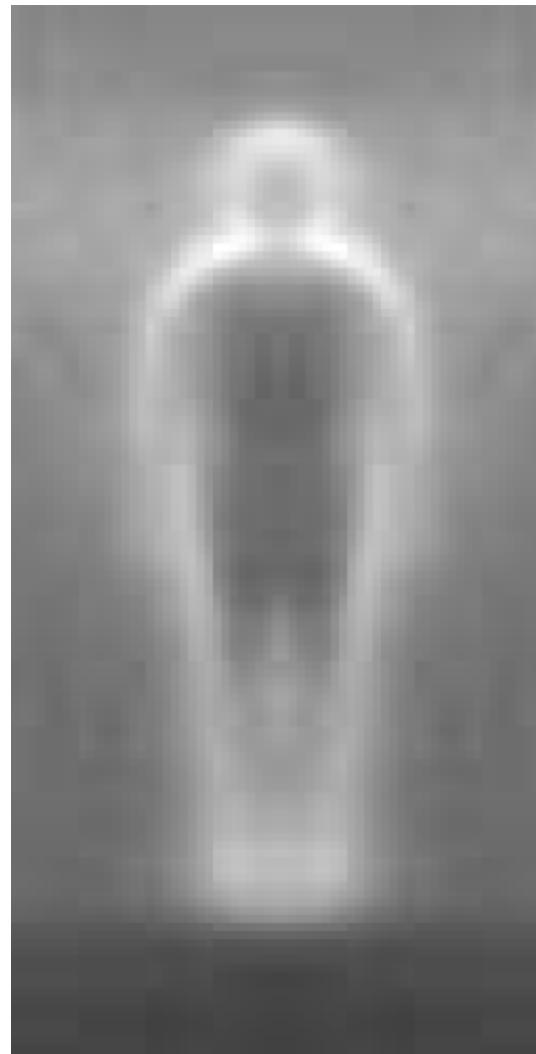


-1	1
----	---

necentrat

1	-8	0	8	-1
---	----	---	---	----

cubic corectat

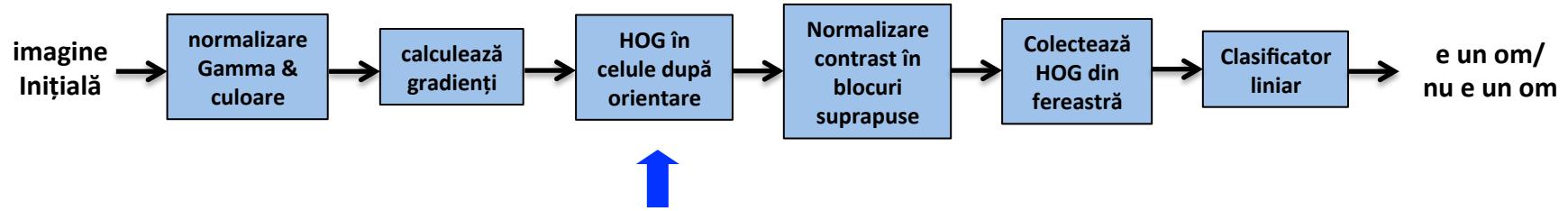


0	1
-1	0

diagonal

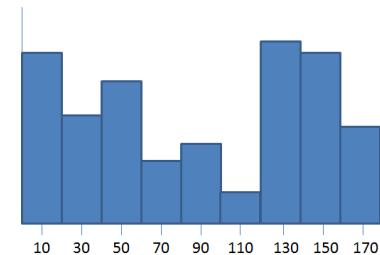
-1	0	1
-2	0	2
-1	0	1

Sobel

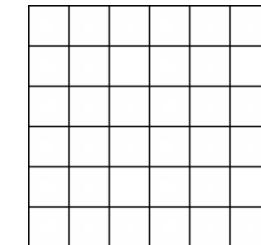


- histogramme de gradients orientés

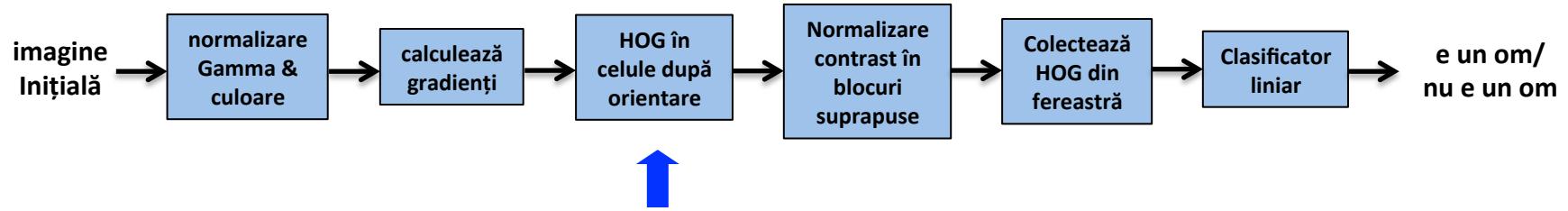
- după orientare: 9 intervale pentru orientări între $[0^\circ, 180^\circ]$



- după celule

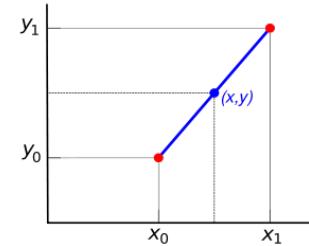
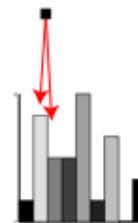


- contribuția fiecărui pixel direct proporțională cu magnitudinea gradientului

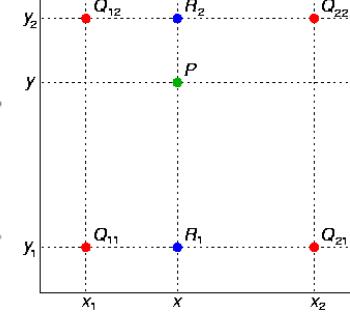
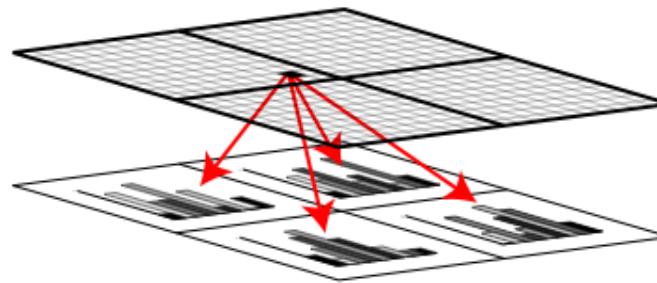


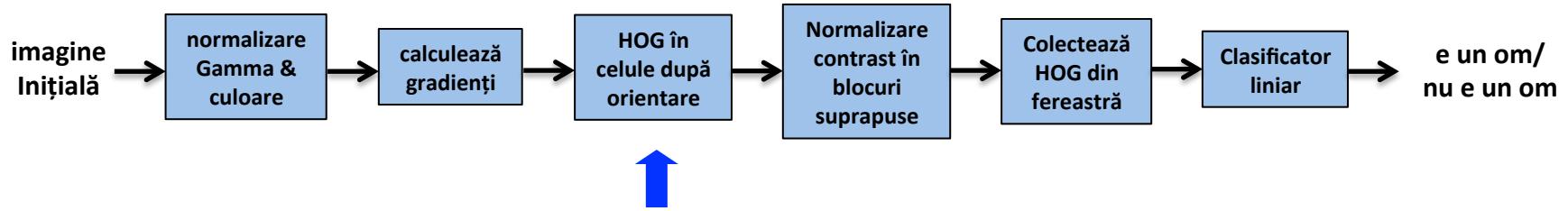
- histograme de gradienții orientați
 - interpolare triliniară:

- liniară după orientare



- biliniară după celule



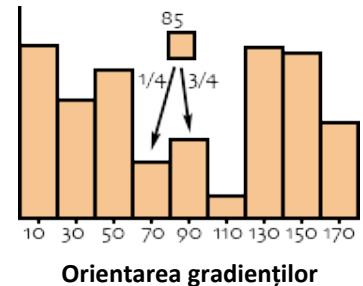


- exemplu de interpolare

- interpolare triliniară:

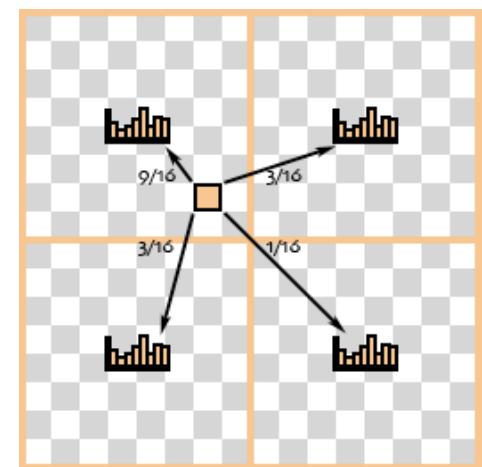
- liniară după orientare

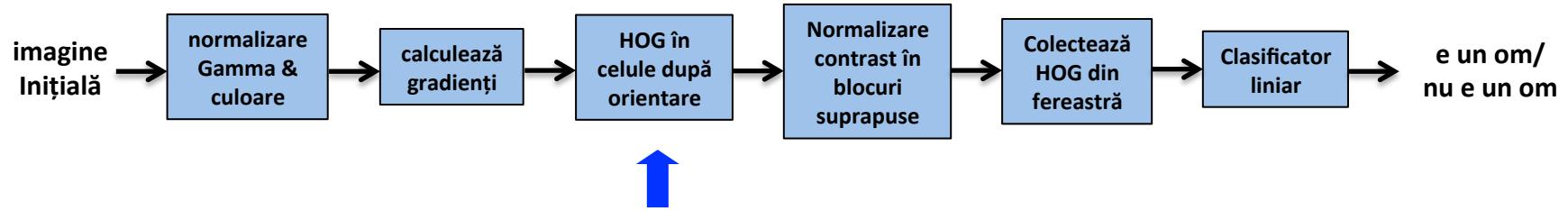
- $\Theta = 85^\circ$
 - distanțe față de centri $70^\circ, 90^\circ$ sunt de $15^\circ, 5^\circ$
 - $5/20 = \frac{1}{4}$ contribuție în intervalul cu centru 70°
 - $15/20 = \frac{3}{4}$ contribuție în intervalul cu centru 90°
 - ‘soft-assignment’



- biliniară după celule

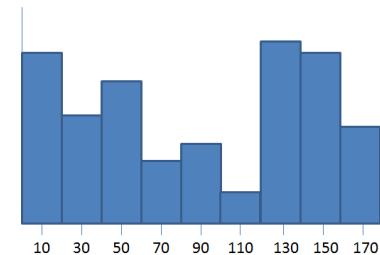
- distanțe față de centri celulelor: stânga – 2, dreapta – 6, sus – 2, jos -6
 - contribuții parțiale: $6/8 = \frac{3}{4}$ - stânga, $2/8 = \frac{1}{4}$ dreapta
 - contribuții parțiale: $6/8 = \frac{3}{4}$ - sus, $2/8 = \frac{1}{4}$ jos
 - contribuții totale:
 - $6/8 * 6/8 = 9/16$ stânga – sus
 - $6/8 * 2/8 = 3/16$ stânga – jos
 - $2/8 * 6/8 = 3/16$ dreapta – sus
 - $2/8 * 2/8 = 1/16$ dreapta -jos



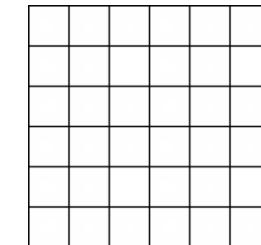


- histogramme de gradients orientés

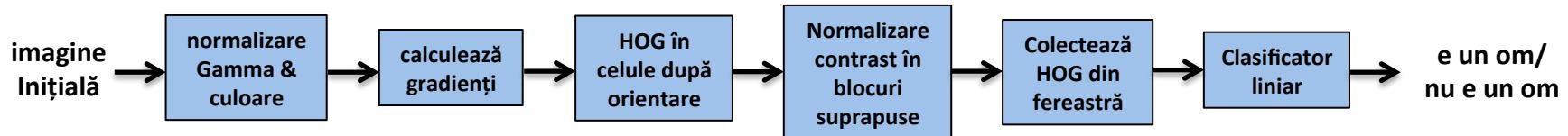
- după orientare: 9 intervale pentru orientări între $[0^\circ, 180^\circ]$



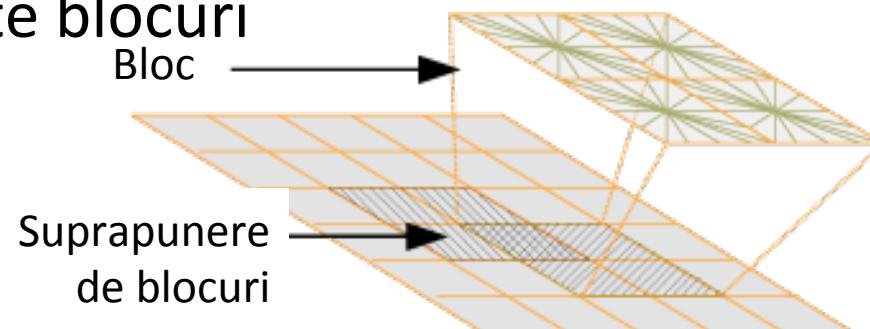
- după celule



- contribuția fiecărui pixel direct proporțională cu magnitudinea gradientului



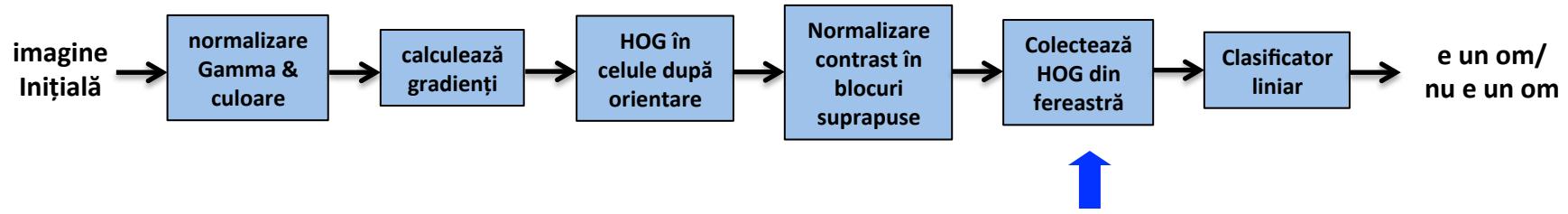
- bloc = grup de 2x2 celule
- histograma blocului = concatenarea histogramelor celor 4 celule componente (histogramele sunt normalizează în funcție de bloc)
- o celulă face parte din mai multe blocuri
- normalizare
 - diferite normalizări posibile



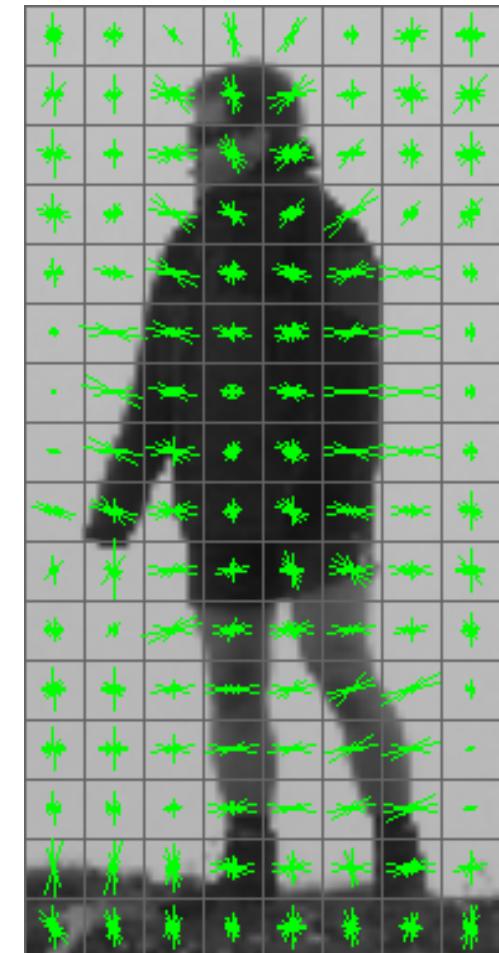
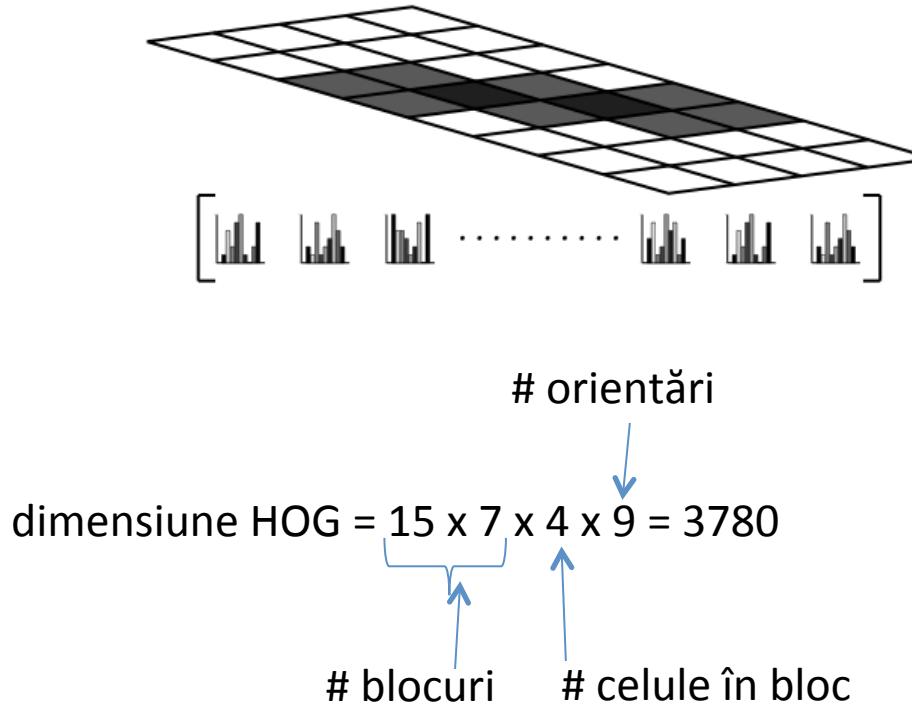
$$\mathbf{v} \rightarrow \frac{\mathbf{v}}{\|\mathbf{v}\|_1 + \epsilon}$$

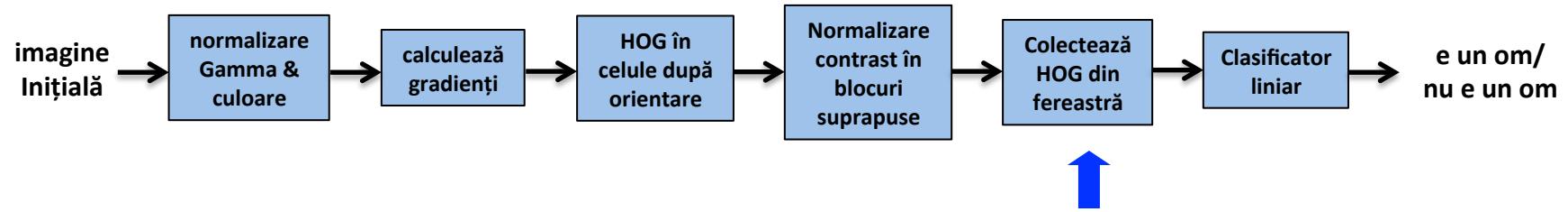
$$\mathbf{v} \rightarrow \frac{\mathbf{v}}{\sqrt{\|\mathbf{v}\|_2^2 + \epsilon}}$$

$$\mathbf{v} \rightarrow \sqrt{\frac{\mathbf{v}}{\|\mathbf{v}\|_1 + \epsilon}}$$

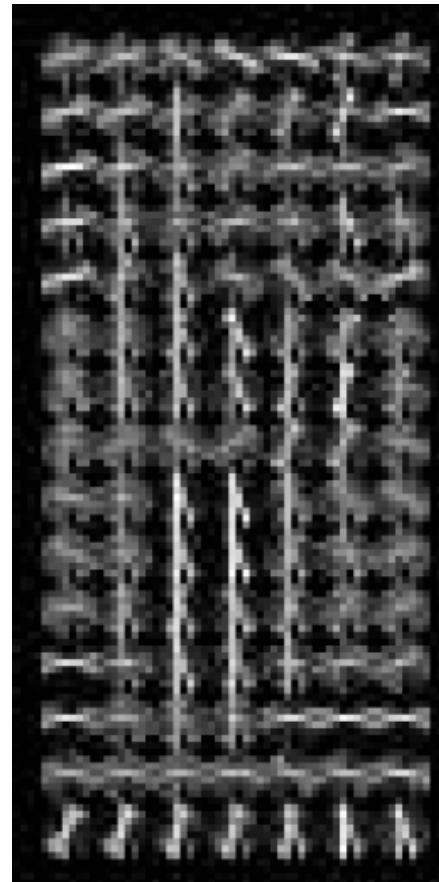
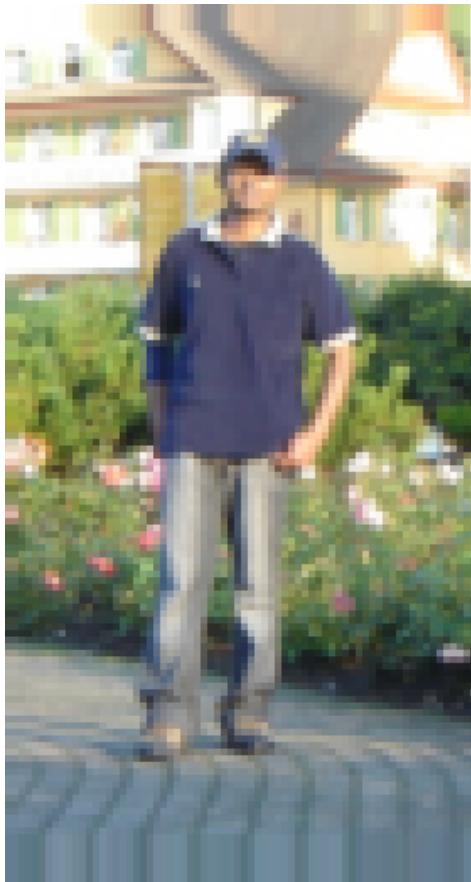


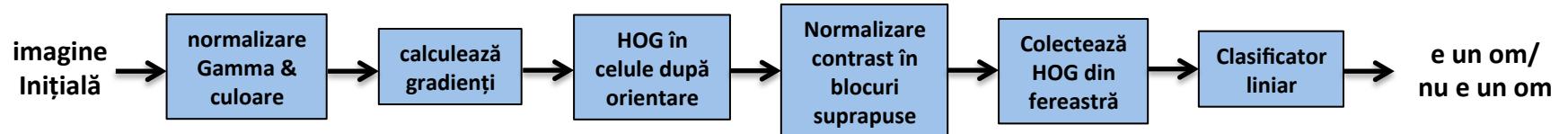
- concatenare de histograme de blocuri





- vizualizare descriptor





Exemple pozitive



+ alte câteva mii...

Exemple negative



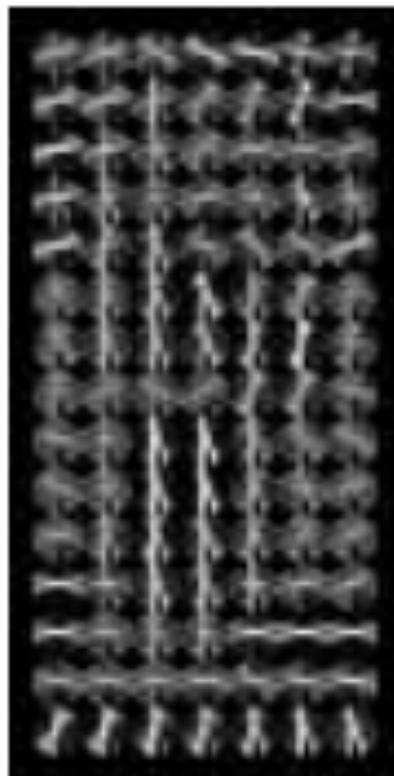
+ alte câteva milioane...

SVM + HOG

- Vizualizare



imagine test



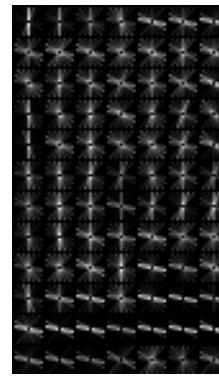
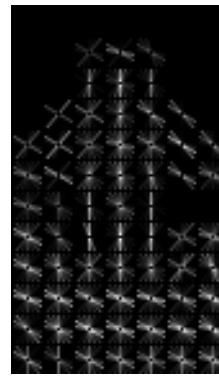
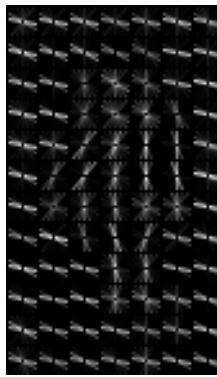
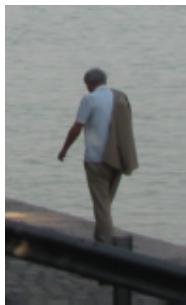
descriptorul HOG



descriptorul HOG
afisam orientarile pt.
ponderi pozitive din w

SVM + HOG

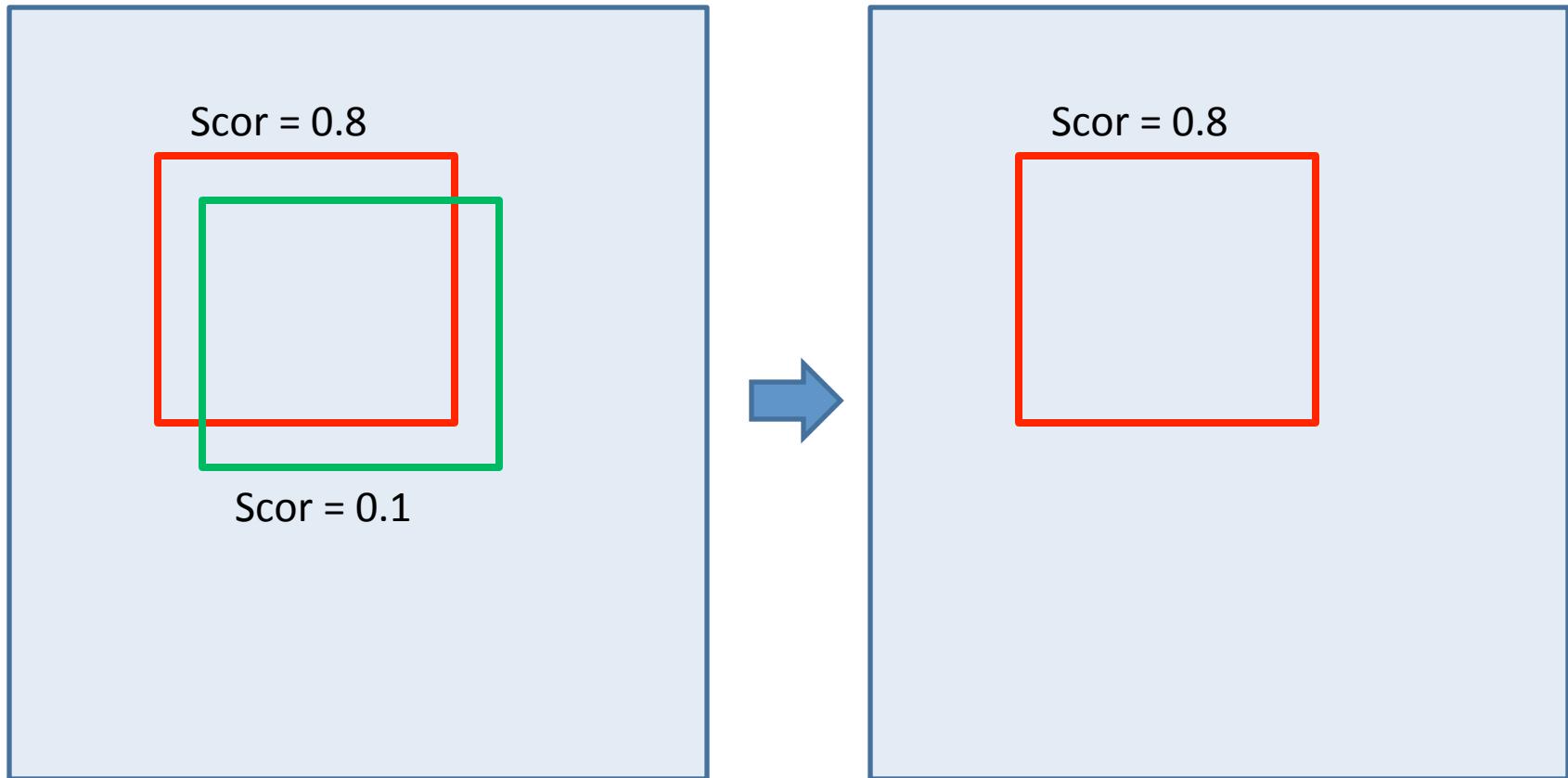
- Vizualizare



Suprimarea non-maximelor



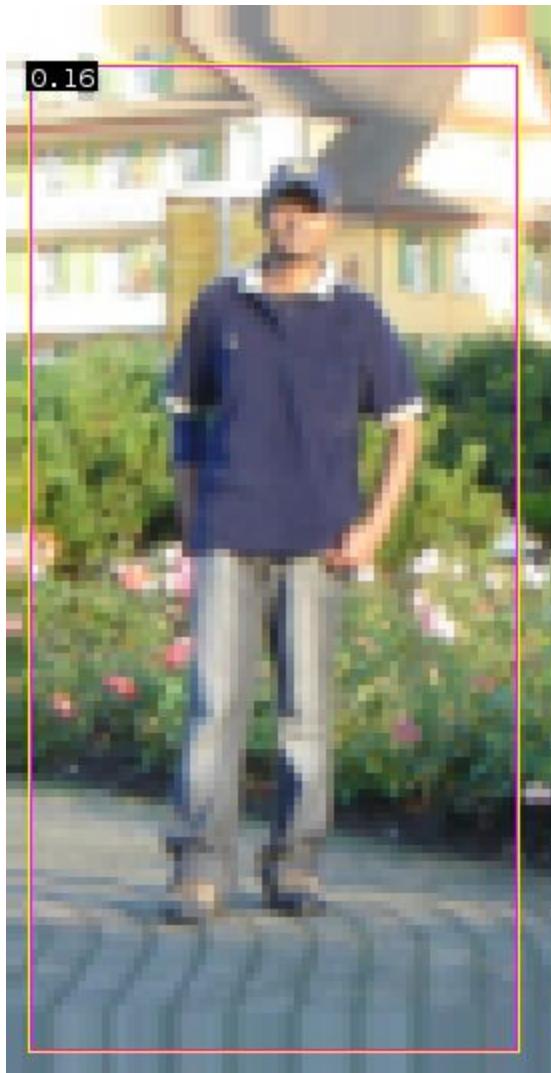
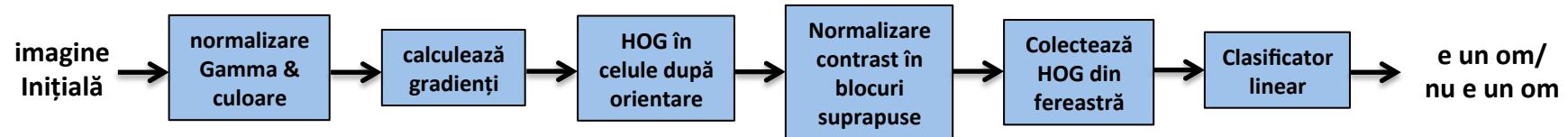
Suprimarea non-maximelor



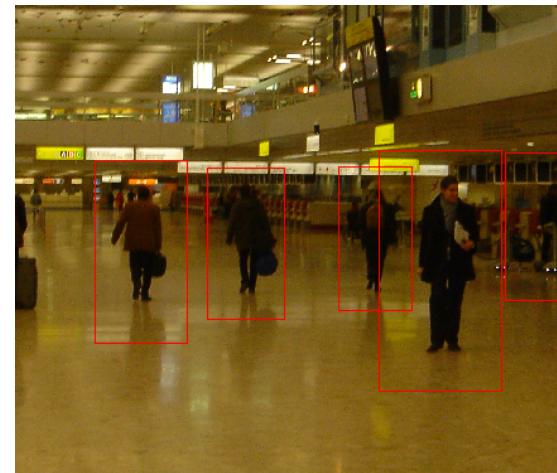
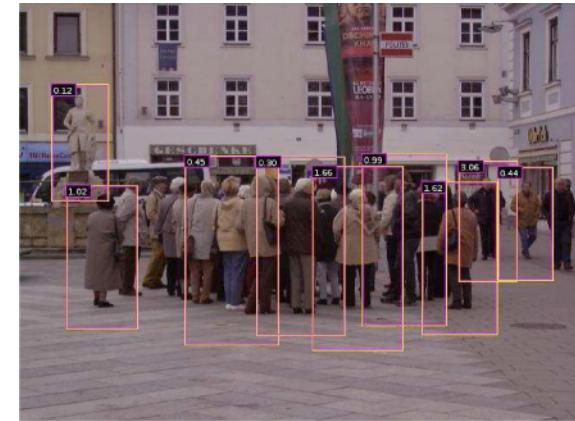
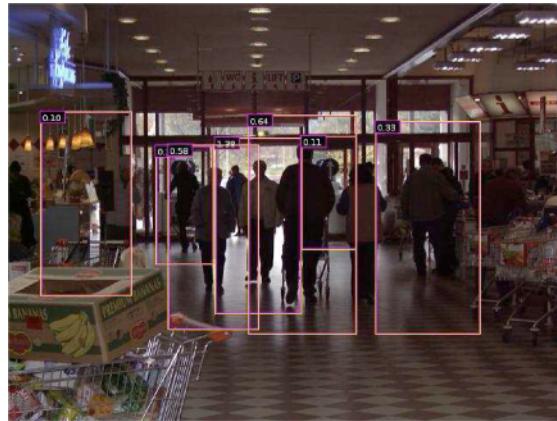
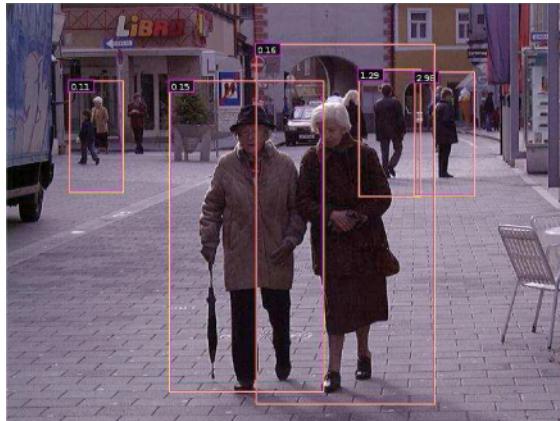
HOG pentru detectarea oamenilor



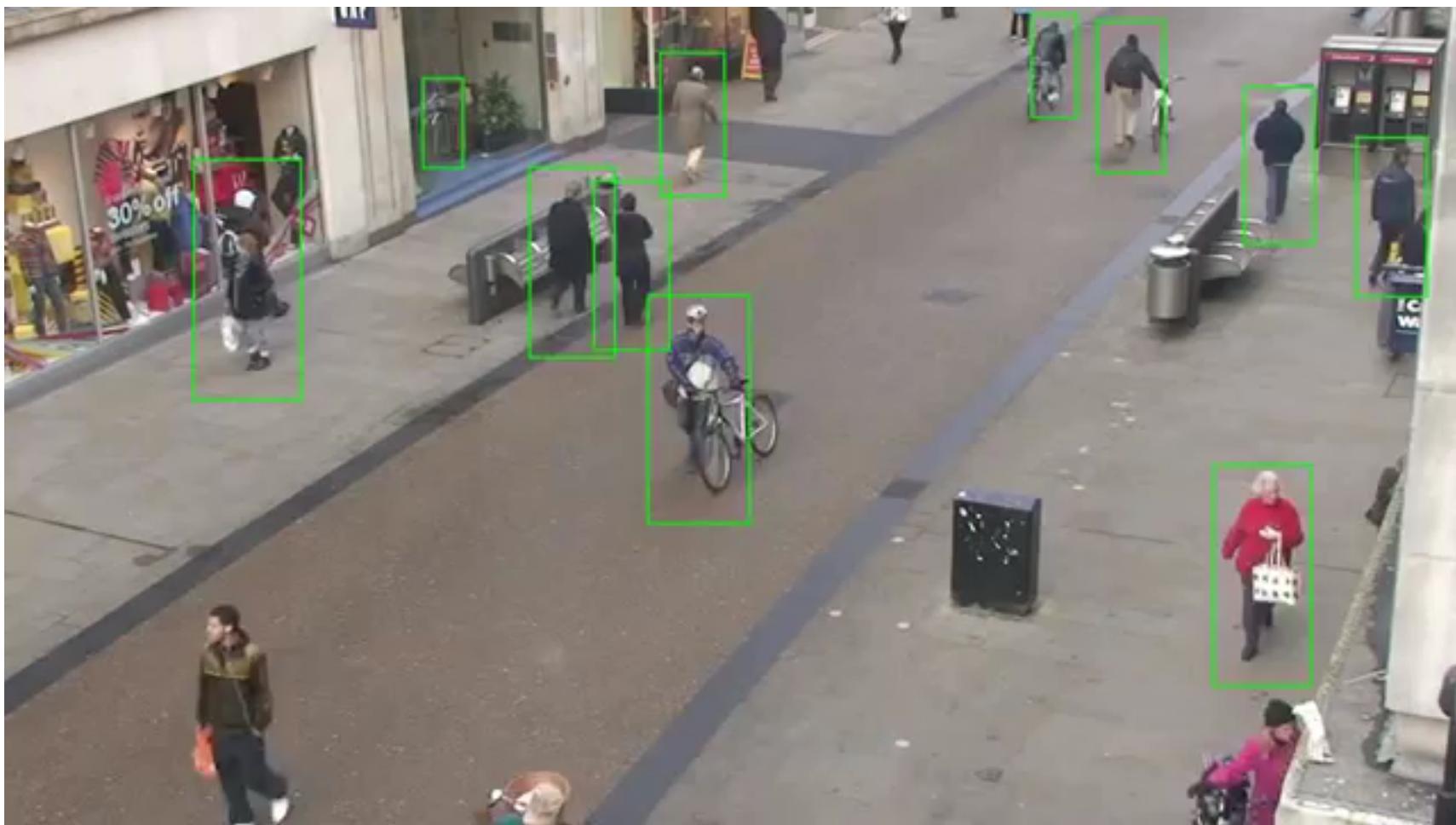
1. Extragă o fereastră de dimensiuni 128x64 pixeli la fiecare punct și scală din imagine.
2. Calculează HOG (histogramă de gradienți orientați) pentru fiecare fereastră.
3. Asociază un scor acestei ferestre pe baza clasificatorului liniar învățat
4. Realizează suprimarea non-maximelor pentru eliminarea detectiilor ce se suprapun și au un scor mai mic.



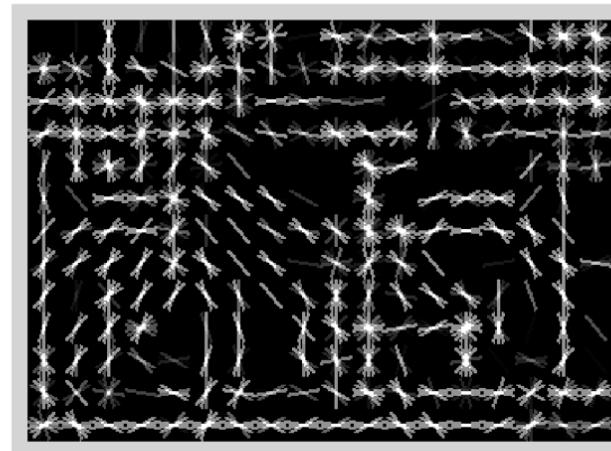
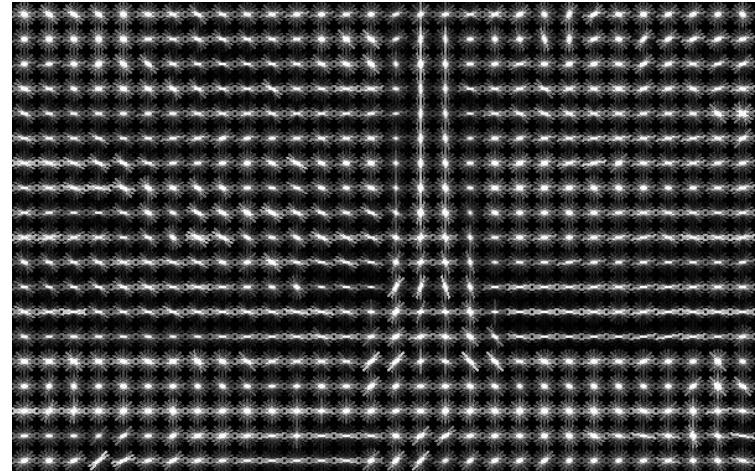
Rezultate



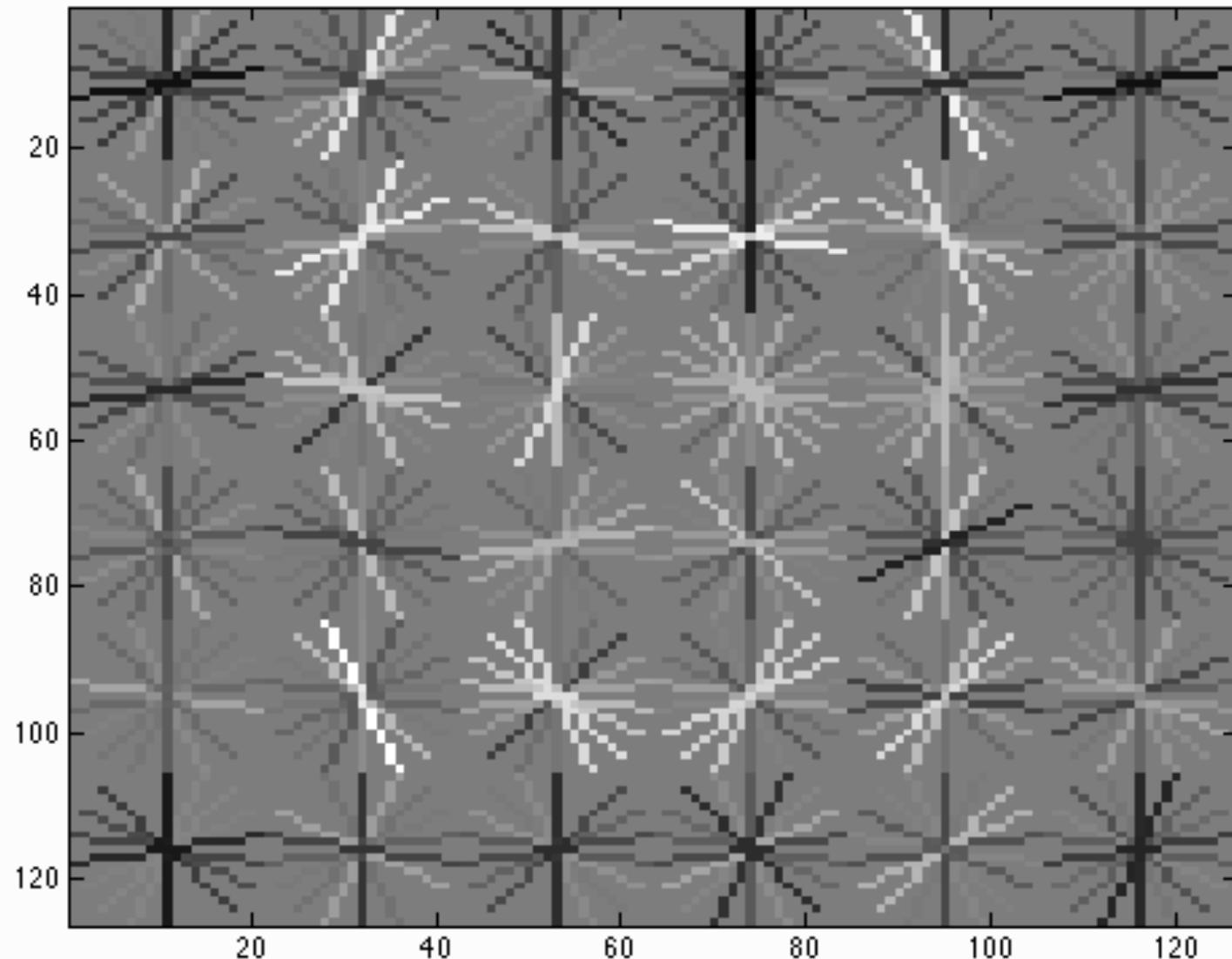
HOG pentru detectarea oamenilor în video



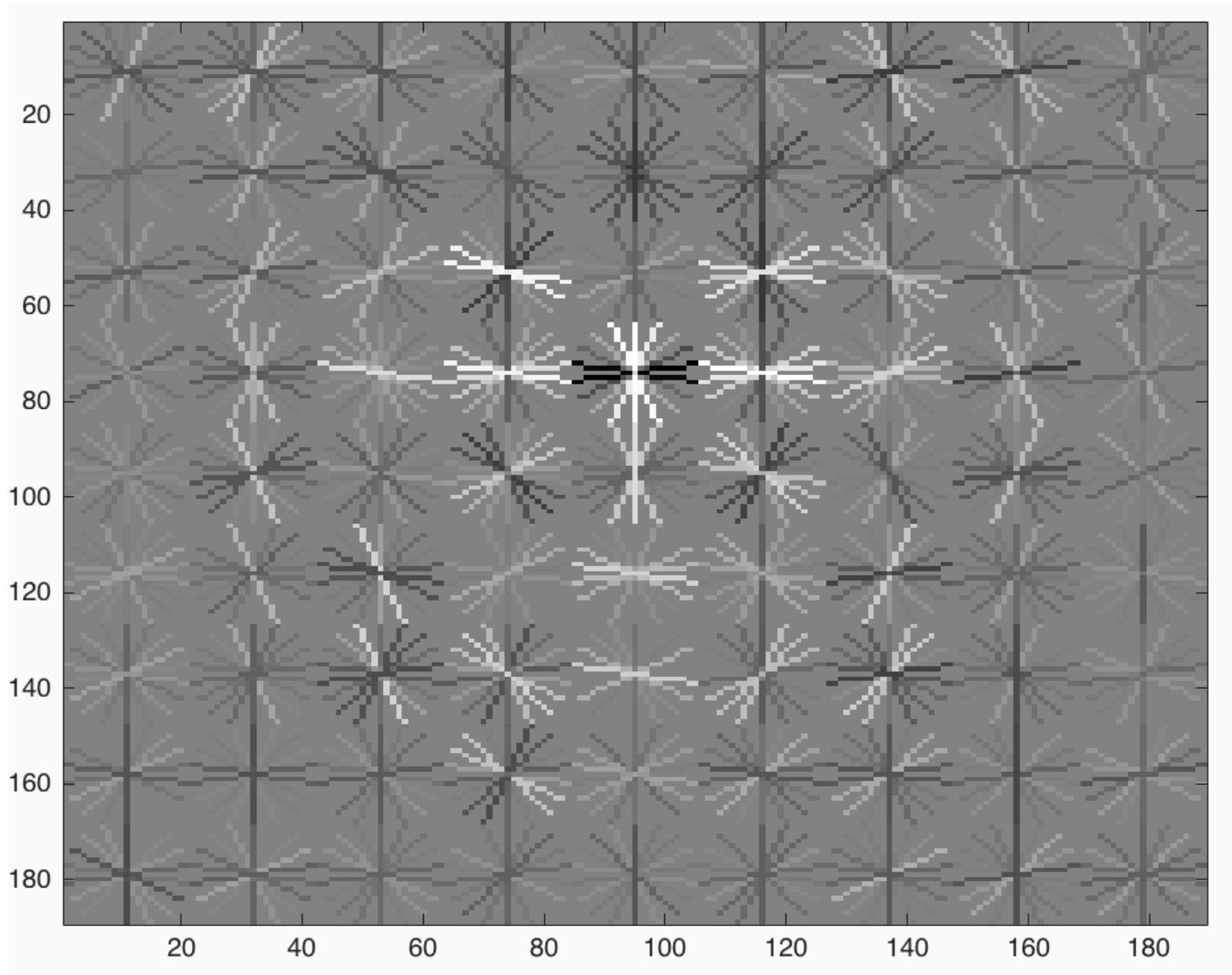
HOG și pentru alte clase



HOG pentru detectare facială

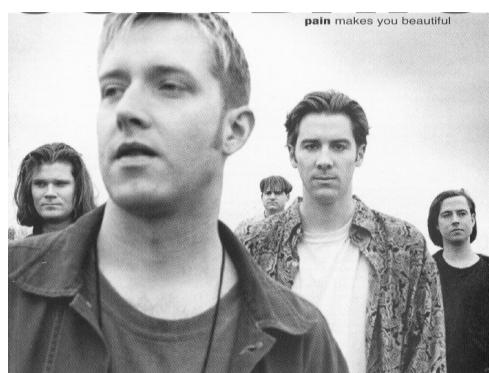


HOG pentru detectare facială



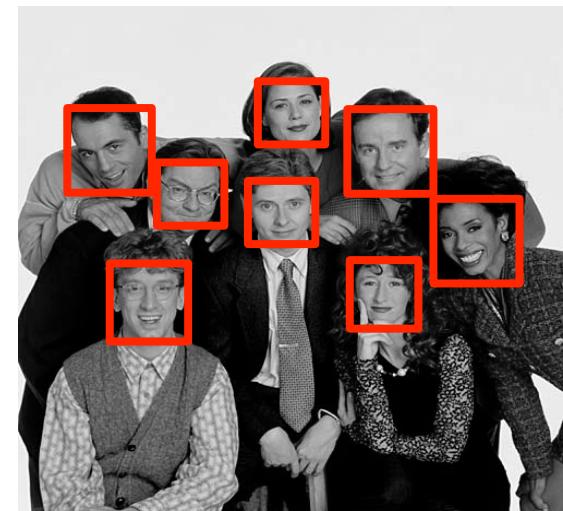
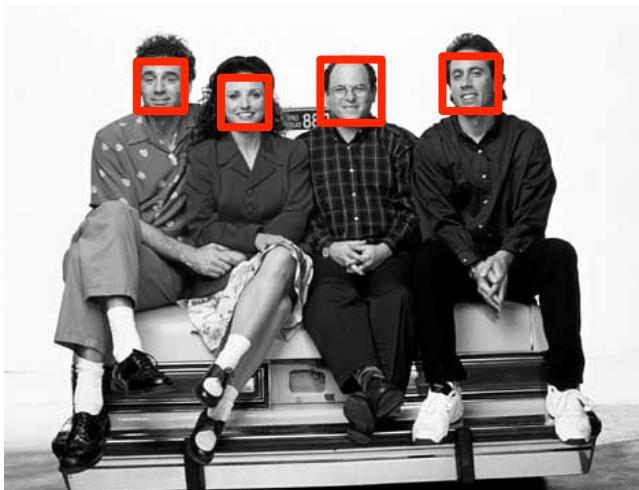
Detectare facială folosind metoda glisării ferestrei și histograme de gradienti orientați

Unde se află în imagine fețele umane?



Detectare facială folosind metoda glisării ferestrei și histograme de gradienți orientați

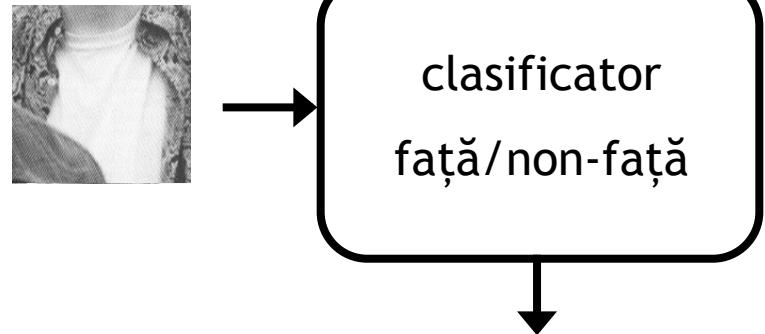
Unde se află în imagine fețele umane?



Localizare la nivel de fereastră dreptunghiulară

Metoda glisării ferestrei

- Localizarea obiectelor la nivel de fereastră:
metoda ferestrei glisante (sliding-window)
 - detectare via clasificare: **clasificator binar** → conține sau nu fiecare fereastră din imagine instanță a clasei de obiecte X (față)?
 - consideră **ferestre poziționate în fiecare pixel**, de mărimi diferite



NDată e față.

Metoda glisării ferestrelor

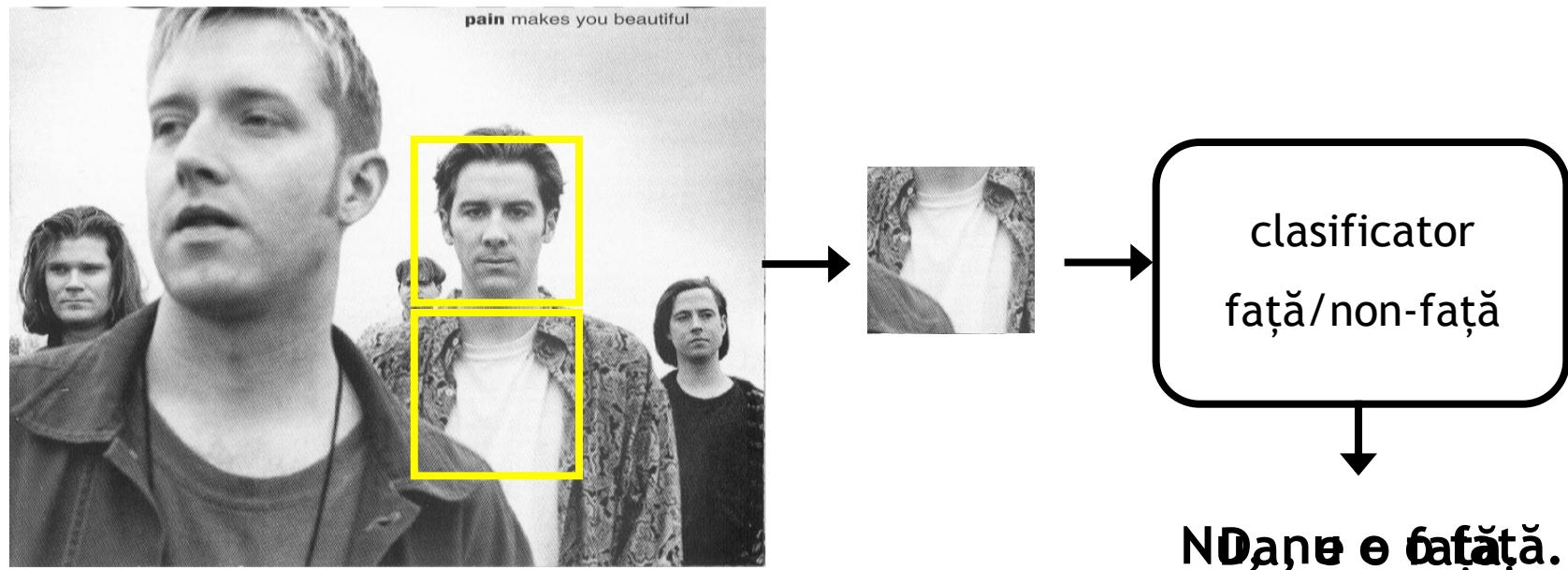
- Localizarea obiectelor la nivel de fereastră:
metoda ferestrelor glisante (sliding-window)
 - detectare via clasificare: **clasificator binar** → conține sau nu fiecare fereastră din imagine instanță a clasei de obiecte X (față)?
 - consideră **ferestre poziționate în fiecare pixel**, de mărimi diferite



Etape în construcția detectorului facial

1. Învățarea unui clasificator

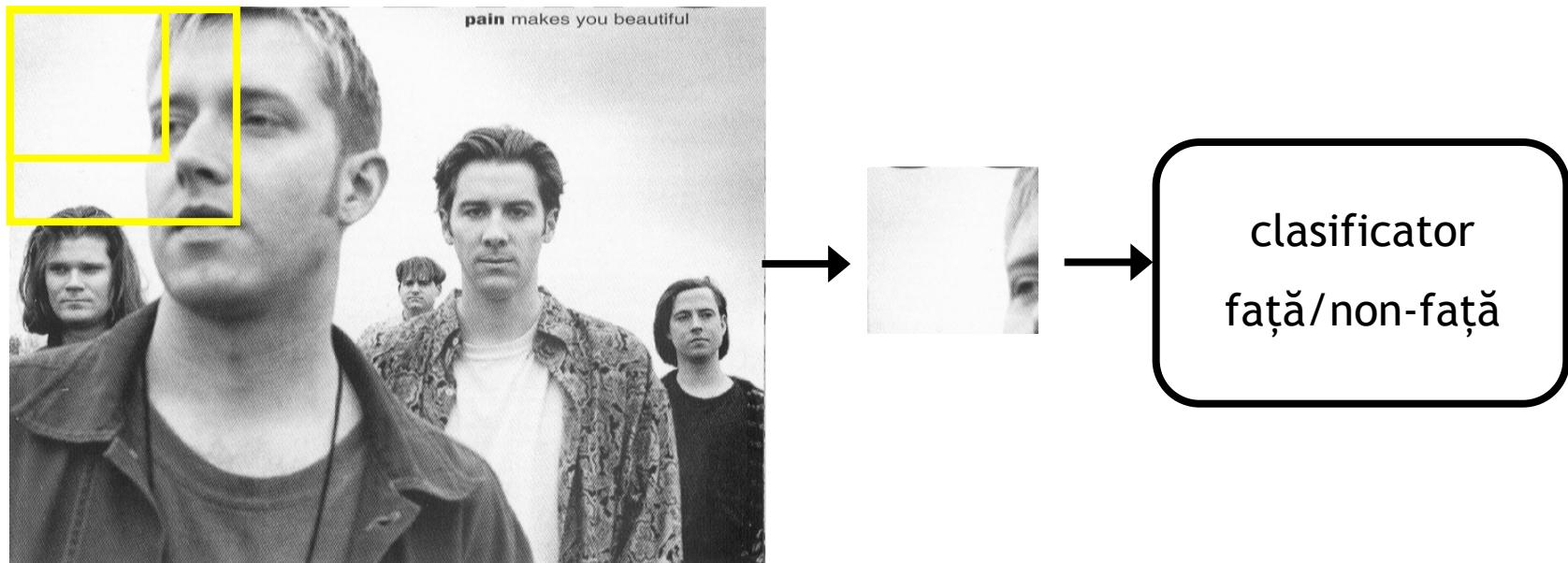
- **clasificator**: funcție care **asignează un scor** unei ferestre pe baza conținutului lor vizual (pixelii din interiorul ei)
- vrem să învățăm un clasificator care distinge ferestrele ce conțin fețe (**ideal, clasificatorul va asigna acestor ferestre un scor > 0**) de ferestrele ce nu conțin fețe (**ideal, clasificatorul va asigna acestor ferestre un scor < 0**)



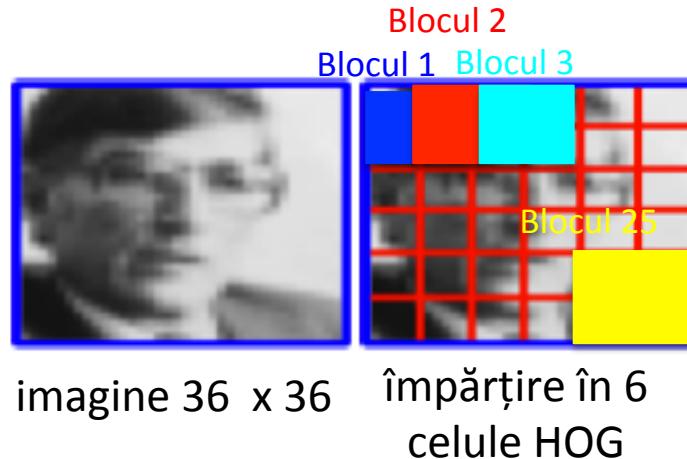
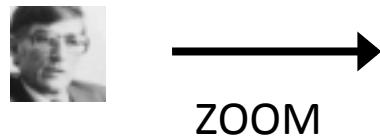
Etape în construcția detectorului facial

2. Implementarea metodei glisării unei ferestre

- glisarea unei ferestre de la stânga la dreapta și de sus în jos
- clasificarea (asignarea unui scor) fiecărei ferestre pe baza clasificatorului învățat la etapa 1
- localizarea fețelor pe baza scorurilor ferestrelor



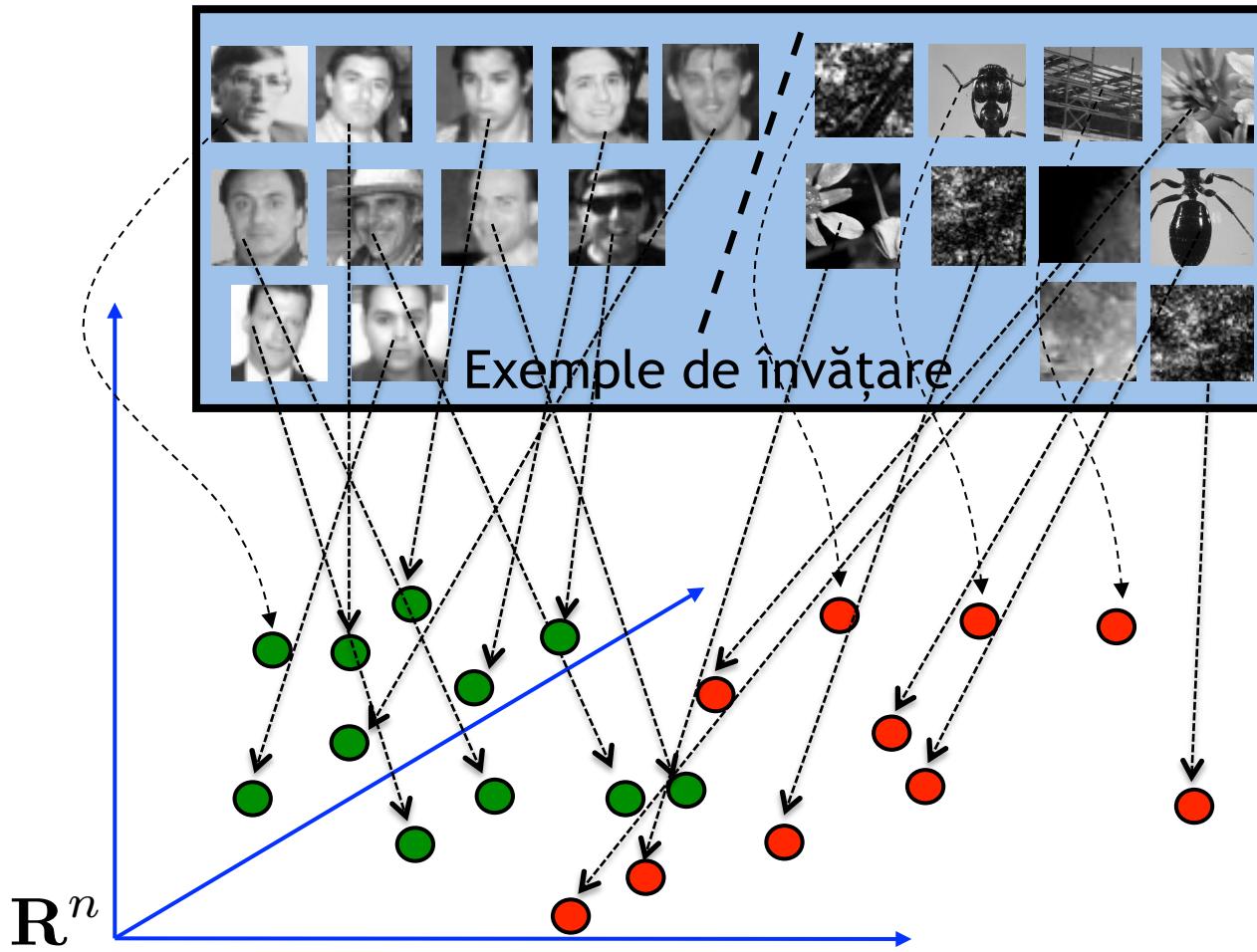
Descriptorul HOG al unei ferestre



- împarte imaginea în celule, fiecare celulă are 6×6 pixeli
- calculează gradientul imaginii (pentru fiecare pixel avem orientare și magnitudine)
- pentru fiecare celulă calculează o histogramă de gradienți orientați (9 bin-uri)
- grupează celule în blocuri, un bloc = 4 celule; fiecare celulă face parte din 4 blocuri = 4 normalizări L2;
- histograma unui bloc = 4 histograme ale celulelor = $4 \times 9 =$ vector de dimensiune 36
- HOG = descriptor obținut prin concatenarea histogramelor a 25 blocuri = $25 \times 36 = 900$

Descriptori pentru exemple de învățare

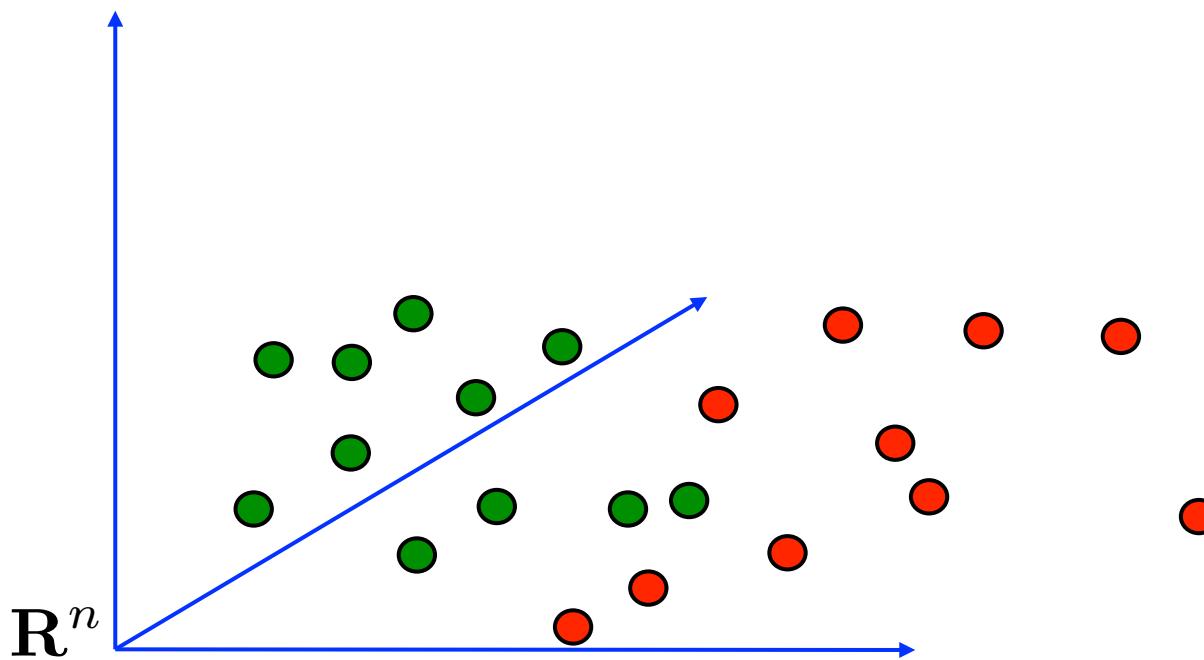
- Învățarea unui clasificator se realizează pe baza descriptorilor pentru **exemple pozitive** și **exemple negative**.



Spațiul descriptorilor

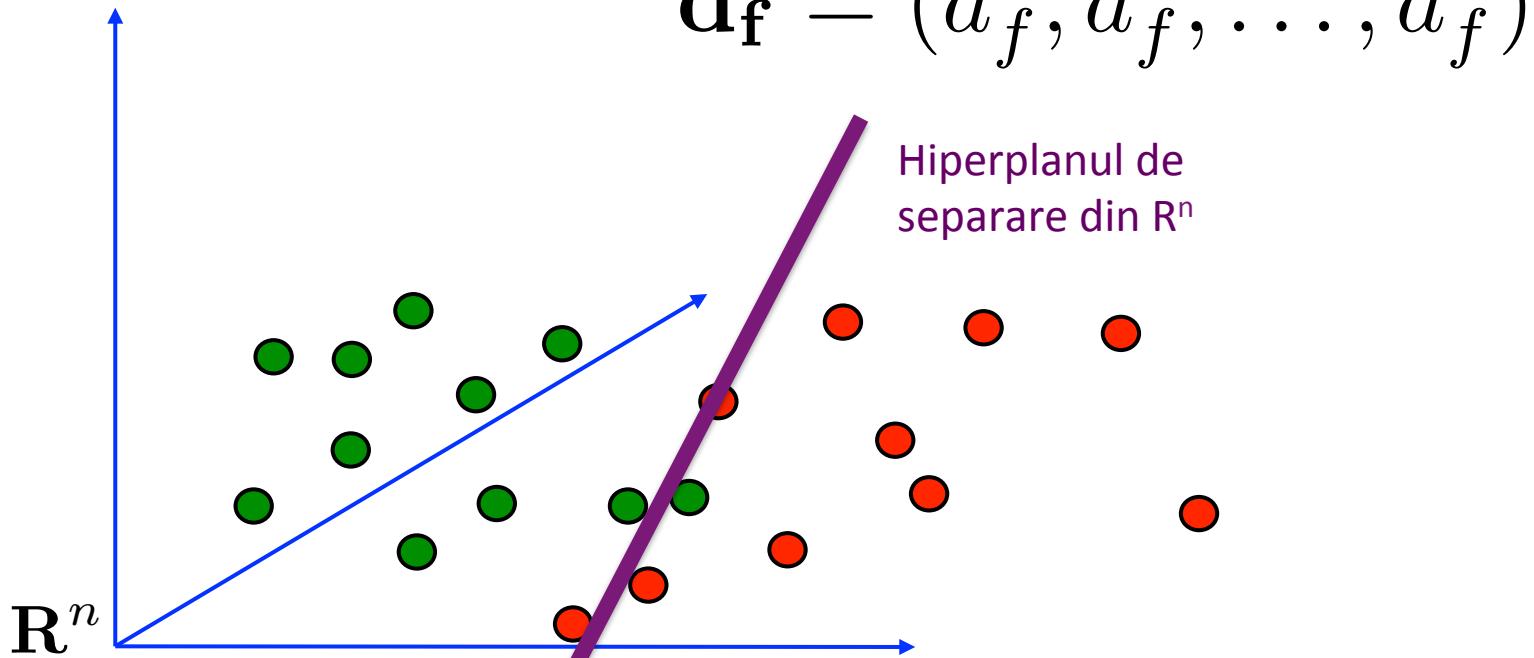
- descriptori pentru **exemple pozitive** și **exemple negative**
- fiecare descriptor \mathbf{d}_f este de dimensiune n

$$\mathbf{d}_f = (d_f^1, d_f^2, \dots, d_f^n)$$



Model de clasificator

- descriptori pentru **exemple pozitive** și **exemple negative**
- fiecare descriptor \mathbf{d}_f este de dimensiune n
- un posibil clasificator: clasificator liniar (hiperplan) care să separe cât mai bine **exemplile pozitive** de **exemplile negative**
- ecuația unui hiperplan în R^n : $\mathbf{w}^t \cdot \mathbf{d}_f + b = 0$
 $\mathbf{d}_f = (d_f^1, d_f^2, \dots, d_f^n)$



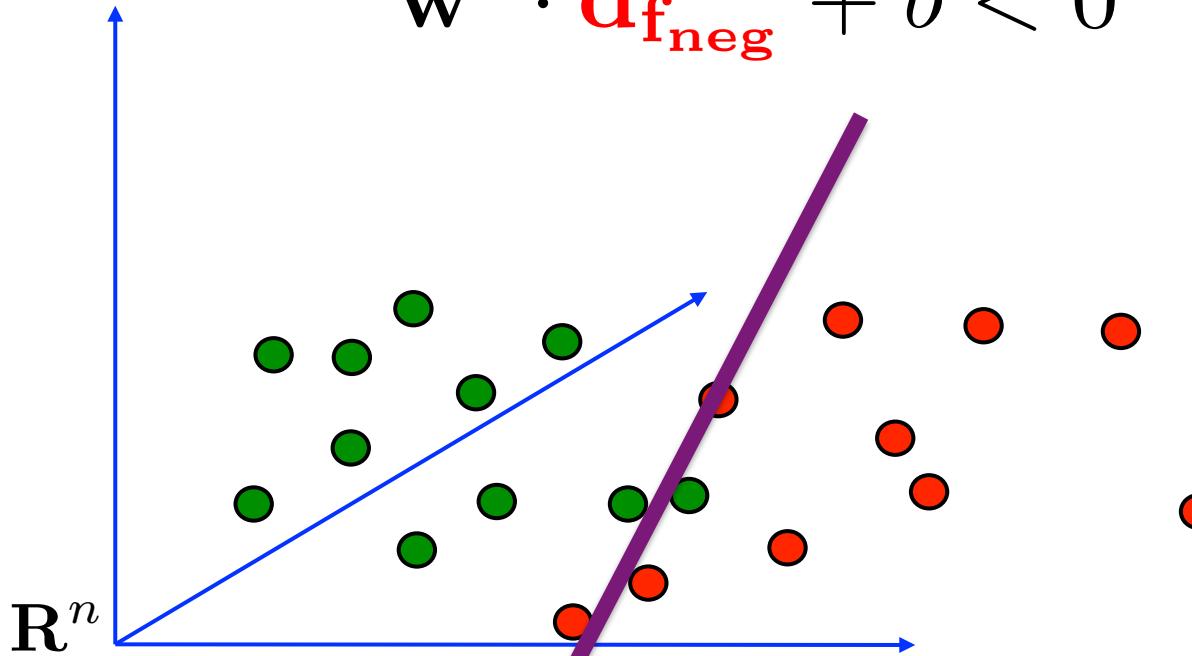
Model de clasificator

- regula de clasificare:
 - pentru **exemple pozitive** vrem să avem:

$$\mathbf{w}^t \cdot \mathbf{d}_{f_{\text{pos}}} + b > 0$$

- pentru exemple negative vrem să avem:

$$\mathbf{w}^t \cdot \mathbf{d}_{f_{\text{neg}}} + b < 0$$



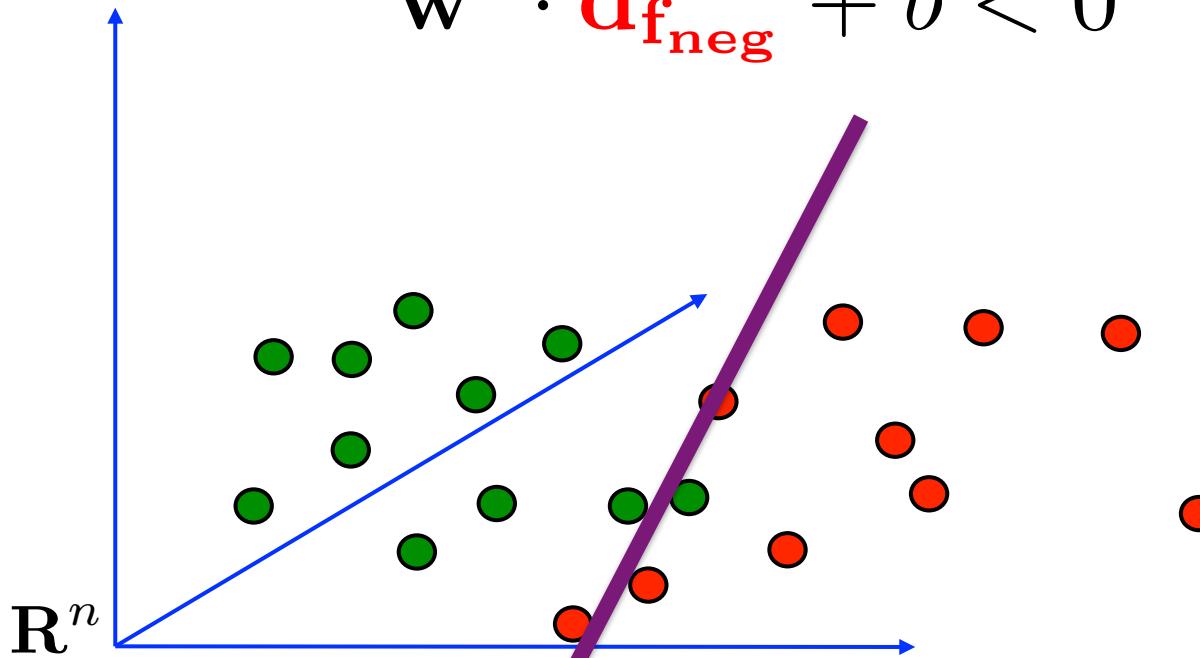
Învățarea parametrilor modelului

- învăță parametri (w, b) ai clasificatorului liniar (= hiperplan) astfel încât
 - pentru **exemple pozitive** vrem să avem:

$$w^t \cdot d_{f_{\text{pos}}} + b > 0$$

- pentru exemple negative vrem să avem:

$$w^t \cdot d_{f_{\text{neg}}} + b < 0$$



Implementarea metodei glisării unei ferestre

1. Pentru o imagine test se calculează descriptorii HOG asociați celulelor



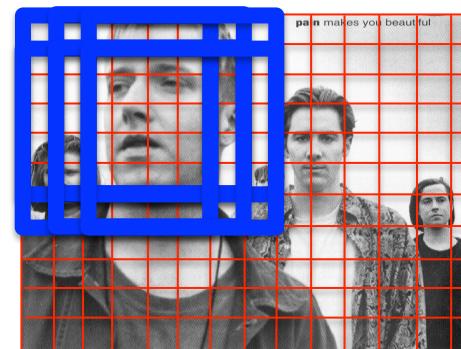
Implementarea metodei glisării unei ferestre

1. Pentru o imagine test se calculează descriptorii HOG asociati celulelor
2. Evaluează toate ferestrele pătratice f de dimensiune 36x36 pixeli
 - evaluarea clasificatorului: $w^t d_f + b$
3. Repetă 1+2 pentru mărimi diferite ale imaginii
 - localizarea fețelor din imagine de mărimi diferite



Implementarea metodei glisării unei ferestre

1. Pentru o imagine test se calculează descriptorii HOG asociati celulelor
2. Evaluează toate ferestrele pătratice f de dimensiune 36x36 pixeli
 - evaluarea clasificatorului: $w^t d_f + b$
3. Repetă 1+2 pentru mărimi diferite ale imaginii
 - localizarea fețelor din imagine de mărimi diferite



Implementarea metodei glisării unei ferestre

1. Pentru o imagine test se calculează descriptorii HOG asociati celulelor
2. Evaluează toate ferestrele pătratice f de dimensiune 36x36 pixeli
 - evaluarea clasificatorului: $w^t d_f + b$
3. Repetă 1+2 pentru mărimi diferite ale imaginii
 - localizarea fețelor din imagine de mărimi diferite
4. Maximele locale ale clasificatorului localizează fețele în imagine



Implementarea metodei glisării unei ferestre

1. Pentru o imagine test se calculează descriptorii HOG asociati celulelor
2. Evaluează toate ferestrele pătratice f de dimensiune 36x36 pixeli
 - evaluarea clasificatorului: $w^t d_f + b$
3. Repetă 1+2 pentru mărimi diferite ale imaginii
 - localizarea fețelor din imagine de mărimi diferite
4. Maximele locale ale clasificatorului localizează fețele în imagine



Maxim local
(fereastra cu scorul cel
mai mare)

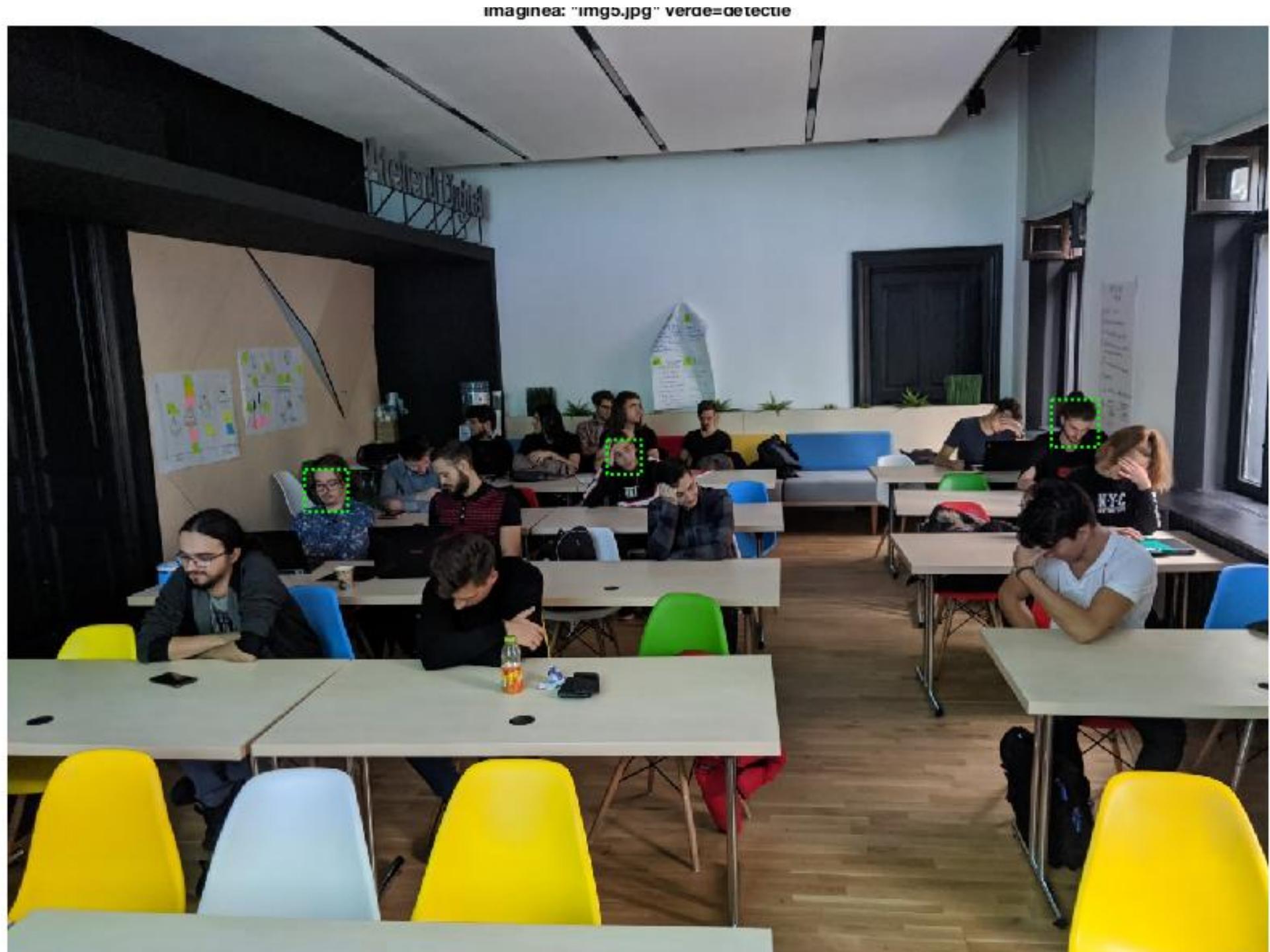








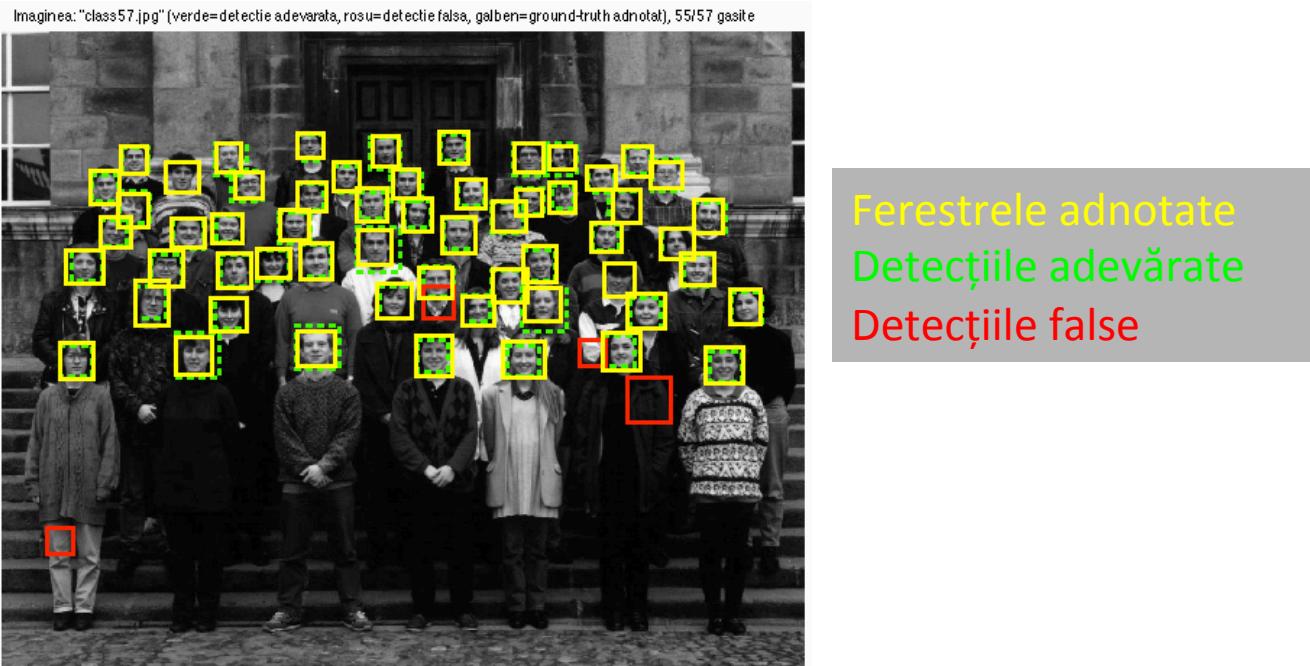
Imaginea: "img5.jpg" verde=detectie



Evaluare – detectii adevărate și false

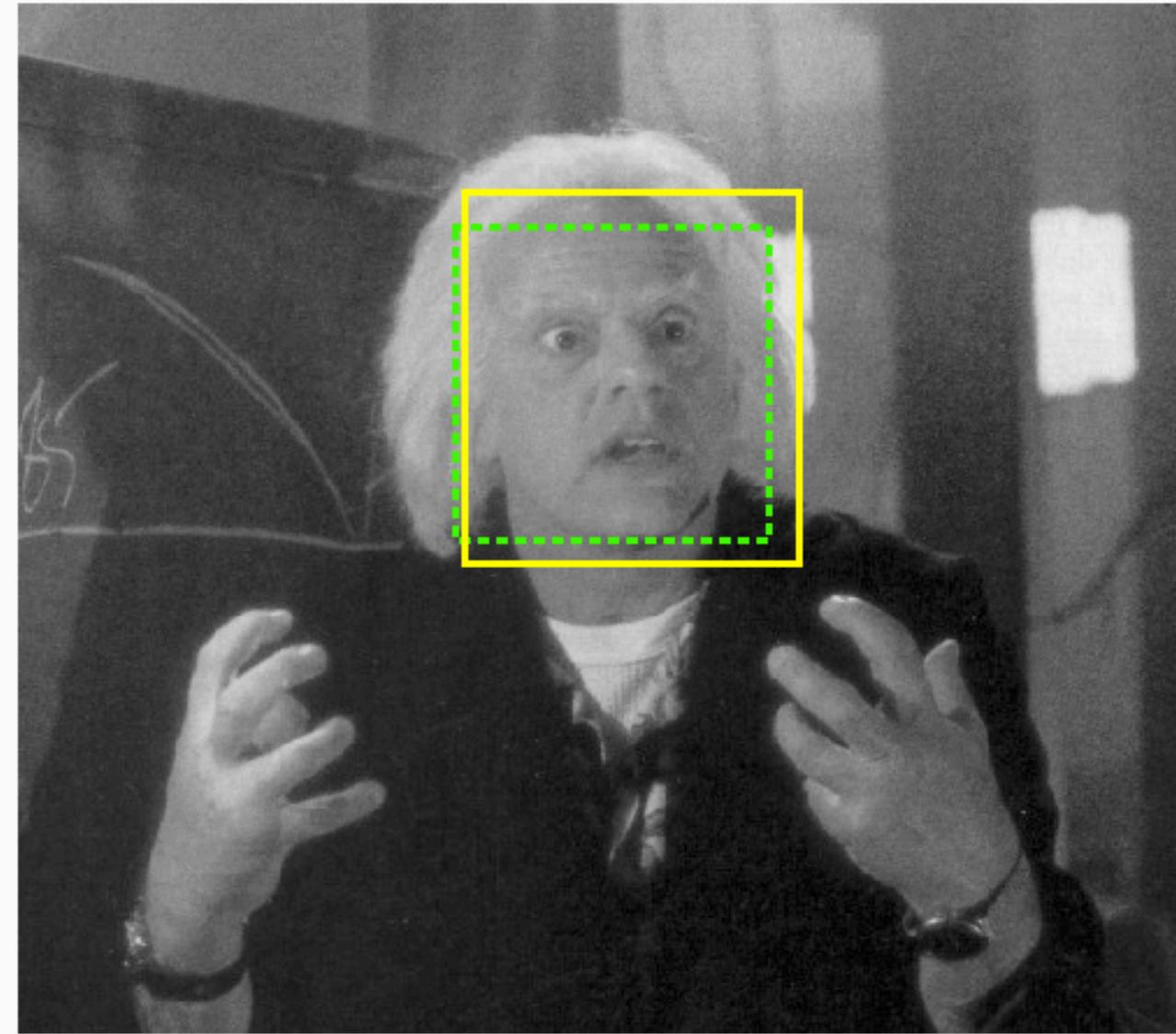
Cum evaluăm performanța unui detector facial?

- comparăm ceea ce am obținut (ferestrele de maxim local se numesc detectii) cu ceea ce trebuia să obținem (ferestre adnotate conținând fețe)
- detectii adevărate** = detectii care se suprapun (intersecție/reuniune) > 0.3 cu o fereastră adnotată
- detectii false** = detectii care nu se suprapun > 0.3 cu o fereastră adnotată sau există o altă detectie de scor mai mare care se suprapune > 0.3 cu fereastra adnotată (se penalizează multiple detectii ale aceleiași fețe)



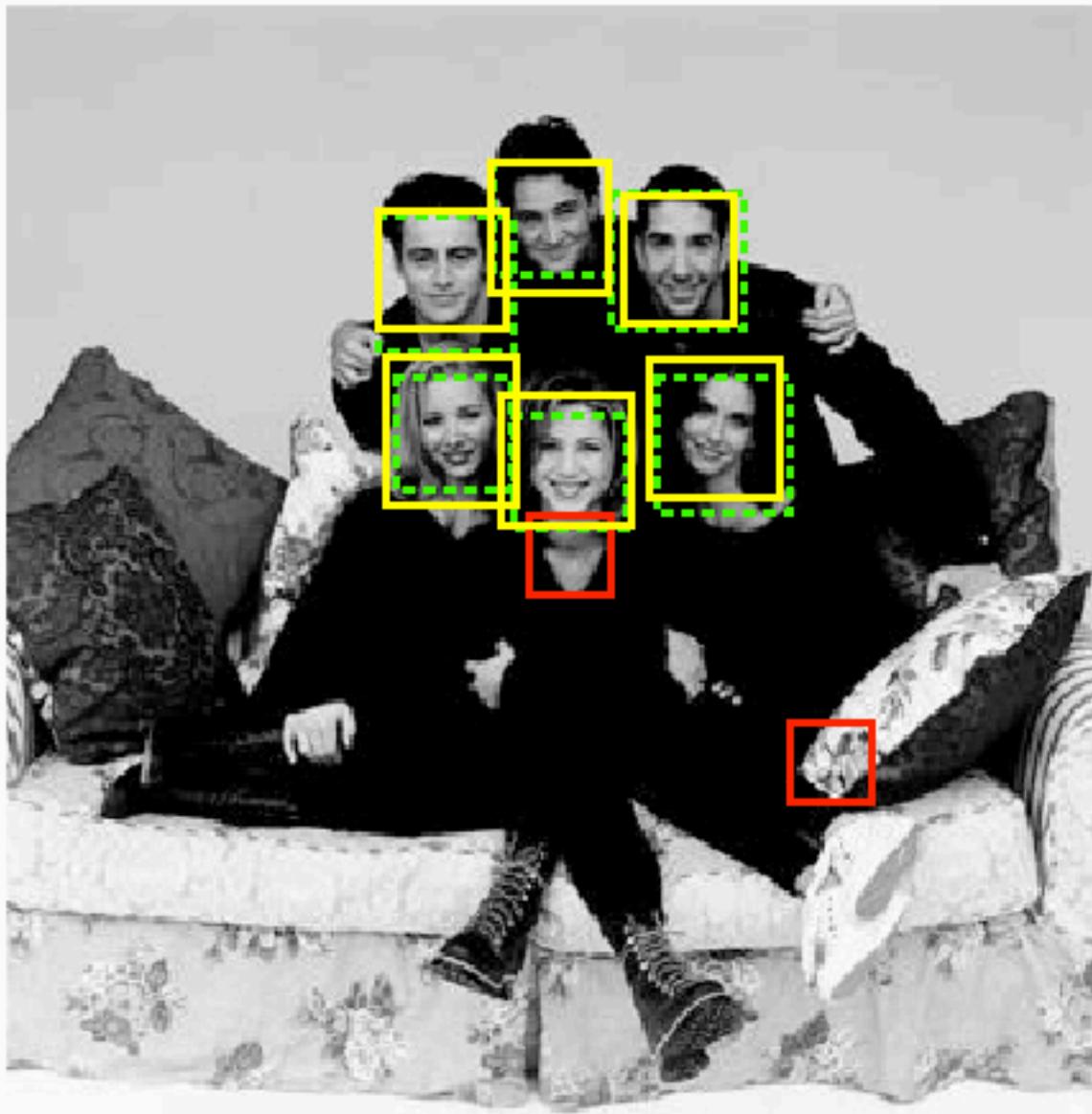
Rezultate

Imaginea: "bttf206.jpg" (verde=detectie adevarata, rosu=detectie falsa, galben=ground-truth adnotat), 1/1 gasite

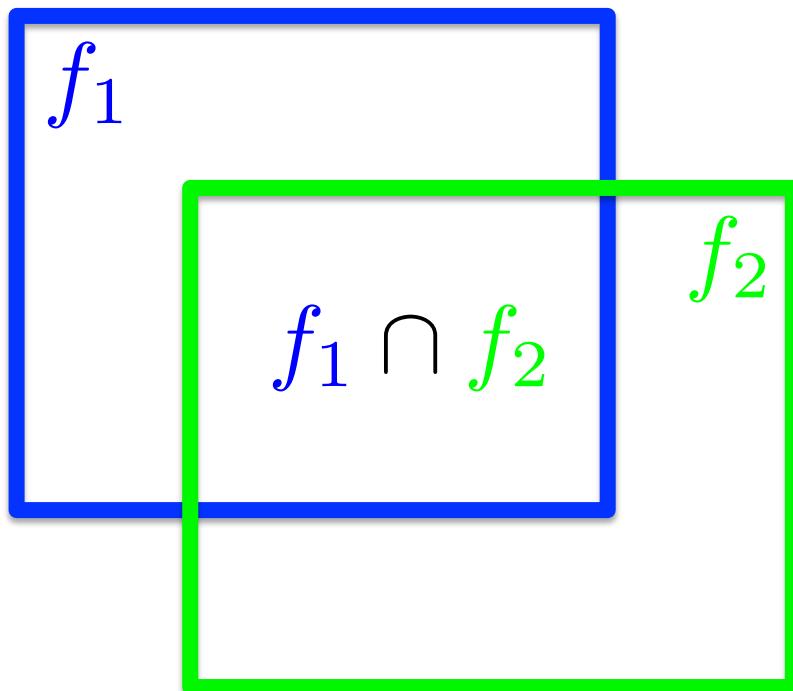


Rezultate

Imaginea: "ew-friends.jpg" (verde=detectie adevarata, rosu=detectie falsa, galben=ground-truth adnotat), 6/6 gasite



Suprapunerea a două ferestre



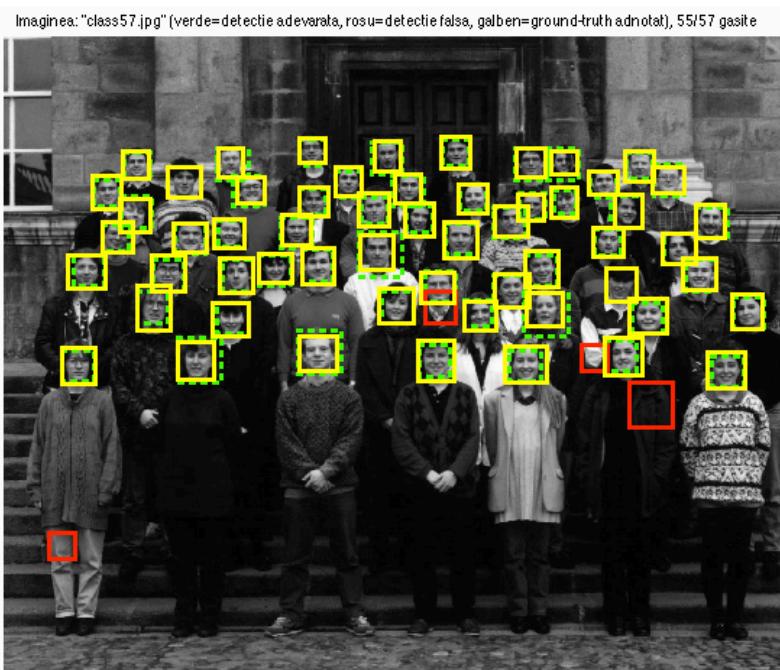
$$suprapunere(f_1, f_2) = \frac{aria(f_1 \cap f_2)}{aria(f_1 \cup f_2)}$$

$$suprapunere(f_1, f_2) = \frac{aria(f_1 \cap f_2)}{aria(f_1) + aria(f_2) - aria(f_1 \cap f_2)}$$

Evaluare – recall și precizie pe o imagine

Cum evaluăm performanța unui detector?

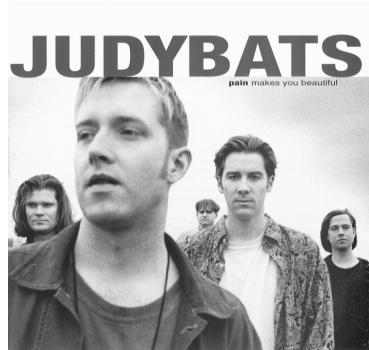
- comparăm ceea ce am obținut (ferestrele de maxim local se numesc detectii) cu ceea ce trebuia să obținem (ferestre adnotate conținând obiectele de interes)
- recall = câte fețe am găsit din cele adnotate =
$$\frac{\#\text{detectii adevarate}}{\#\text{ferestre adnotate}}$$
- precizie = câte detectii sunt adevărate =
$$\frac{\#\text{detectii adevarate}}{\#\text{detectii adevarate} + \#\text{detectii false}}$$



Ferestrele adnotate
Detectiile adevărate
Detectiile false
recall = $55/57 = 0.965$
precizie = $55/59 = 0.932$

Compararea a doi algoritmi

- se face pe numite baze de date pentru care se cunoaște ceea ce trebuia să întoarcă algoritmul de detectare facială (imaginile sunt adnotate)

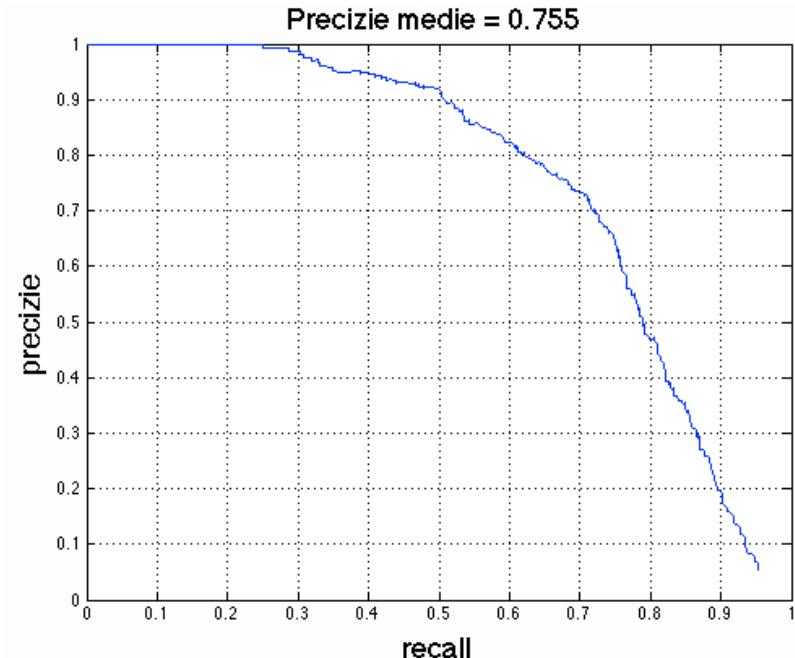


- se compară în funcție de precizie și recall sumarizate în graficul precizie-recall care oferă un număr: precizia medie a detectorului de fețe

Evaluare – recall și precizie pe toate imaginile

Se construiește graficul precizie-recall în felul următor:

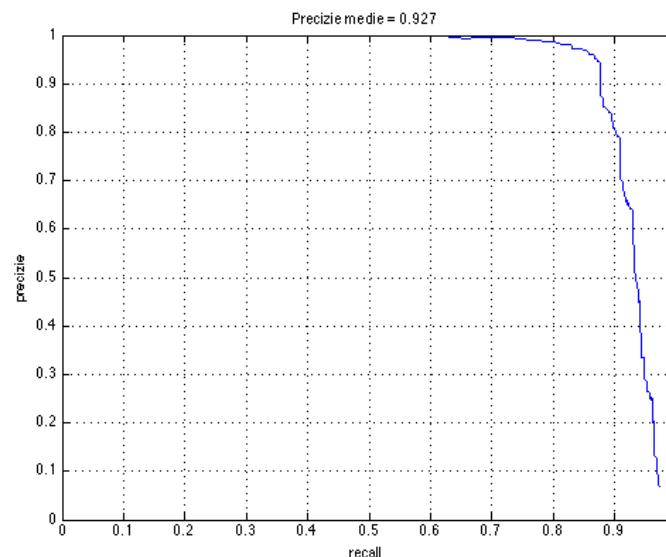
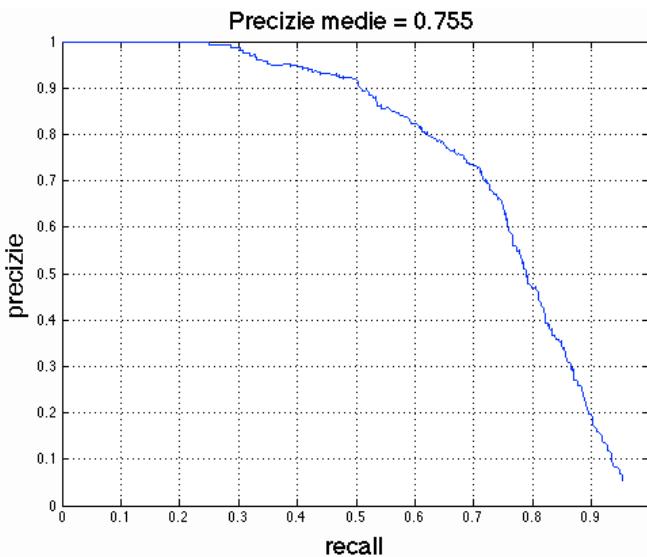
1. se rulează detectorul facial pe toate imaginile și se obține pentru fiecare imagine o mulțime de detectii;
2. se stabilește pentru fiecare imagine (pentru care avem adnotări) care sunt detectiile adevărate și cele false;
3. se ordonează descrescător toate detectiile în funcție de scorul lor;
4. pentru un threshold care descrește de la scorul cel mai mare la scorul cel mai mic se calculează precizia și recall-ul pentru toate detectiile cu scorul $>$ threshold



Valorea precizie medie sumarizează graficul precizie-recall prin aria de sub grafic
Pe baza acestei valori evaluăm performanța unui algoritm

Compararea a doi algoritmi

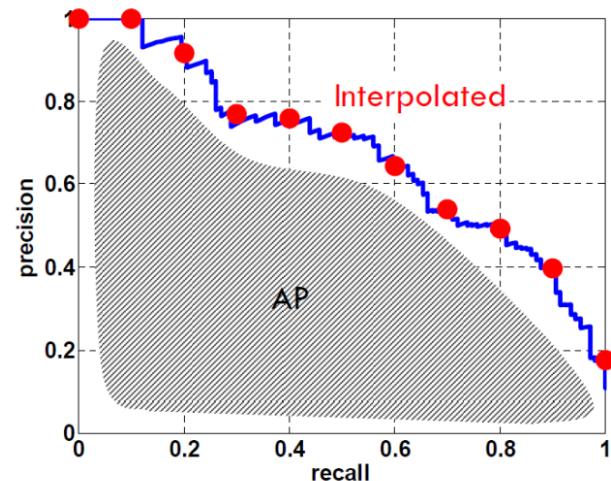
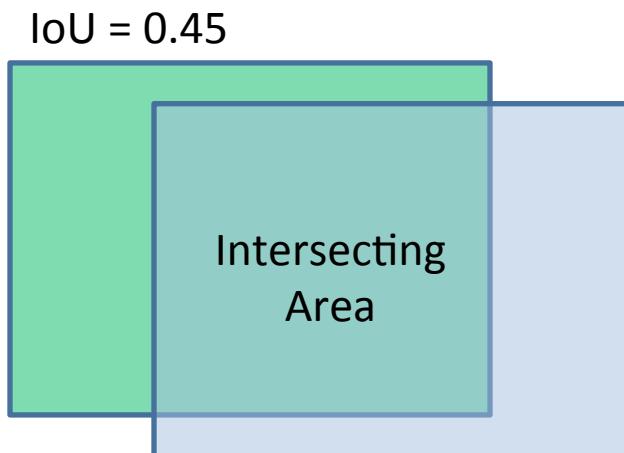
- se face pe numite baze de date pentru care se cunoaște ceea ce trebuia să întoarcă algoritmul de detectare facială (imaginile sunt adnotate)
- se compară în funcție de precizie, recall sumarizate în graficul precizie-recall care oferă un precizia medie (AP – Average Precision)



- algoritmul din dreapta este mai bun decât algoritmul din stânga

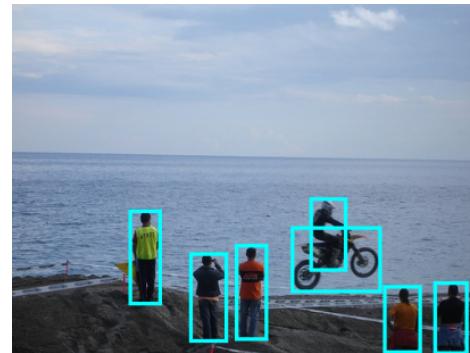
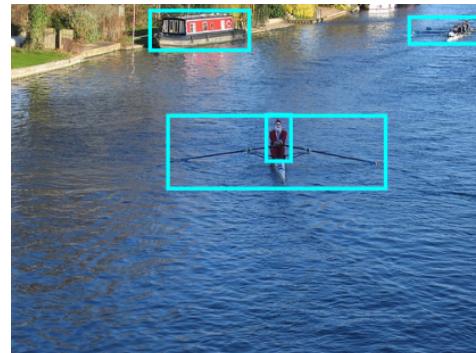
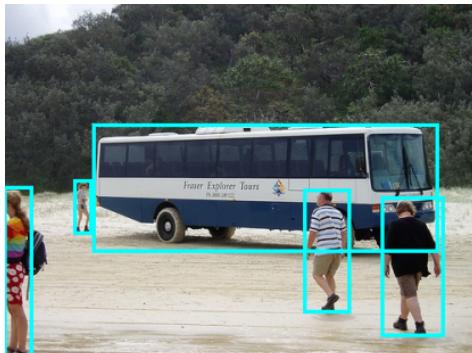
Evaluarea detectării de obiecte

- Seturi de date:
 - [PASCAL VOC](#) (2005-2012): 20 de clase, ~20,000 imagini
 - [MS COCO](#) (2014-prezent): 60 clase, ~300,000 imagini
 - [IMAGENET](#) (2012-prezent): 1000 clase, ~1,200,000 imagini
- Evaluare
 - output: pentru fiecare clasă, prezice ferestre (x_1, y_1, x_2, y_2) cu scoruri
 - metrică:
 - detecție adevărată: ≥ 0.5 (IoU) + nu e duplicat
 - graficul precizie-recall
 - Average Precision: aria de sub grafic (obținut prin interpolare)

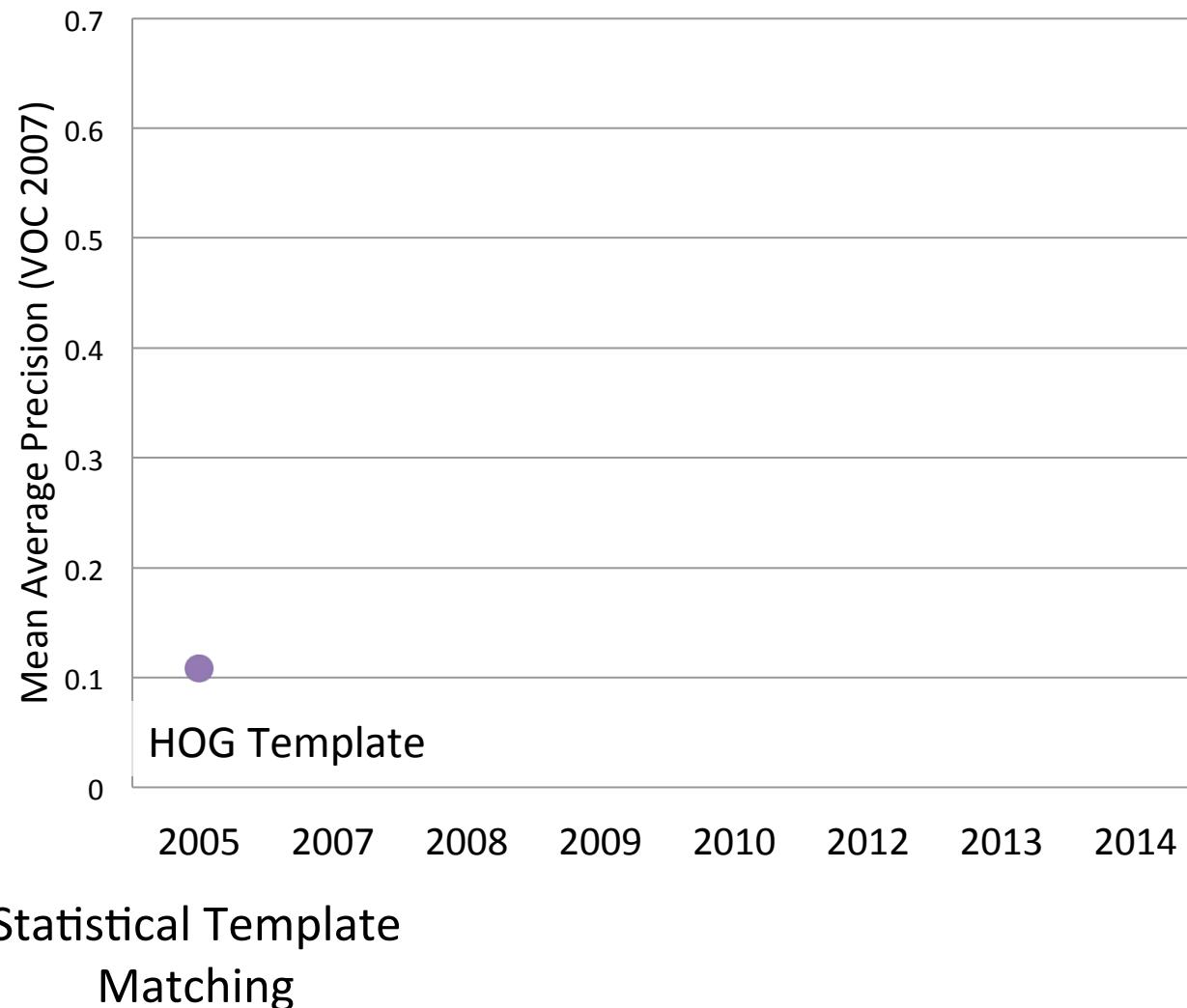


PASCAL VOC 2005-2012

- PASCAL Visual Object Classes Challenge
- 20 classes, ~10K images, ~25K annotated objects
- Training, validation, test data sets.
- Evaluation:
 - Average Precision (AP) per class
 - mean Average Precision

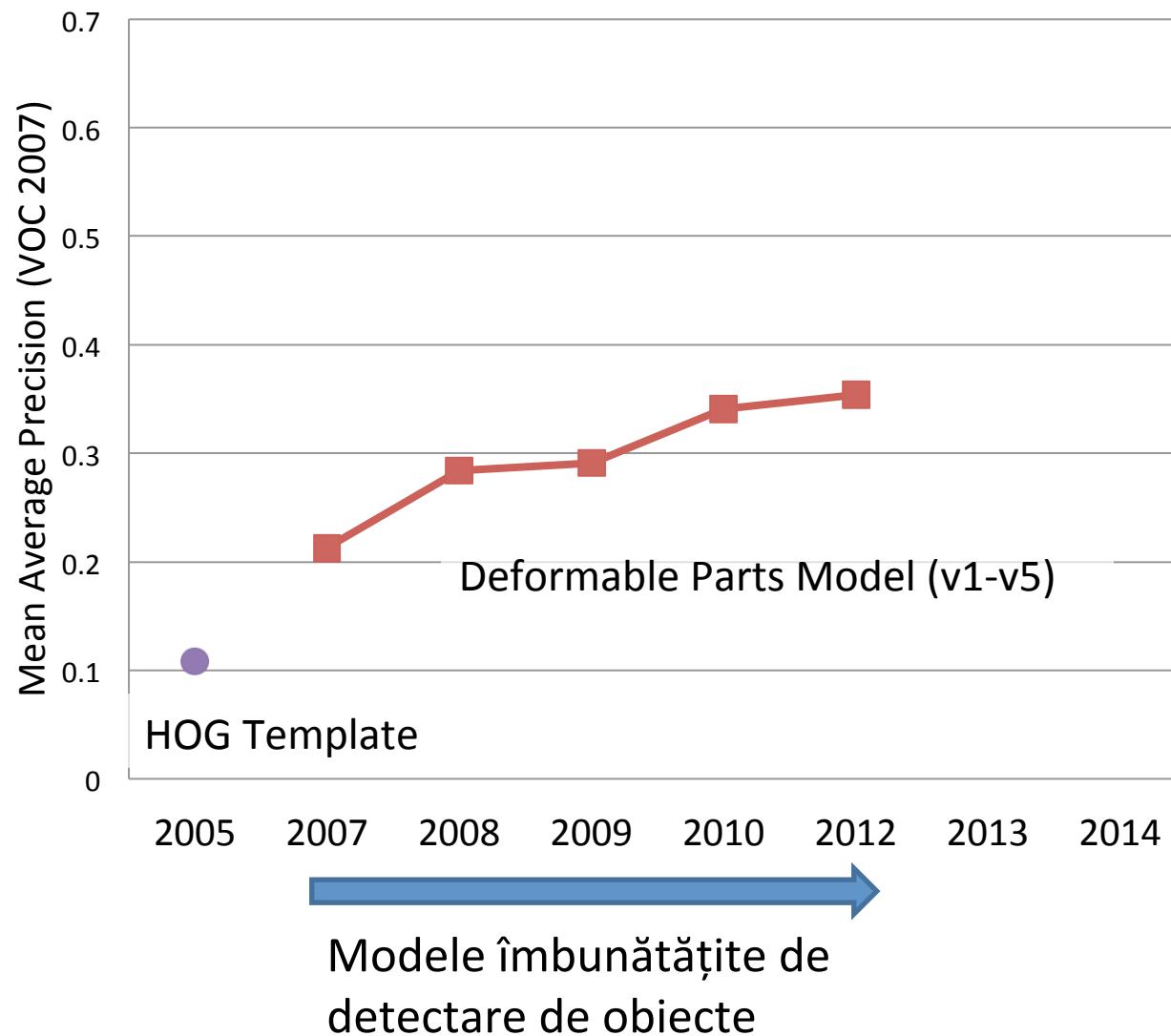


Evoluția performanței detectării de obiecte



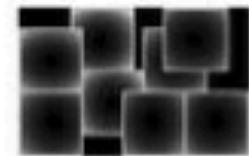
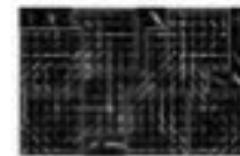
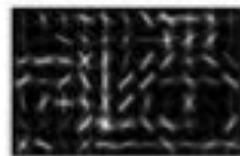
Statistical Template
Matching

Evoluția performanței detectării de obiecte



Modele deformabile cu părți

- folosește drept caracteristici HOG
- învățare discriminativă cu variabile latente (poziția părților)
- mAP: 33.4% - 33.7%

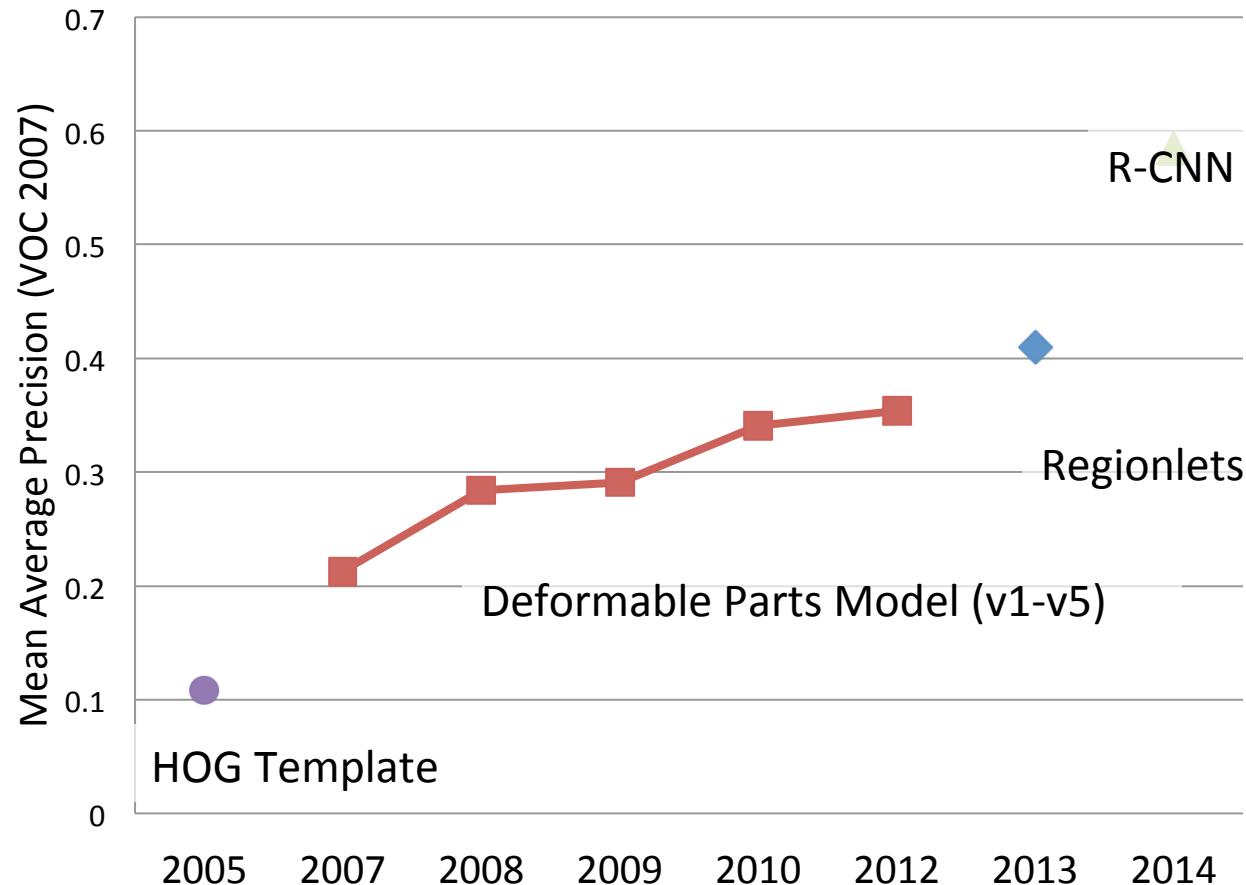


P.F. Felzenszwalb et al., "Object Detection with Discriminatively Trained Part-Based Models", PAMI 2010.

J.J. Lim et al., "Sketch Tokens: A Learned Mid-level Representation for Contour and Object Detection", CVPR 2013.

X. Ren et al., "Histograms of Sparse Codes for Object Detection", CVPR 2013.

Evoluția performanței detectării de obiecte



Punct cheie: învață caracteristici mai bune (discriminative) dintr-un volum de date mare (adnotate) și folosește la probleme cu puține date

Caracteristici mai bune