

Computer Vision 3

Ş.I. dr. ing. Mihai DOGARIU

www.mdogariu.aimultimedialab.ro

Structura cursului



- M1. Introducere
- M2. Fundamentele Învățării Adânci (Deep Learning Fundamentals)
- M3. Învățare Adâncă Supervizată (Supervised Deep Learning)
- M4. Învățare Adâncă Nesupervizată (Unsupervised Deep Learning)
- M5. Învățare Consolidată (Reinforcement Learning)

M3. Învățare Adâncă Supervizată (Supervised Deep Learning)

3.1. Concept Supervised Deep Learning

3.2. Clasificarea imaginilor

3.2. Detectia obiectelor

M3.1. Concept Supervised Deep Learning

Supervised Deep Learning

Învățarea mașinilor (**machine learning**) = spunem despre un sistem că „învață” din experiența E cu privire la o clasă de sarcini de lucru T și o măsură de performanță P, dacă performanța sa în rezolvarea sarcinilor T, măsurată prin P, crește cu experiența E.

Bază de date = o grupare de elemente cu proprietăți comune. Reprezintă „experiență” pe care o întâlnește un algoritm de învățare conform definiției de mai sus.

$$D = \{((x_i, y_i) | T), 1 \leq i \leq M\}$$

input output sarcina dimensiunea

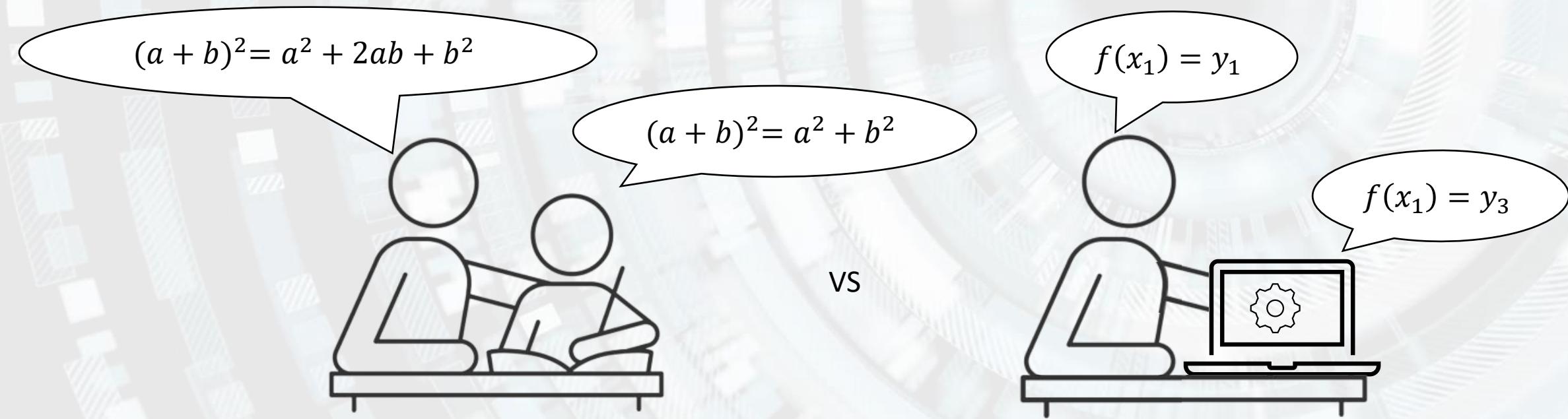
Supervised Deep Learning

$$D = \{((x_i, y_i)|T), 1 \leq i \leq M\} = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), \dots, (x_M, y_M)\}$$

$$\left. \begin{array}{l} f(x_1) = y_1 \\ f(x_2) = y_2 \\ f(x_3) = y_3 \\ \dots \\ f(x_M) = y_M \end{array} \right\} f = ?$$

- Fiecare pereche (x_M, y_M) se mai numește și exemplu de antrenare;
- x_M = vector de intrare;
- y_M = ieșirea reală/eticheta.

Supervised Deep Learning



Supervised Deep Learning

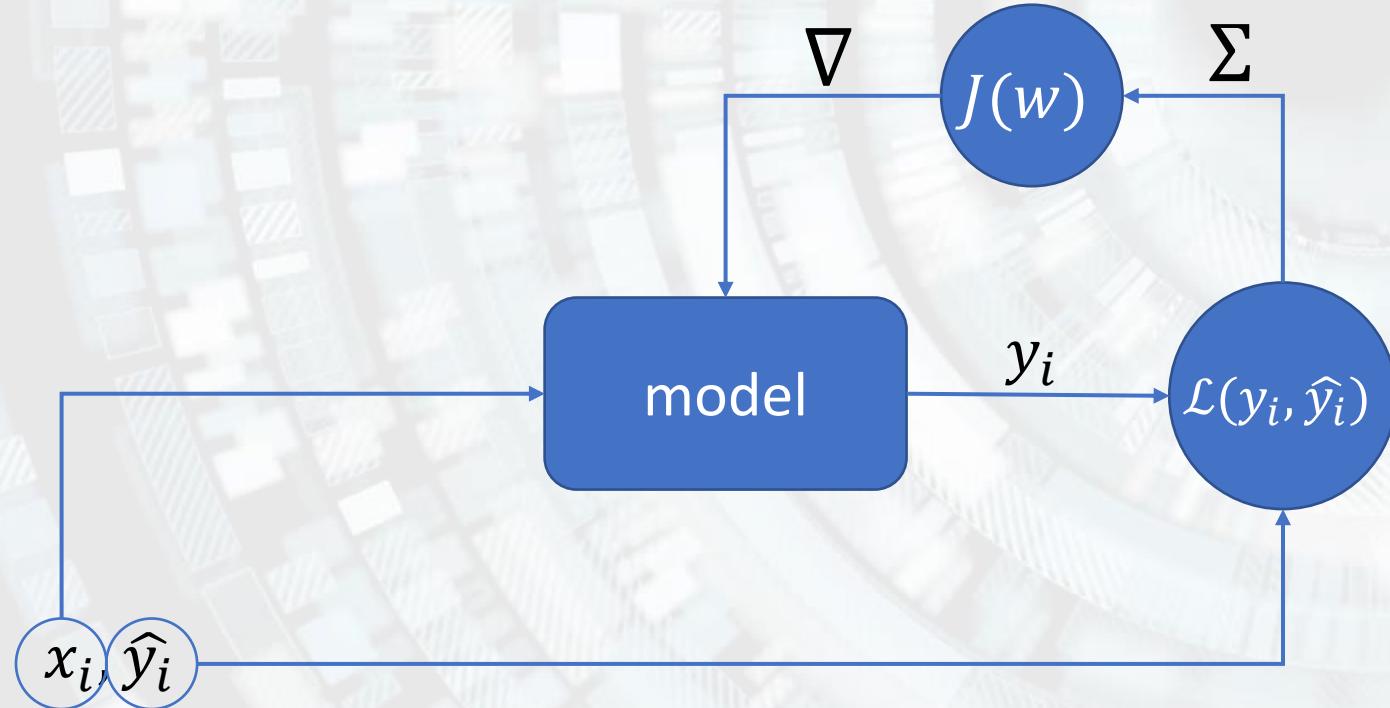
Învățarea supervizată = paradigmă de învățare a mașinilor în care datele de antrenare sunt etichetate. Fiecare exemplu de antrenare este format dintr-un descriptor de trăsături și o etichetă. Scopul învățării supervizate este de a învăța funcția de asociere dintre trăsăturile de intrare și eticheta corespondentă.

Învățarea supervizată adâncă = paradigma învățării supervizate aplicată pe rețele neuronale adânci (adânci = mai multe (>3 , >7 , >30 , >50) straturi).

Supervised Deep Learning

- Toate datele de antrenament conțin o etichetă proprie;
- 2 subcategorii principale:
 1. Clasificare – ne dorim să prezicem o valoare discretă reprezentând cărei clase îi aparține un eșantion de intrare.
 2. Regresie – ne dorim să prezicem valori continue adaptate modelului care descrie baza de date.
- Coliziuni ale definițiilor:
 - Un clasificator poate prezice o valoare continuă sub forma unei distribuții de probabilitate.
 - Un regresor poate prezice o valoare discretă sub forma unei cantități întregi.

Supervised Deep Learning



M3.2. Clasificarea imaginilor

Clasificarea imaginilor

Clasificarea imaginilor = sarcina de a atribui o etichetă/clasă unei imagini.

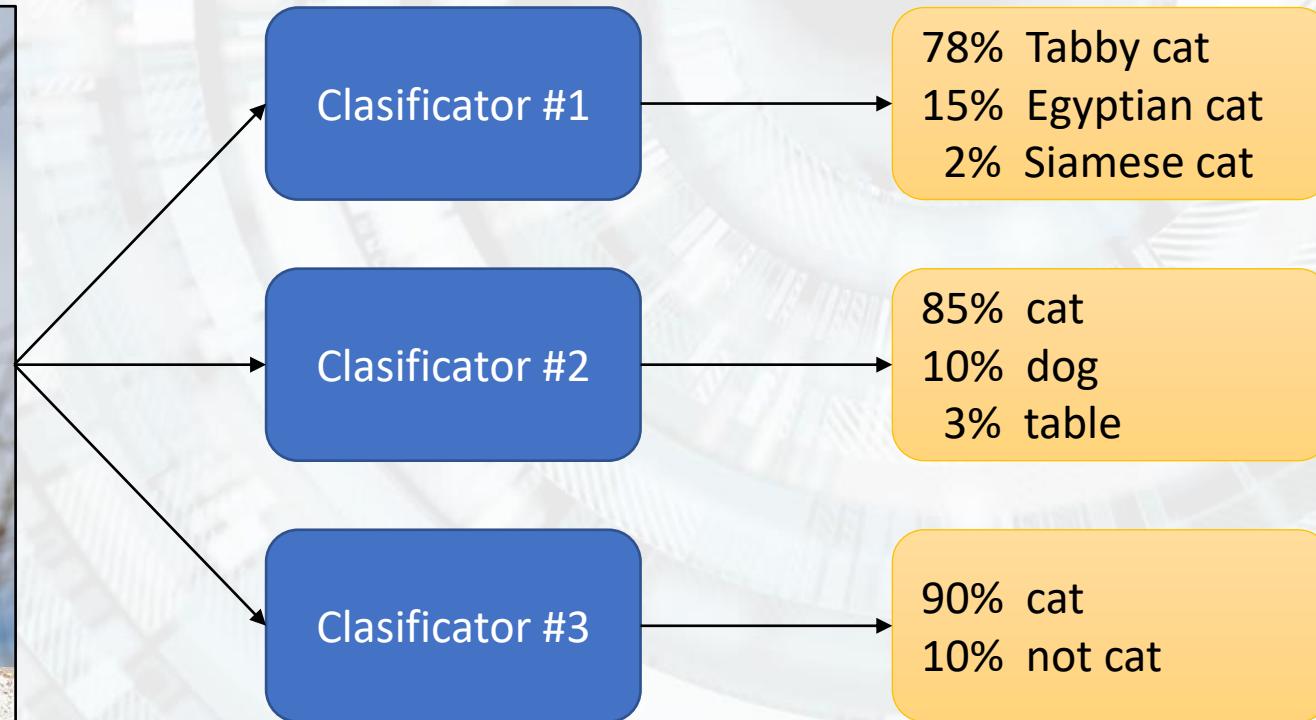
Caracteristici:

- Conținutul imaginii este tratat ca un întreg \Leftrightarrow eticheta descrie întreaga imagine, nu doar o porțiune din ea.
- Orice imagine aparține unei singure clase.
- Ieșirea unui astfel de clasificator este, de obicei, o probabilitate, nu o decizie categorică.
- De obicei, sunt clasificate doar imagini în care obiectul/conceptul de interes ocupă o pondere semnificativă din imagine sau în care se găsește doar obiectul/conceptul de interes.
- Depinde foarte mult de aspectele calitative și cantitative ale bazei de date de antrenare.
- Clasificatorii multi-clasă au, de obicei, ultimul strat complet conectat și activare de tipul softmax.

Clasificarea imaginilor



Baza de date



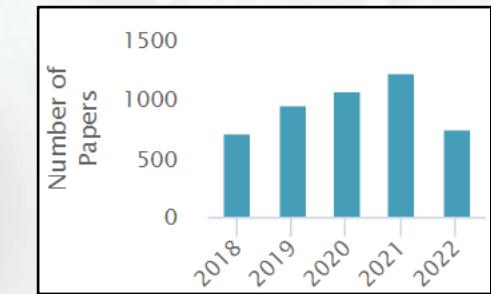
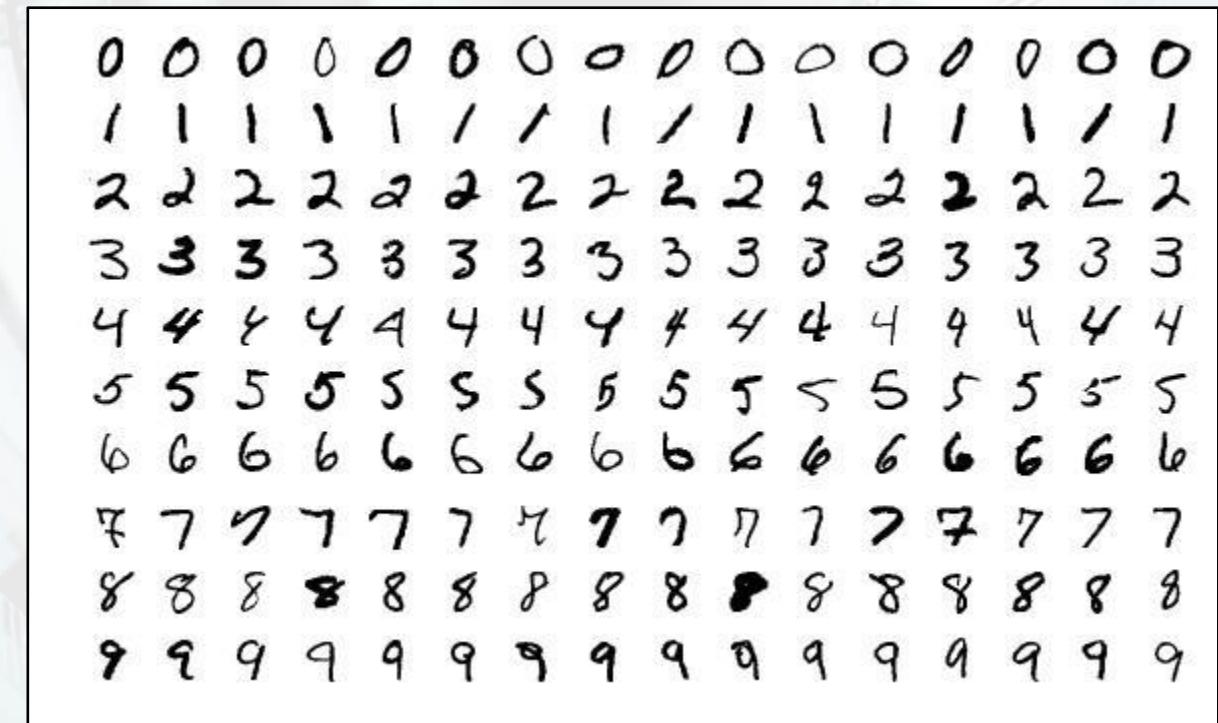
Modele

Metrici

Clasificarea imaginilor – datasets

MNIST [1]

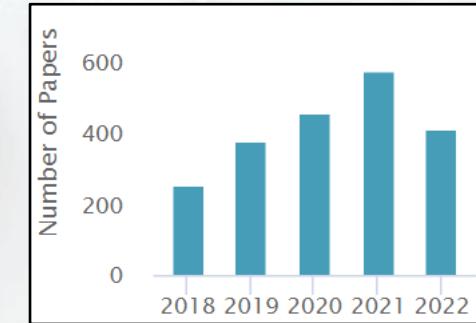
- Contine cifre scrise de mâнă;
- Train: 60k exemple;
- Test: 10k exemple;
- Dimensiune: 28x28 pixeli;
- Nuanțe de gri;
- Dimensiune normalizată;
- Imagini centrate;



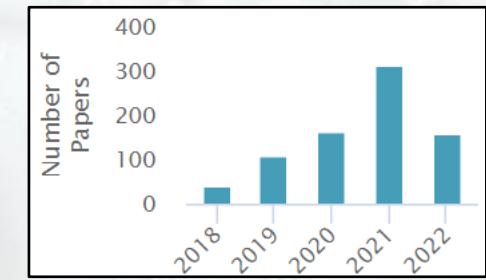
Clasificarea imaginilor – datasets

SVHN [2]

- Conține numere ale caselor de pe stradă;
- Train: 73k exemple;
- Test: 26k exemple;
- Extra train: 531k exemple;
- Dimensiune: 32x32 pixeli;
- Imagini color;
- Imagini decupate;
- Conține și parțial distractori.

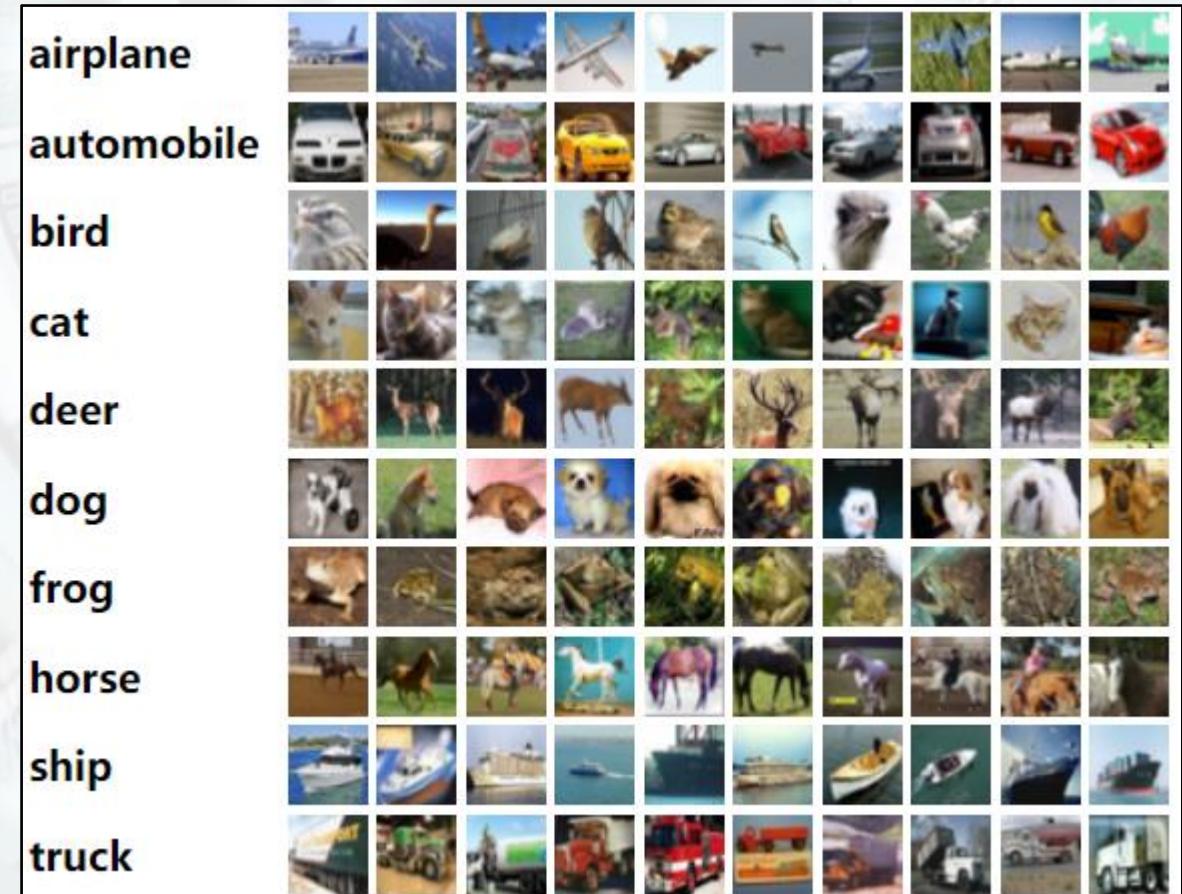


Clasificarea imaginilor – datasets

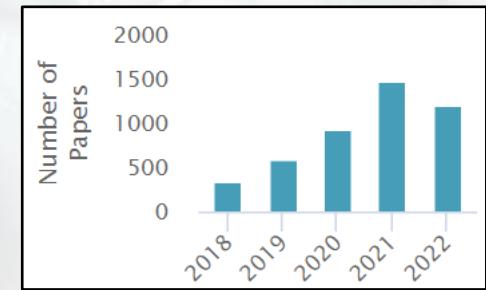


CIFAR-10 (Plantere) [3]

- Subset al Tiny Images Dataset;
- Conține imagini ale unor obiecte din viață reală;
- Train: 50k exemple;
- Test: 10k exemple;
- Dimensiune: 32x32 pixeli;
- 10 clase diferite;
- 6000 exemple/clasă;
- Imagini color;
- Clase echilibrate.



Clasificarea imaginilor – datasets



CIFAR-100 [3]

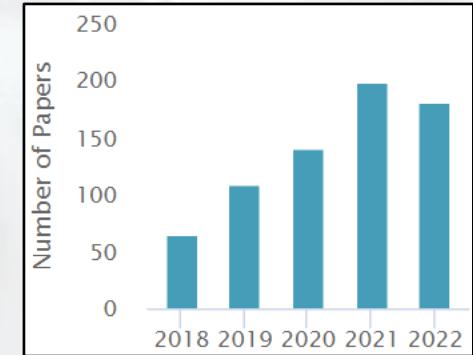
- Subset al Tiny Images Dataset;
- La fel ca CIFAR-10;
- 100 clase diferite;
- 600 exemple/clasă;
- 20 superclase (grosiere);
- 5 subclase (fine) /superclasă;

Superclass	Classes
aquatic mammals	beaver, dolphin, otter, seal, whale
fish	aquarium fish, flatfish, ray, shark, trout
flowers	orchids, poppies, roses, sunflowers, tulips
food containers	bottles, bowls, cans, cups, plates
fruit and vegetables	apples, mushrooms, oranges, pears, sweet peppers
household electrical devices	clock, computer keyboard, lamp, telephone, television
household furniture	bed, chair, couch, table, wardrobe
insects	bee, beetle, butterfly, caterpillar, cockroach
large carnivores	bear, leopard, lion, tiger, wolf
large man-made outdoor things	bridge, castle, house, road, skyscraper
large natural outdoor scenes	cloud, forest, mountain, plain, sea
large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
medium-sized mammals	fox, porcupine, possum, raccoon, skunk
non-insect invertebrates	crab, lobster, snail, spider, worm
people	baby, boy, girl, man, woman
reptiles	crocodile, dinosaur, lizard, snake, turtle
small mammals	hamster, mouse, rabbit, shrew, squirrel
trees	maple, oak, palm, pine, willow
vehicles 1	bicycle, bus, motorcycle, pickup truck, train
vehicles 2	lawn-mower, rocket, streetcar, tank, tractor

Clasificarea imaginilor – datasets

Places-365 [4]

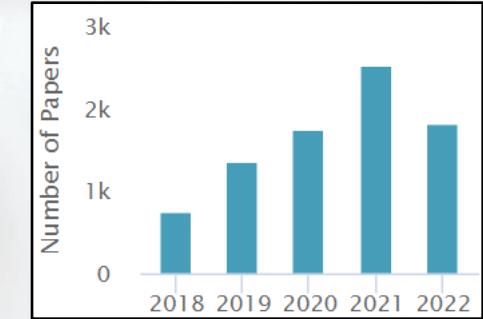
- Utilizată pentru recunoașterea scenelor;
- >10M imagini;
- 434 clase;
- 5-30k imagini/clasă;
- Imagini color;
- Dimensiuni variabile;
- Generată automat => există zgomot.



Clasificarea imaginilor – datasets

ImageNet [5]

- Conține imagini corespunzătoare synsets ale ierarhiei WordNet;
- 14M imagini color;
- 21841 clase;
- ~650 imagini/clasă;
- Dimensiuni variabile;
- Utilizată pentru competiția ILSVRC (ImageNet Large Scale Visual Recognition Challenge): 1.4M imagini, 1000 clase;
- Generată automat => zgomot;
- Cea mai populară în procesarea imaginilor.

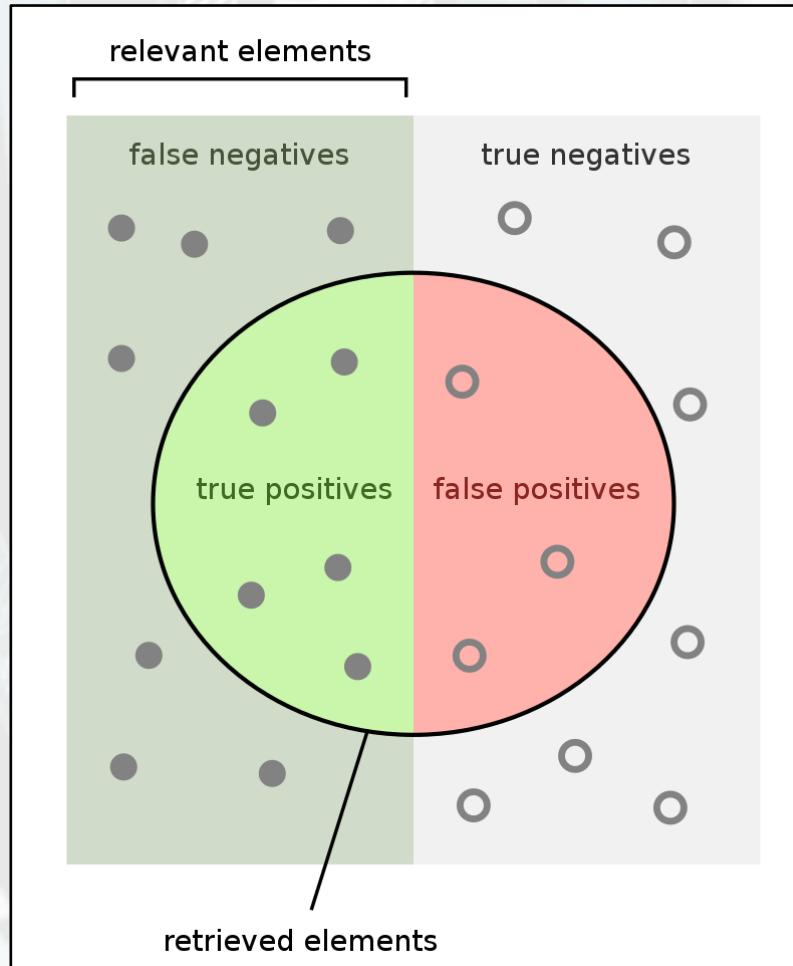


Clasificarea imaginilor – metrici

Metrică = metodă cantitativă de a măsura performanța unui sistem. Oferă valoari numerice ordonabile pentru cuantificarea progresului făcut de un model antrenabil.

- Se calculează la finalul unei epoci, pe întreaga bază de date (train, val, test).
- Depinde de sarcina de lucru – diferite metrici pentru diferite tipuri de sisteme.
- De obicei, nu este diferențiabilă, deci nu poate fi utilizată pentru a conduce procesul de învățare.
- În unele cazuri, poate fi sinonimă cu funcția de cost (e.g. MAE, MSE).
- Trebuie interpretată în context. De obicei, sunt menționate valorile/intervalele de valori ce reprezintă situația dorită.

Clasificarea imaginilor – metrici



How many retrieved items are relevant?

Precision = $\frac{\text{true positives}}{\text{true positives} + \text{false positives}}$

How many relevant items are retrieved?

Recall = $\frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$

*Doar pentru clasificare binară!

Clasificarea imaginilor – metrici

Matrice de confuzie:

		Prezis	
		Pozitiv	Negativ
Real	Pozitiv	True positive (tp)	False negative (fn)
	Negativ	False positive (fp)	True negative (tn)

$$\text{precizie}(P) = \frac{tp}{tp + fp}$$

$$\text{reamintire}(R) = \frac{tp}{tp + fn} = \text{rata TP (TPR)}$$

$$\text{rata TN (TNR)} = \frac{tn}{tn + fp}$$

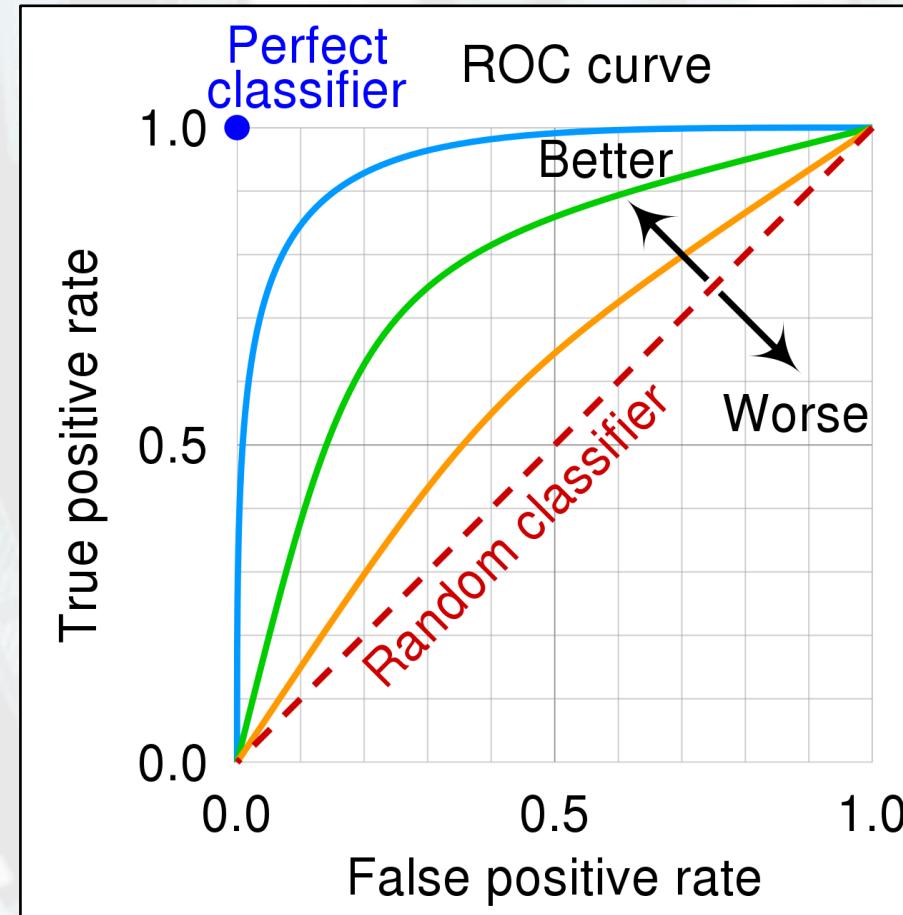
$$\text{rata FP (FPR)} = \frac{fp}{tn + fp}$$

$$\text{acuratețea (ACC)} = \frac{tp + tn}{tp + tn + fp + fn}$$

$$F - score (F) = 2 \frac{PR}{P + R}$$

*Doar pentru clasificare binară!

Clasificarea imaginilor – metrici



ROC = Receiver Operating Characteristic
AUC = Area Under the (ROC) Curve
AUC=1 => clasificator perfect
AUC=0.5 => clasificator complet aleator
AUC=0 => clasificator anti-perfect

Clasificarea imaginilor – metrici

Acuratețea pentru clase dezechilibrate (imbalanced dataset)

- Pentru un clasificator binar, acuratețea este definită astfel:

$$ACC = \frac{tp + tn}{tp + tn + fp + fn}$$

- Scenariu:

- Considerăm o bază de date cu 95 exemple negative și 5 exemple pozitive.
- Un clasificator care prezice clasa negativă în 100% din cazuri obține o acuratețe de 95%, ceea ce este înselător.
- Soluție: folosim acuratețea echilibrată (balanced accuracy).

Clasificarea imaginilor – metrici

Acuratețea pentru clase dezechilibrate (imbalanced dataset)

- Acuratețea echilibrată este definită astfel:

$$ACC = \frac{TPR + TNR}{2} = \frac{\frac{tp}{tp + fn} + \frac{tn}{tn + fp}}{2}$$

- Scenariu:

- Considerăm o bază de date cu 95 exemple negative și 5 exemple pozitive.
- Un clasificator care prezice clasa negativă în 100% din cazuri

		Prezis	
		Pozitiv	Negativ
Real	Pozitiv	tp = 0	fn = 5
	Negativ	fp = 0	tn = 95

$$ACC = \frac{TPR + TNR}{2} = \frac{\frac{0}{0+5} + \frac{95}{95+0}}{2} = 0.5$$

Clasificarea imaginilor – metrici

Acuratețea pentru clasificarea multi-clasă

$$ACC = \frac{\text{clasificări corecte}}{\text{clasificări totale}}$$

- E.g.: din 100 de exemple analizate, 83 au fost clasificate corect => 83% acuratețe.

Acuratețea top-k

- Pentru un exemplu din baza de date se calculează probabilitatea relativă de apartenență la fiecare clasă.
- Se ordonează probabilitățile de ieșire.
- Dacă n dintre primele k cel mai bine cotate clase se numără și clasa reală => clasificare corectă.

Clasificarea imaginilor – metrici



Clasificator

35% cat
3% table
1% velociraptor
12% airplane
6% truck
40% dog
2% baseball
1% mouse

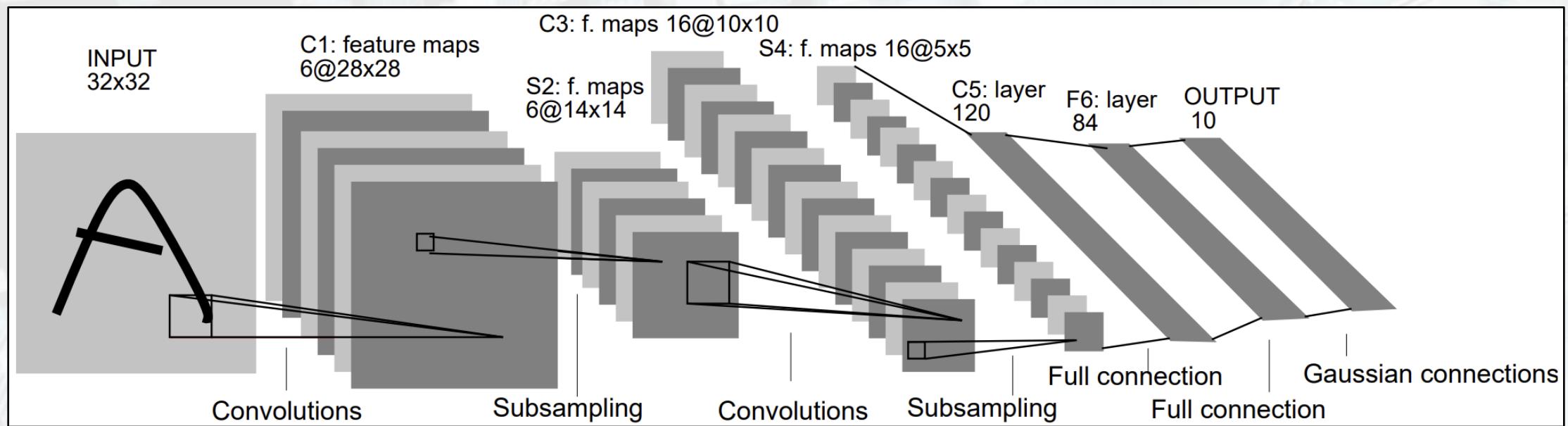
40% dog]]
35% cat]]
12% airplane]]
6% truck]]
3% table]]
2% baseball]]
1% velociraptor]]
1% mouse]]

top-1: miss
top-2: hit
top-3: hit
top-5: hit

Clasificarea imaginilor – modele

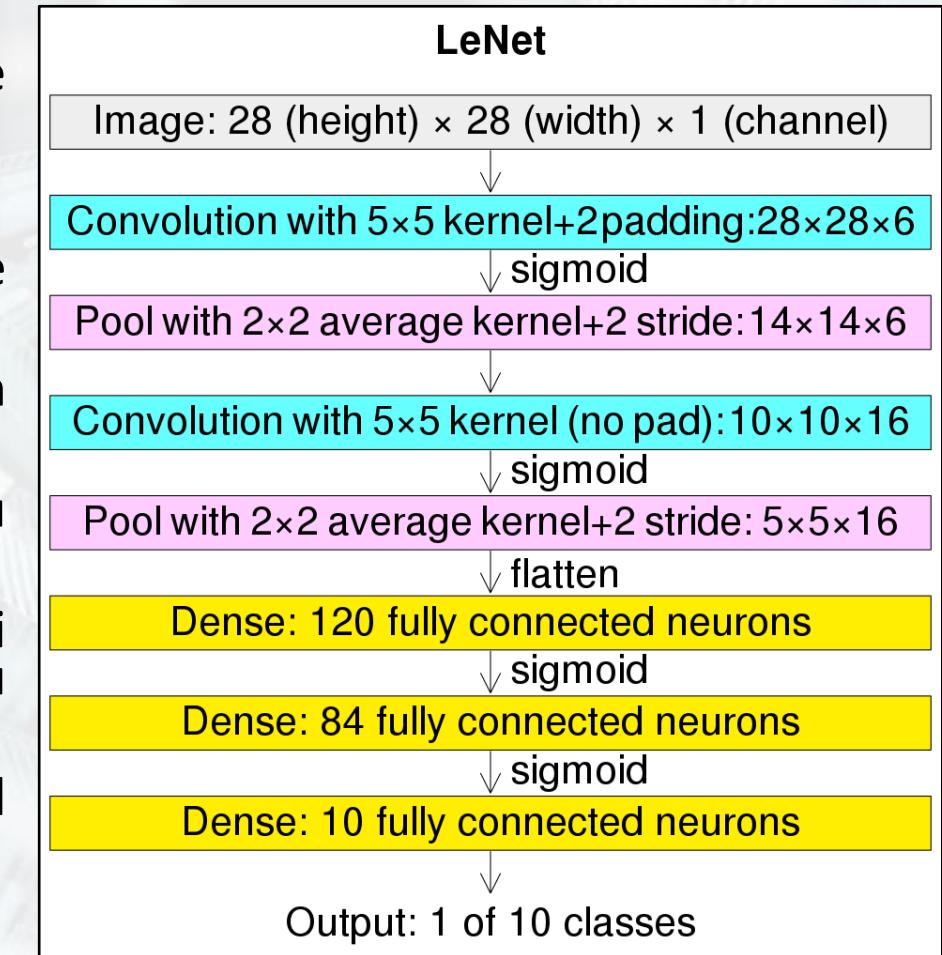
- Modelele de rețele neuronale reprezintă partea centrală a sistemelor de clasificare a imaginilor.
- Tradițional, s-au concentrat pe rețele convoluționale (complet convoluționale sau conv + fully-connected).
- Au reprezentat punctul de atracție al domeniului de deep learning.
- Au fost preluate și în alte domenii (audio, text, meta).
- Au o gamă largă de aplicații, nu doar clasificare.

Clasificarea imaginilor – LeNet [5]

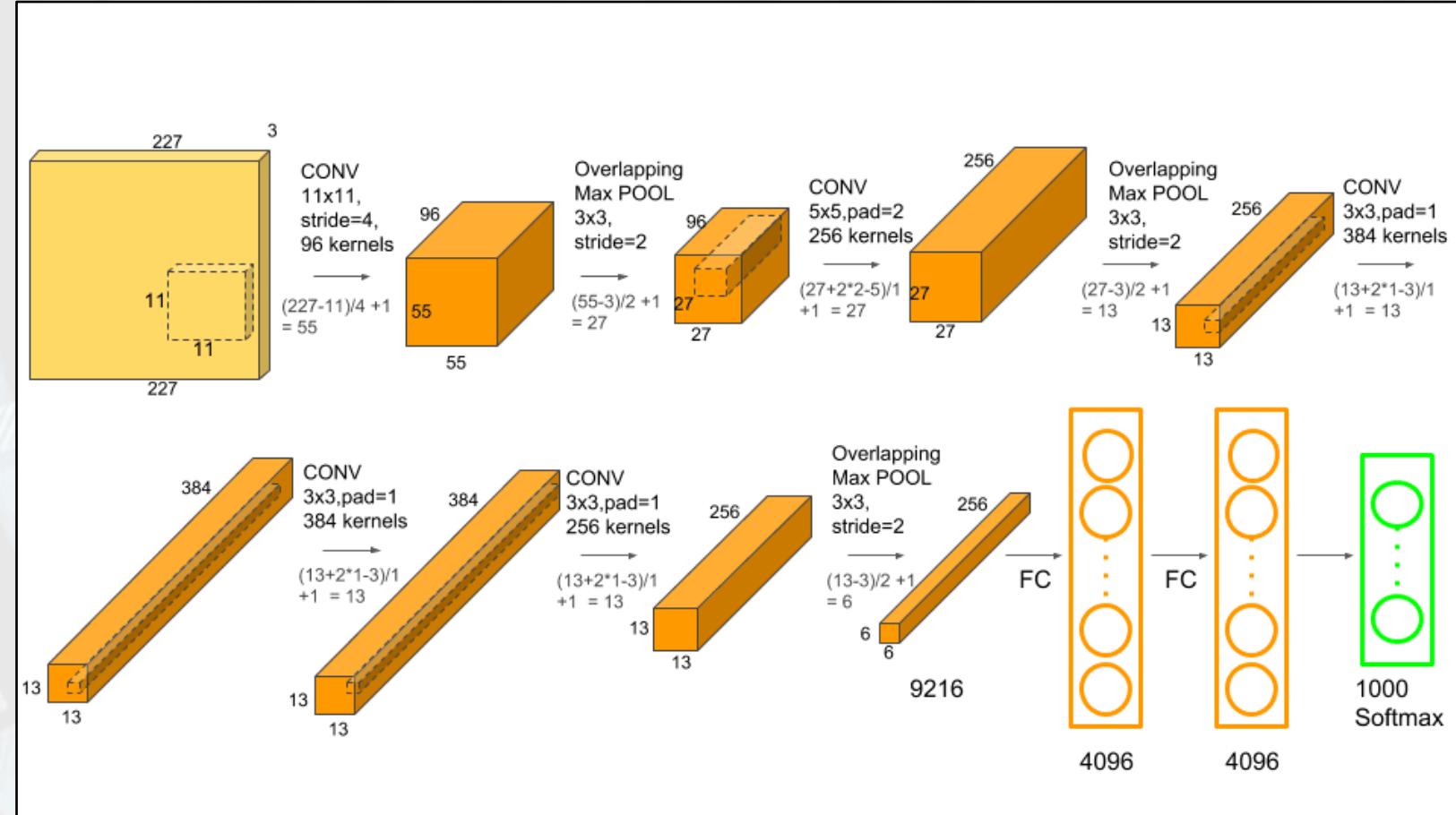


Clasificarea imaginilor – LeNet

- Utilizată pentru a detecta automat codurile poștale scrise de mâna de pe plicuri poștale;
- Prima rețea în care s-a folosit propagarea înapoi;
- A fost introdusă în același timp cu baza de date MNIST (Digits);
- Fiecare strat convecțional este format din convecție, activare și pooling;
- Este utilizat mean/average pooling pentru subeșantionare;
- C5 este un strat convecțional cu nucleul de aceeași dimensiune cu trăsăturile de intrare, echivalent cu un strat fully connected.
- Pe ultimul nivel este folosit un strat fully connected pentru clasificare.
- Varianta LeNet-5 a obținut o acuratețe de 99.2%.

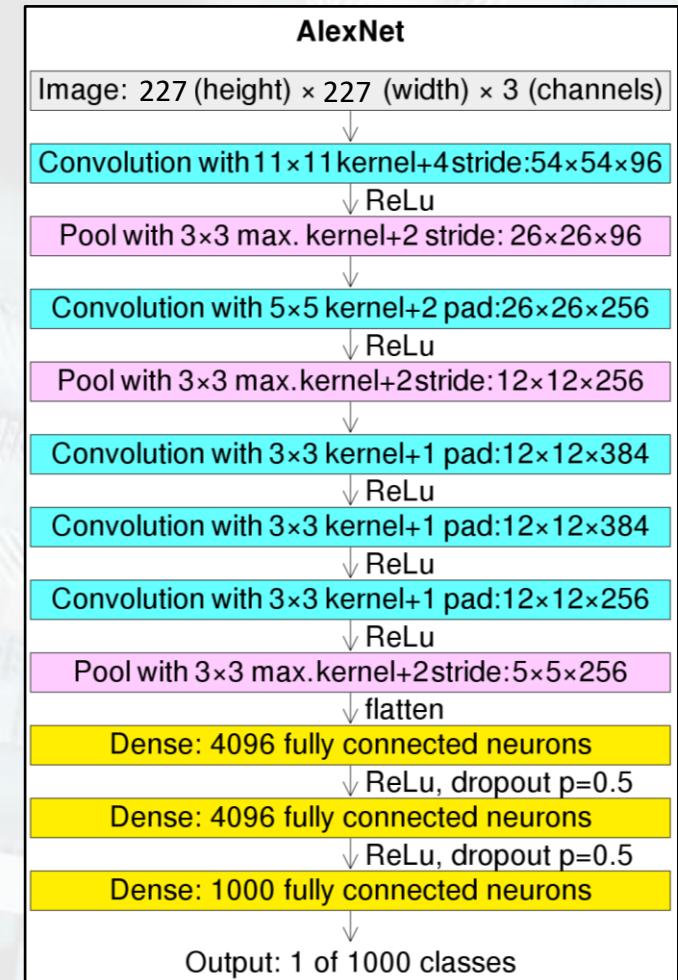


Clasificarea imaginilor – AlexNet [6]

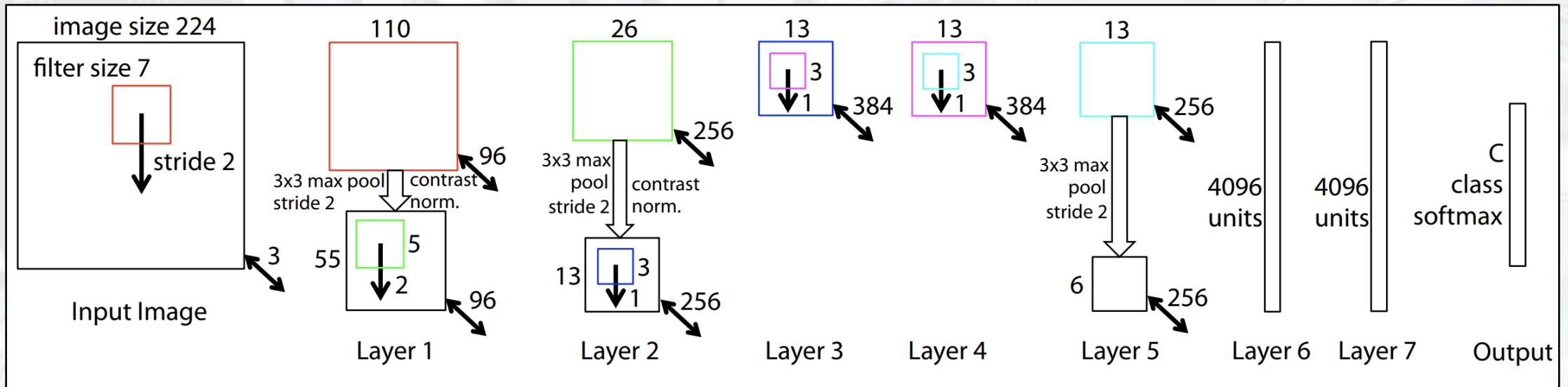


Clasificarea imaginilor – AlexNet

- Utilizată pentru clasificarea imaginilor naturale în cadrul competiției ILSVRC 2012 (câștigător);
- O variantă îmbunătățită a LeNet-5;
- Este utilizat max pooling pentru subeșantionare;
- Utilizează funcția de activare ReLU, în defavoarea sigmoid și tanh;
- A obținut o eroare top-5 de 15.3% (locul 2 – 26.1%);
- Utilizează dropout ca mod de regularizare;
- A demonstrat superioritatea mai multor elemente cheie: adâncimea rețelelor neuronale, funcția de activare ReLU, antrenarea distribuită pe GPU. A adus deep learning în atenția cercetătorilor.



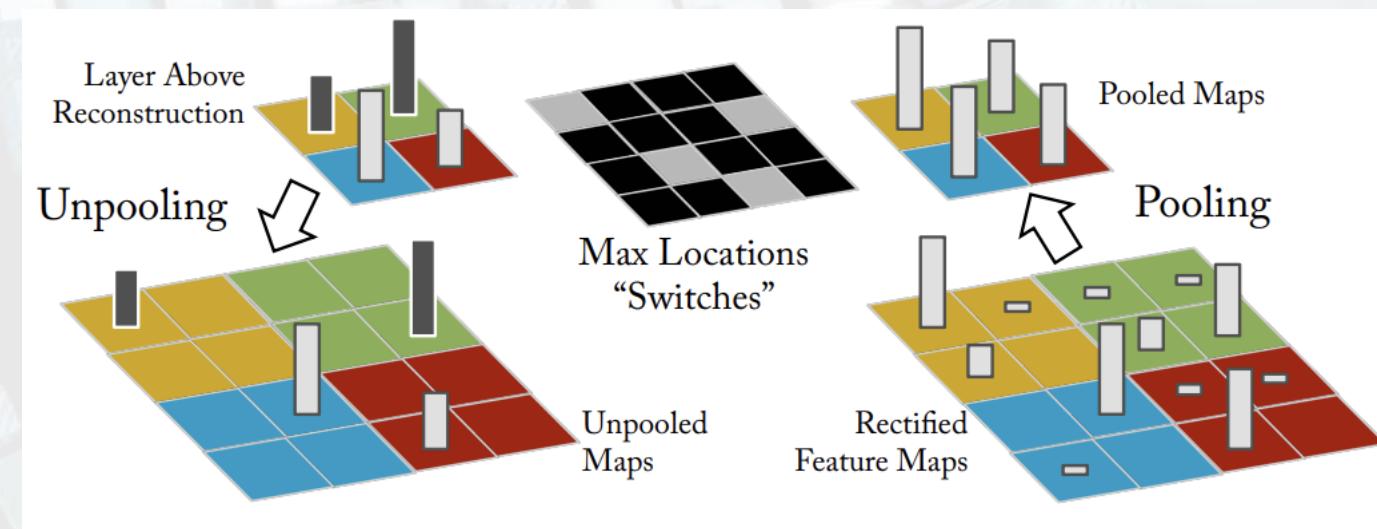
Clasificarea imaginilor – ZFNet [7]



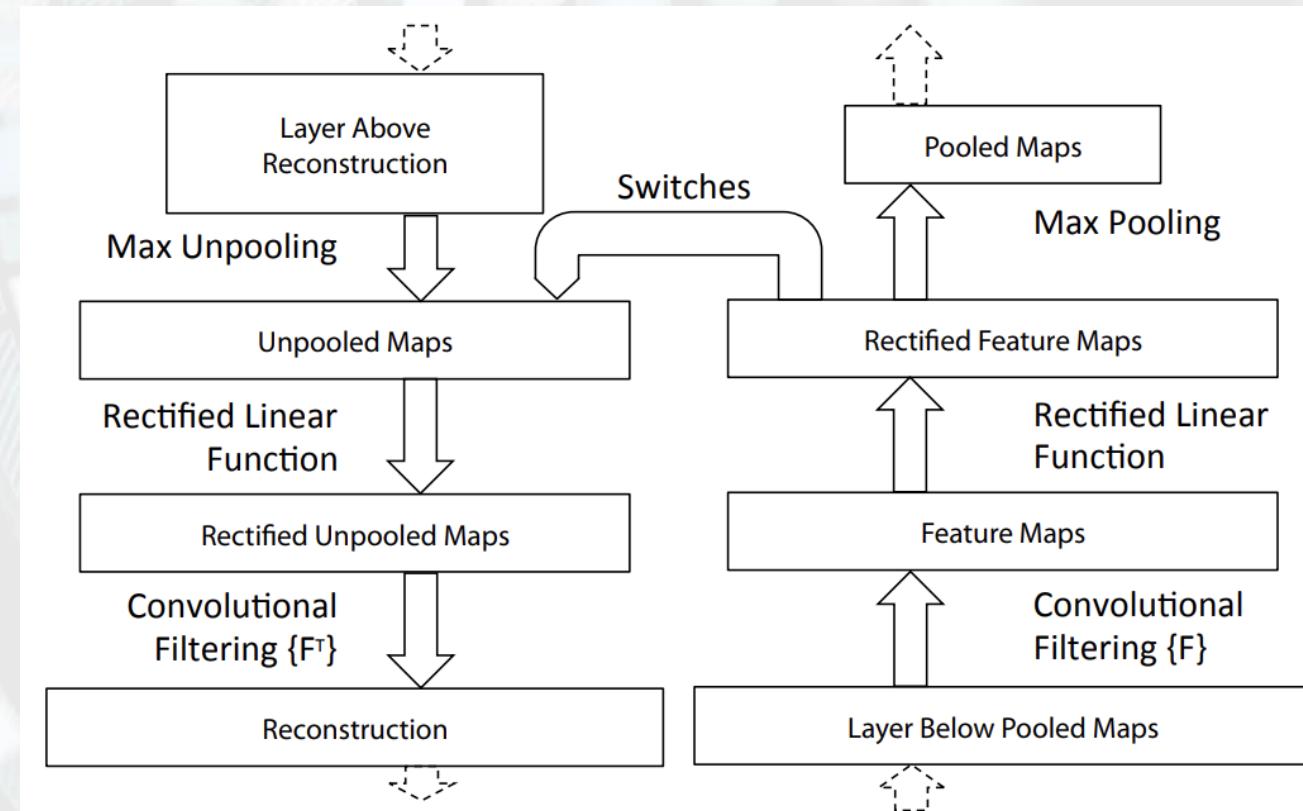
Clasificarea imaginilor – ZFNet

- Câștigător al ILSVRC 2013;
- Arhitectură asemănătoare cu AlexNet – convoluții atât cu filtre, cât și pași de dimensiuni mai mici;
- Introduc vizualizarea componentelor învățate de hărțile de trăsături intermediare cu ajutorul operației de deconvoluție – rețelei de feed-forward î se atașează o rețea complementară, ce execută inversul operațiilor din rețea, în paralel. Reușesc să obțină o aproximare a intrării care a produs o activare anume.

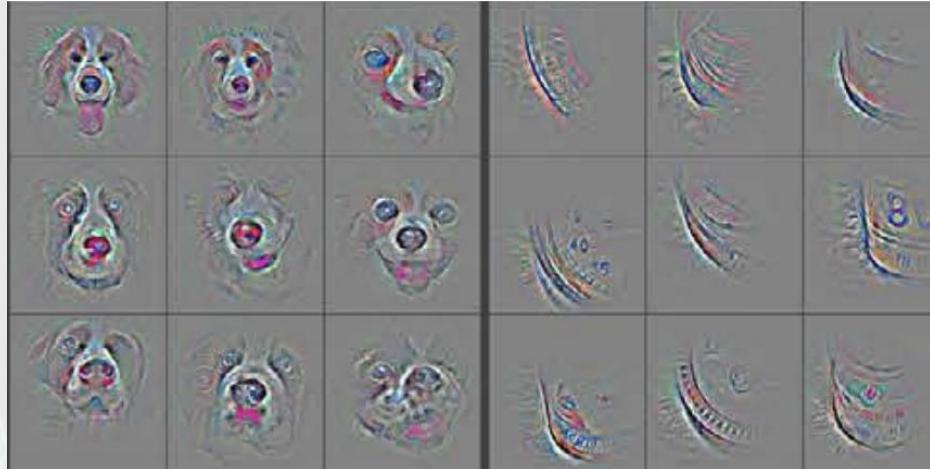
Clasificarea imaginilor – ZFNet



Clasificarea imaginilor – ZFNet

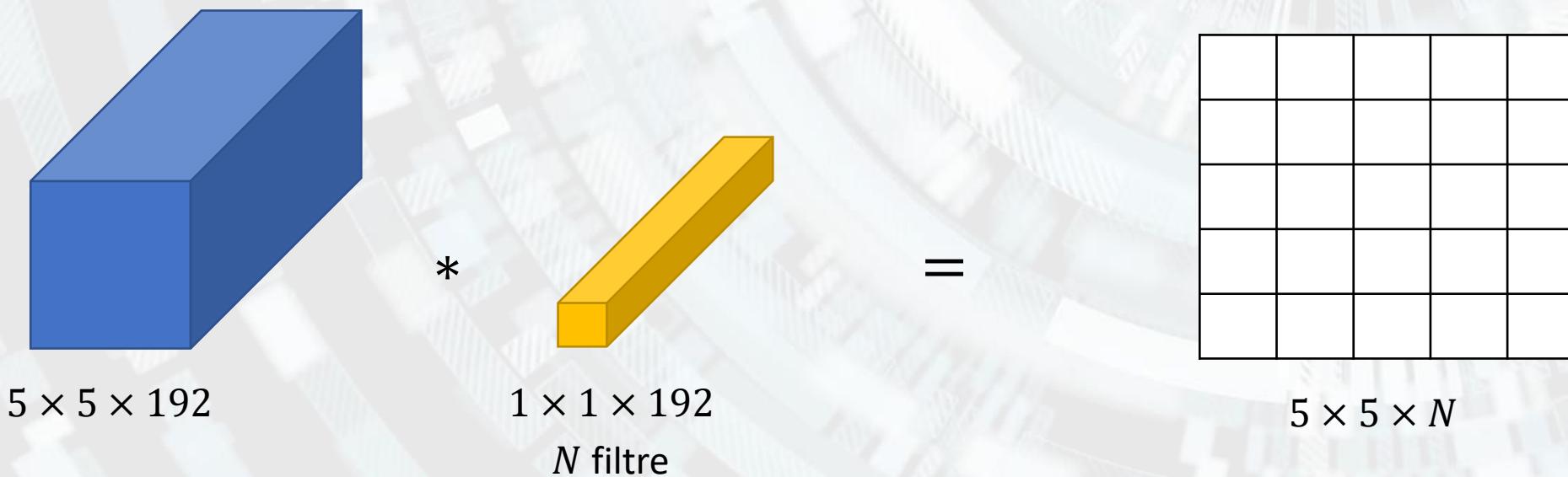


Clasificarea imaginilor – ZFNet

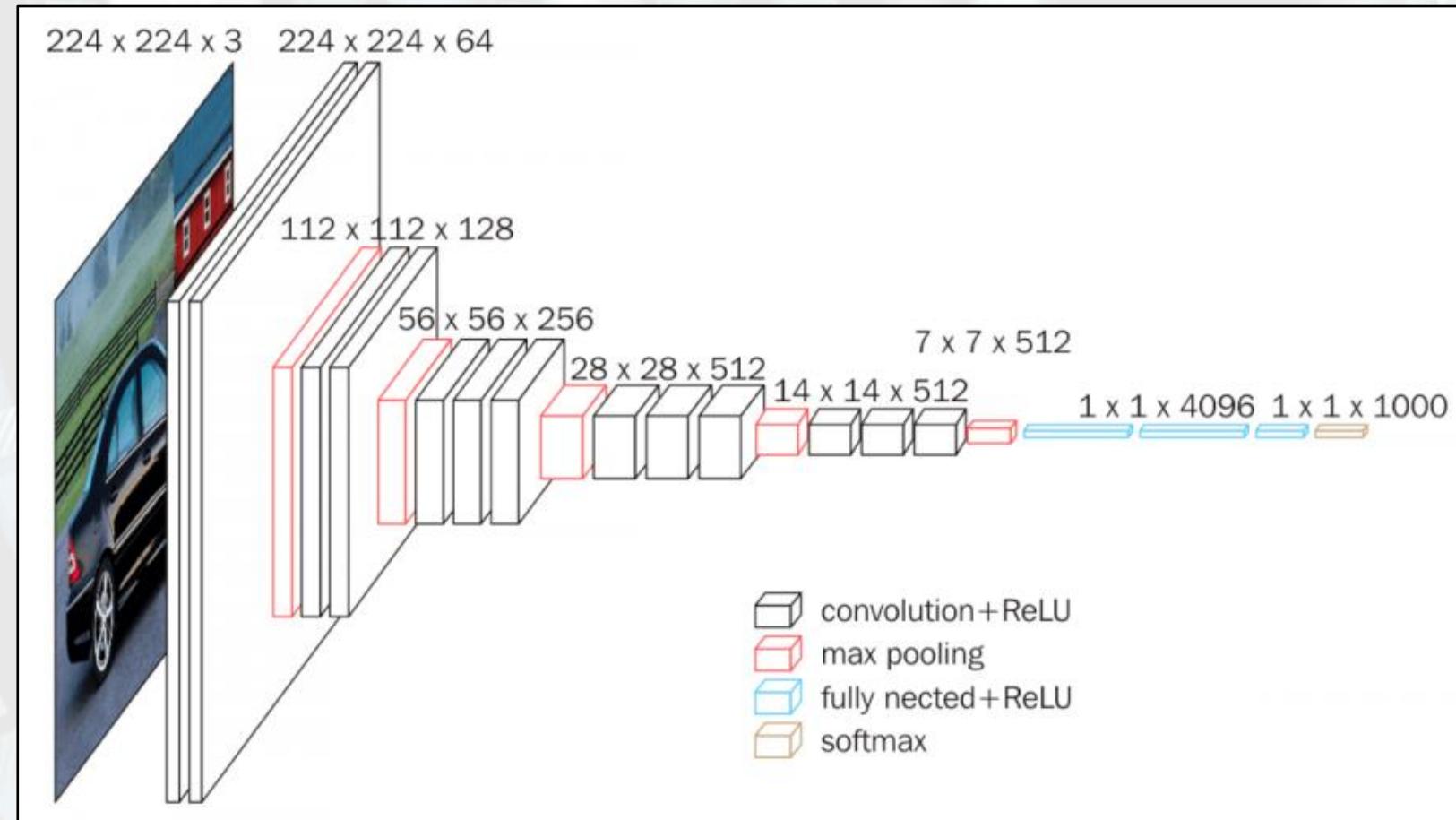


Clasificarea imaginilor – NIN [8]

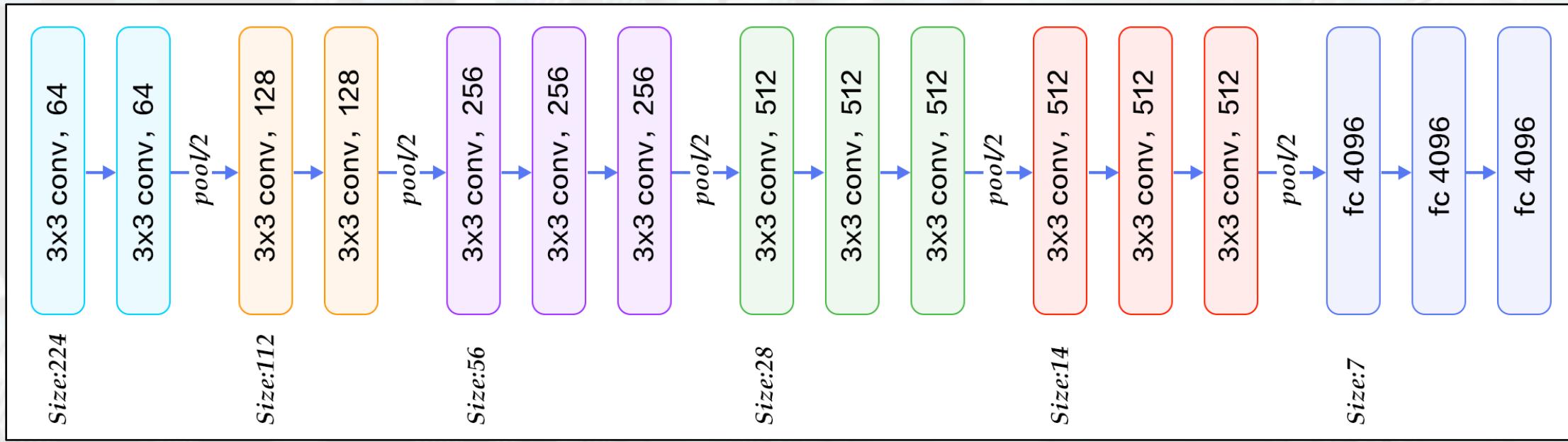
- Network in Network (NIN) introduce ideea de a folosi convoluții 1×1 pentru a micșora dimensionalitatea datelor.
- Convoluția 1×1 este echivalentă cu un strat fully connected.



Clasificarea imaginilor – VGG [9]



Clasificarea imaginilor – VGG



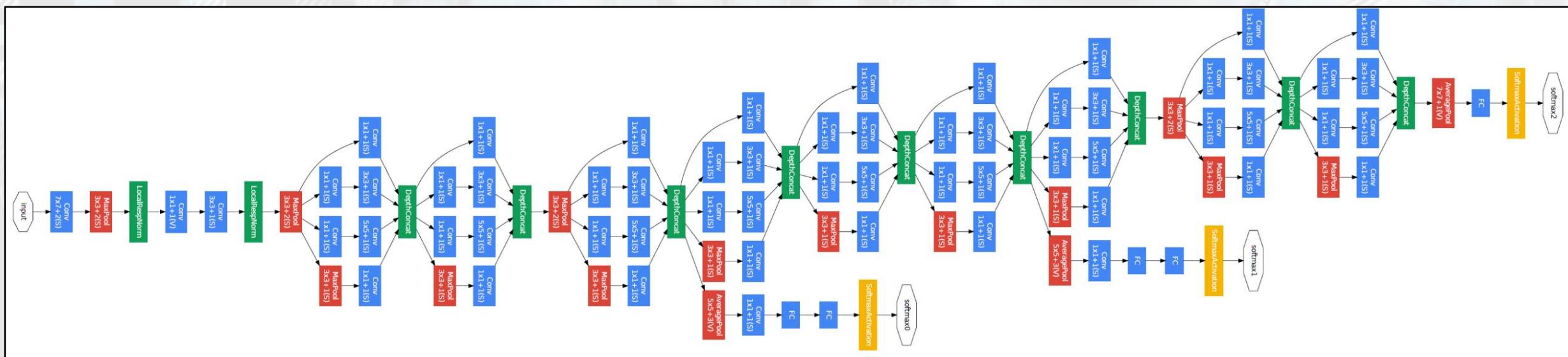
Clasificarea imaginilor – VGG

- Model similar cu AlexNet;
- Folosește convoluții cu nucleu de dimensiune mică, asemănător ZFNet;
- Adaugă mai multă adâncime rețelei, demonstrând că rețelele adânci reușesc să capteze mai multă informație;
- Introduc o modularizare a rețelei prin repetarea unei configurații de convoluții de mai multe ori, într-un singur modul.
- 2 variante populare: VGG-16, VGG-19.
- Procesarea intrărilor multi-scală: imaginea este inițial scalată la o valoare între 256 și 512, după care se decupează ferestre de 224 x 224 pixeli care sunt folosite împreună la antrenare, obținând un fel de regularizare.

Clasificarea imaginilor – GoogLeNet [10]

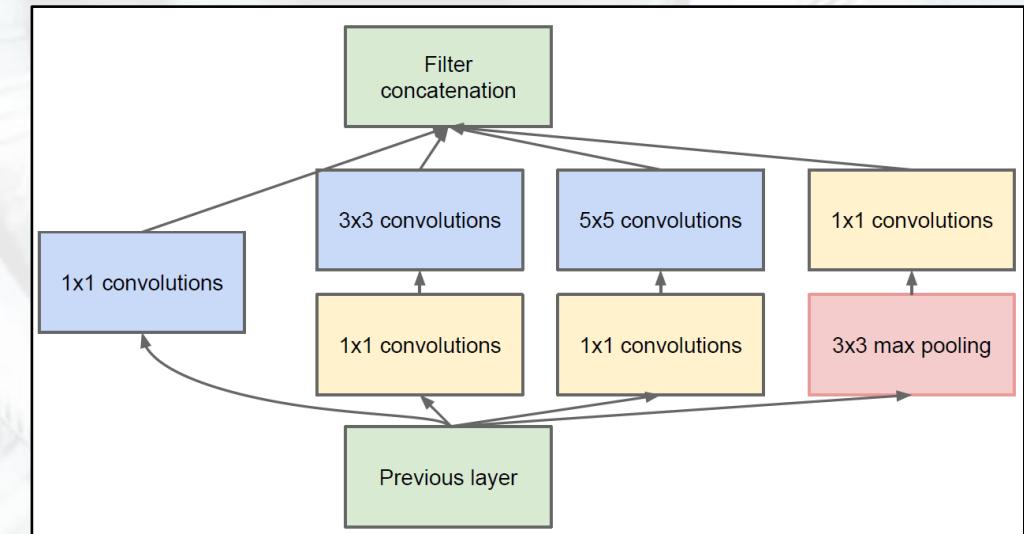


Clasificarea imaginilor – GoogLeNet [9]

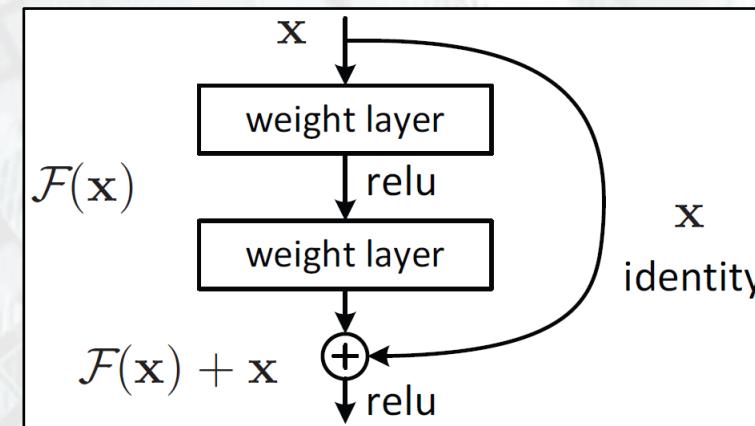
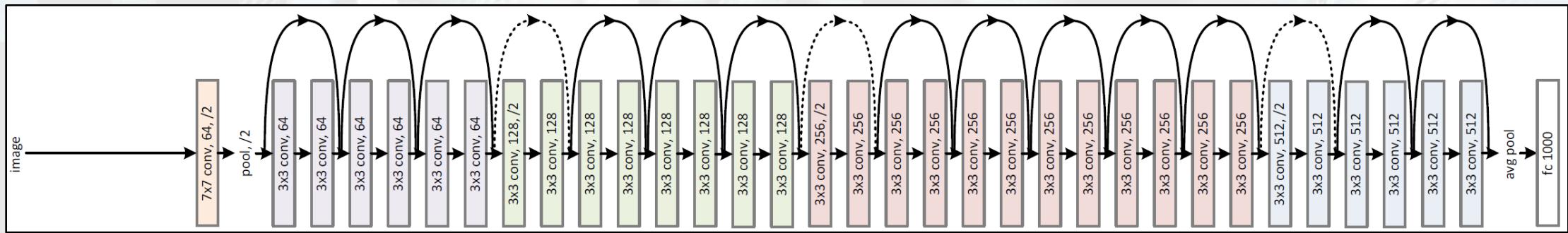


Clasificarea imaginilor – GoogLeNet

- Câștigător al ILSVRC 2014;
- Cunoscută și sub denumirea de Inception (v1, v2, v3, v4);
- Se bazează pe modulele Inception;
- Clasificatorii auxiliari sunt utilizati pentru a combate vanishing gradient în timpul antrenării. În timpul testării se renunță la ei.
- Utilizează extensiv conoluțiile 1×1 pentru a reduce dimensionalitatea și a economisi resurse de calcul;
- Reușește să depășească AlexNet în performanțe, iar numărul de parametri este redus de 12 ori.



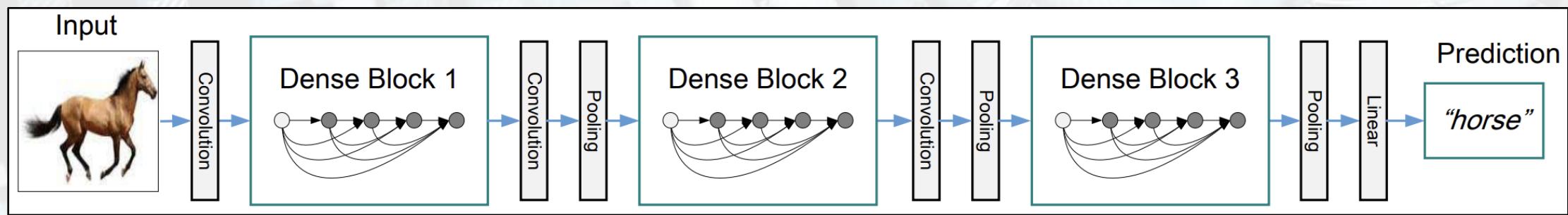
Clasificarea imaginilor – ResNet [11]



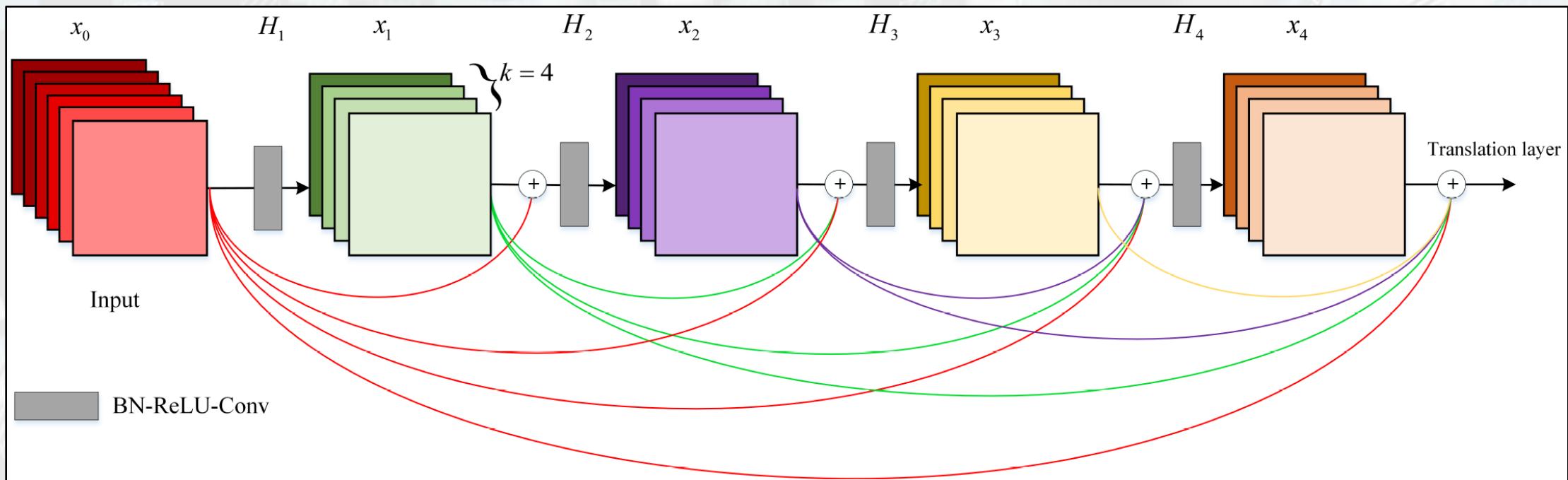
Clasificarea imaginilor – ResNet [11]

- Câștigător al ILSVRC 2015;
- Cel mai popular model la ora actuală (140k citări);
- Utilizează scurtături (skip connections) pentru a sări peste unele straturi;
- Folosește module, asemănător VGG și Inception;
- Rețelele prea adânci nu mai reușesc să propage cu succes gradienții înapoi și ajung să se „degradeze” – soluția: utilizarea skip connections. Acestea ajută și în cazul problemei „vanishing gradients”.
- Are multe variante asociate: ResNe[x]t-34/50/101/152.

Clasificarea imaginilor – DenseNet [12]



Clasificarea imaginilor – DenseNet

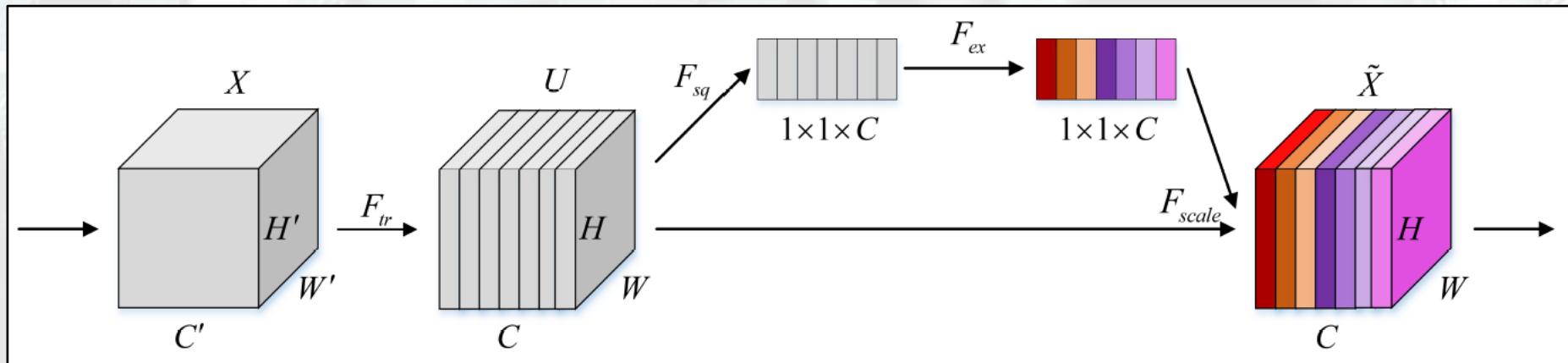


Structura unui bloc dens conectat cu un factor de creștere $k=4$ [13]

Clasificarea imaginilor – DenseNet

- Continuă ideea blocurilor modulare, asemănător ResNet;
- Introduc conexiunile dense: fiecare strat dintr-un bloc este conectat la toate straturile care îi urmează din blocul respectiv;
- Blocurile dense sunt utilizate pentru a rezolva problema „vanishing gradient”: fiecare strat are acces direct la gradienții din straturile care îi urmează => nu mai există pericolul ca aceștia să se diminueze excesiv până când sunt propagați către straturile incipiente;
- Adaptarea numărului de canale se realizează cu convoluții 1×1 ;
- Micșorarea hărților de trăsături se realizează cu straturi de pooling.

Clasificarea imaginilor – SENet [14]

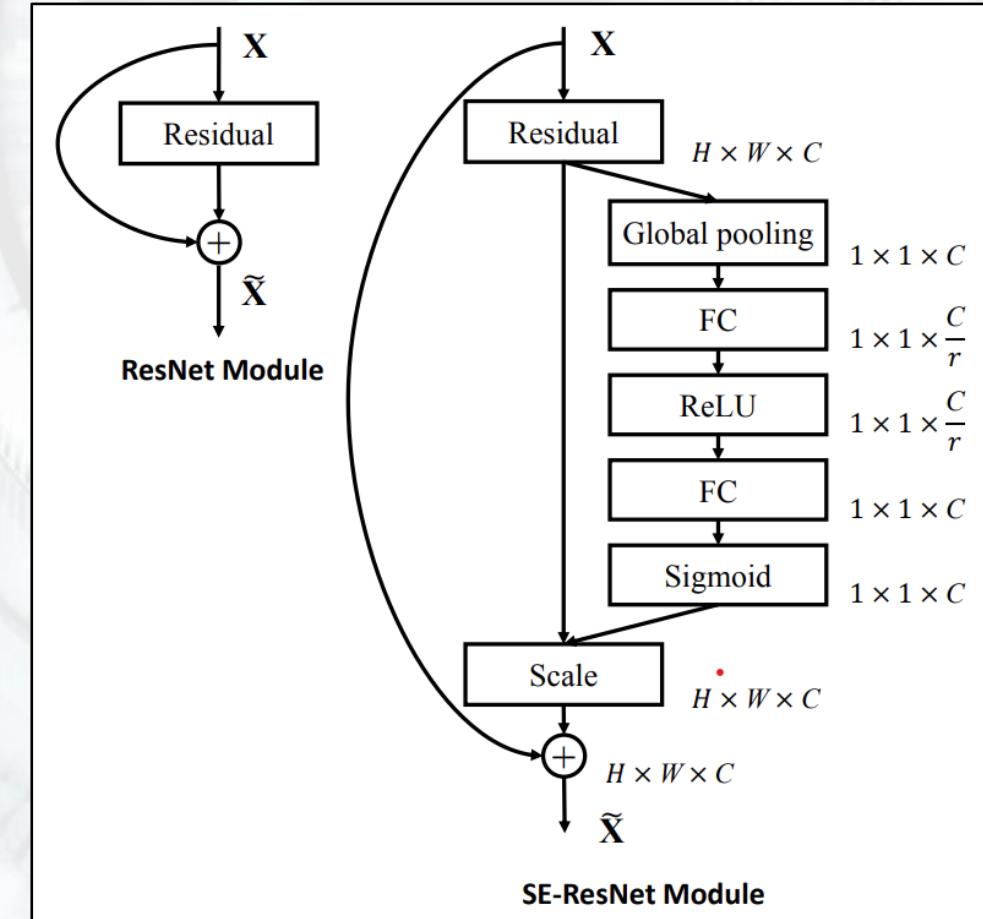
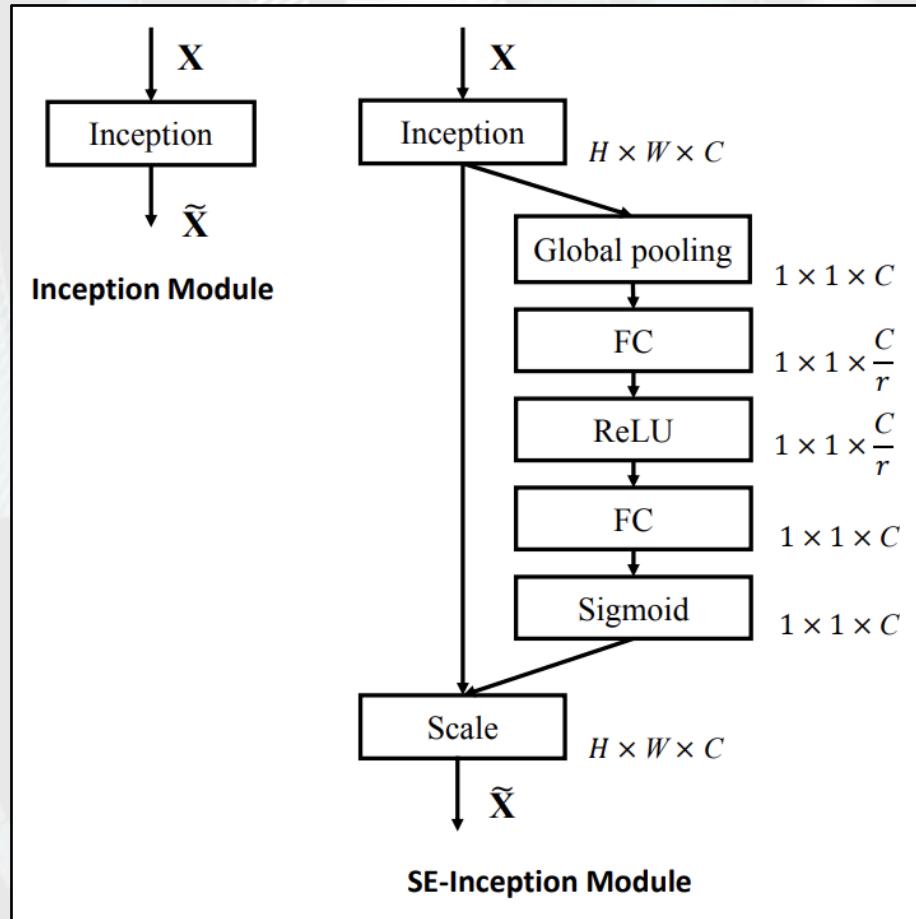


$$F_{sq}(\mathbf{u}_c) = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W u_c(i, j)$$

$$s = F_{ex}(\mathbf{z}, \mathbf{W}) = \sigma(g(\mathbf{z}, \mathbf{W}))$$

$$F_{scale} = (\mathbf{u}_c, s_c) = s_c \cdot u_c$$

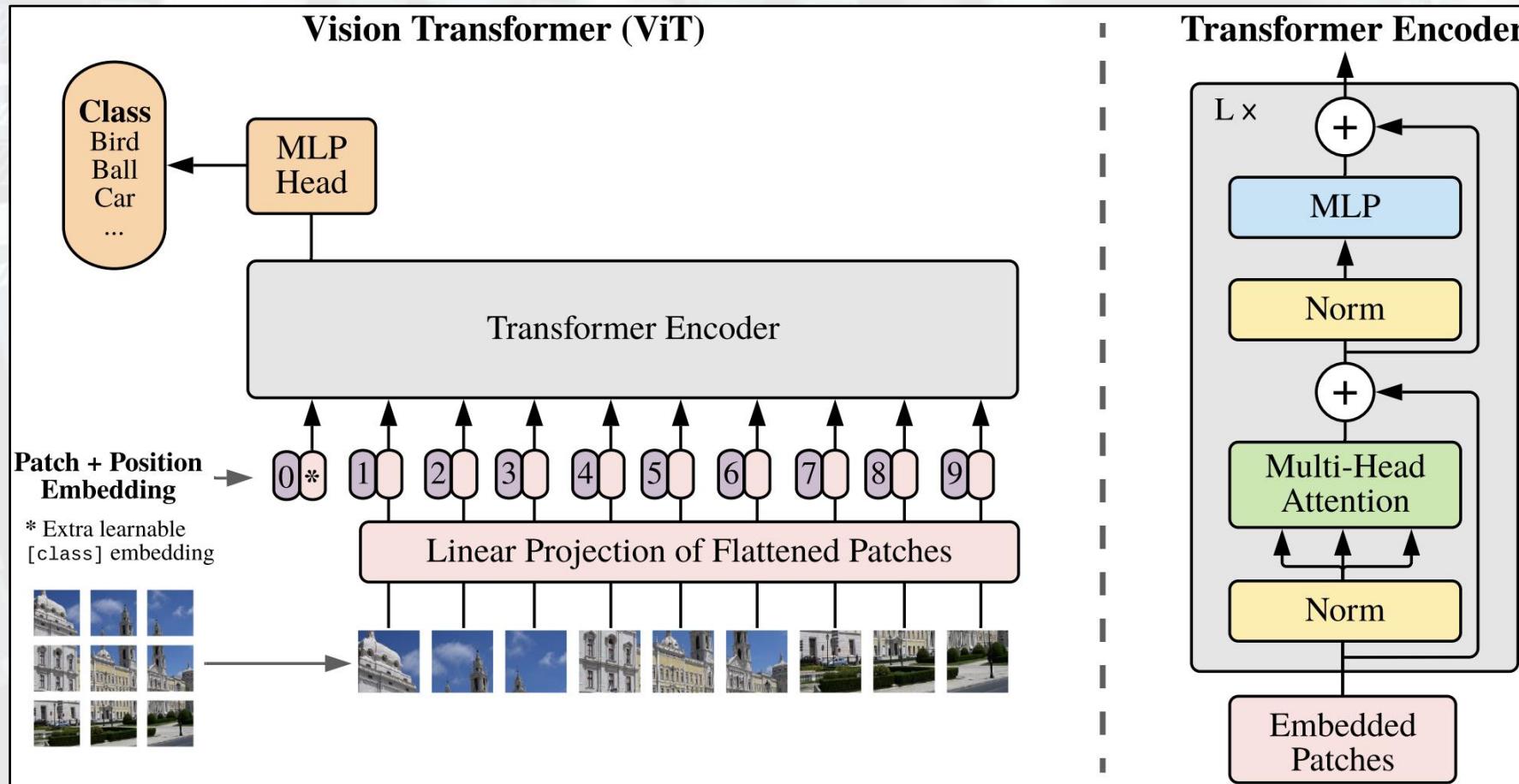
Clasificarea imaginilor – SENet



Clasificarea imaginilor – SENet

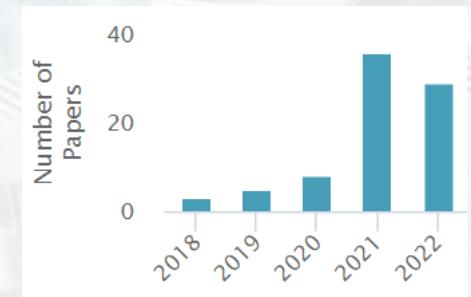
- Câștigător al ILSVRC 2017 (ultima ediție);
- Introduc un bloc ce îmbunătățește interdependențele între canale fără aproape niciun cost adăugat;
- Modulele Squeeze-and-Excitation pot fi adăugate la orice arhitectură existentă;
- Scalează în mod diferit ponderile canalelor cu ajutorul mecanismului de „attenție”;
- Printre primele implementări de succes ale „attenției” în domeniul de computer vision.

Clasificarea imaginilor – ViT [15]



Clasificarea imaginilor – ViT

- Vision Transformer reprezintă adaptarea arhitecturii recurente de tip Transformer la sarcini de computer vision;
- ViT funcționează mai bine decât clasicele rețele convoluționale doar pentru baze de date foarte mari (>100M exemple);
- ViT este mai rapid decât ResNet;
- Majoritatea modelelor de tip Transformer sunt antrenate pe JFT-300M – bază de date internă (closed source) a Google. Acest lucru reprezintă o mare constrângere pentru cercetare.
- Resursele de calcul necesare pentru reproducerea rezultatelor nu sunt disponibile publicului larg.



Clasificarea imaginilor – modele embedded

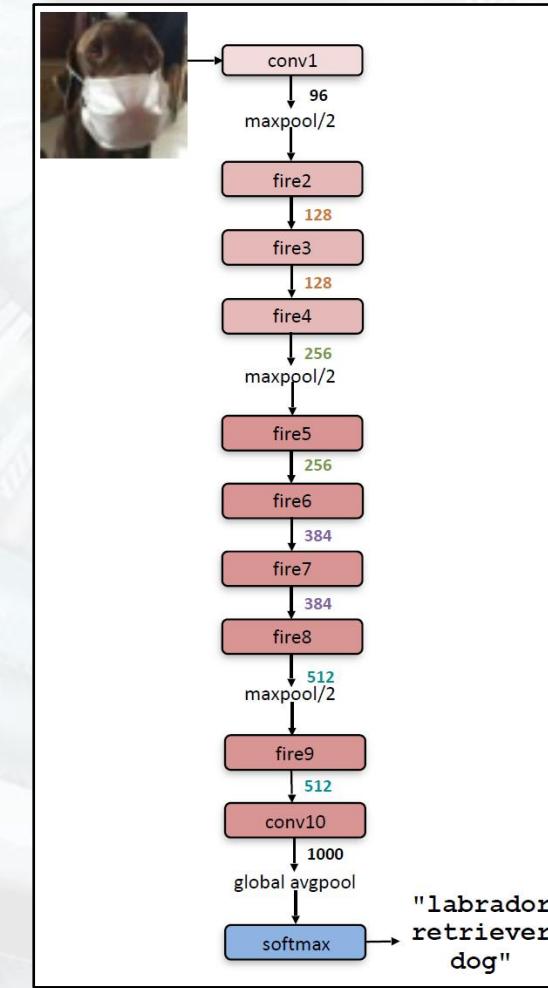
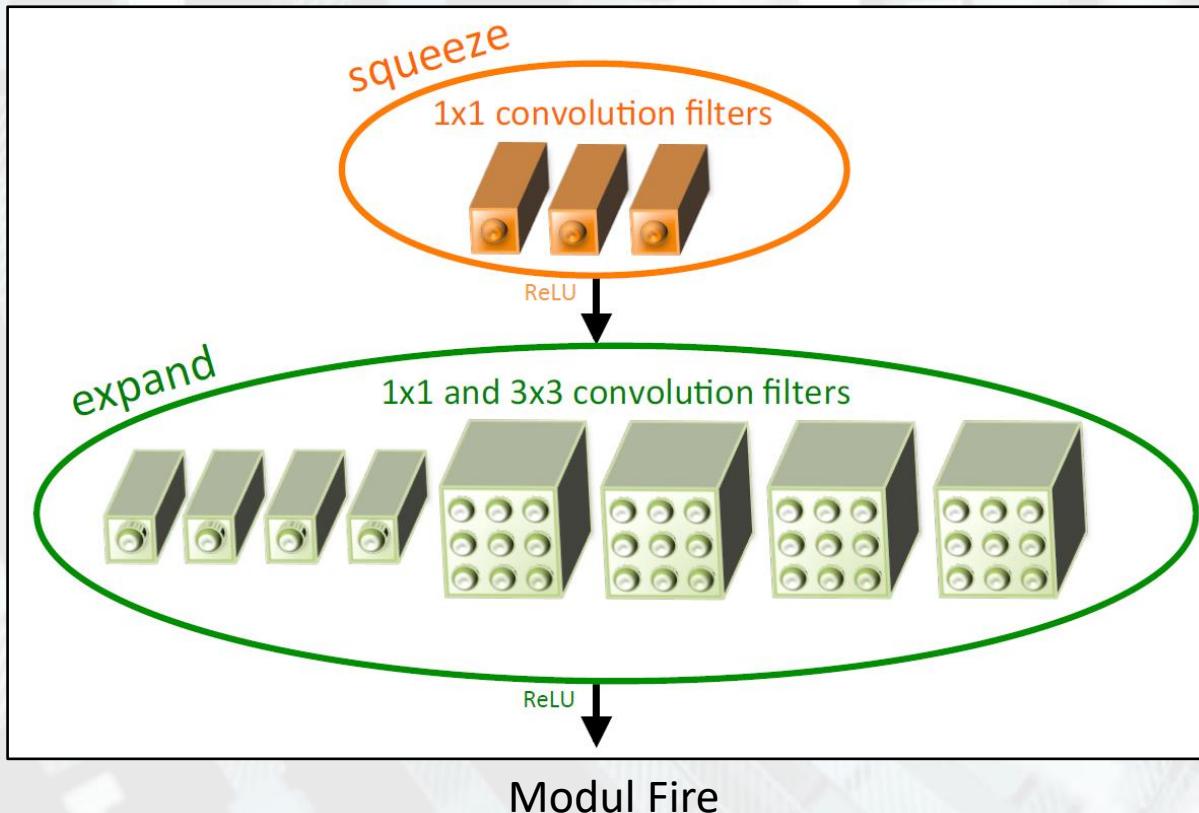
➤ Motivație:

- Număr mare de dispozitive conectate (IoT);
- Industrii bazate pe subansamble cât mai compacte: drone, telefoane mobile, roboți, mașini autonome etc. => platforme mobile cu putere redusă de calcul, baterie limitată;

➤ Necesitate:

- Număr mic de parametri;
- Dimensiune redusă a modelului (în MB);
- Timp de inferență cât mai redus;
- Performanțe comparabile cu sisteme full-scale.

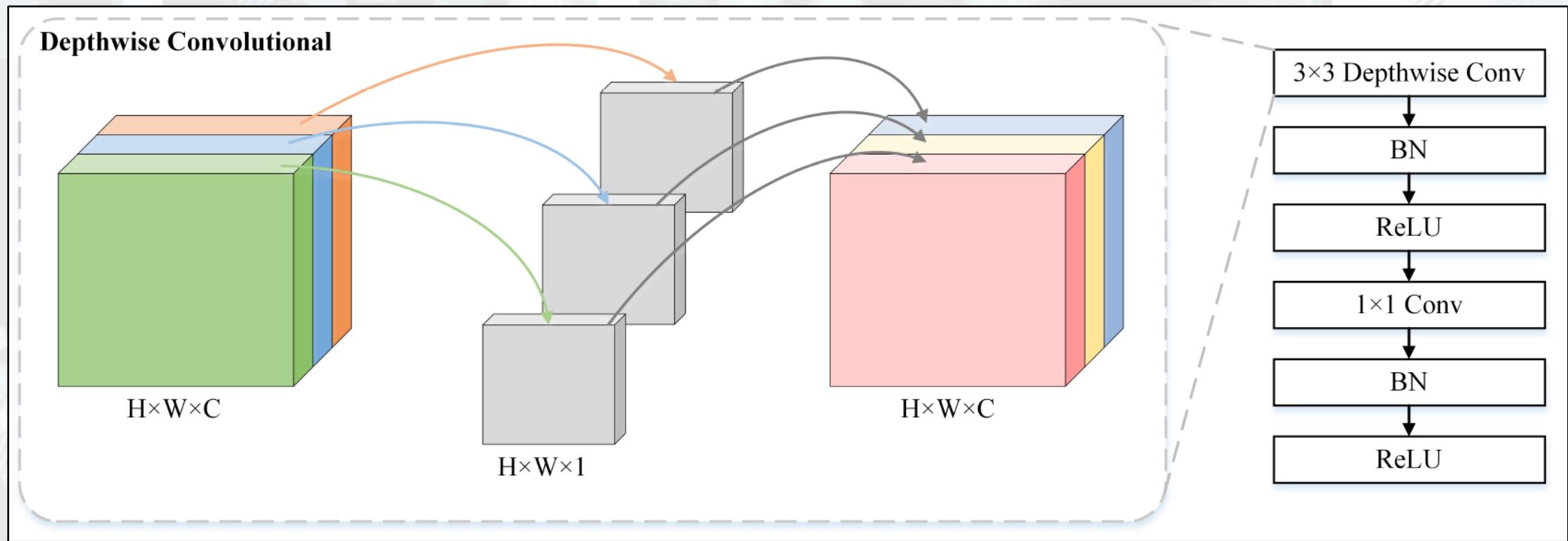
Clasificarea imaginilor – SqueezeNet [18]



Clasificarea imaginilor – SqueezeNet

- Datorită conoluțiilor 1×1 în locul celor de 3×3 , se reduc parametrii rețelei;
- Este introdus un strat de global average pooling în partea de final a rețelei, astfel încât straturile conoluționale să aibă harta de activări cât mai mare (transformă trăsăturile de dimensiuni $N \times N \times c$ în trăsături de dimensiuni $1 \times 1 \times c$);
- Utilizează DeepCompression pentru a reduce volumul modelului AlexNet cu un factor de 510x (prin cuantizare pe 6 biți în loc de 32 biți), de la 240MB la 0.47MB;
- A redus numărul de parametri cu un factor de aproape 50;
- Păstrează performanțele modelului original.

Clasificarea imaginilor – MobileNet [20]

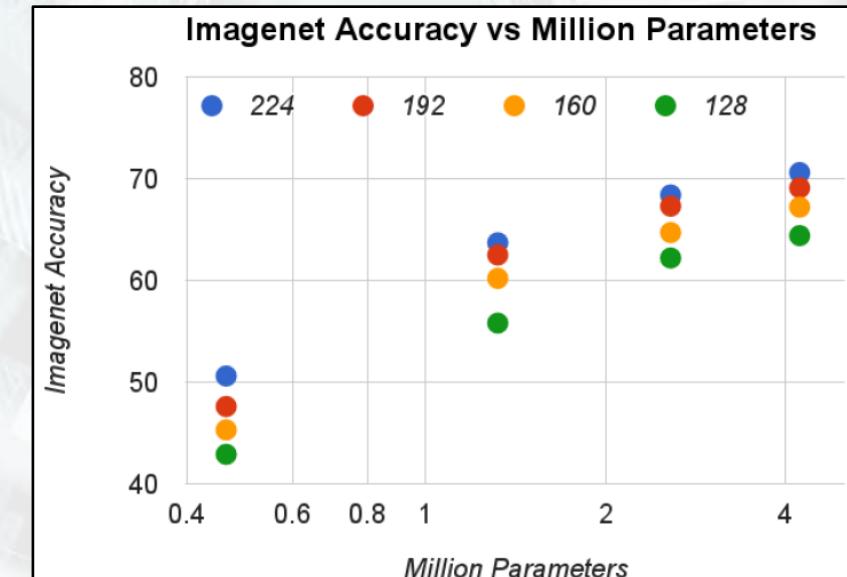


Clasificarea imaginilor – MobileNet

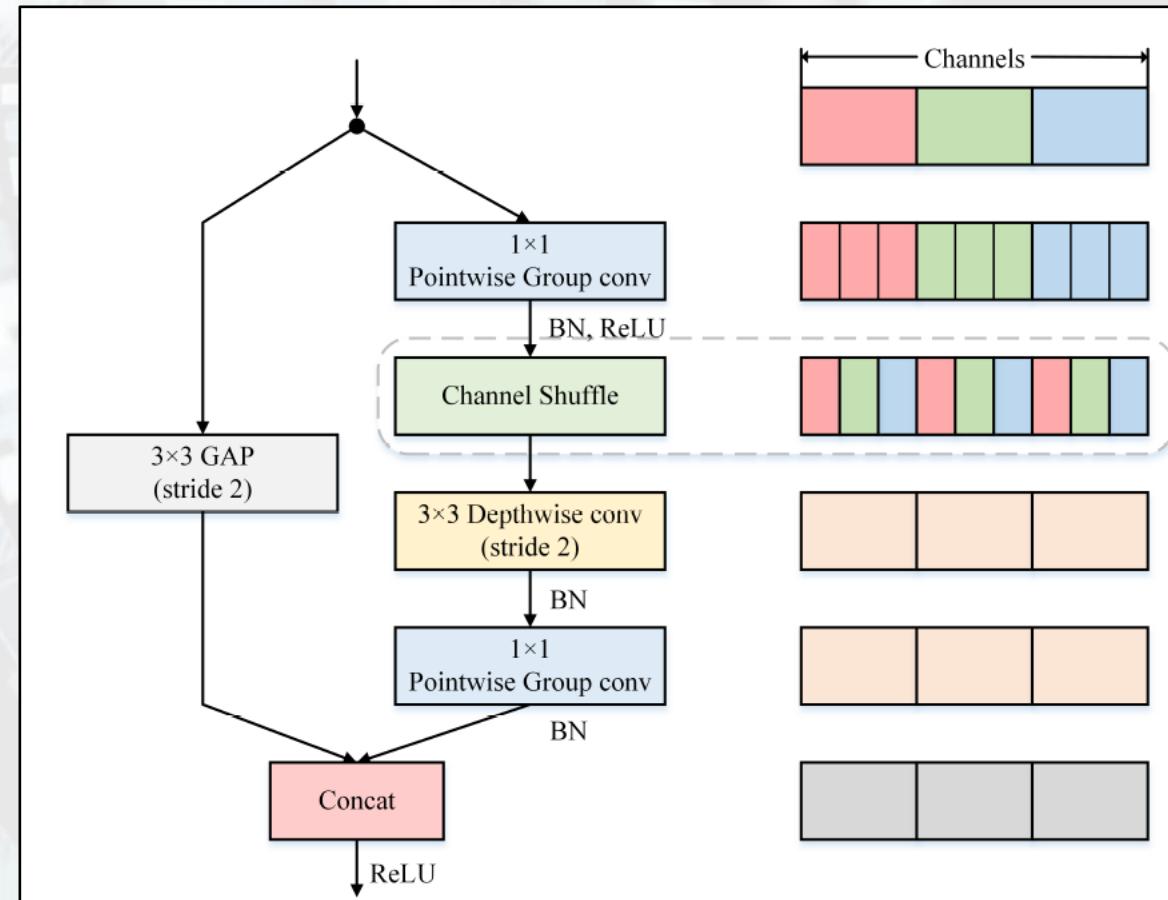
- Folosește convoluții separabile pe adâncime formate din convoluții pe adâncime și convoluții punctuale (1×1);
- Folosește doi hiperparametri, multiplicator de lățime, respectiv multiplicator de rezoluție pentru a controla compromisul dintre timpul de procesare și acuratețe.

Table 8. MobileNet Comparison to Popular Models

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138



Clasificarea imaginilor – ShuffleNet [21]



Clasificarea imaginilor – ShuffleNet

- Folosește convoluțiile punctuale (1×1) de grup pentru a combate problema convoluțiilor punctuale normale, care reduc excesiv de mult numărul de canale => limitează complexitatea reprezentării datelor => rezultate slabe.
- Folosește amestecarea canalelor pentru a combate problema convoluțiilor de grup, care blochează schimbul de informații între canalele dintre grupuri diferite și limitează puterea de reprezentare.

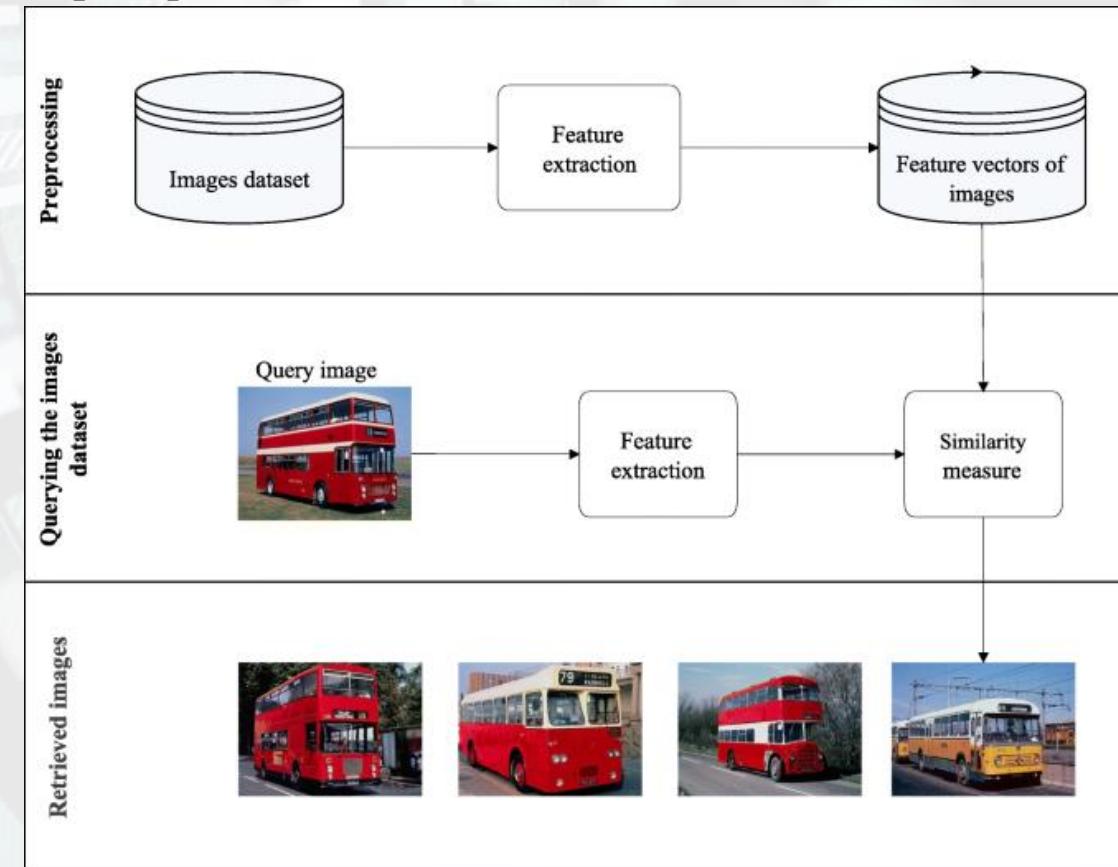
Clasificarea imaginilor – aplicații

1. Image captioning

A young boy is playing basketball. 	Two dogs play in the grass. 	A dog swims in the water. 	A little girl in a pink shirt is swinging. 
A group of people walking down a street. 	A group of women dressed in formal attire. 	Two children play in the water. 	A dog jumps over a hurdle. 

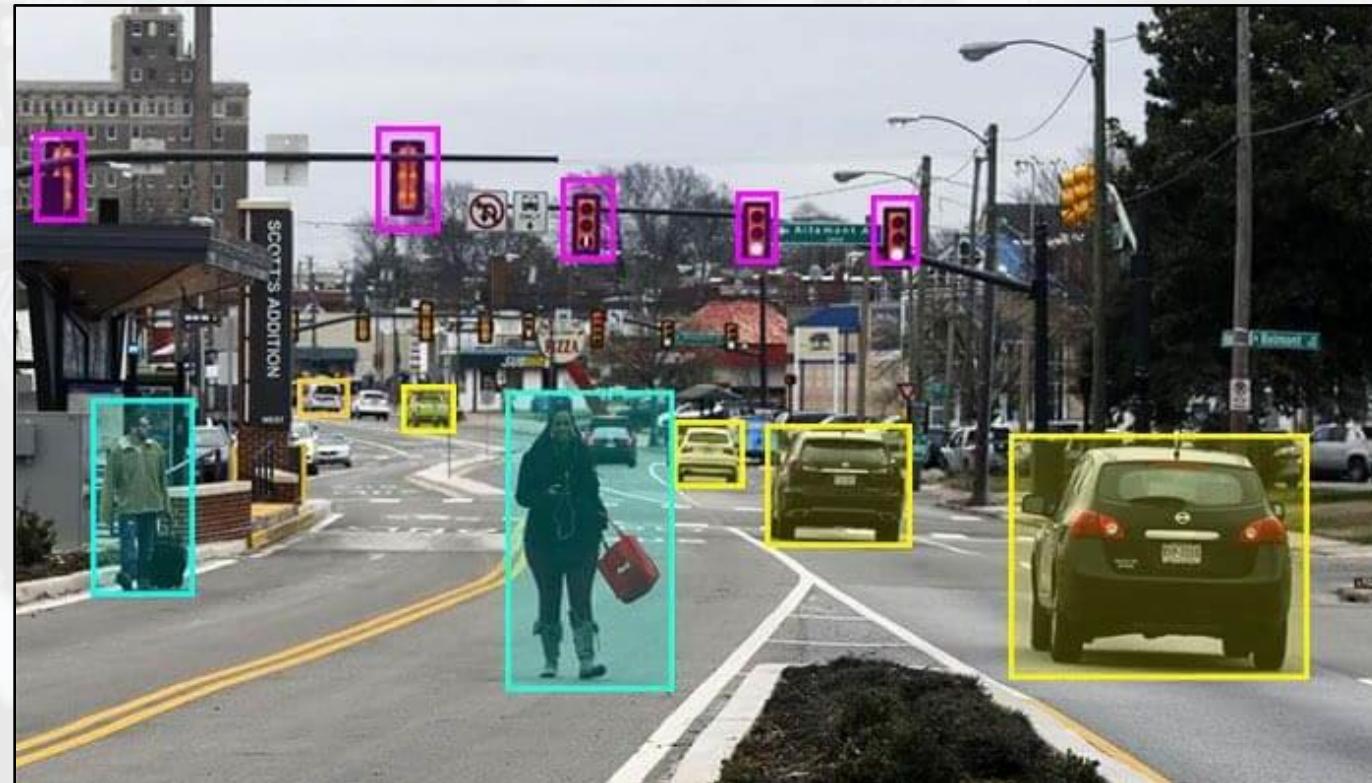
Clasificarea imaginilor – aplicații

2. Image retrieval [22]



Clasificarea imaginilor – aplicații

3. Detectia automată a obiectelor



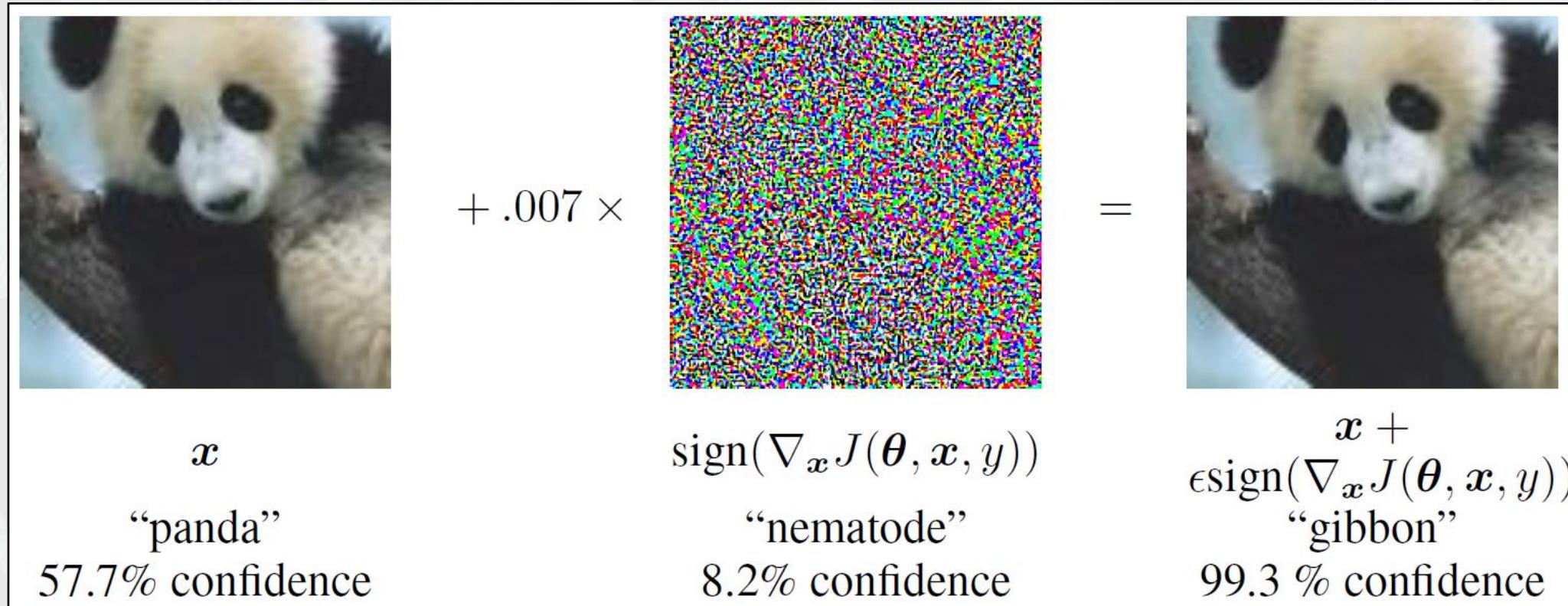
Clasificarea imaginilor – aplicații

4. Segmentare semantică



Clasificarea imaginilor - limitări

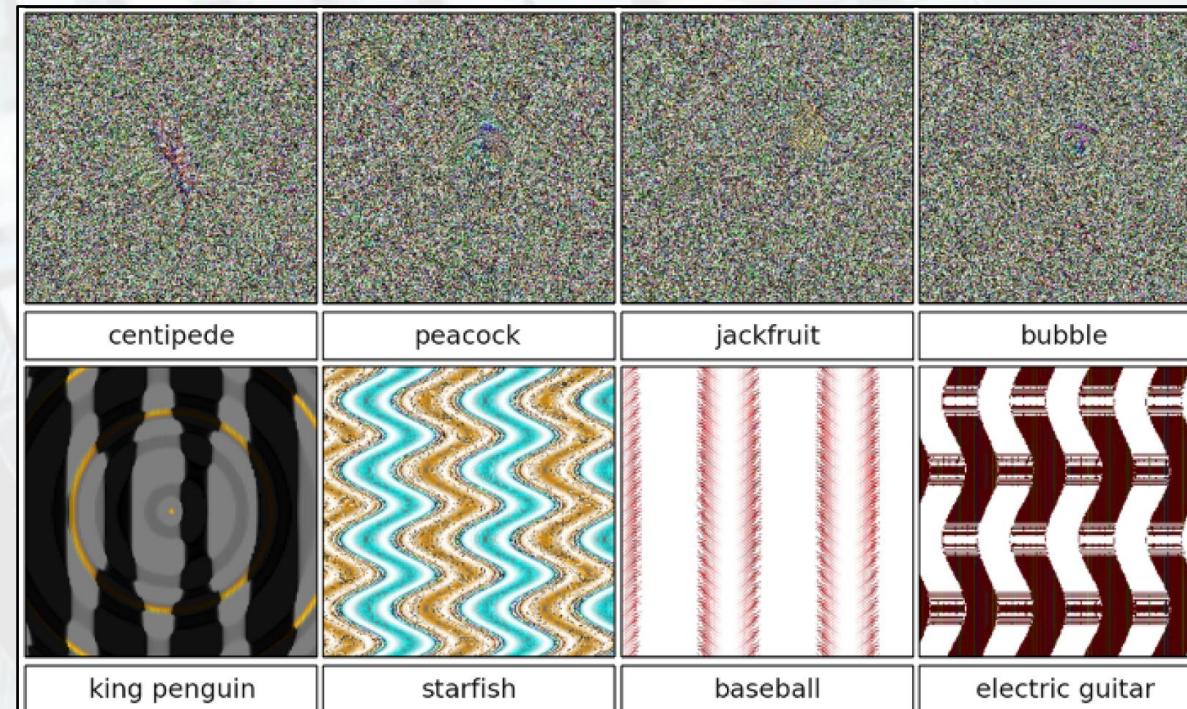
- Rețelele neuronale sunt sensibile la atacuri adversariale (GoogLeNet):



Perturbație liniară insesizabilă pentru oameni asupra unei imagini [16]

Clasificarea imaginilor - limitări

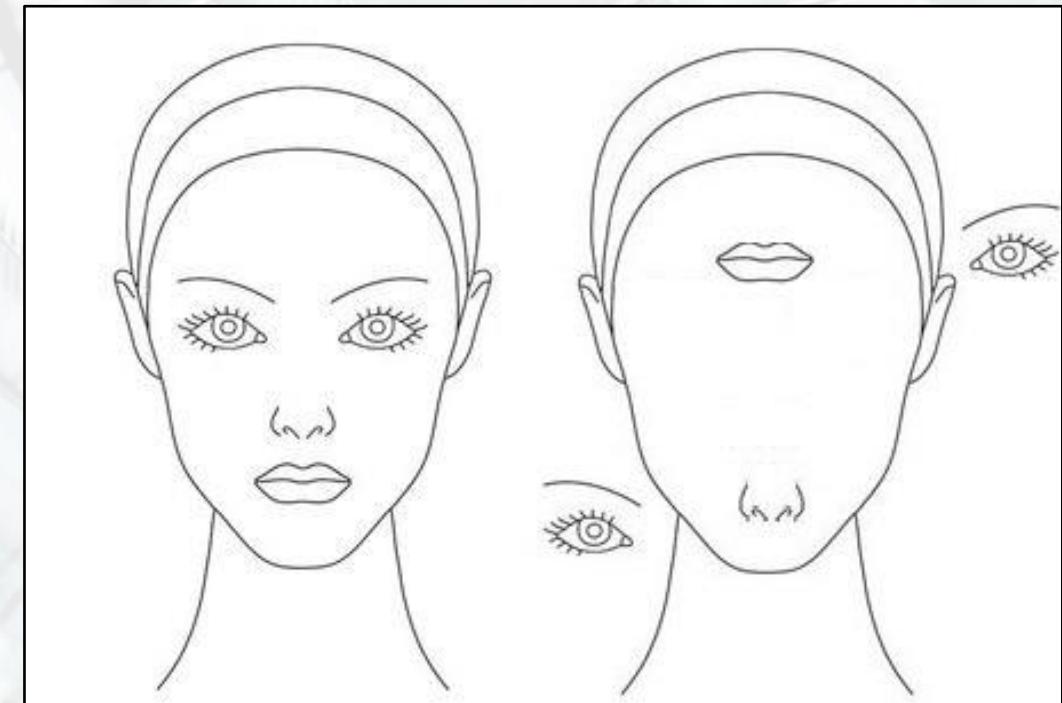
- Rețelele neuronale pot fi păcălite cu imagini absurde generate de algoritmi evolutivi:



Imagini recunoscute greșit, cu scoruri >99.6% de către rețele state-of-the-art [17]

Clasificarea imaginilor - limitări

- Rețelele neuronale sunt sensibile la translații globale, rotații și scalare.



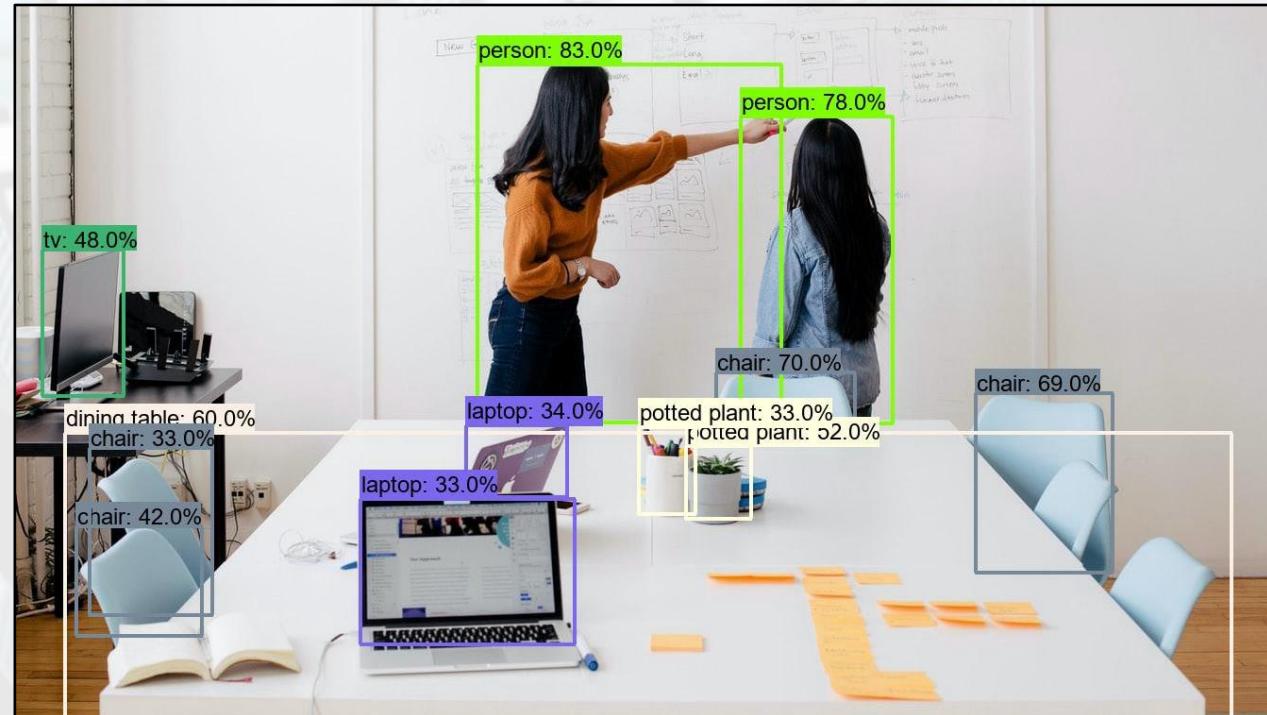
Clasificarea imaginilor - concluzii

- Sistemul de clasificare a imaginilor presupune 2 componente majore:
 - Extragerea unor descriptori de trăsături din date;
 - Clasificarea descriptorilor de trăsături.
- Sistemele de clasificare a imaginilor ocupă un rol central în dezvoltarea rețelelor neuronale cu aplicații în computer vision;
- Versatilitate mare – baza de date determină tipul aplicației;
- Susceptibile la „atacuri”;
- Potențial comercial ridicat.

M3.3. Detectia obiectelor

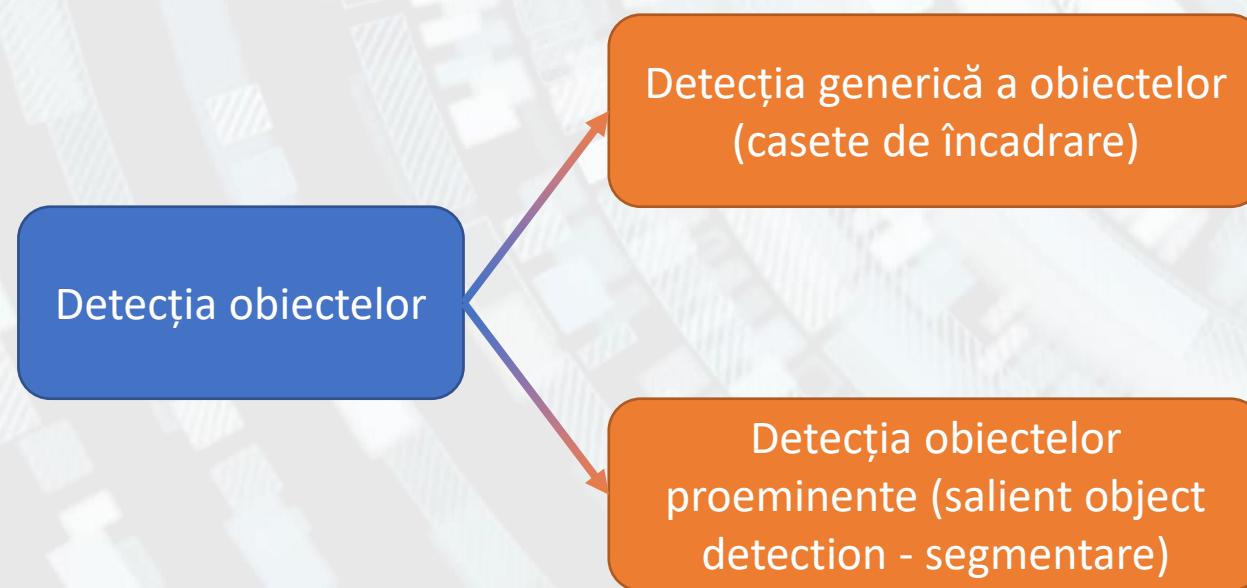
Detectia obiectelor

Detectia obiectelor = sarcina de a determina unde anume se găsesc obiecte în imagine și de a determina cărei categorii aparțin.



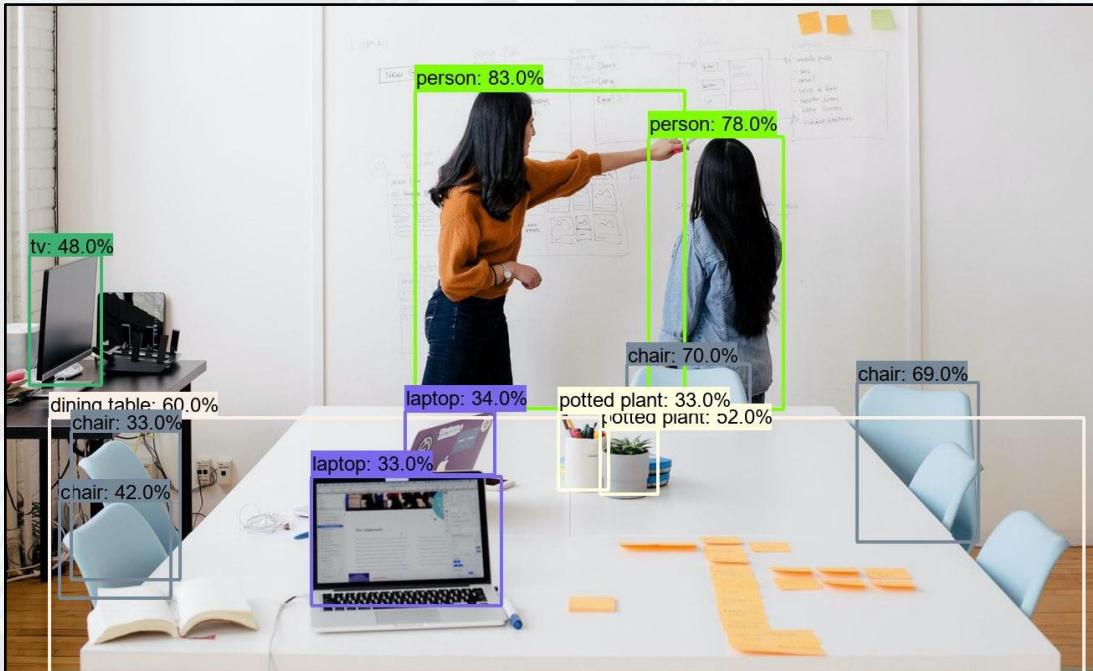
Detectia obiectelor

- Procesul de detectie a obiectelor poate fi împărtit în 3 pași:
 1. Găsirea regiunii informative (unde se află obiectul?);
 2. Extragerea descriptorului asociat obiectului (cum este descris obiectul?);
 3. Clasificarea obiectului (ce fel de obiect este?)

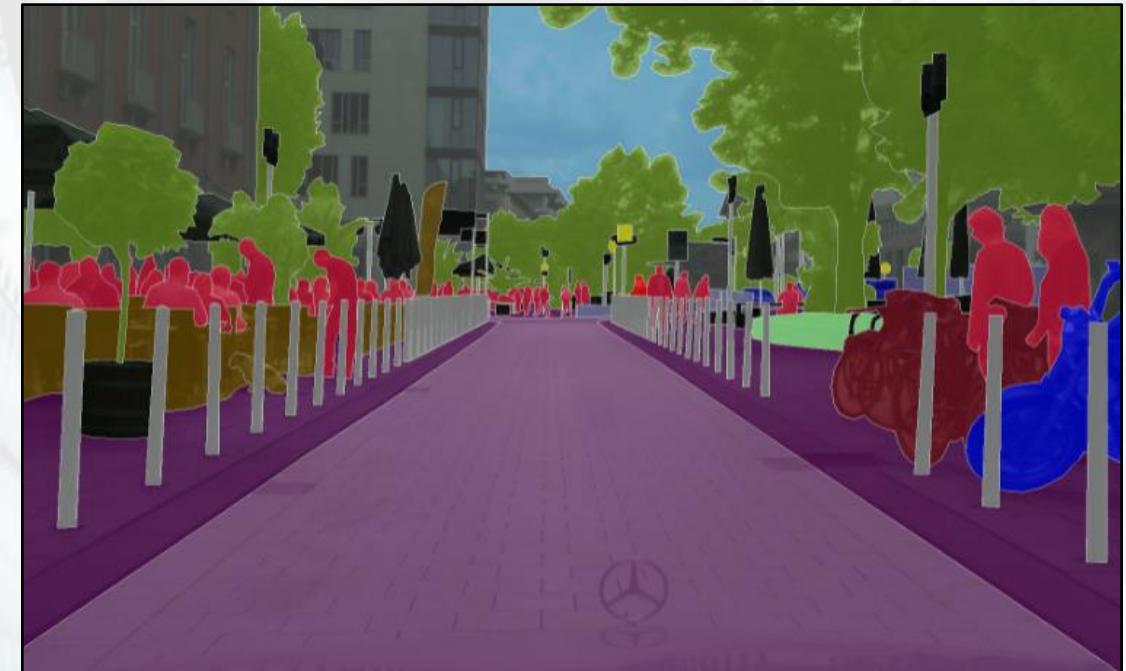


Detectia obiectelor

Detectia generică a obiectelor – M3.3.



Detectia obiectelor proeminente – M3.4.



Detectia obiectelor

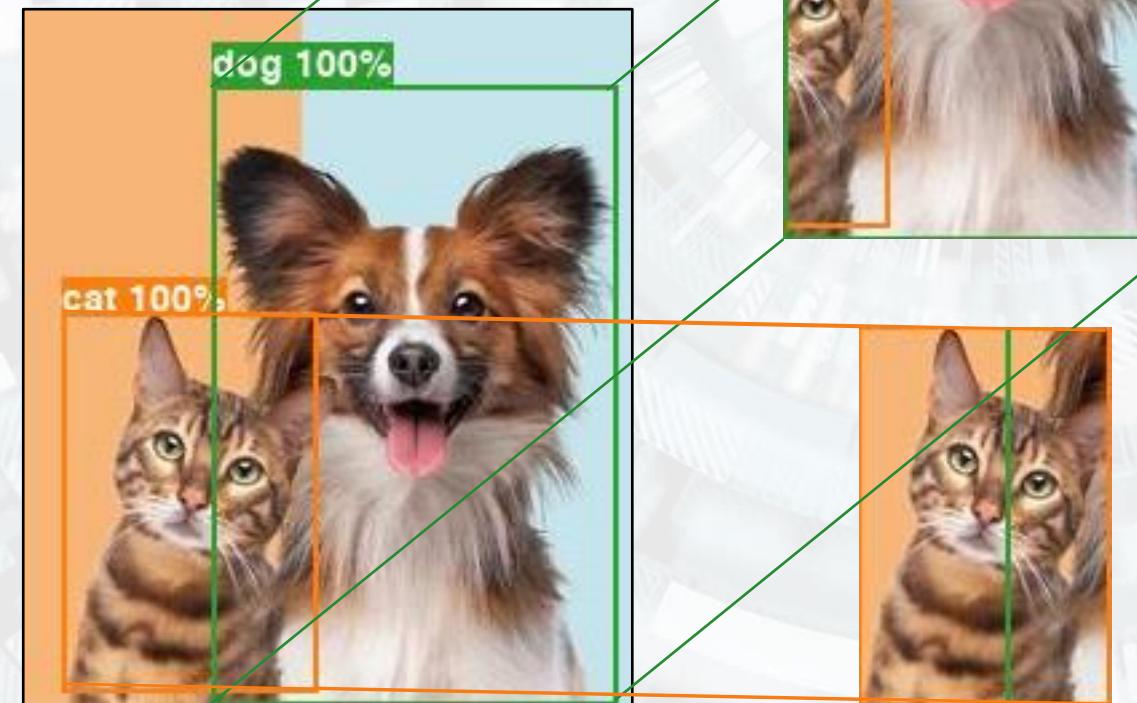
- Detectia generică a obiectelor – caracteristici:
 - Trebuie recunoscute toate obiectele (cu care a fost antrenat modelul) dintr-o imagine => baza de date este un punct critic;
 - Fiecare obiect detectat este descris de:
 - Casetă de încadrare – 4 coordonate (2 colțuri opuse sau coordonatele unui colt/punct central + lățime și înălțime);
 - Clasa obiectului;
 - Scor $\in [0, 1]$;
 - Conțează atât scorul, cât și poziționarea casetei.

Detectia obiectelor

➤ Detectia generică a obiectelor



Input



Output

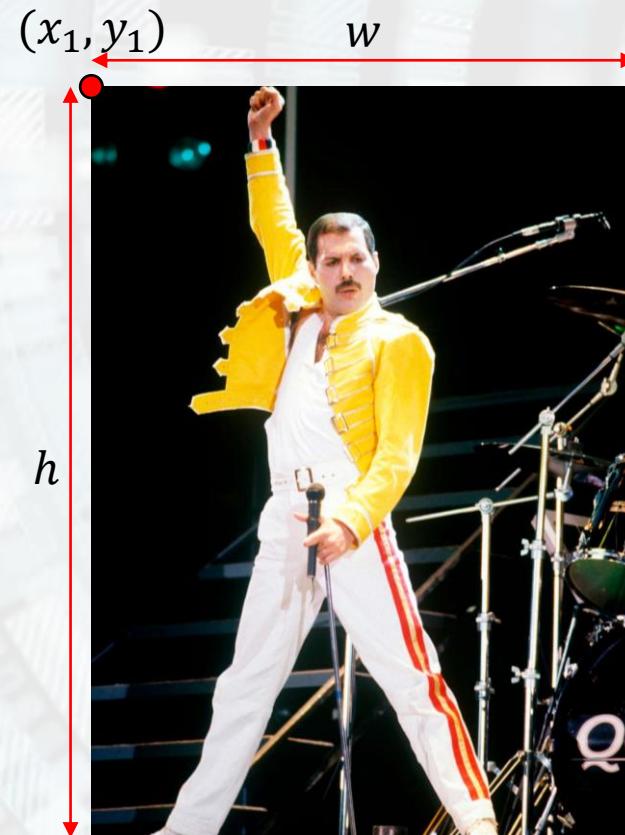
- Casetă de încadrare:
 - 1) x_1, y_1, x_2, y_2 ;
 - 2) x, y, w, h ;
- Clasa: „dog”;
- Scorul: 1.0;

- Casetă de încadrare:
 - 1) x_1, y_1, x_2, y_2 ;
 - 2) x, y, w, h ;
- Clasa: „cat”;
- Scorul: 1.0;

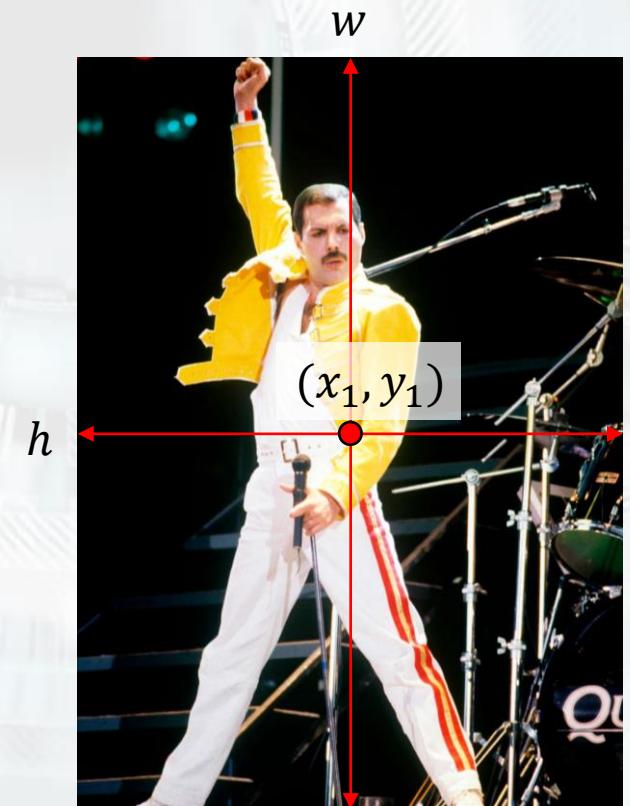
Detectia obiectelor



$$bbox_1 = (x_1, y_1, x_2, y_2)$$



$$bbox_2 = (x_1, y_1, w, h)$$

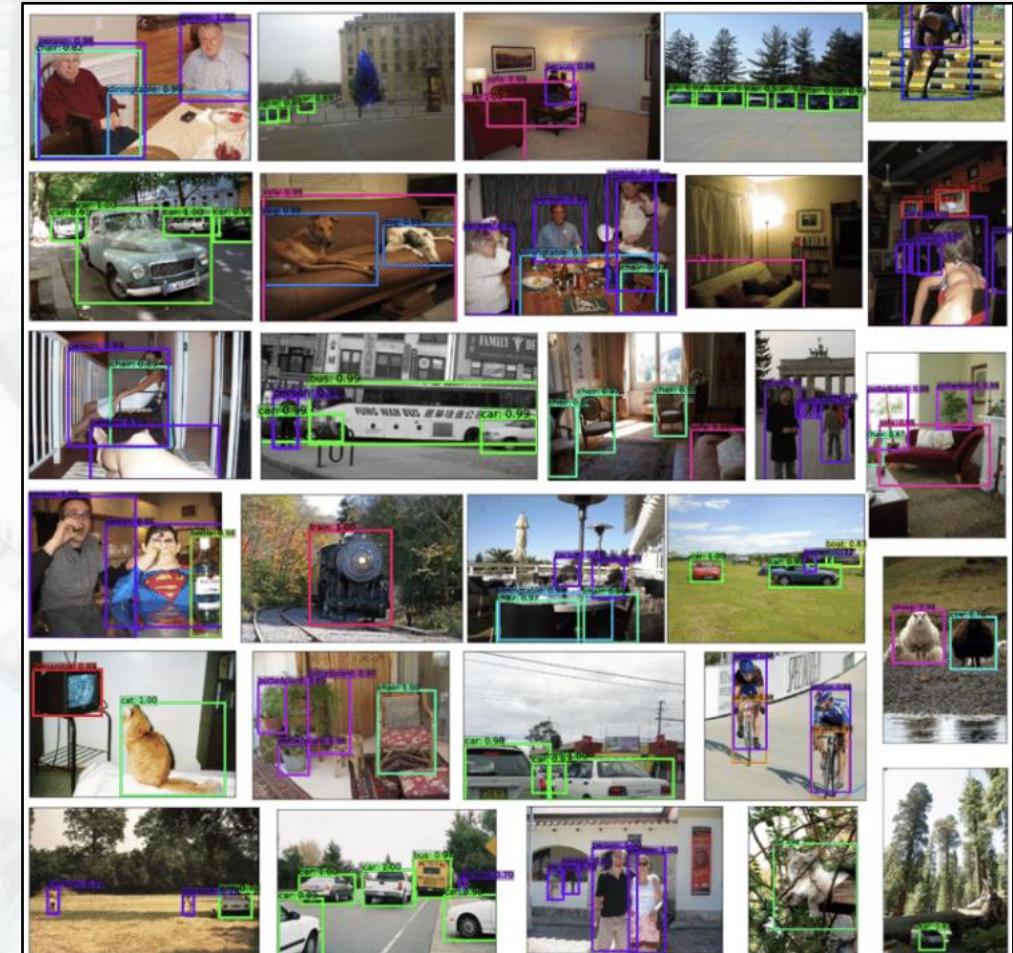
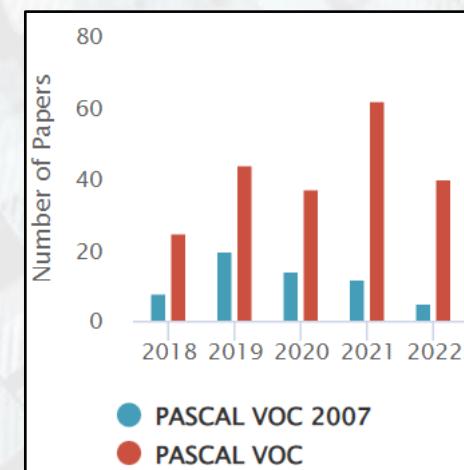


$$bbox_3 = (x_1, y_1, w, h)$$

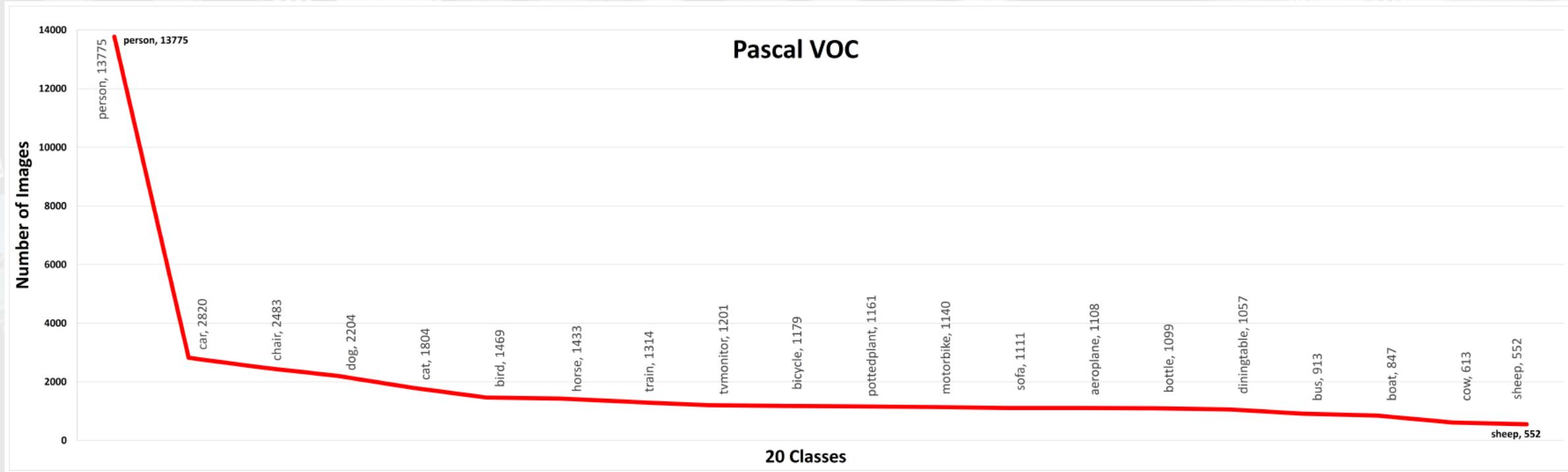
Detectia obiectelor – datasets

PASCAL VOC 07/12 [23]

- Contine imagini naturale;
- 20 clase annoteate;
- Train: 5.7k imagini;
- Val: 5.8k imagini;
- Test: 10.9k imagini;
- 2.37 obiecte/img;
- Introduce mAP;



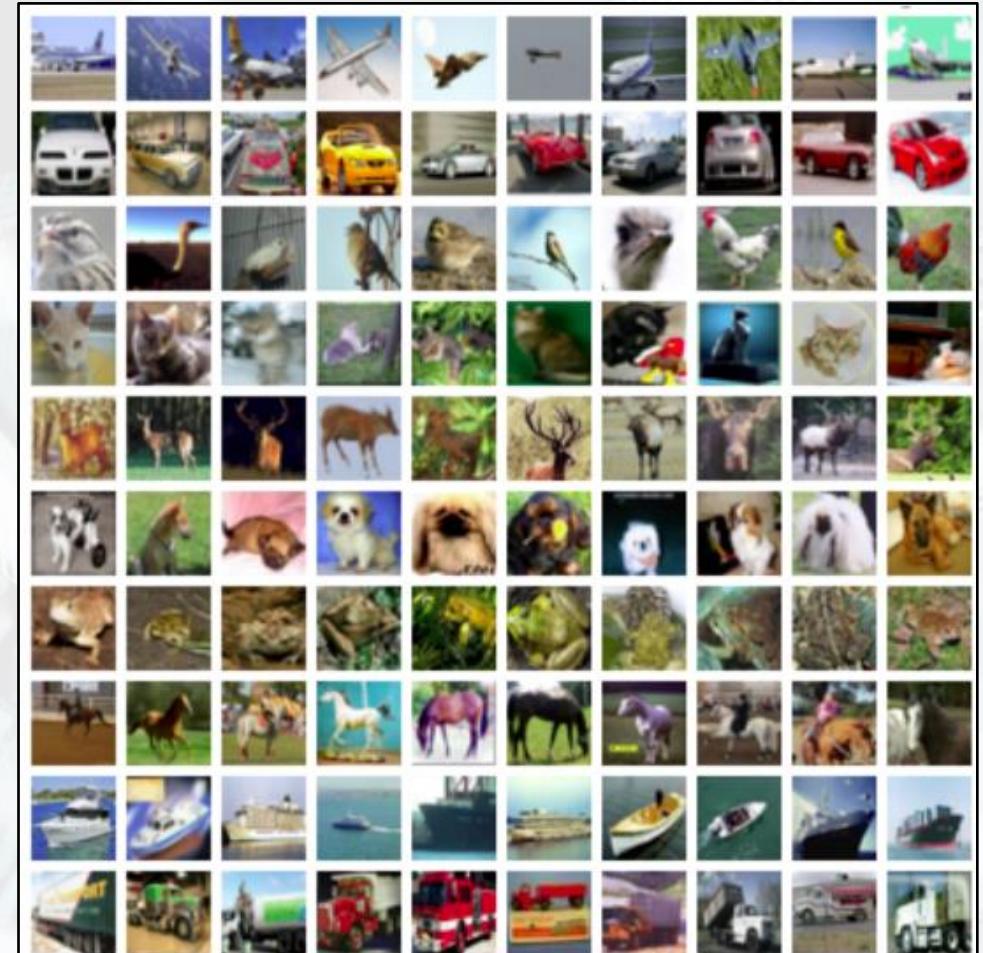
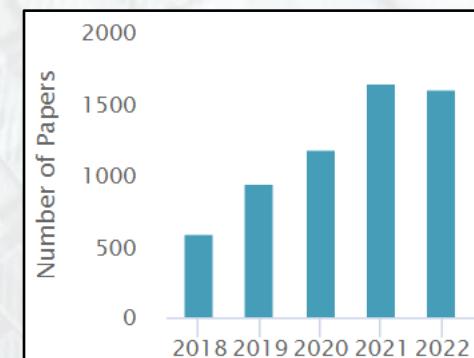
Detectia obiectelor – datasets



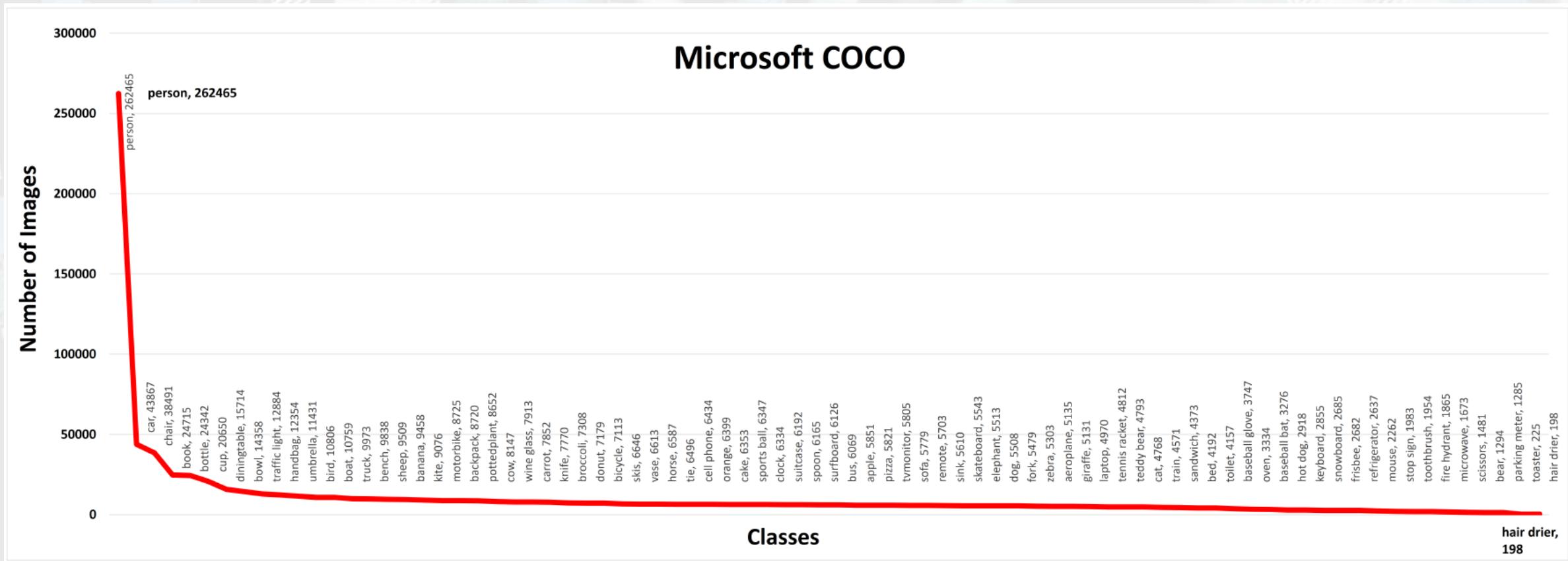
Detectia obiectelor – datasets

MS-COCO [24]

- Contine imagini naturale;
- 80 clase;
- 160k imagini adnotate;
- 940k obiecte adnotate;
- 7.27 obiecte/img;
- Cea mai populară;



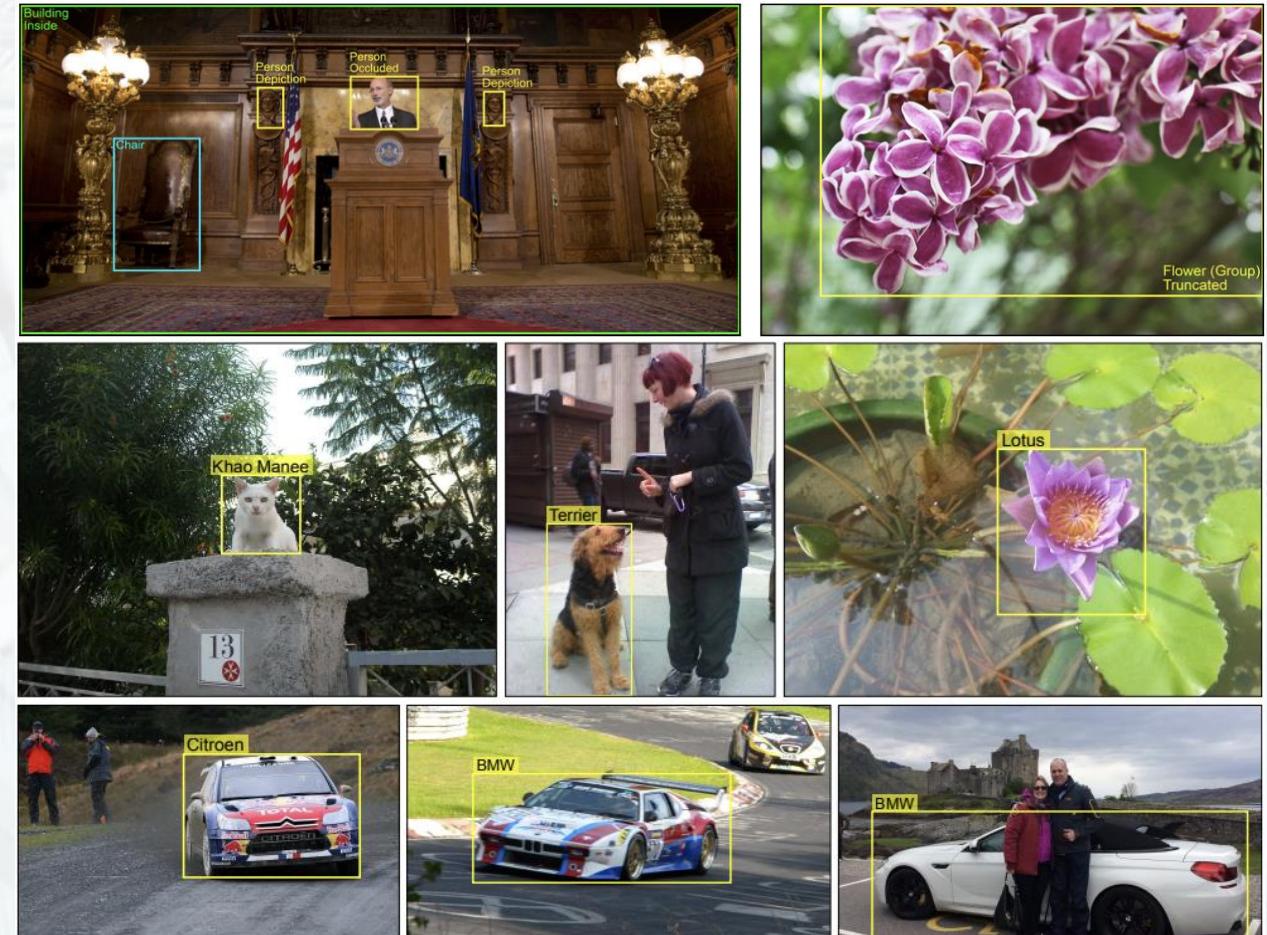
Detectia obiectelor – datasets



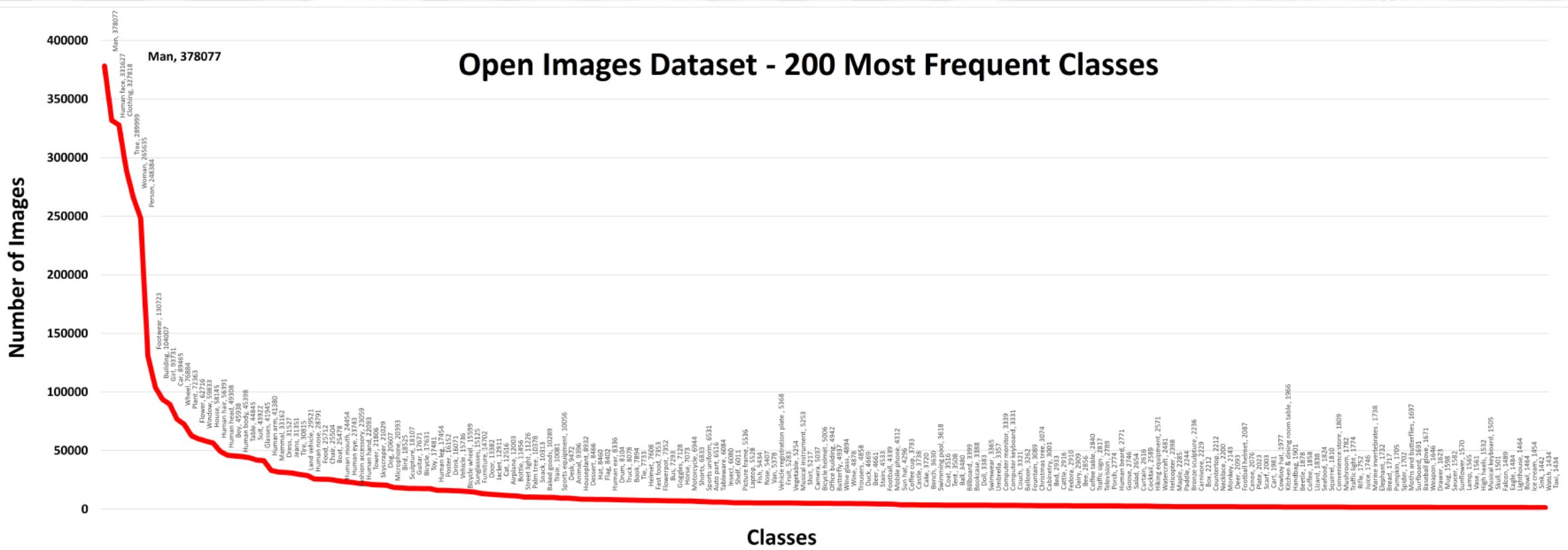
Detectia obiectelor – datasets

Open Images [25]

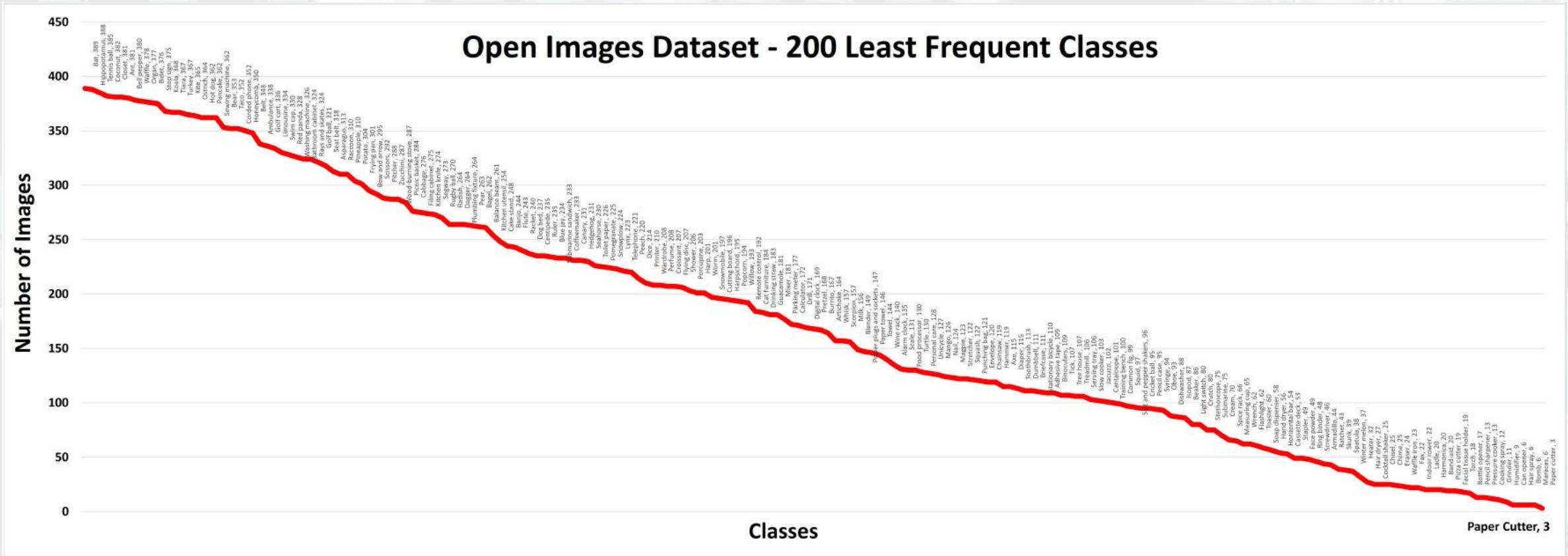
- Contine imagini naturale;
- 600 clase;
- 1.9M imagini adnotate;
- 15M obiecte adnotate;
- 8.38 obiecte/img;
- Cea mai mare.



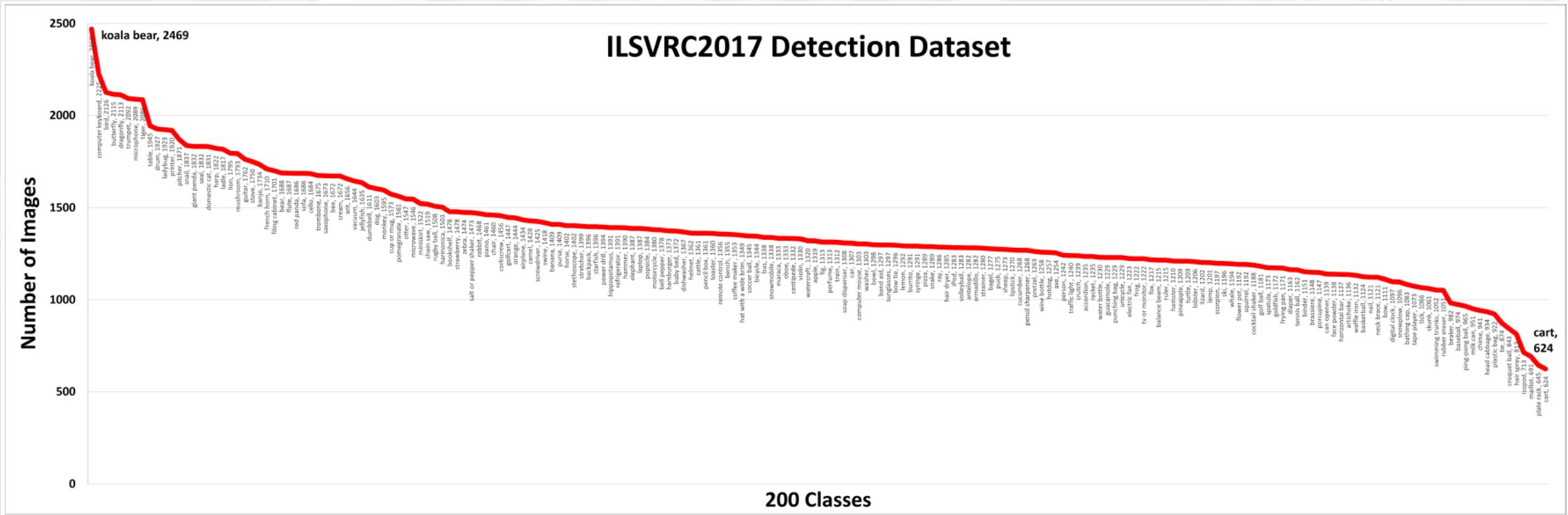
Detectia obiectelor – datasets



Detectia obiectelor – datasets



Detectia obiectelor – datasets



Detectia obiectelor – datasets

Dataset	Classes	Train			Validation			Test
		Images	Objects	Objects/Image	Images	Objects	Objects/Image	
PASCAL VOC 12	20	5,717	13,609	2.38	5,823	13,841	2.37	10,991
MS-COCO	80	118,287	860,001	7.27	5,000	36,781	7.35	40,670
ILSVRC	200	456,567	478,807	1.05	20,121	55,501	2.76	40,152
OpenImage	600	1,743,042	14,610,229	8.38	41,620	204,621	4.92	125,436

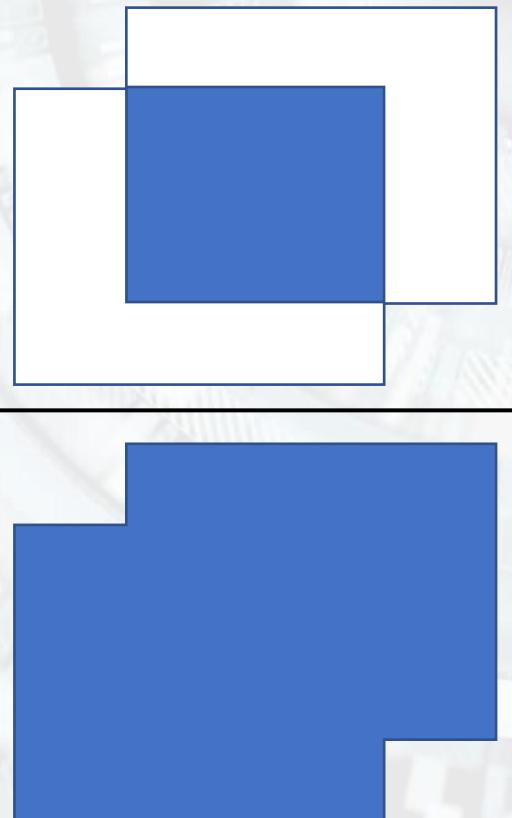
Detectia obiectelor – mAP

- Metrica folosită pentru a evalua sistemele de detectie a obiectelor se numește mean Average Precision (mAP).
- În componența ei, se iau în calcul mai multe metrici:
 - Intersecția supra reuniunea;
 - Precizia;
 - Reamintirea;
 - Precizia medie.

Detectia obiectelor – mAP

- Intersecția supra reuniunea (Intersection over Union – IoU):

$$IoU = \frac{\text{Aria intersecției}}{\text{Aria reuniunii}} =$$



Detectia obiectelor – mAP

- Se stabilește un prag (e.g. 0.5) în funcție de care se determină precizia și reamintirea pentru o clasă anume.

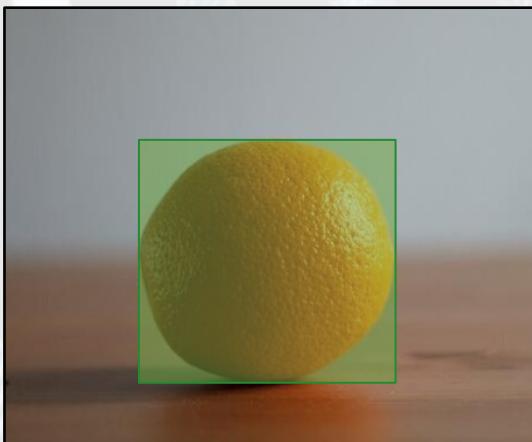
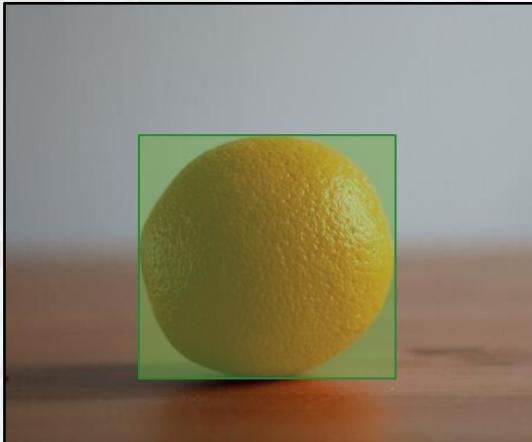
$$precizie(P) = \frac{tp}{tp + fp}$$

$$reamintire(R) = \frac{tp}{tp + fn}$$

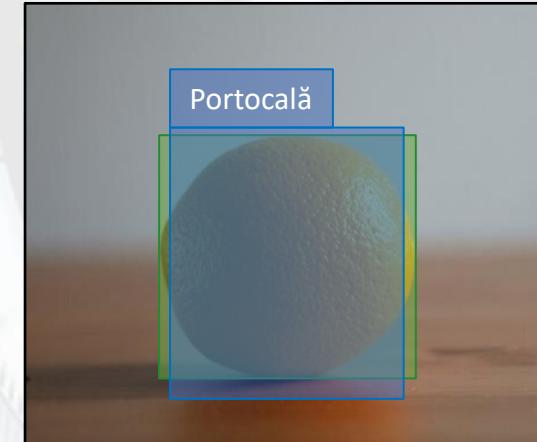
- Cum definim tp, fp, fn ?

Detectia obiectelor – mAP

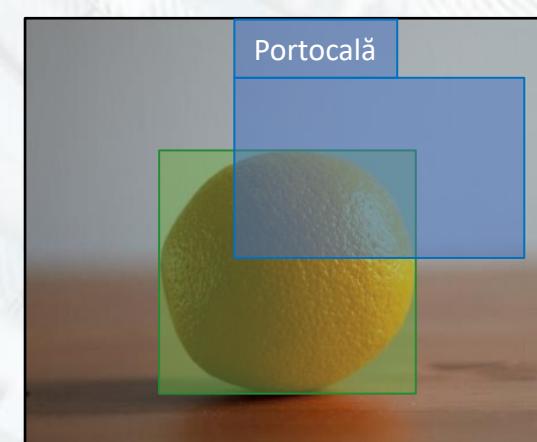
Groundtruth



Lipsă detectie /
Detectie clasa greșită
 $\text{IoU} = 0 \Rightarrow \text{false negative}$



Detectie clasa corectă
 $\text{IoU} \approx 0.9 > \text{prag} \Rightarrow \text{true positive}$



Detectie clasa corectă
 $\text{IoU} \approx 0.3 < \text{prag} \Rightarrow \text{false positive}$

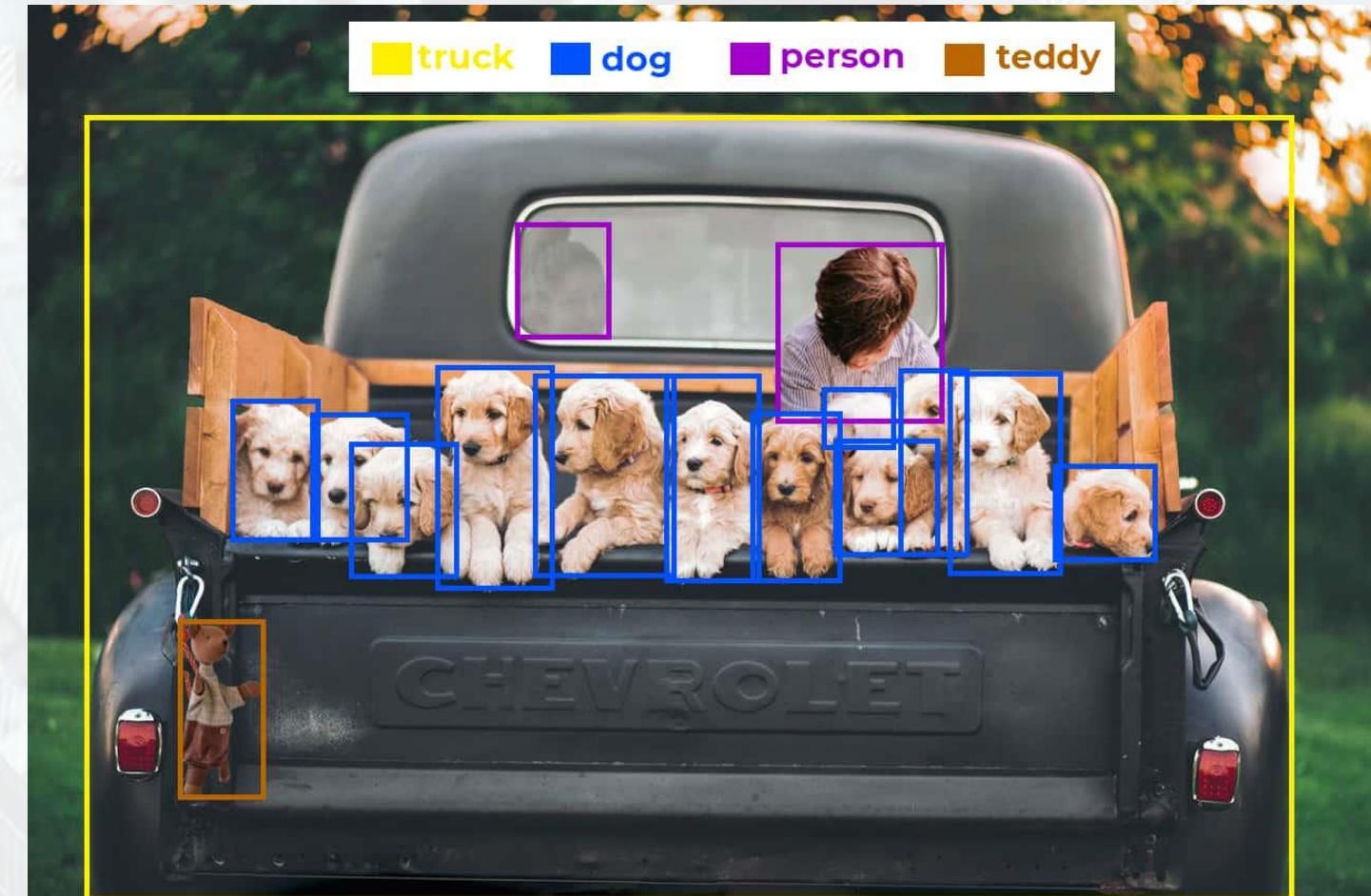
Detectia obiectelor – mAP

1. Se selectează o imagine;



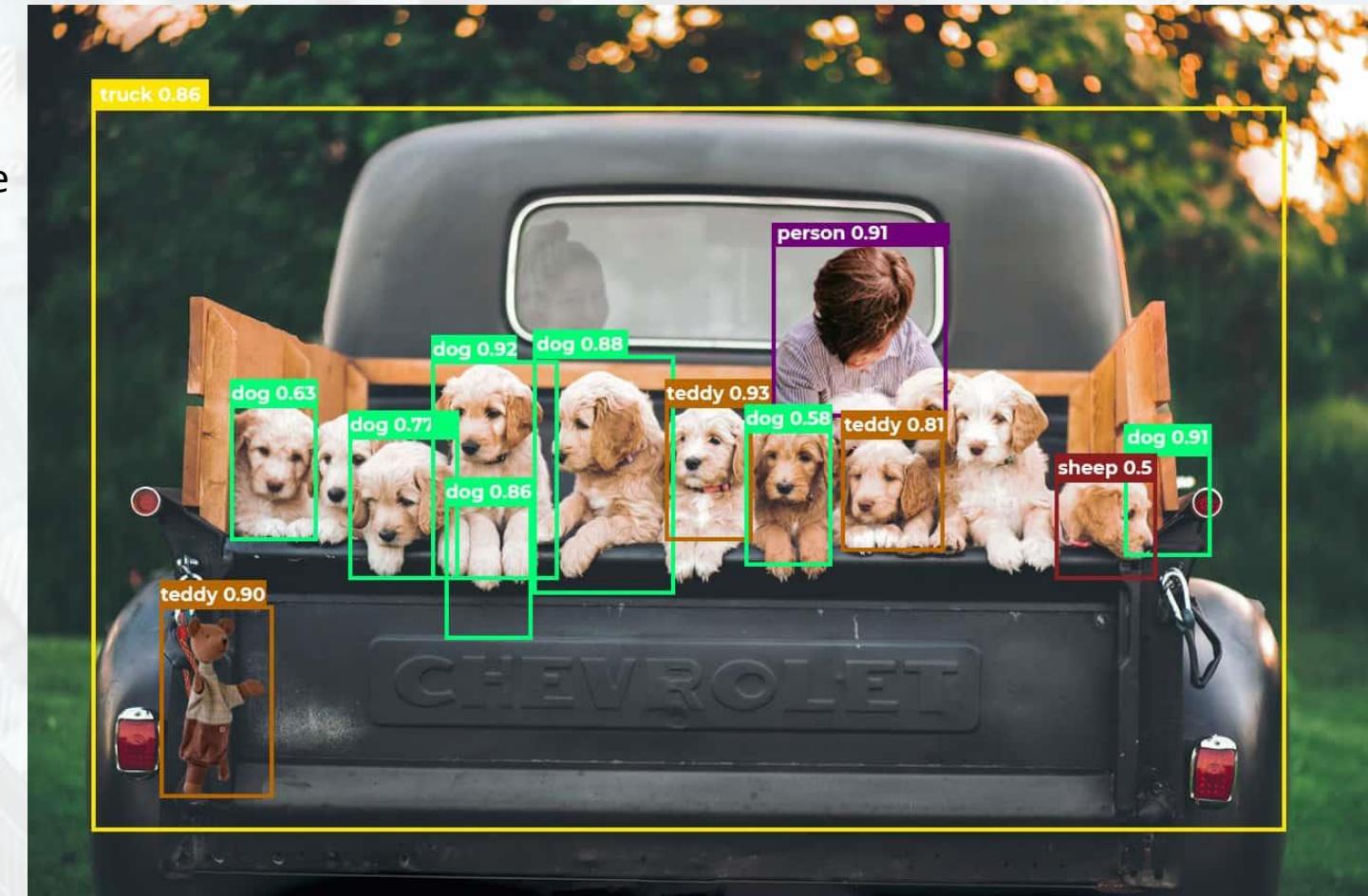
Detectia obiectelor – mAP

1. Se selectează o imagine;
2. Se rețin etichetele imaginii;



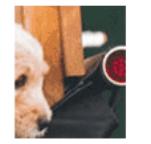
Detectia obiectelor – mAP

1. Se selectează o imagine;
2. Se rețin etichetele imaginii;
3. Se rulează detectia obiectelor pe imagine;



Detectia obiectelor – mAP

1. Se selectează o imagine;
2. Se rețin etichetele imaginii;
3. Se rulează detectia obiectelor pe imagine;
4. Se extrag detectiile pentru clasa de interes (dog);

Detections							
Conf.	0.63	0.77	0.92	0.86	0.88	0.58	0.91
Matches GT by IoU?	TP	TP	TP	FP	TP	TP	FP

Detectia obiectelor – mAP

1. Se selectează o imagine;
2. Se rețin etichetele imaginii;
3. Se rulează detectia obiectelor pe imagine;
4. Se extrag detectiile pentru clasa de interes (dog);
5. Se sortează tabelul în ordinea descrescătoare a scorului (confidence);

Detections								
Conf.	0.92	0.91	0.88	0.86	0.77	0.63	0.58	
Matches GT by IoU?	TP	FP	TP	FP	TP	TP	TP	TP

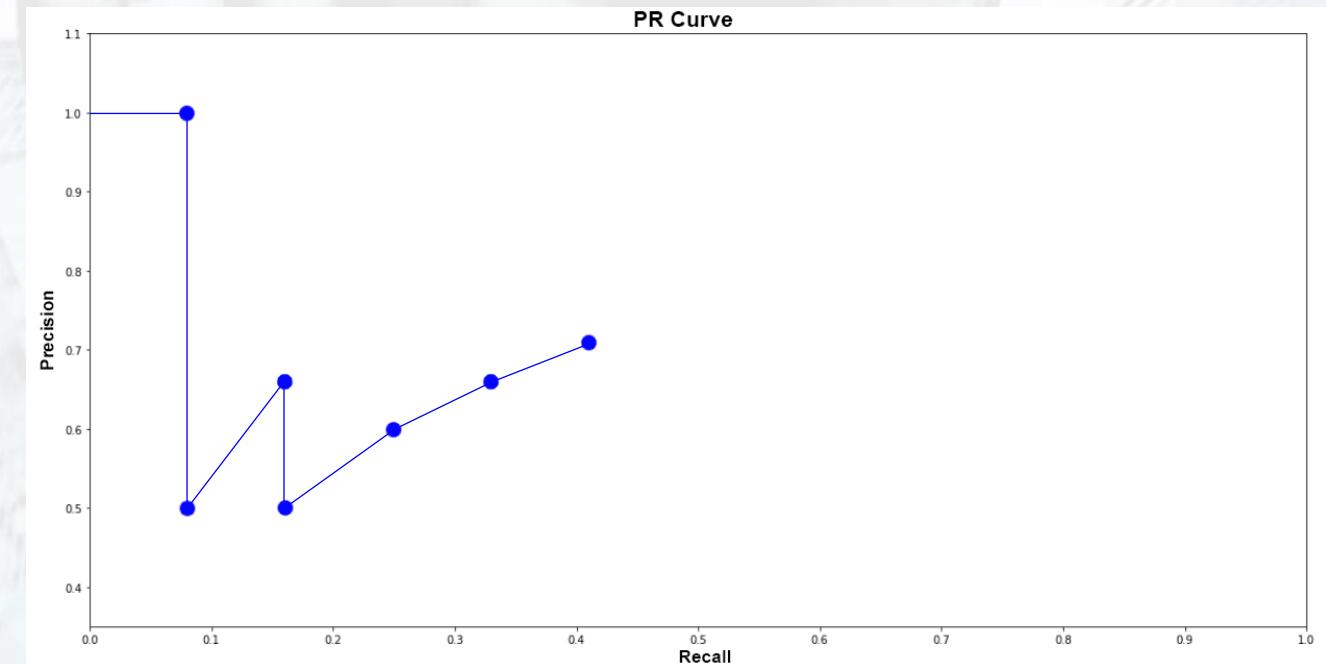
Detectia obiectelor – mAP

1. Se selectează o imagine;
2. Se rețin etichetele imaginii;
3. Se rulează detectia obiectelor pe imagine;
4. Se extrag detectiile pentru clasa de interes (dog);
5. Se sortează tabelul în ordinea descrescătoare a scorului (confidence);
6. Se calculează precizia și reamintirea pentru un prag IoU;

Preds.	Conf.	Matches	Cumulative TP	Cumulative FP	Precision	Recall
	0.92	TP	1	0	$1/(1+0) = 1$	$1/16 = 0.06$
	0.91	FP	1	1	$1/(1+1) = 0.5$	$1/16 = 0.06$
	0.88	TP	2	1	$2/(2+1) = 0.66$	$2/16 = 0.12$
	0.86	FP	2	2	0.5	0.12
	0.77	TP	3	2	0.6	0.18
	0.63	TP	4	2	0.66	0.25
	0.58	TP	5	2	0.71	0.31

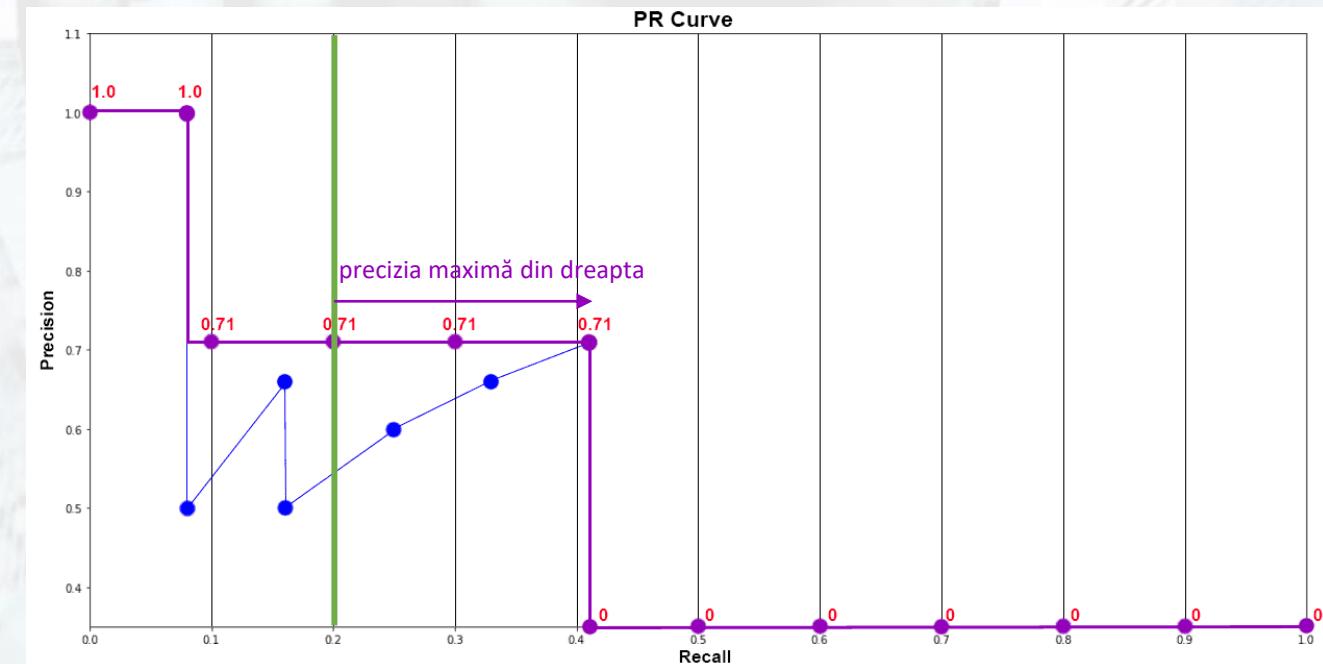
Detectia obiectelor – mAP

1. Se selectează o imagine;
2. Se rețin etichetele imaginii;
3. Se rulează detectia obiectelor pe imagine;
4. Se extrag detectiile pentru clasa de interes (dog);
5. Se sortează tabelul în ordinea descrescătoare a scorului (confidence);
6. Se calculează precizia și reamintirea pentru un prag IoU;
7. Se trasează graficul PR;



Detectia obiectelor – mAP

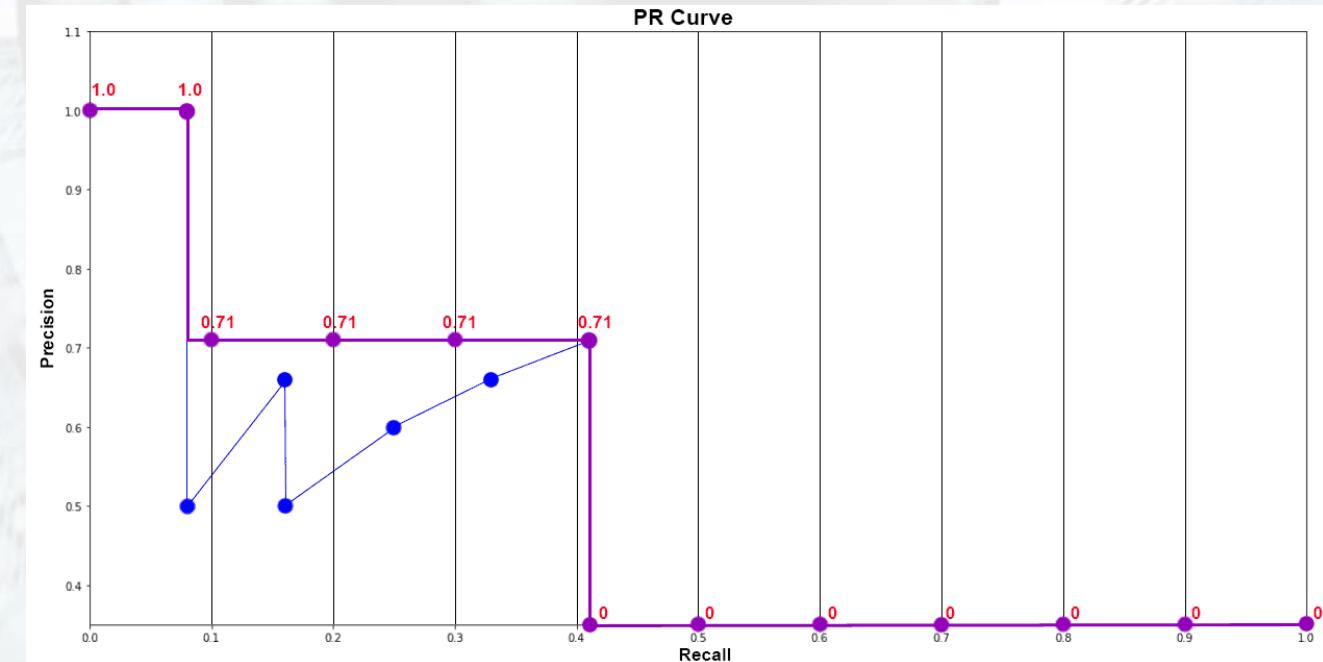
1. Se selectează o imagine;
2. Se rețin etichetele imaginii;
3. Se rulează detectia obiectelor pe imagine;
4. Se extrag detectiile pentru clasa de interes (dog);
5. Se sortează tabelul în ordinea descrescătoare a scorului (confidence);
6. Se calculează precizia și reamintirea pentru un prag IoU;
7. Se trasează graficul PR;
8. Se interpolează graficul PR în 11 (101) puncte;



Se selectează precizia maximă corespunzătoare valorilor de reamintire mai mari decât reamintirea curentă

Detectia obiectelor – mAP

1. Se selectează o imagine;
2. Se rețin etichetele imaginii;
3. Se rulează detectia obiectelor pe imagine;
4. Se extrag detectiile pentru clasa de interes (dog);
5. Se sortează tabelul în ordinea descrescătoare a scorului (confidence);
6. Se calculează precizia și reamintirea pentru un prag IoU;
7. Se trasează graficul PR;
8. Se interpolează graficul PR în 11 (101) puncte;
9. Se calculează precizia medie;



$$\begin{aligned}
 AP_{dog} &= \frac{1}{11} (P(0) + P(0.1) + P(0.2) + \dots + P(1.0)) \\
 &= \frac{1}{11} (1 + 4 * 0.71 + 6 * 0) \\
 &= 34.9\%
 \end{aligned}$$

Detectia obiectelor – mAP

1. Se selectează o imagine;
2. Se rețin etichetele imaginii;
3. Se rulează detectia obiectelor pe imagine;
4. Se extrag detectiile pentru clasa de interes (dog);
5. Se sortează tabelul în ordinea descrescătoare a scorului (confidence);
6. Se calculează precizia și reamintirea pentru un prag IoU;
7. Se trasează graficul PR;
8. Se interpolează graficul PR în 11 (101) puncte;
9. Se calculează precizia medie;
10. Se calculează media preciziilor medie pe toate clasele.

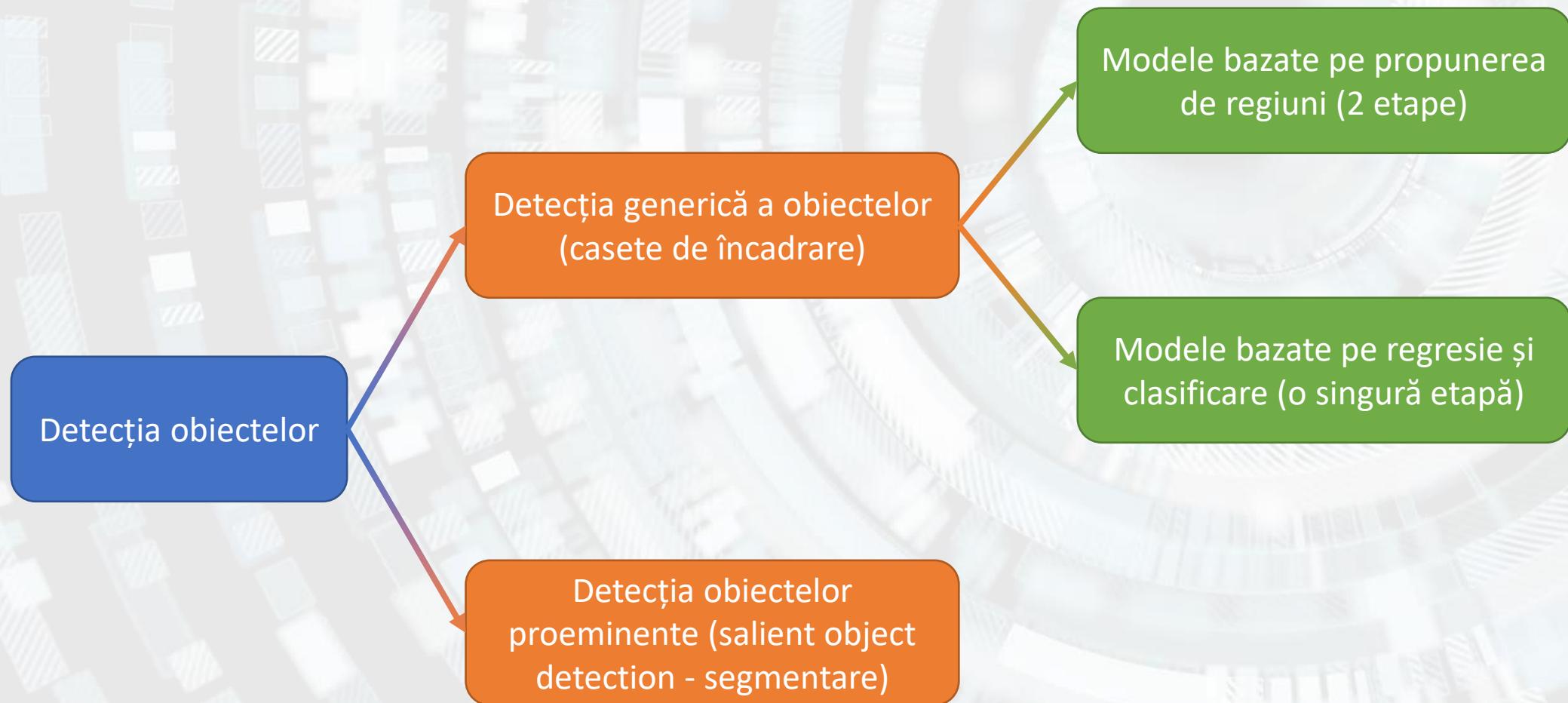
CLASS	dog	person	sheep	truck	teddy
AP	0.349	0.545	0.00	1.00	0.50

$$\begin{aligned}
 mAP &= \frac{1}{5} (AP_{dog} + AP_{person} + AP_{sheep} + AP_{truck} + AP_{teddy}) \\
 &= \frac{1}{5} (0.349 + 0.545 + 0.00 + 1.00 + 0.50) \\
 &= 47.88\%
 \end{aligned}$$

Detectia obiectelor – modele

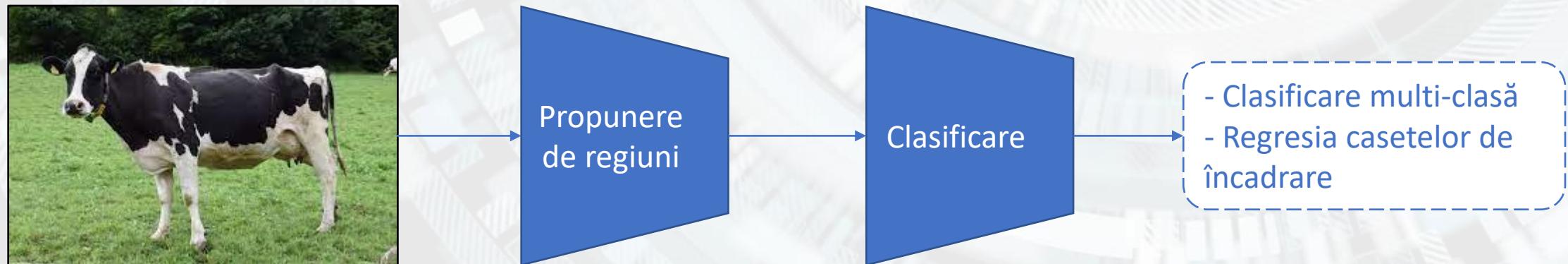
- Fiecare model de detectie a obiectelor are nevoie de o arhitectura specializata in extragerea unui set de traseuri cat mai distinctiv – elementul in jurul caruia se construieste modelul (backbone). Arhitecturi “backbone” populare:
 - AlexNet [6];
 - VGG [9];
 - GoogLeNet/Inception [10];
 - ResNet [11];
 - ResNeXt [26];
 - CSPNet [27];
 - EfficientNet [28];

Detectia obiectelor – modele



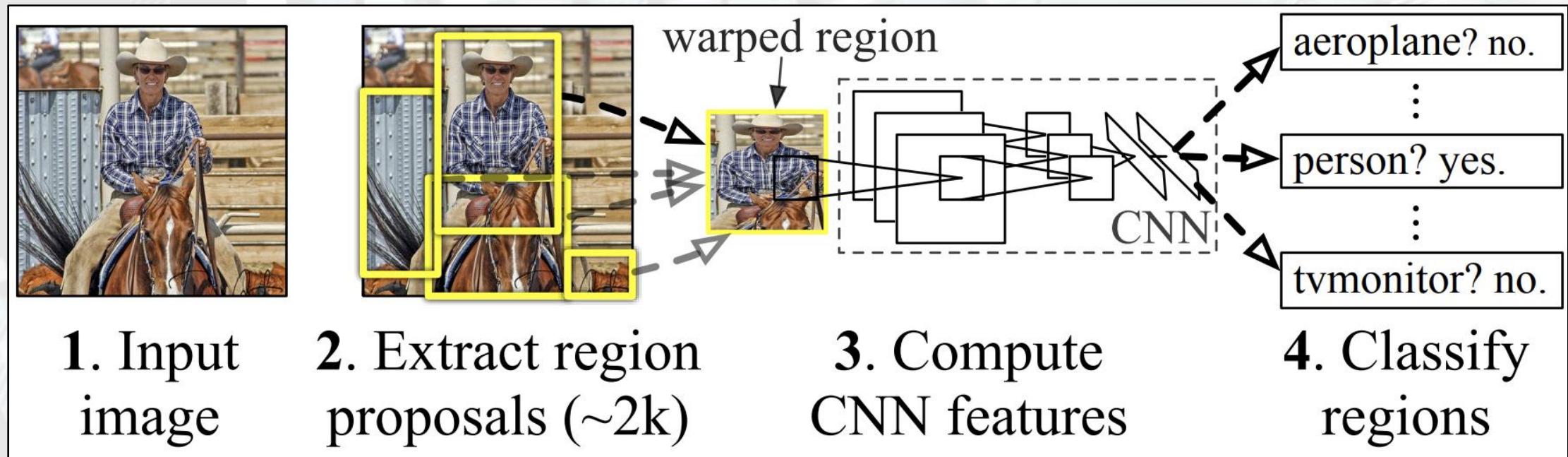
Detectia obiectelor – modele în 2 etape

- Modelele în 2 etape presupun parcurgerea următoarelor... 2 etape:
 1. Propunerea și extragerea unor regiuni de interes (potențiale obiecte);
 2. Clasificarea regiunilor propuse + regresia casetelor de încadrare.



Detectia obiectelor – modele în 2 etape

Regions with CNN features (R-CNN) [29]



Detectia obiectelor – modele în 2 etape

R-CNN – extragerea propunerilor de regiuni:

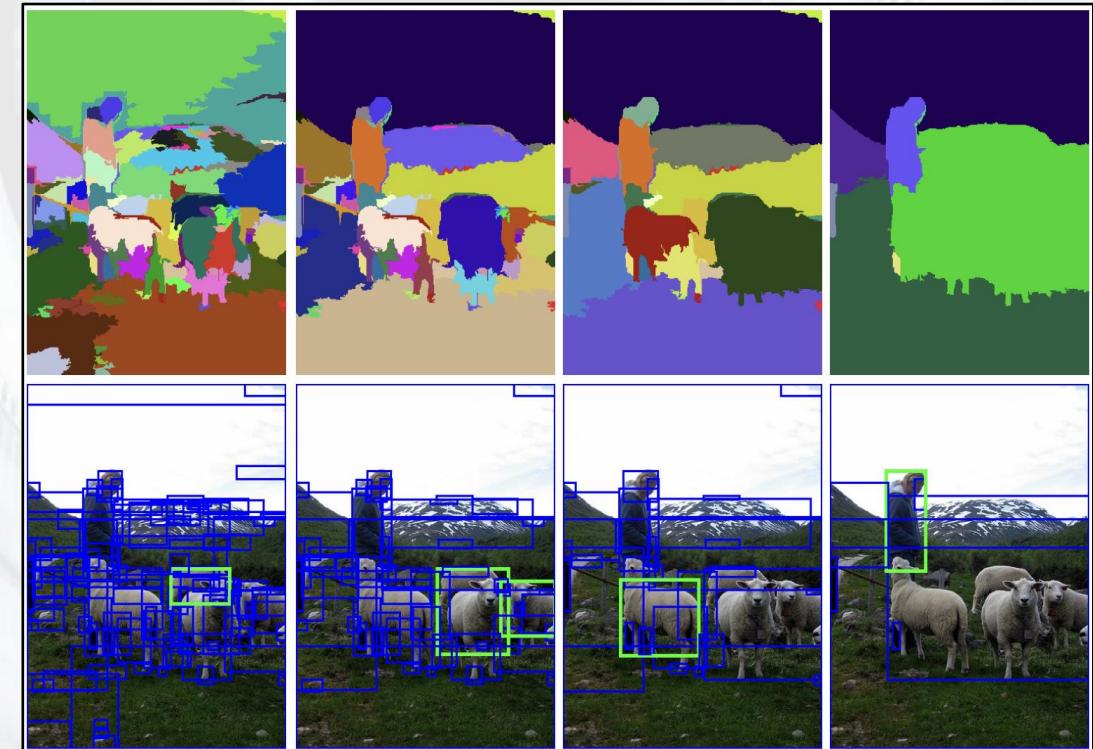
1. Se realizează o suprasegmentare a fiecărei imagini de intrare (algoritmul Felzenszwalb & Huttenlocher [30])



Detectia obiectelor – modele în 2 etape

R-CNN – extragerea propunerilor de regiuni:

2. Regiunile segmentate se grupeaza aglomerativ, ierarhic, în funcție de similitudine (algoritmul Selective Search [31]), până la obtinerea numărului dorit de regiuni propuse (2k).



Detectia obiectelor – modele în 2 etape

R-CNN – extragerea trăsăturilor din regiunile propuse:

- Fiecare regiune este redimensionată (warped) la 227×227 pixeli;
- Regiunile sunt propagate prin rețeaua de extragere de trăsături – backbone (AlexNet);
- Se obține un vector de trăsături de dimensiune 4096 pentru fiecare regiune.

180 x 430



227 x 227



Detectia obiectelor – modele în 2 etape

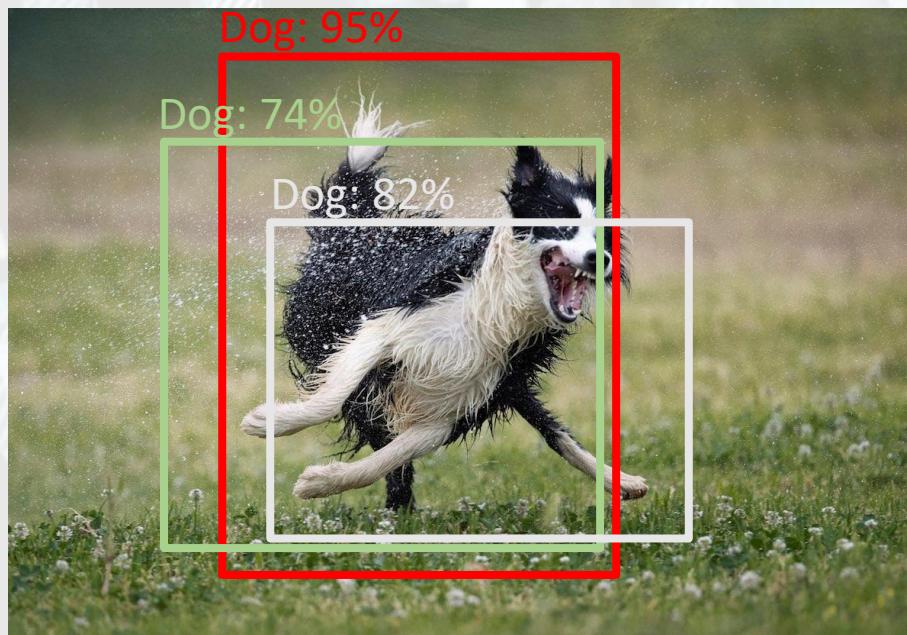
R-CNN – clasificarea regiunilor propuse:

- Fiecare vector de trăsături este procesat de către un clasificator SVM antrenat pentru fiecare clasă, în parte => N clasificatori ($N = \text{nr. clase}$);
- Fiecare clasificator dă o decizie asupra aceleiași regiuni => N scoruri diferite. Se alege clasa corespunzătoare celui mai mare scor. Dacă acest scor este mai mare decât un prag predefinit (e.g. 0.7), atunci se consideră că a fost detectat un obiect din acea clasă. Altfel, se consideră ca fiind „background”.

Detectia obiectelor – modele în 2 etape

R-CNN – clasificarea regiunilor propuse:

- Există situații când un obiect are asociate mai multe casete de încadrare asemănătoare:



Se utilizează mecanismul Non-Maximum Supression (NMS):

1. Se selectează caseta de încadrare cu cel mai mare scor (95%);
2. Se calculează IoU dintre caseta de la pct. 1 și toate celelalte casete asociate aceleiași clase;
3. Dacă IoU dintre caseta de scor maxim și o casetă țintă depășește un prag stabilit (e.g. 0.3), se elimină caseta țintă. Altfel, se trece mai departe.
4. Se repetă algoritmul până la epuizarea tuturor casetelor de încadrare.

Detectia obiectelor – modele în 2 etape

R-CNN – clasificarea regiunilor propuse:

- După stabilirea detectiilor, se rulează și o ramură de regresie a casetei de încadrare pentru rafinarea încadrării.



Detectia obiectelor – modele în 2 etape

R-CNN – antrenare:

1. Pre-train: rețeaua backbone (AlexNet) este pre-antrenată pe ImageNet, folosind adnotări la nivel de imagine (nu obiect);
2. Fine-tune: se înlocuiește ultimul strat al clasificatorului, de dimensiune 1000, cu un clasificator de dimensiune N+1 (N clase + background) și se antrenează rețeaua backbone pentru a o adapta la noua sarcină (obiecte decupate din imagini și redimensionate);
3. Se înlocuiește clasificatorul final cu N clasificatori SVM, câte unul pentru fiecare clasă și se optimizează.

Detectia obiectelor – modele în 2 etape

R-CNN – rezultate:

VOC 2010 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
DPM v5	49.2	53.8	13.1	15.3	35.5	53.4	49.7	27.0	17.2	28.8	14.7	17.8	46.4	51.2	47.7	10.8	34.2	20.7	43.8	38.3	33.4
UVA	56.2	42.4	15.3	12.6	21.8	49.3	36.8	46.1	12.9	32.1	30.0	36.5	43.5	52.9	32.9	15.3	41.1	31.8	47.0	44.8	35.1
Regionlets	65.0	48.9	25.9	24.6	24.5	56.1	54.5	51.2	17.0	28.9	30.2	35.8	40.2	55.7	43.5	14.3	43.9	32.6	54.0	45.9	39.7
SegDPM	61.4	53.4	25.6	25.2	35.5	51.7	50.6	50.8	19.3	33.8	26.8	40.4	48.3	54.4	47.1	14.8	38.7	35.0	52.8	43.1	40.4
R-CNN	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	50.2
R-CNN BB	71.8	65.8	53.0	36.8	35.9	59.7	60.0	69.9	27.9	50.6	41.4	70.0	62.0	69.0	58.1	29.5	59.4	39.3	61.2	52.4	53.7

Detectia obiectelor – modele în 2 etape

R-CNN – concluzii:

- A depășit toate modelele existente de detectie a obiectelor la acea vreme cu o diferență semnificativă;
- Algoritmul este încet (47s/imagină pentru VGG16);
- Antrenarea este complexă – proces în 3 pași și durează mult (84 ore);
- Propunerea de regiuni se face conform unui algoritm fix, fără învățare.

Detectia obiectelor – modele în 2 etape

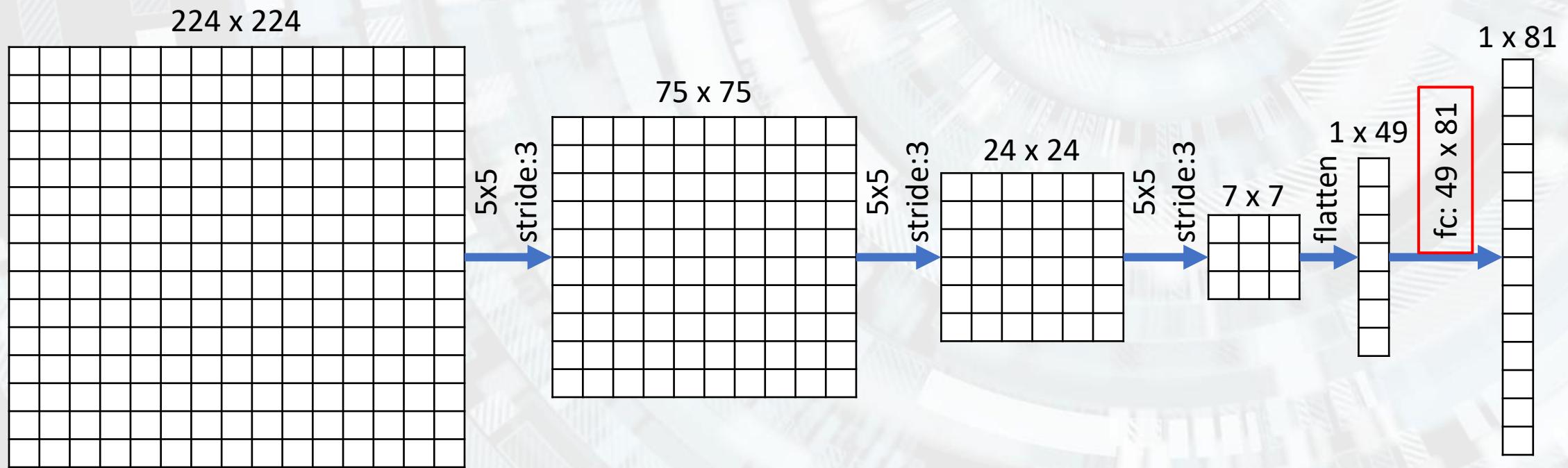
Spatial Pyramid Pooling Net – SPP-Net [31]

➤ Se abordează 2 probleme:

1. Redimensionarea imaginilor de intrare – toate imaginile sunt redimensionate la 227×227 pixeli => se pierde raportul de aspect;
2. Rularea inferenței pentru fiecare regiune propusă – modelul de clasificare se rulează individual pentru fiecare dintre cele 2000 de regiuni propuse.

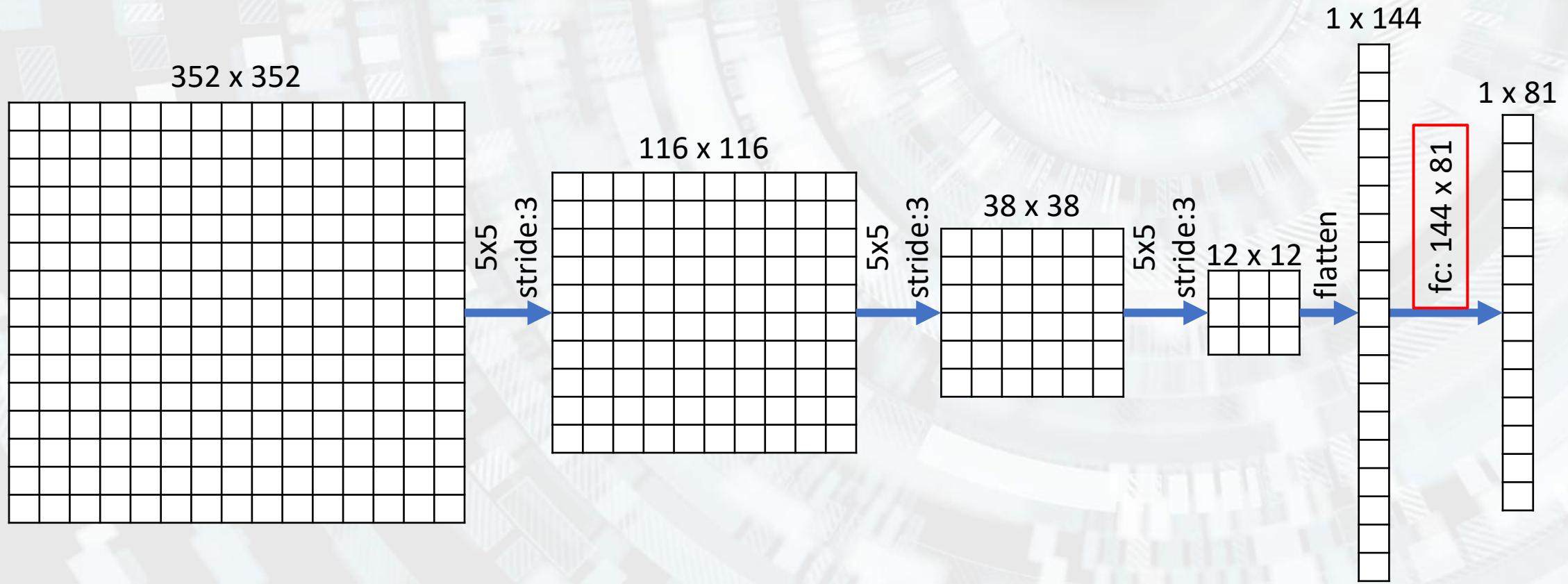
Detectia obiectelor – modele în 2 etape

SPP-Net – redimensionarea imaginilor de intrare



Detectia obiectelor – modele în 2 etape

SPP-Net – redimensionarea imaginilor de intrare



Detectia obiectelor – modele în 2 etape

SPP-Net – redimensionarea imaginilor de intrare

- Imagine de dimensiune $227 \times 227 \Rightarrow$ strat complet conectat cu 49 de neuroni de intrare și 81 de ieșire;
- Imagine de dimensiune $352 \times 352 \Rightarrow$ strat complet conectat cu 144 de neuroni de intrare și 81 de ieșire;

Straturi cu intrări de dimensiuni diferite, dar ieșiri de aceeași dimensiune \Rightarrow imposibil pentru straturi complet conectate \Rightarrow modele diferite.

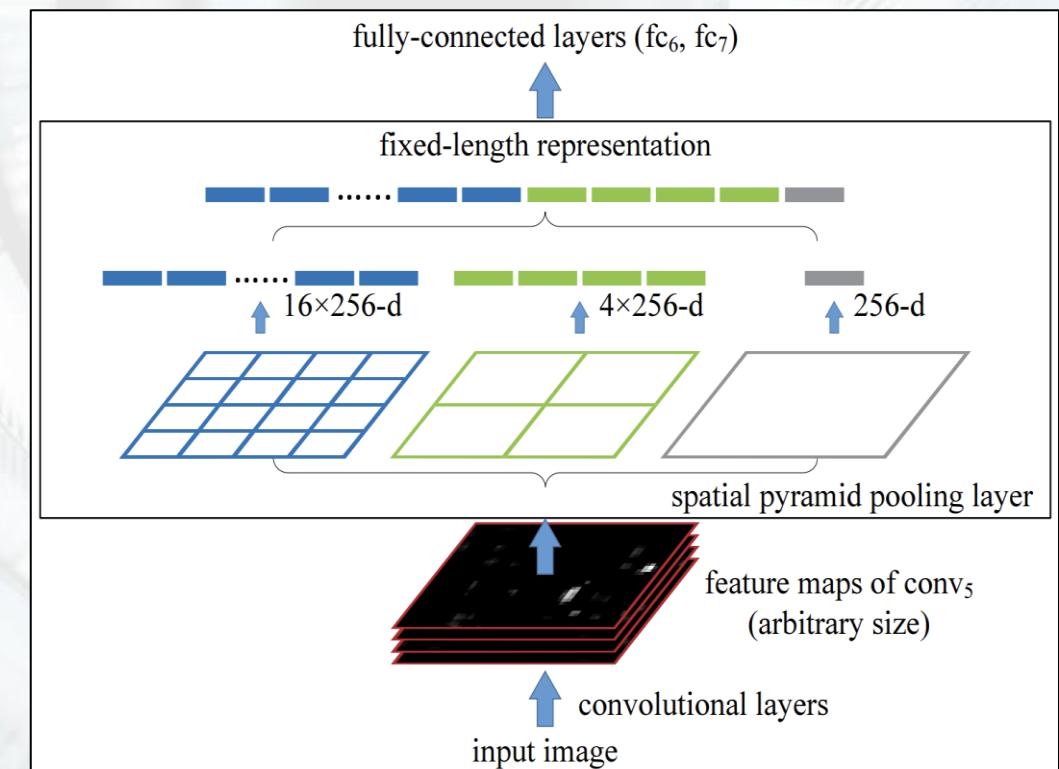
Cum procedăm ca să prelucrăm ambele imagini cu același model?

1. Redimensionăm imaginea de intrare \Rightarrow se pierde raportul de aspect.
2. Folosim un strat agnostic la dimensiunea intrării, cu ieșire fixă.

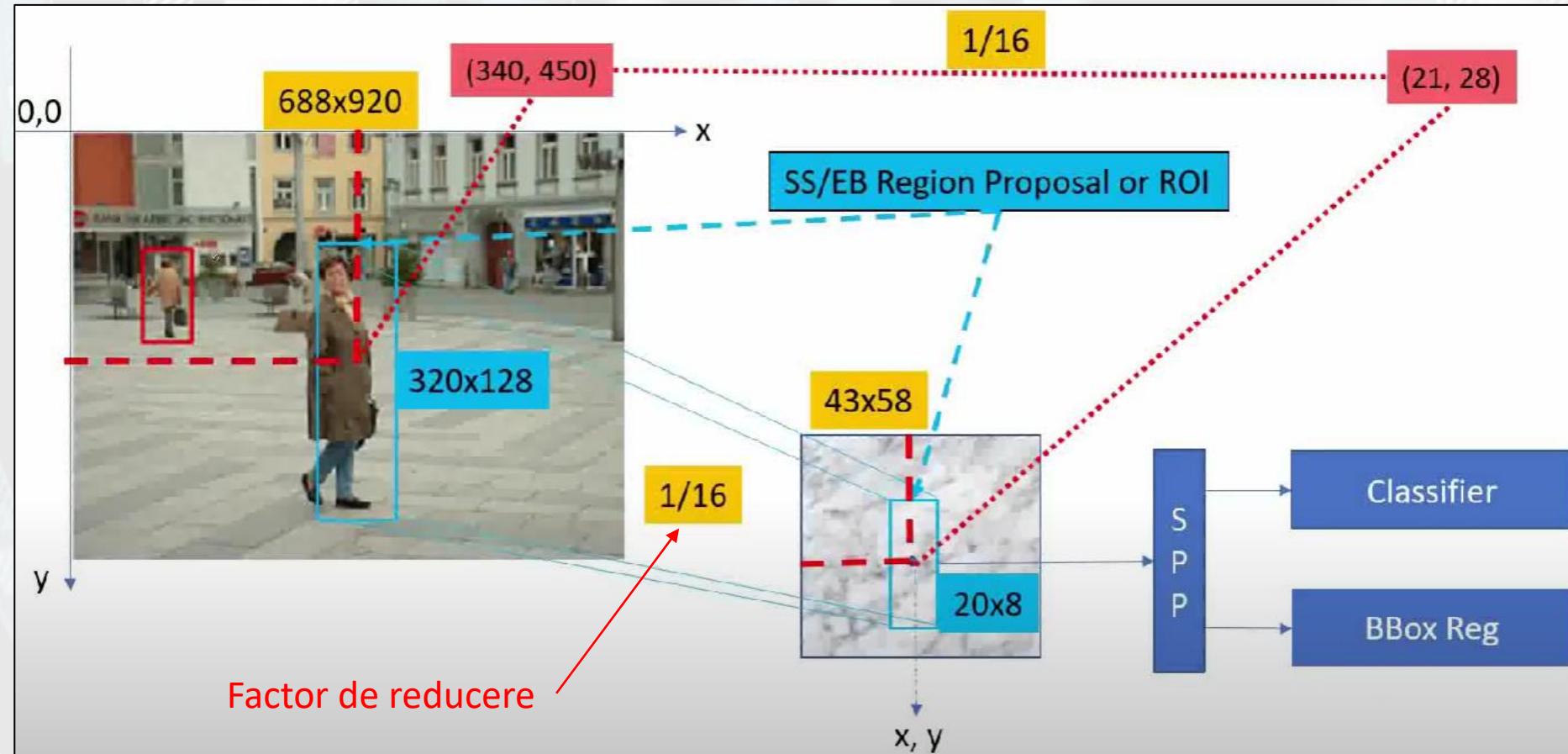
Detectia obiectelor – modele în 2 etape

SPP-Net – Spatial Pyramid Pooling (SPP)

- Harta de trăsături are 256 de canale;
- Harta de trăsături este împărțită într-un număr fix de chenare (bins);
- Se aplică max pooling pe fiecare chenar;
- Se obțin 21 de valori pe fiecare canal;
- La antrenare, se folosesc imagini de intrare de diferite dimensiuni;
- Este introdus ca un nou strat – poate fi cuplat la orice model.



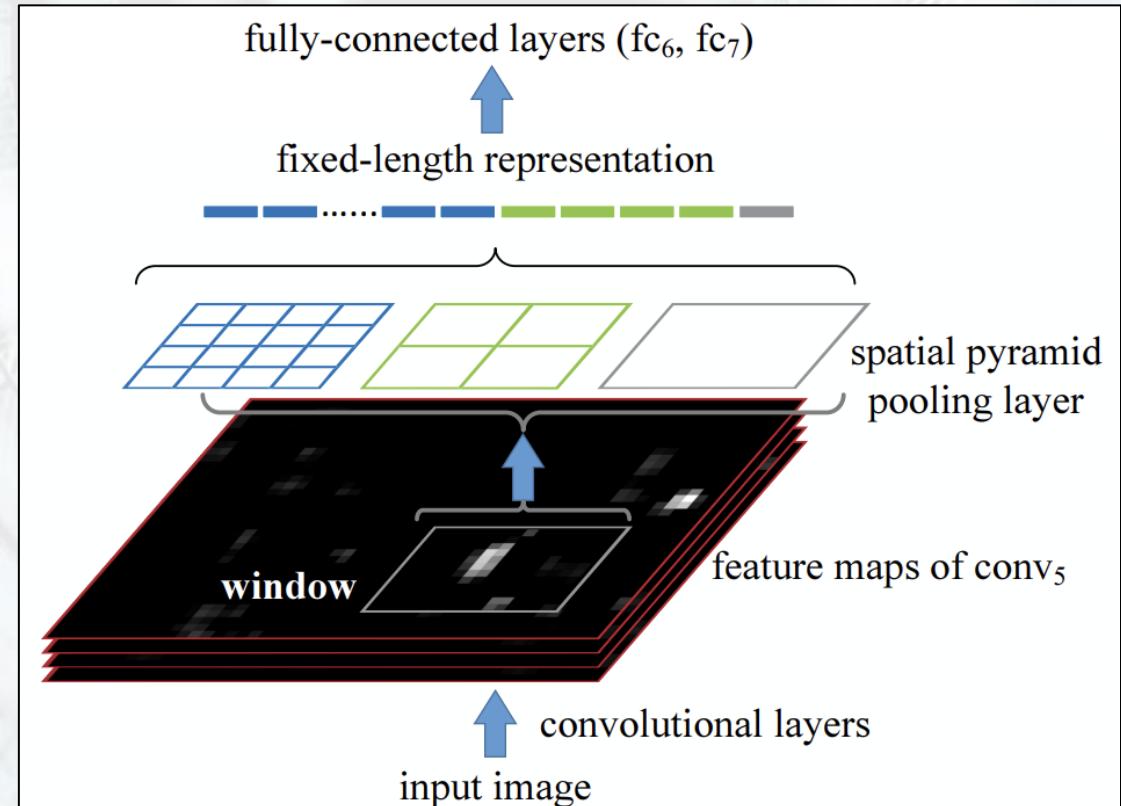
Detectia obiectelor – modele în 2 etape



Detectia obiectelor – modele în 2 etape

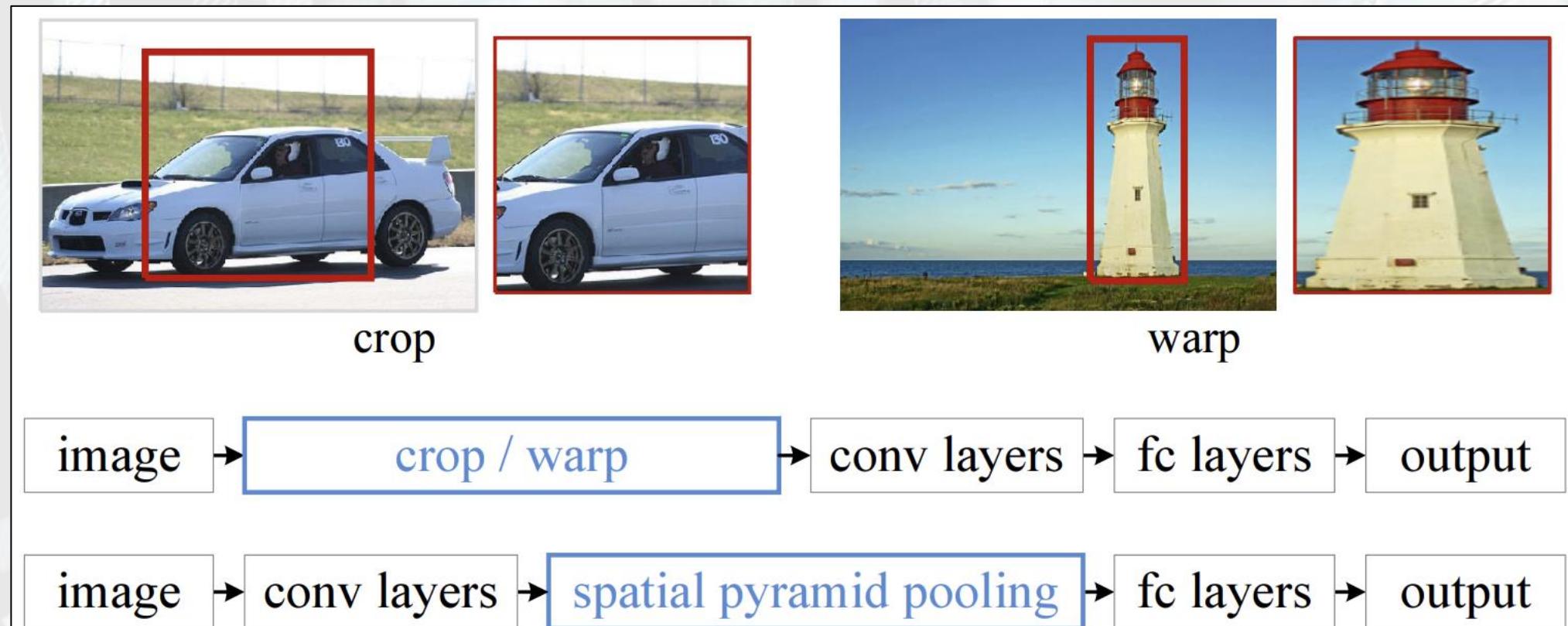
SPP-Net – detectia obiectelor

- Se propun regiuni de interes pe imaginea initială (Selective Search);
- Regiunea de interes este proiectată pe harta de trăsături;
- SPP se aplică pe regiunea proiectată (fără redimensionare);
- Harta de trăsături este extrasă o singură dată și utilizată pentru toate regiunile de interes propuse.



Detectia obiectelor – modele în 2 etape

SPP-Net – comparație cu RCNN



Detectia obiectelor – modele în 2 etape

SPP-Net – rezultate

	SPP (1-sc) (ZF-5)	SPP (5-sc) (ZF-5)	R-CNN (Alex-5)
pool ₅	43.0	<u>44.9</u>	44.2
fc ₆	42.5	<u>44.8</u>	<u>46.2</u>
ftfc ₆	52.3	<u>53.7</u>	53.1
ftfc ₇	54.5	<u>55.2</u>	54.2
ftfc ₇ bb	58.0	59.2	58.5
conv time (GPU)	0.053s	0.293s	8.96s
fc time (GPU)	0.089s	0.089s	0.07s
total time (GPU)	0.142s	0.382s	9.03s
speedup (vs. RCNN)	64×	24×	-

Detectia obiectelor – modele în 2 etape

SPP-Net – concluzii

- Nu mai este nevoie de redimensionarea imaginilor de intrare;
- Nu mai este nevoie de redimensionarea regiunilor de inters;
- Nu mai este nevoie de extragerea trăsăturilor din fiecare regiune de interes – harta de trăsături este calculată o singură dată și utilizată pentru toate regiunile propuse;
- Stratul SPP se poate aplica la orice rețea neuronală;
- Păstrează același algoritm de antrenare cu R-CNN - extragerea de trăsături se antrenează separat de clasificator;
- Performanțe similare cu R-CNN.

Detectia obiectelor – modele în 2 etape

Fast R-CNN [32]

Dorește să rezolve mai multe probleme ale R-CNN și SPPNet:

- Antrenarea în mai multe etape;
- Antrenarea prea complexă;
- Detectia lentă a obiectelor;

Fast R-CNN = R-CNN + SPP

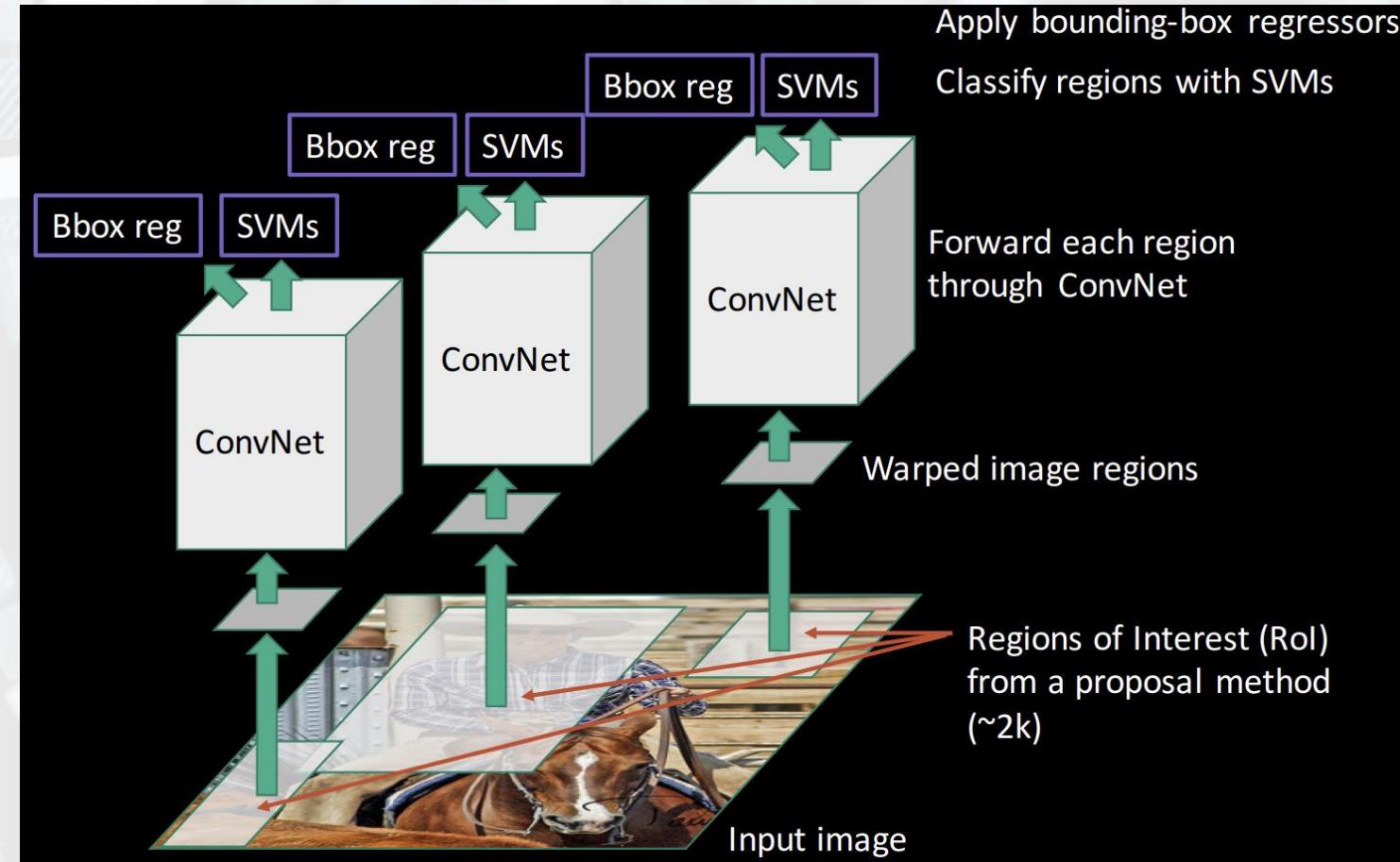
Detectia obiectelor – modele în 2 etape

Fast R-CNN = R-CNN + SPP

- Se folosește RoI Pooling = SPP cu o singură scală (e.g. 7x7), în loc de piramidă multi-scală;
- Se înlocuiesc SVM cu straturi softmax;
- Se face antrenare end-to-end;
- Se păstrează o ramură comună între clasificator și regresor;
- Se adaugă câte un strat complet conectat atât înainte de clasificator, cât și de regresor pentru a îmbunătăți performanțele.

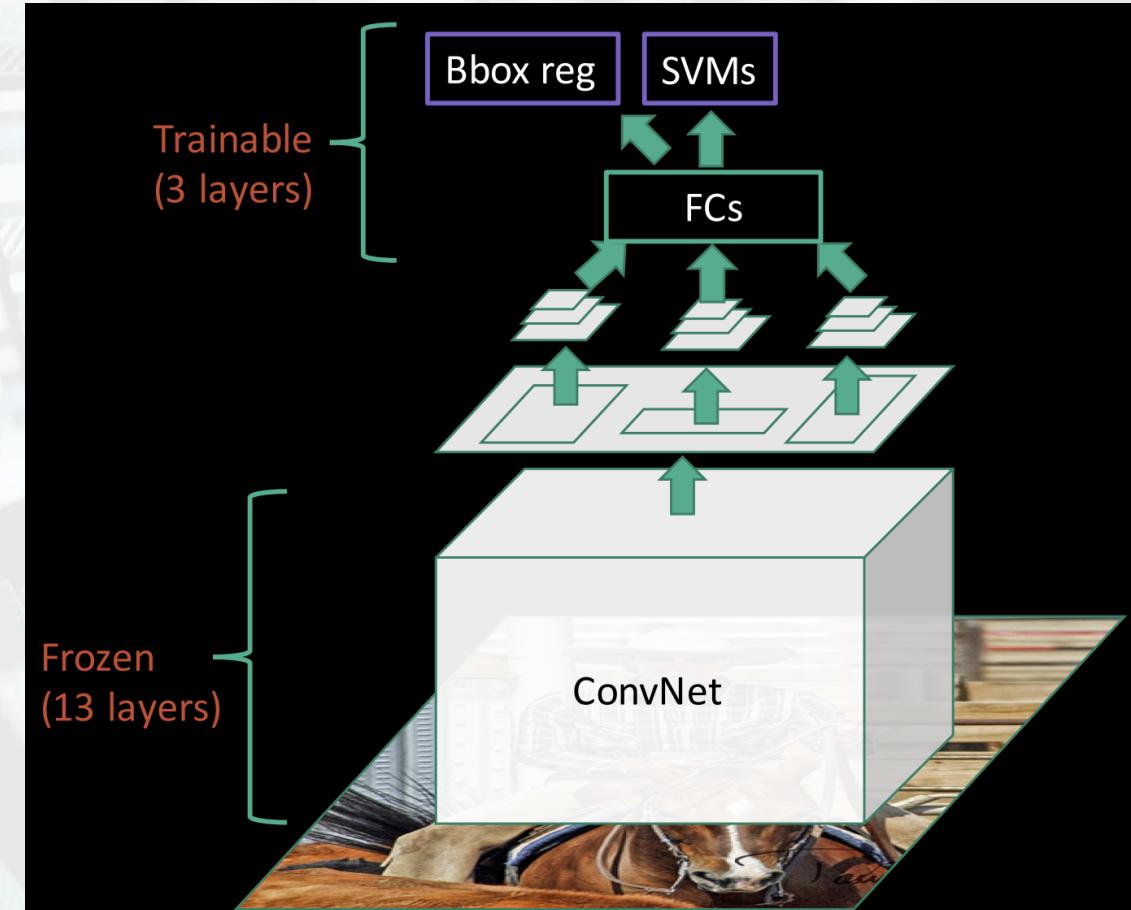
Detectia obiectelor – modele în 2 etape

R-CNN



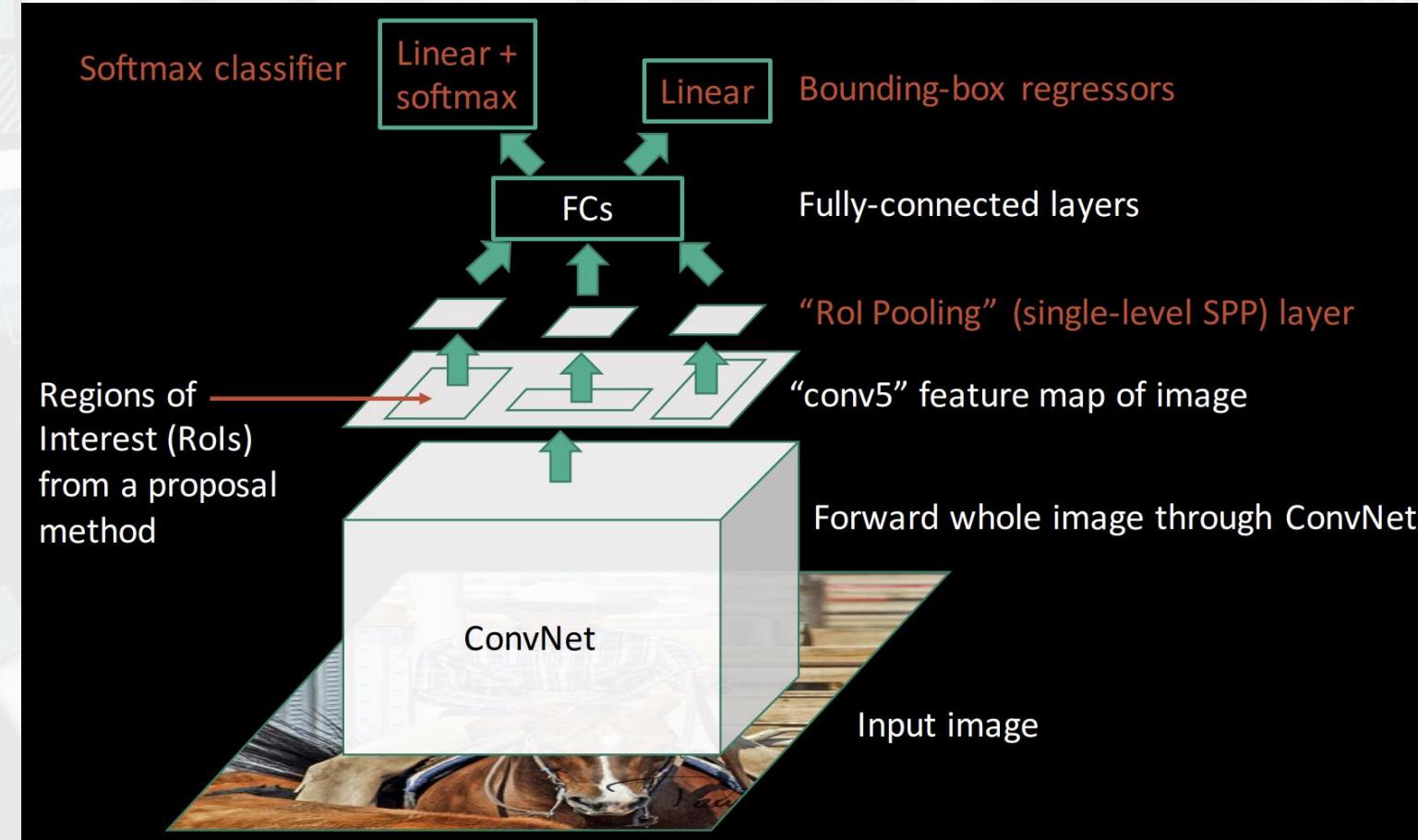
Detectia obiectelor – modele în 2 etape

SPP-Net



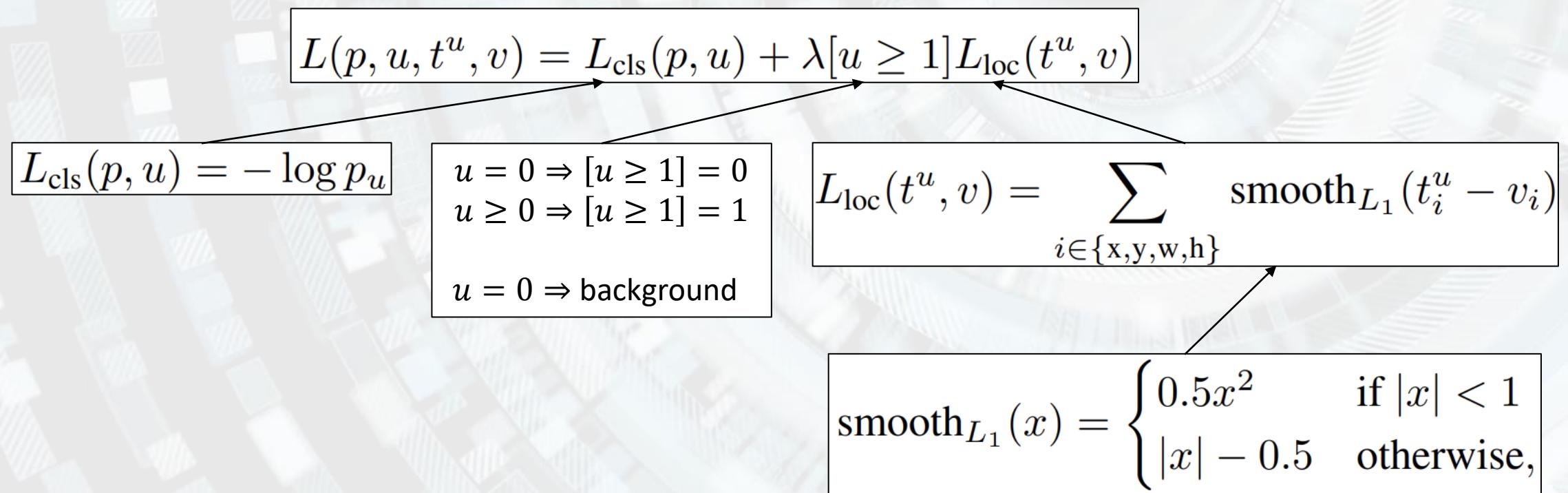
Detectia obiectelor – modele în 2 etape

Fast R-CNN



Detectia obiectelor – modele în 2 etape

Fast R-CNN – multi-task loss



Detectia obiectelor – modele în 2 etape

Fast R-CNN – rezultate

	Fast R-CNN			R-CNN			SPPnet
	S	M	L	S	M	L	$\dagger L$
train time (h)	1.2	2.0	9.5	22	28	84	25
train speedup	18.3×	14.0×	8.8×	1×	1×	1×	3.4×
test rate (s/im)	0.10	0.15	0.32	9.8	12.1	47.0	2.3
▷ with SVD	0.06	0.08	0.22	-	-	-	-
test speedup	98×	80×	146×	1×	1×	1×	20×
▷ with SVD	169×	150×	213×	-	-	-	-
VOC07 mAP	57.1	59.2	66.9	58.5	60.2	66.0	63.1
▷ with SVD	56.5	58.7	66.6	-	-	-	-

S = AlexNet

M = VGG_CNN_M_1024 (acceași adâncime cu S, dar mai lată)

L = VGG16

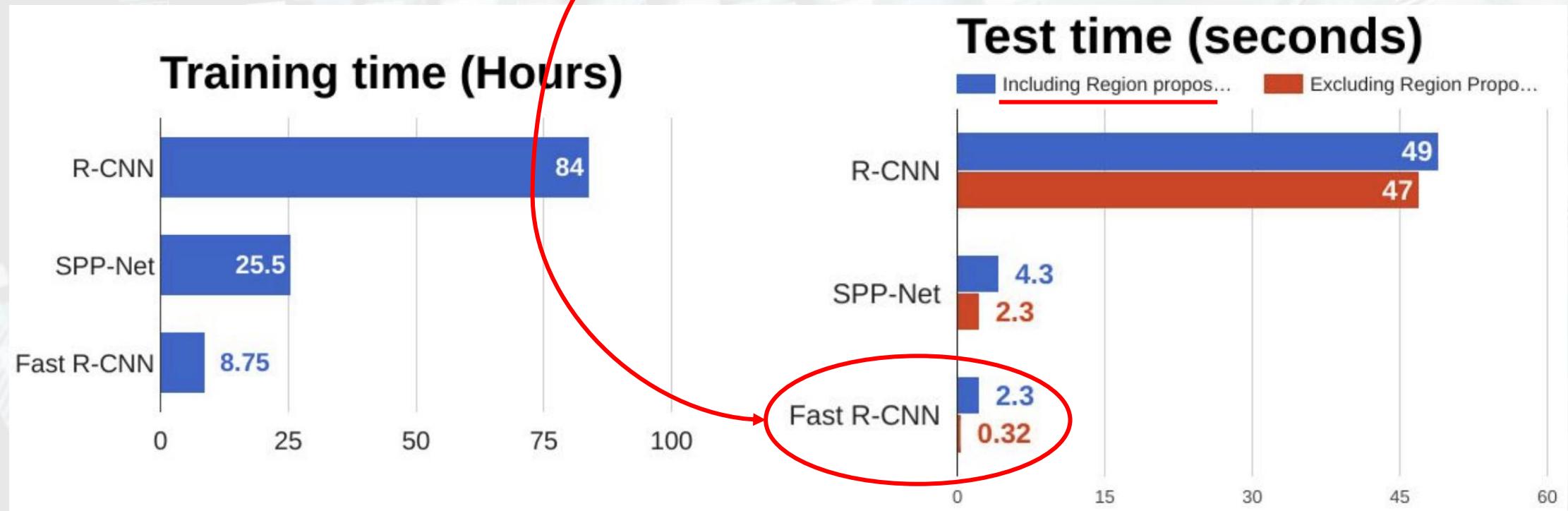
Detectia obiectelor – modele în 2 etape

Fast R-CNN – concluzii

- Antrenarea detectorului se face end-to-end;
- Implementează RoI pooling
- Propunerea de regiuni este, în continuare, externă detectorului;
- Model mai rapid decât variantele precedente (near real-time);
- Model mai performant decât variantele precedente;
- Se testează mai multe arhitecturi backbone.

Detectia obiectelor – modele în 2 etape

Spot the problem!



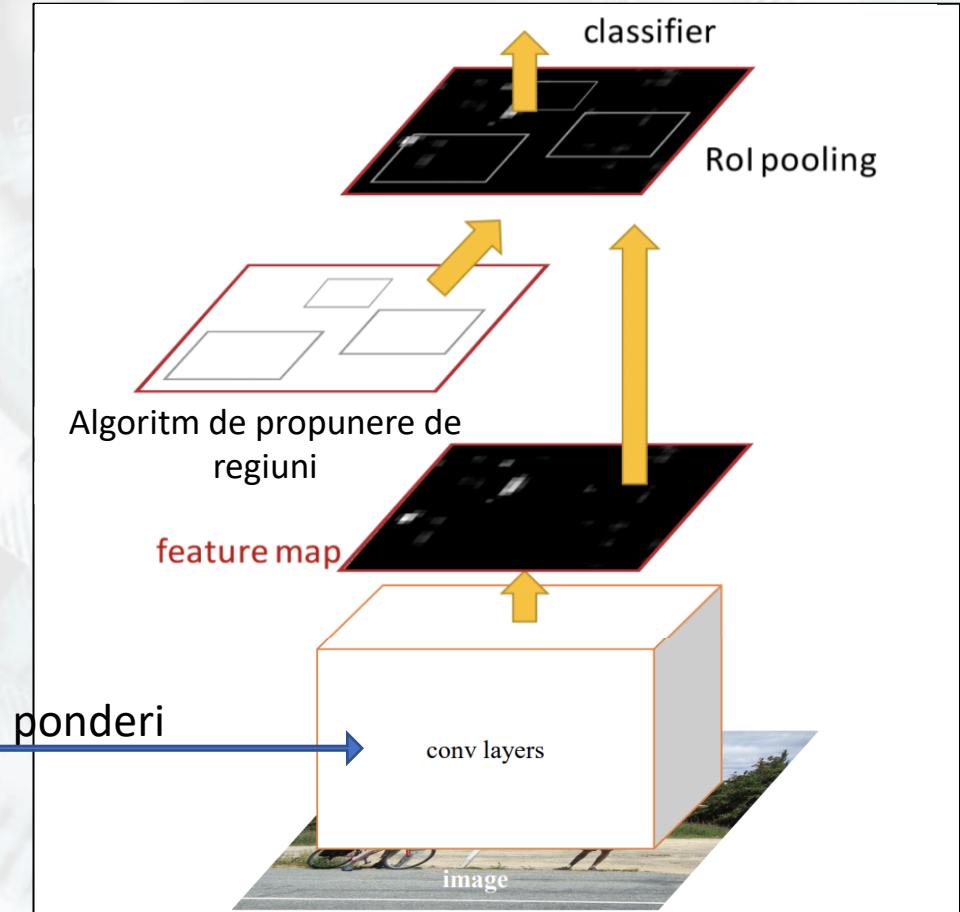
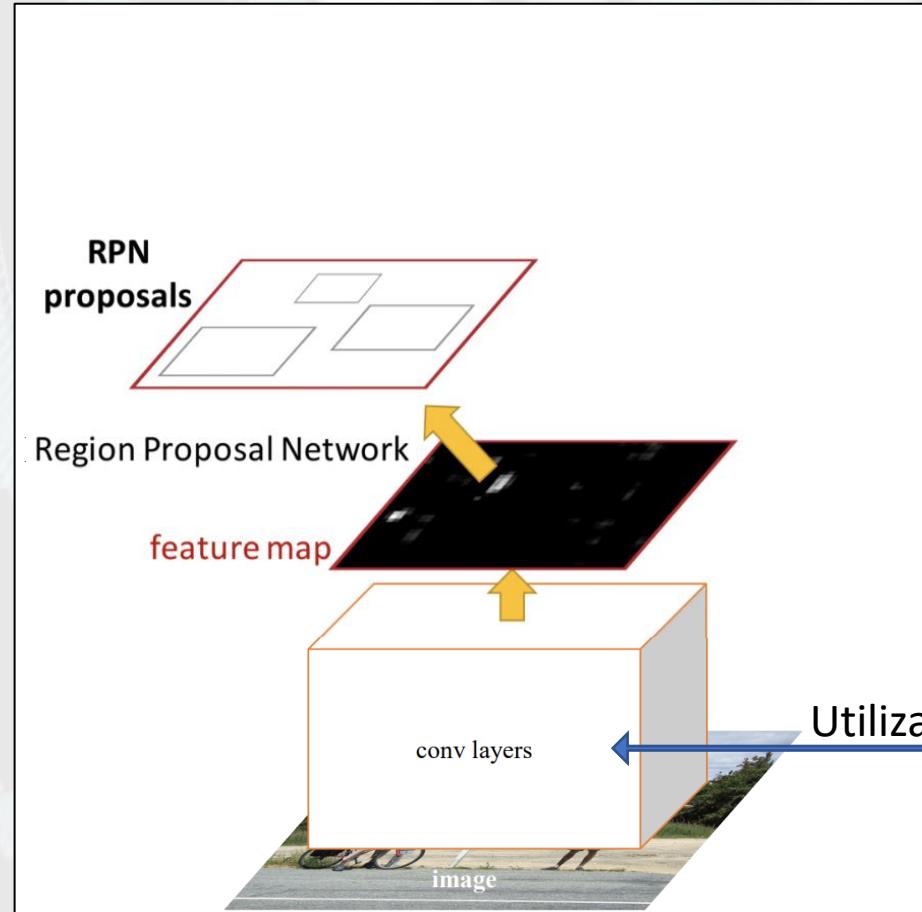
Detectia obiectelor – modele în 2 etape

Faster R-CNN [33]

- Își propune să rezolve problema propunerii de regiuni => propunerea de regiuni se va face cu o rețea neuronală
- Păstrează performanțele ridicate;
- Antrenarea întregului modul se face end-to-end.

Faster R-CNN = Fast R-CNN + RPN (Region Proposal Network)

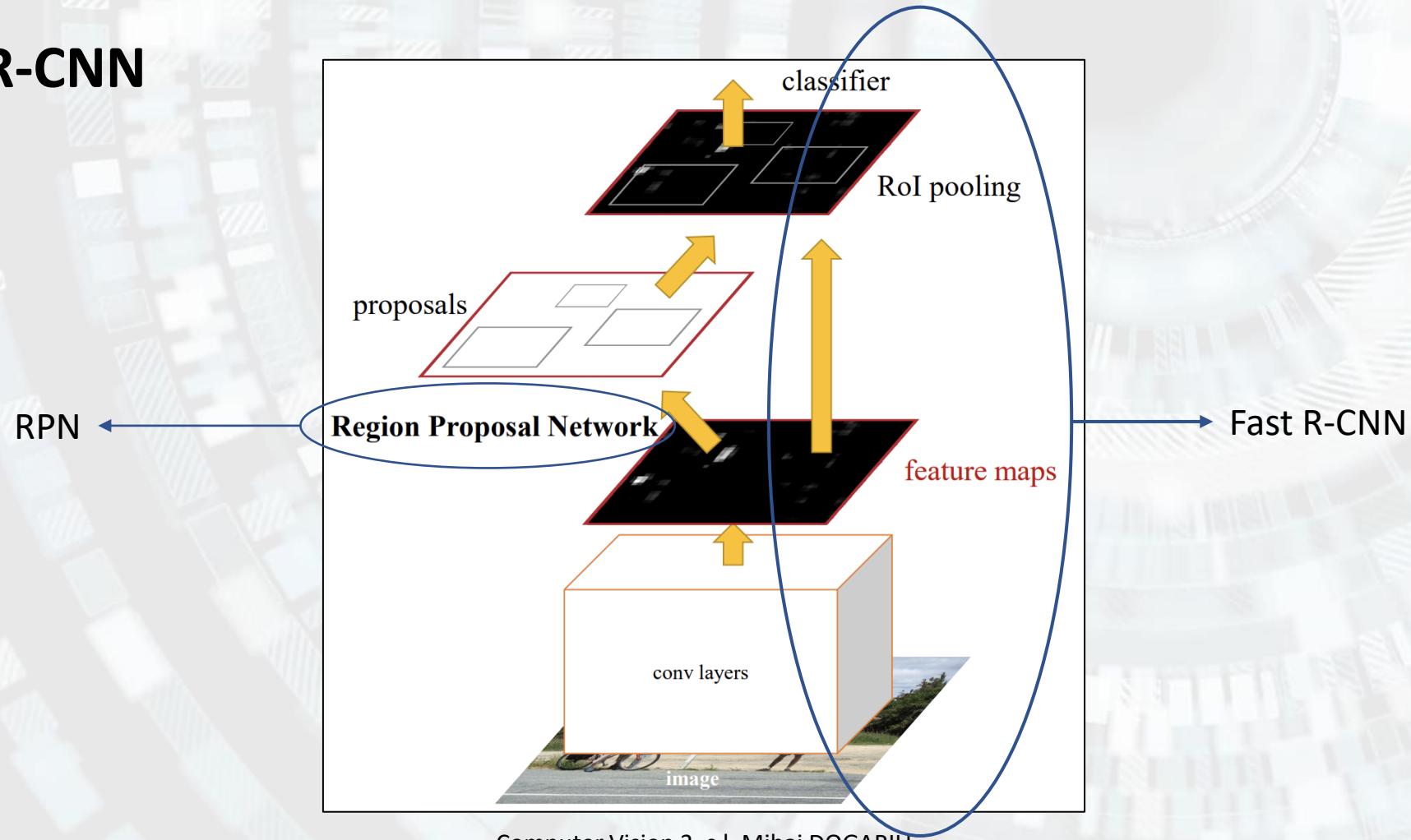
Detectia obiectelor – modele în 2 etape



Fast R-CNN

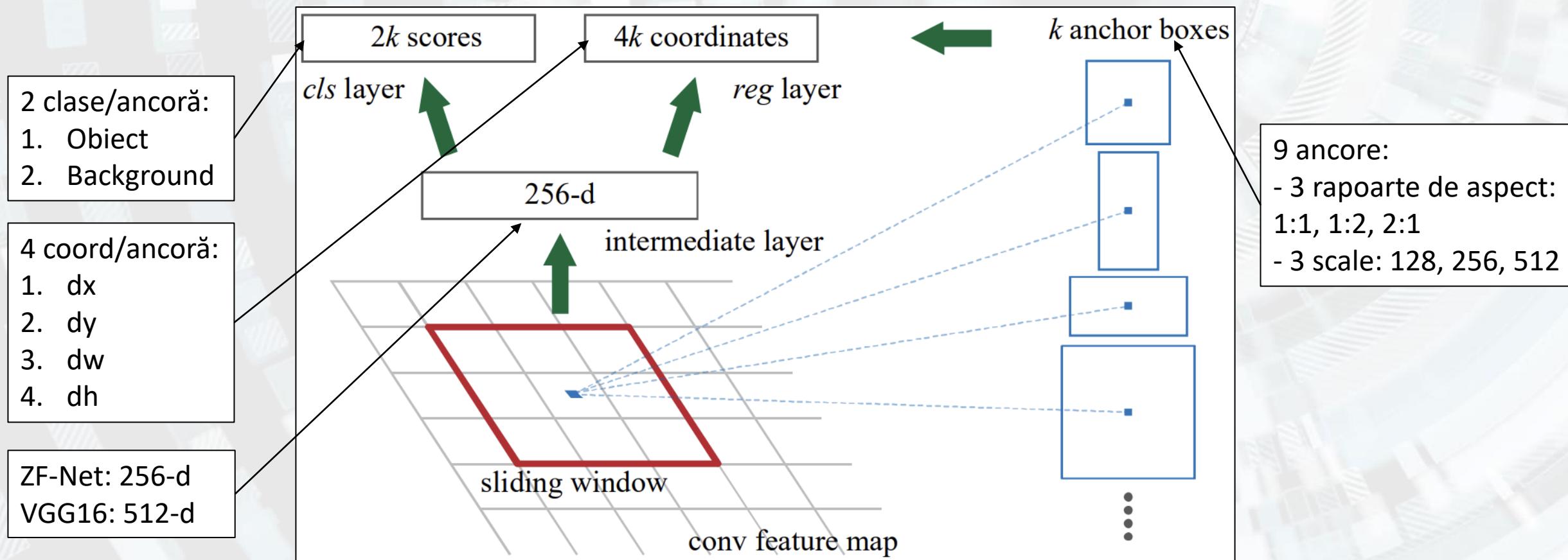
Detectia obiectelor – modele în 2 etape

Faster R-CNN



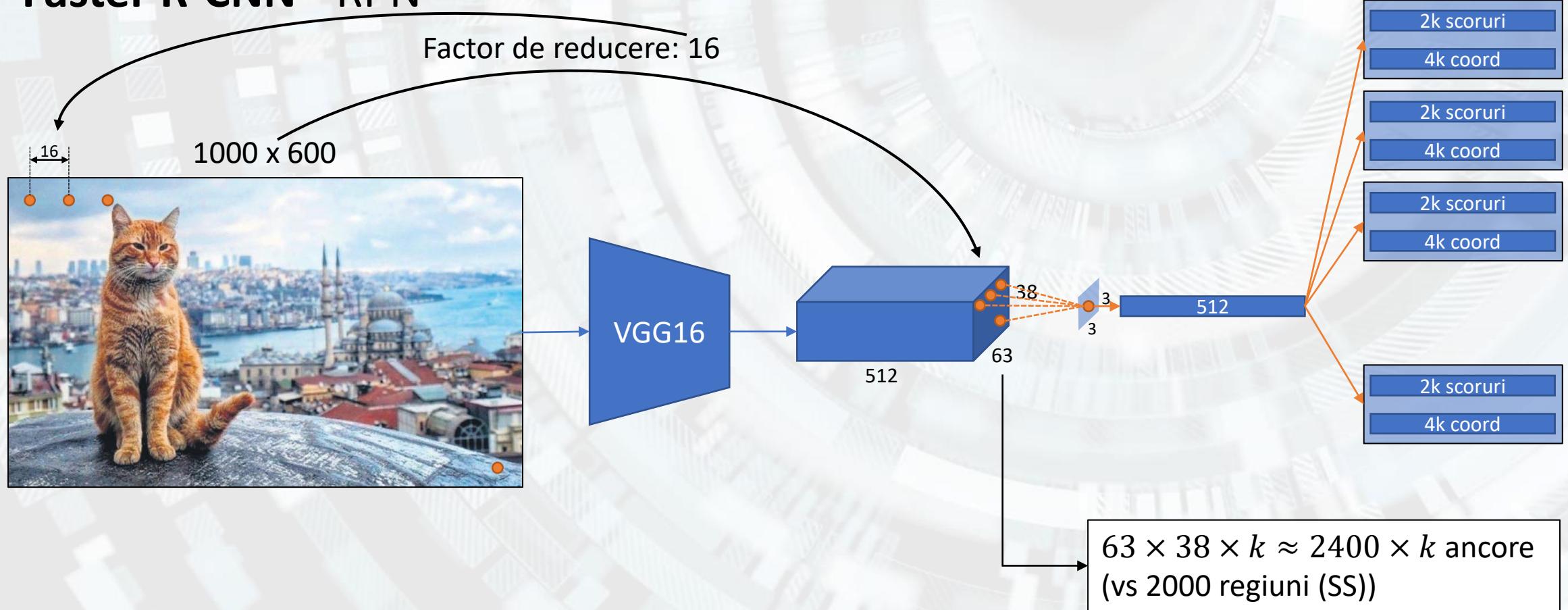
Detectia obiectelor – modele în 2 etape

Faster R-CNN - RPN



Detectia obiectelor – modele în 2 etape

Faster R-CNN - RPN



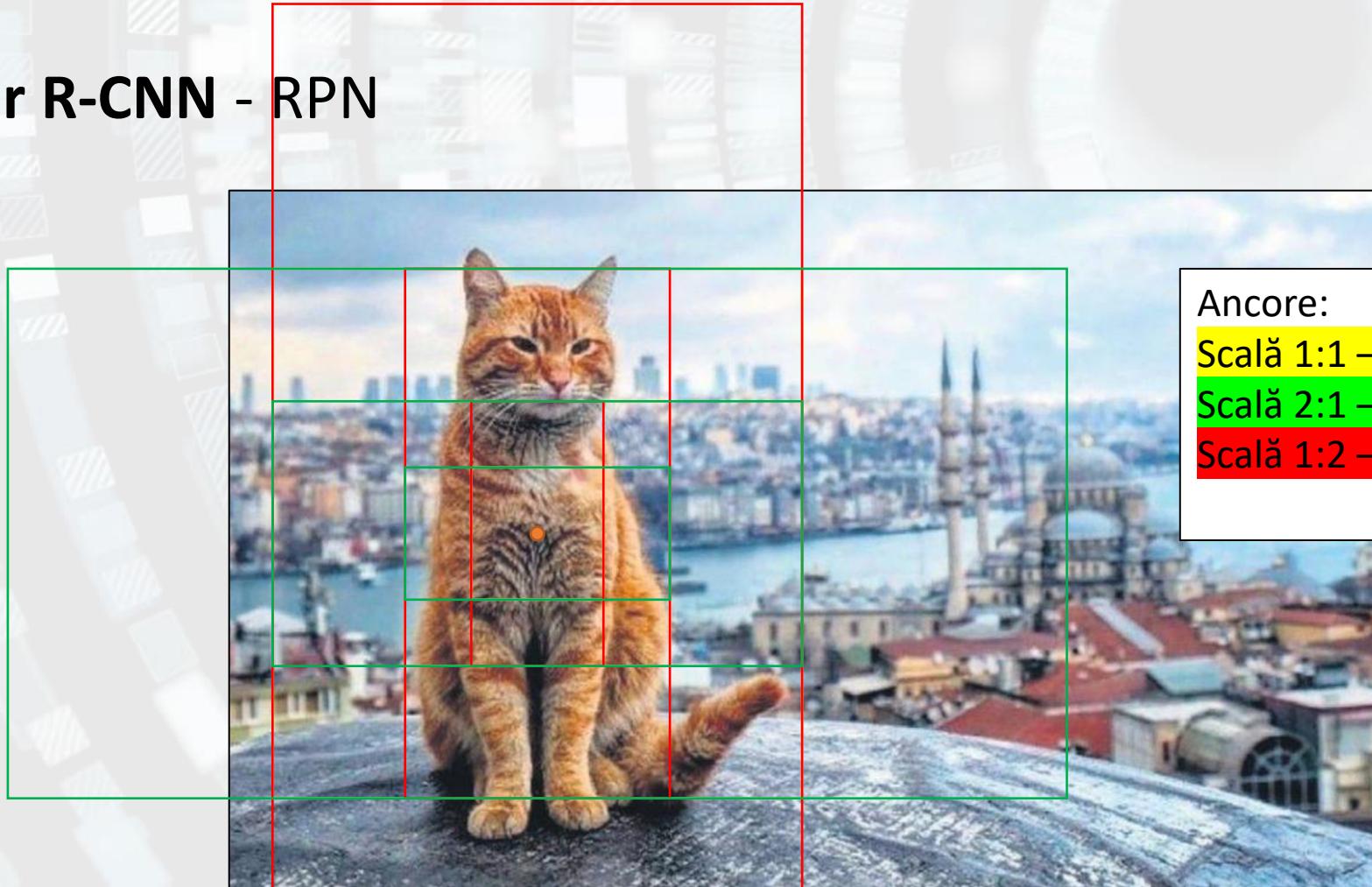
Detectia obiectelor – modele în 2 etape

Faster R-CNN - RPN



Detectia obiectelor – modele în 2 etape

Faster R-CNN - RPN



Ancore:

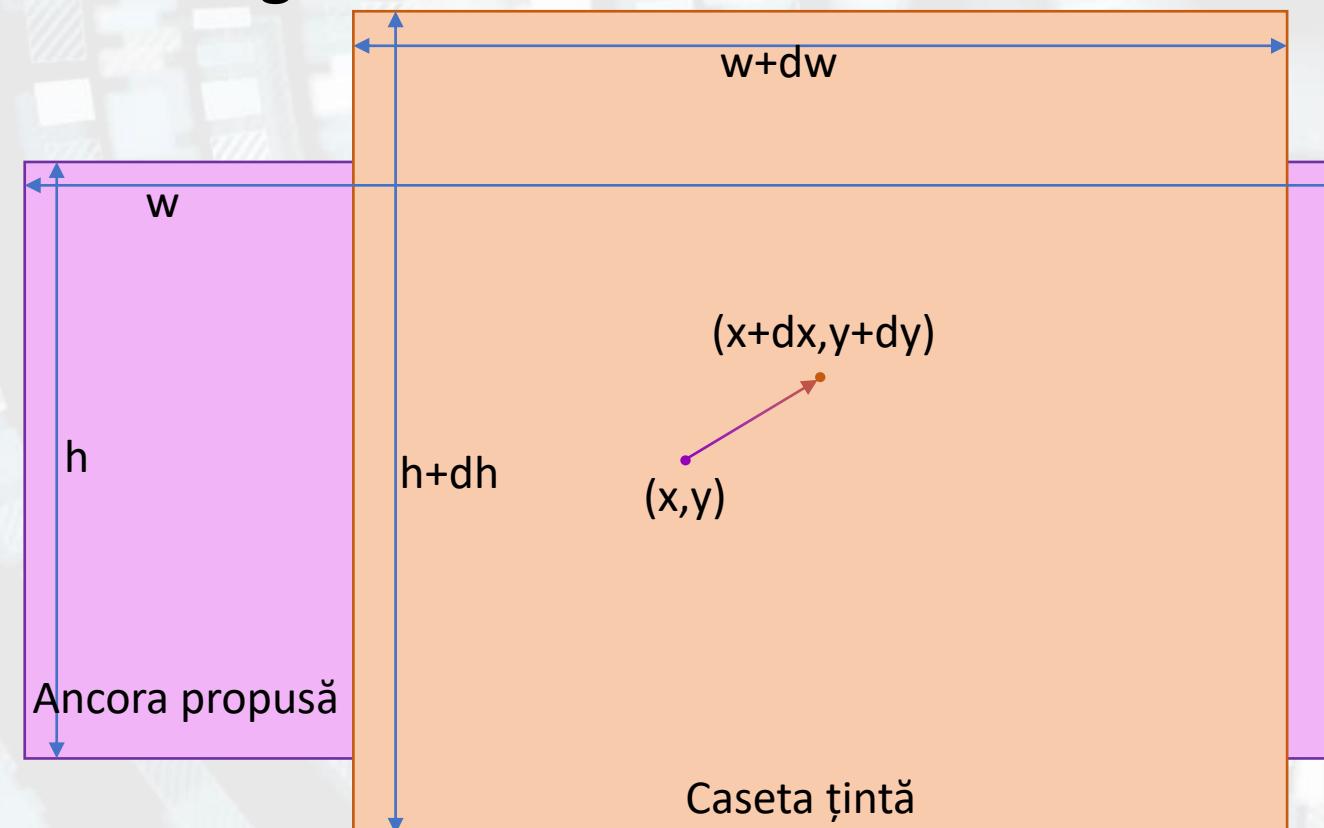
Scală 1:1 – 128x128, 256x256, 512x512

Scală 2:1 – 256x128, 512x256, 1024x512

Scală 1:2 – 128x256, 256x512, 512x1024

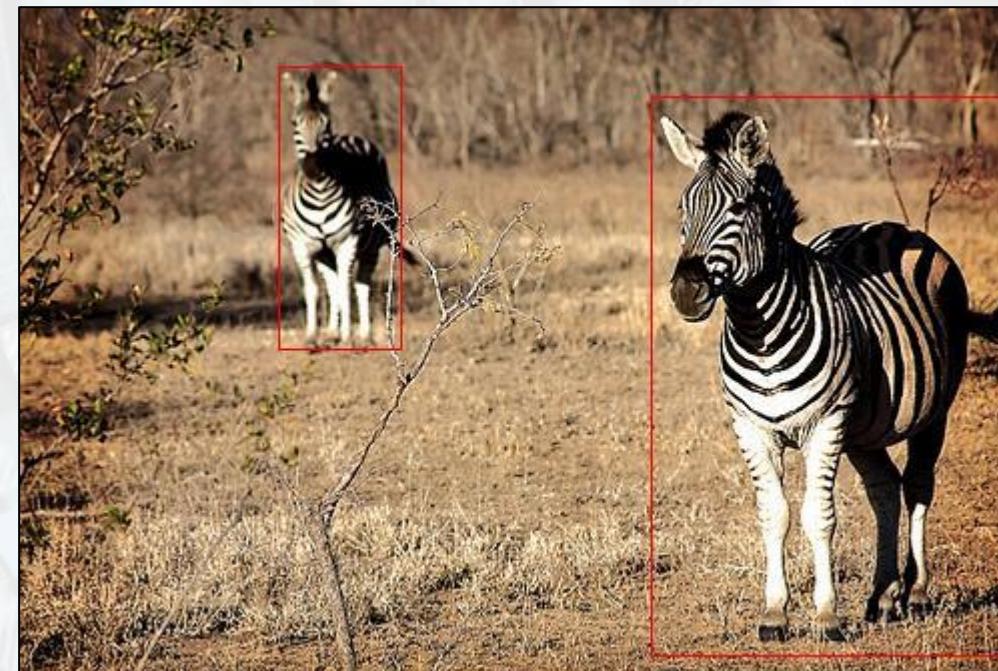
Detectia obiectelor – modele în 2 etape

Faster R-CNN – regresia Rol



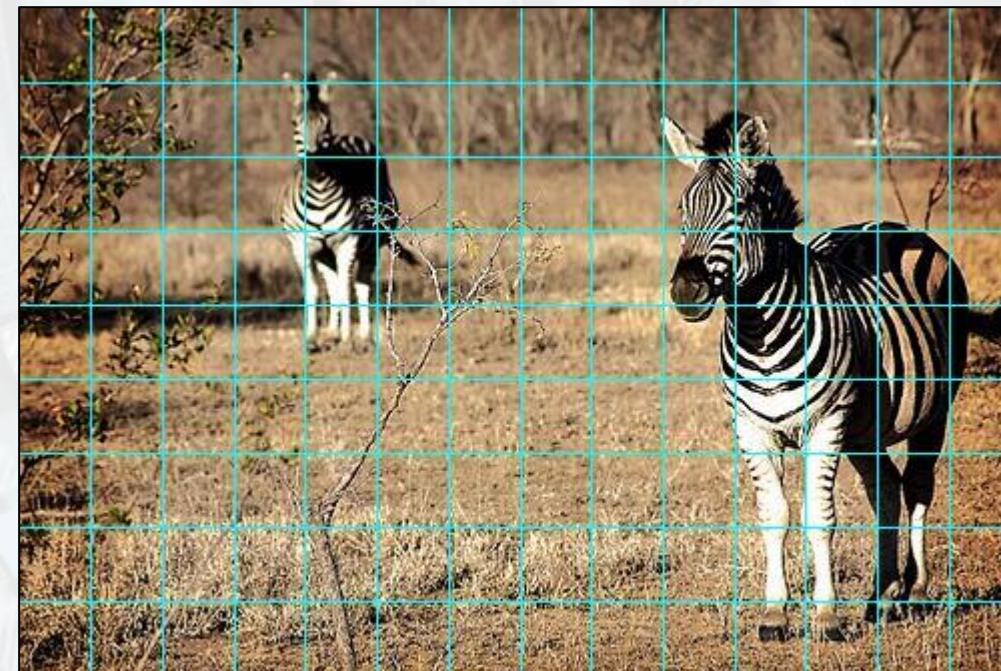
Detectia obiectelor – modele în 2 etape

Faster R-CNN – RPN (exemplu): groundtruth



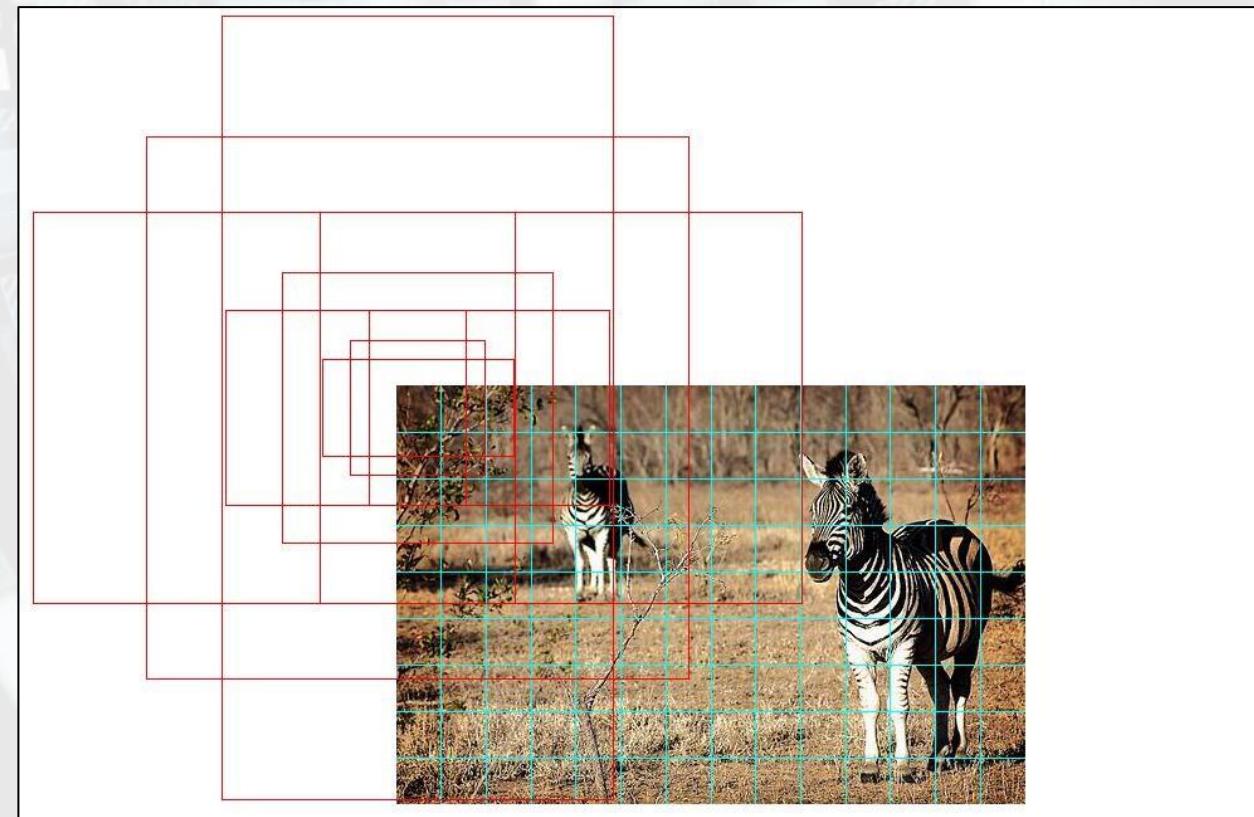
Detectia obiectelor – modele în 2 etape

Faster R-CNN – RPN (exemplu): generarea centrelor ancorelor



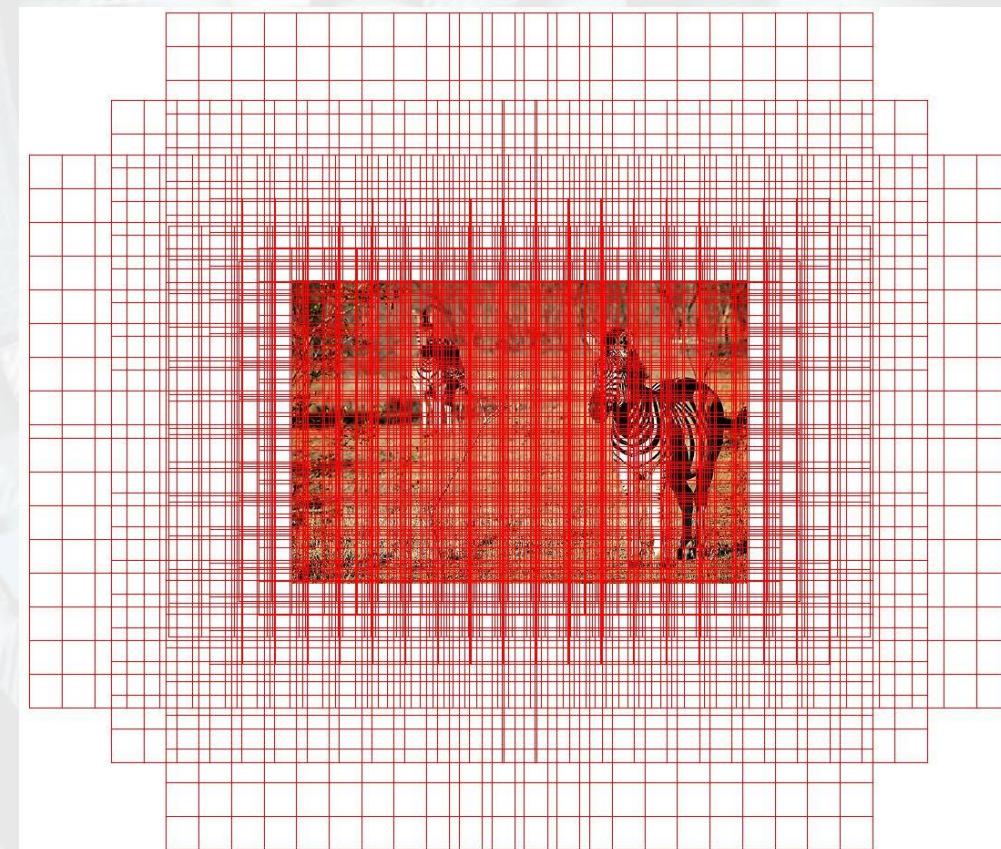
Detectia obiectelor – modele în 2 etape

Faster R-CNN – RPN (exemplu): ancorele pentru primul punct



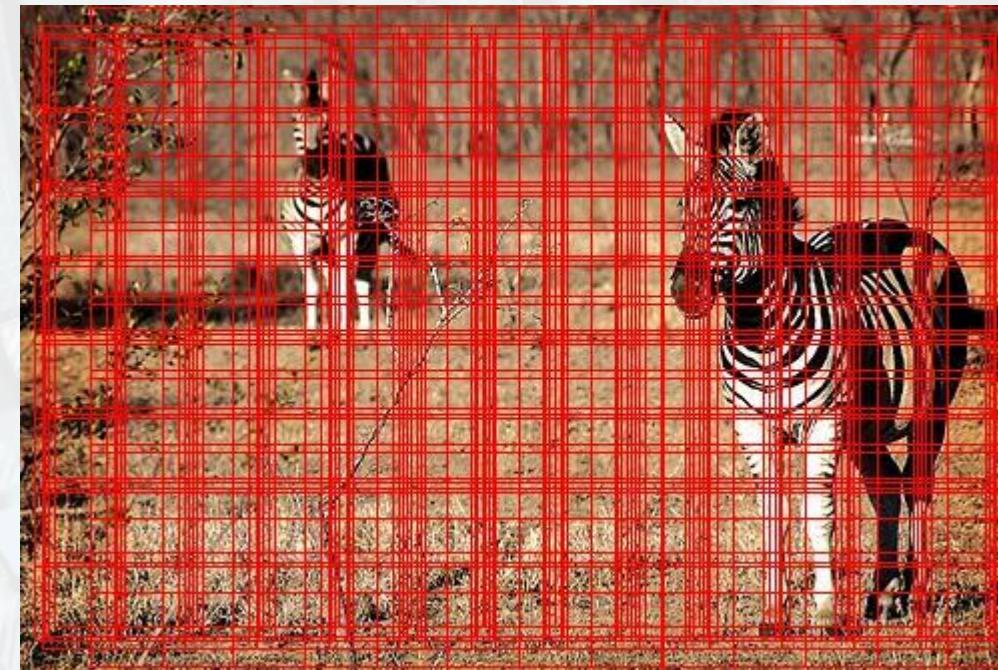
Detectia obiectelor – modele în 2 etape

Faster R-CNN – RPN (exemplu): ancorele pentru toate punctele



Detectia obiectelor – modele în 2 etape

Faster R-CNN – RPN (exemplu): eliminarea ancorelor din afara imaginii



Detectia obiectelor – modele în 2 etape

Faster R-CNN – rezultate PASCAL VOC 2007 test

method	# proposals	data	mAP (%)
SS	2000	07	66.9 [†]
SS	2000	07+12	70.0
RPN+VGG, unshared	300	07	68.5
RPN+VGG, shared	300	07	69.9
RPN+VGG, shared	300	07+12	73.2
RPN+VGG, shared	300	COCO+07+12	78.8

Detectia obiectelor – modele în 2 etape

Faster R-CNN – rezultate PASCAL VOC 2012 test

method	# proposals	data	mAP (%)
SS	2000	12	65.7
SS	2000	07++12	68.4
RPN+VGG, shared [†]	300	12	67.0
RPN+VGG, shared [‡]	300	07++12	70.4
RPN+VGG, shared [§]	300	COCO+07++12	75.9

Detectia obiectelor – modele în 2 etape

Faster R-CNN – rezultate MS-COCO (backbone: VGG16)

method	proposals	training data	COCO val		COCO test-dev	
			mAP@.5	mAP@[.5, .95]	mAP@.5	mAP@[.5, .95]
Fast R-CNN [2]	SS, 2000	COCO train	-	-	35.9	19.7
Fast R-CNN [impl. in this paper]	SS, 2000	COCO train	38.6	18.9	39.3	19.3
Faster R-CNN	RPN, 300	COCO train	41.5	21.2	42.1	21.5
Faster R-CNN	RPN, 300	COCO trainval	-	-	42.7	21.9

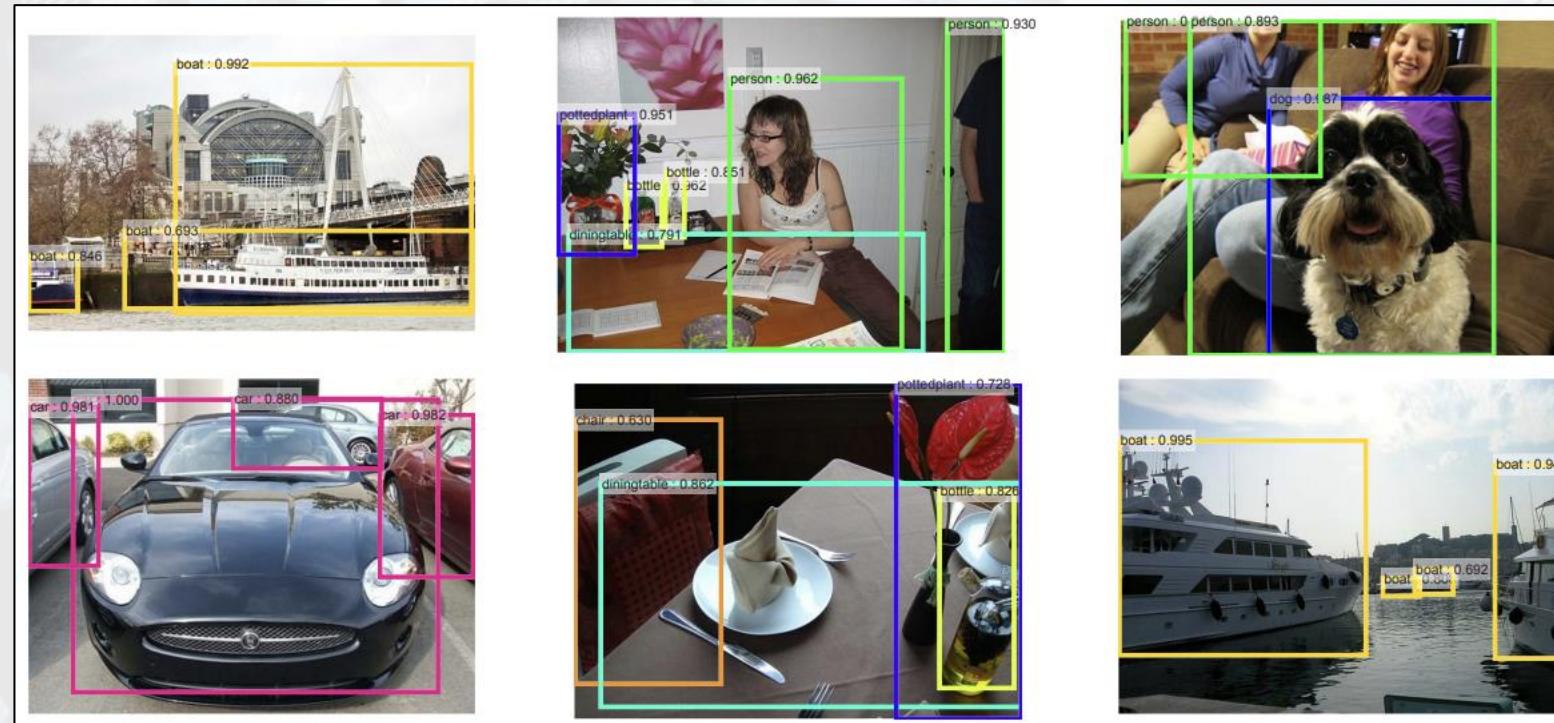
Detectia obiectelor – modele în 2 etape

Faster R-CNN – viteză (ms)

model	system	conv	proposal	region-wise	total	rate
VGG	SS + Fast R-CNN	146	1510	174	1830	0.5 fps
VGG	RPN + Fast R-CNN	141	10	47	198	5 fps
ZF	RPN + Fast R-CNN	31	3	25	59	17 fps

Detectia obiectelor – modele în 2 etape

Faster R-CNN – exemple



Detectia obiectelor – modele în 2 etape

Faster R-CNN – concluzii

- Introduce Region Proposal Network (RPN) aproape „cost-free”. Antrenarea se poate face end-to-end;
- Arhitectura backbone are un impact major asupra rezultatului;
- Antrenare/testare pe MS-COCO;
- Inferență [near] real-time;
- Marchează un punct de referință în dezvoltarea detectoarelor de obiecte.

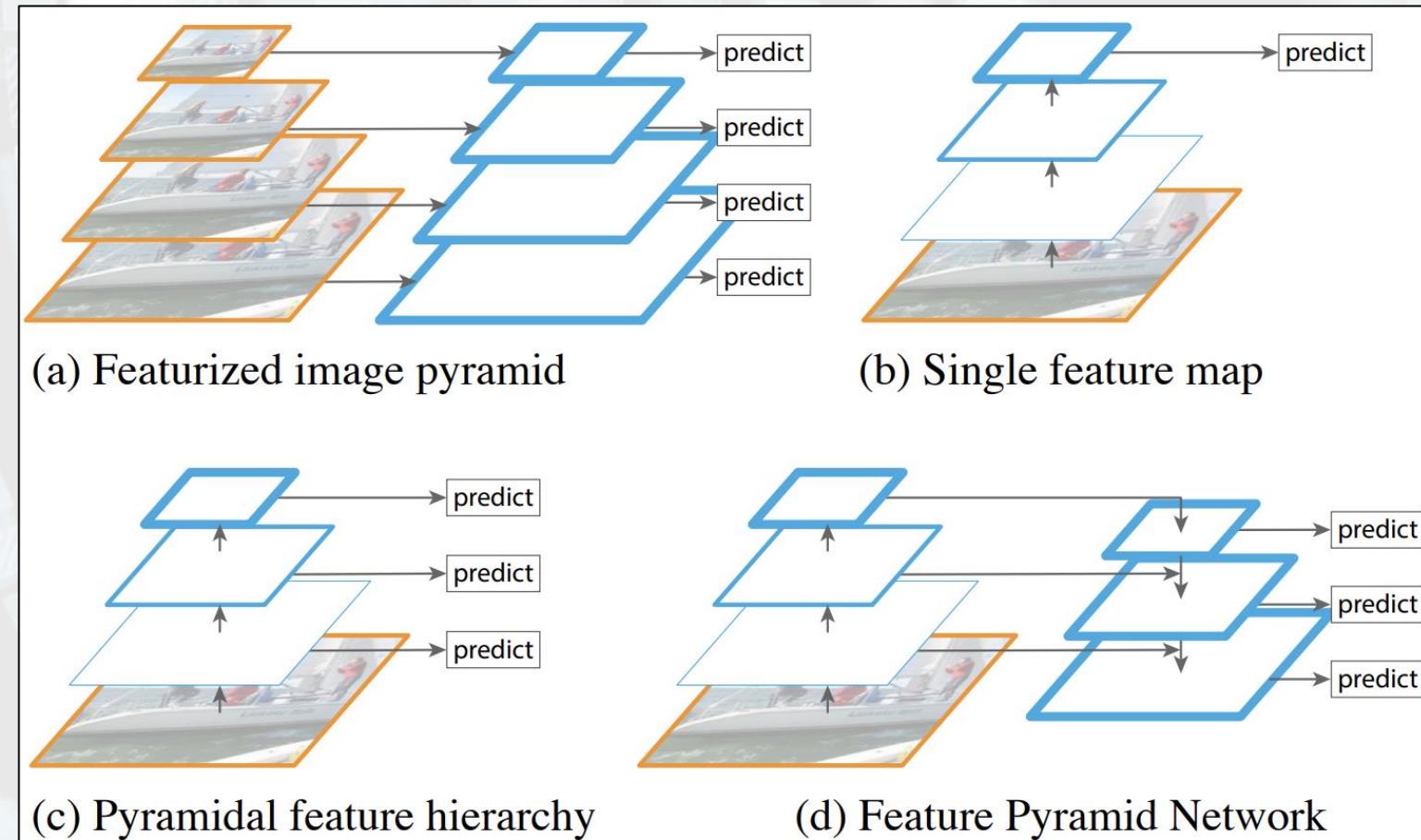
Detectia obiectelor – modele în 2 etape

Feature Pyramid Networks – FPN [34]

- Detectia obiectelor la scale diferite este încă problematică (obiecte foarte mici sau foarte mari);
- **Scop:** găsirea unei scale adecvate pentru reprezentarea obiectelor de dimensiuni diferite;
- Reprezentarea obiectelor la scale diferite este operație costisitoare – memorie și procesare;
- Se inspiră din structura piramidală a hărților de trăsături – conoluțiile creează piramide de trăsături în mod nativ;
- Se poate aplica în orice rețea convețională (ConvNet).

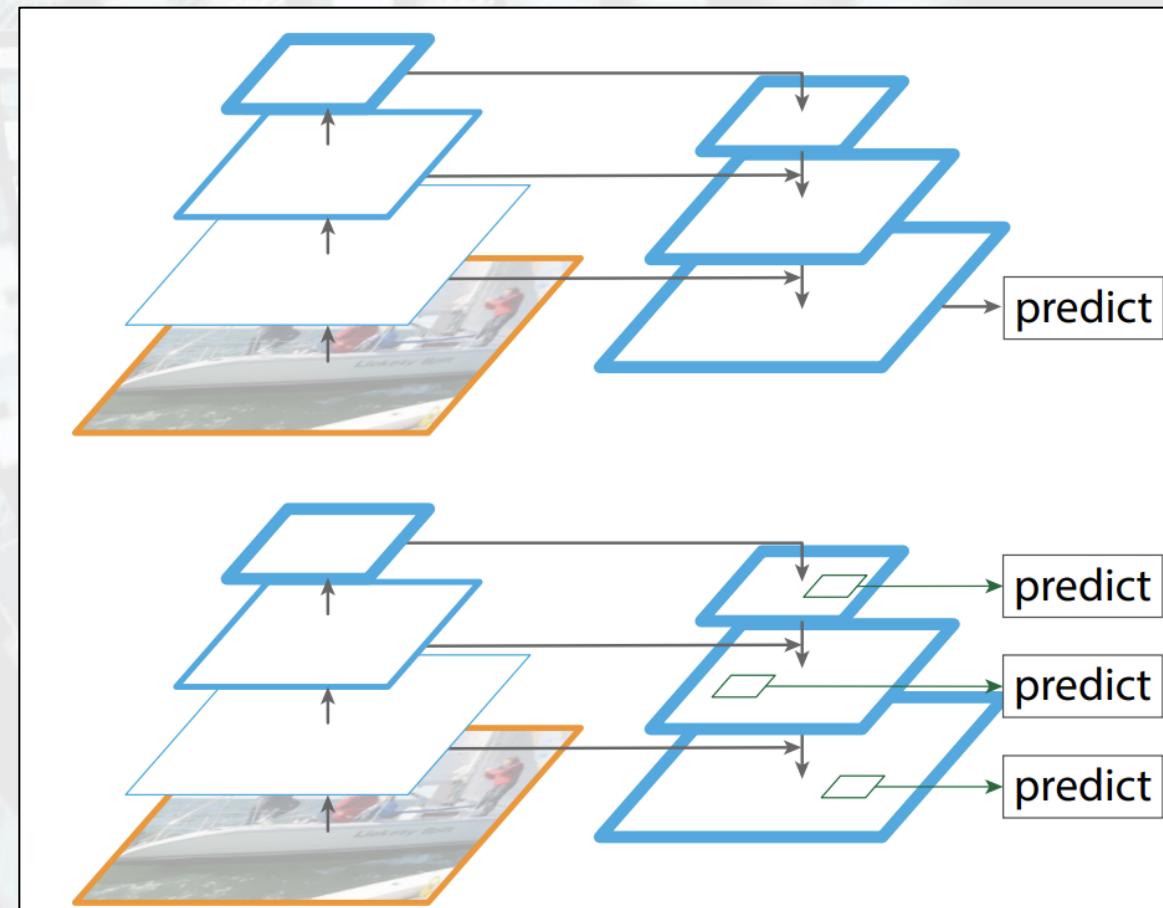
Detectia obiectelor – modele în 2 etape

FPN



Detectia obiectelor – modele în 2 etape

FPN



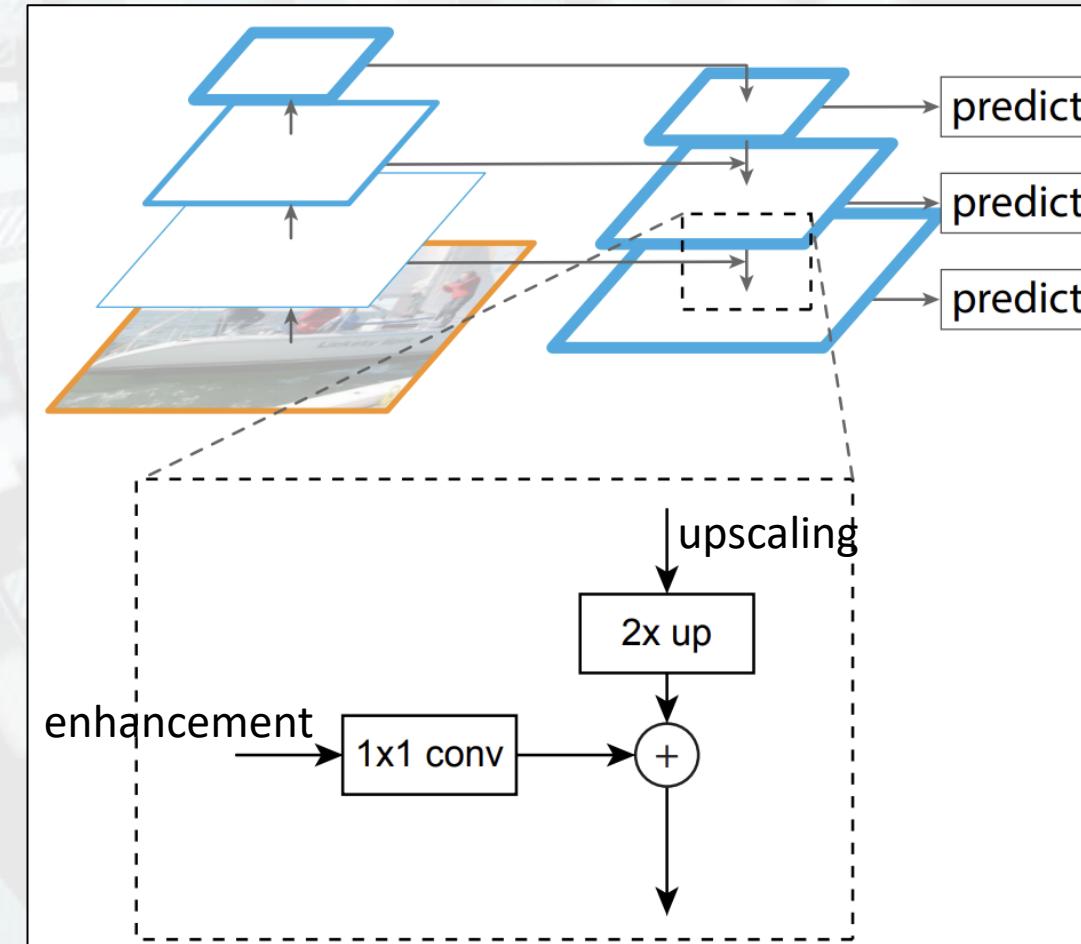
Metodă existentă:
Predictii pe cel mai „fin” nivel

Metodă nouă - FPN:
Predictii pe fiecare nivel

Detectia obiectelor – modele în 2 etape

FPN

Bottom-up: convoluții cu pași din ce în ce mai mari (e.g. straturi diferite din ResNet)



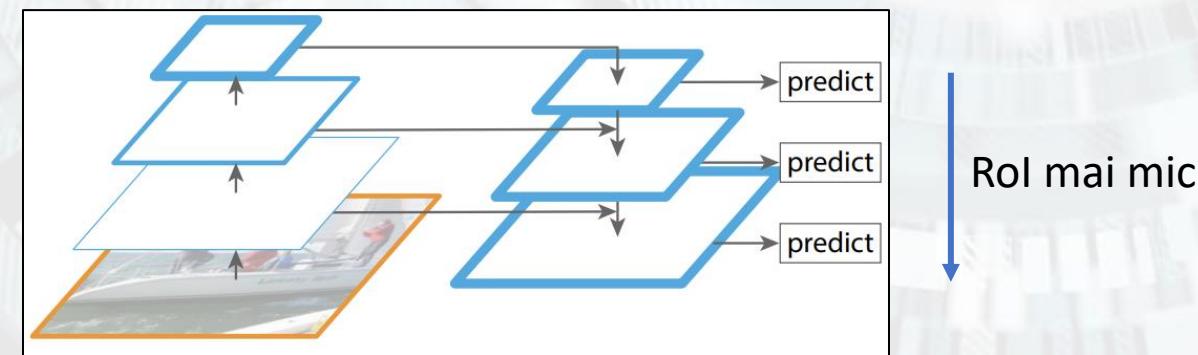
Top-down: scalare de la straturile superioare (nearest neighbor) și rafinare de la straturile laterale

Detectia obiectelor – modele în 2 etape

FPN – nivelul Fast R-CNN

➤ Pentru un Rol de dimensiuni w și h relativ la imaginea de intrare, nivelul din piramida de trăsături din care se va alege descriptorul asociat Rol-ului este:

$$k = \lfloor k_0 + \log_2(\sqrt{wh}/224) \rfloor$$



Detectia obiectelor – modele în 2 etape

FPN – rezultate (testare pe COCO mini-val)

RPN	feature	# anchors	lateral?	top-down?	AR ¹⁰⁰	AR ^{1k}	AR ^{1k} _s	AR ^{1k} _m	AR ^{1k} _l
(a) baseline on conv4	C_4	47k			36.1	48.3	32.0	58.7	62.2
(b) baseline on conv5	C_5	12k			36.3	44.9	25.3	55.5	64.2
(c) FPN	$\{P_k\}$	200k	✓	✓	44.0	56.3	44.9	63.4	66.2

Fast R-CNN	proposals	feature	head	lateral?	top-down?	AP@0.5	AP	AP _s	AP _m	AP _l
(a) baseline on conv4	RPN, $\{P_k\}$	C_4	conv5			54.7	31.9	15.7	36.5	45.5
(b) baseline on conv5	RPN, $\{P_k\}$	C_5	2fc			52.9	28.8	11.9	32.4	43.4
(c) FPN	RPN, $\{P_k\}$	$\{P_k\}$	2fc	✓	✓	56.9	33.9	17.8	37.7	45.8

Faster R-CNN	proposals	feature	head	lateral?	top-down?	AP@0.5	AP	AP _s	AP _m	AP _l
(*) baseline from He <i>et al.</i> [16] [†]	RPN, C_4	C_4	conv5			47.3	26.3	-	-	-
(a) baseline on conv4	RPN, C_4	C_4	conv5			53.1	31.6	13.2	35.6	47.1
(b) baseline on conv5	RPN, C_5	C_5	2fc			51.7	28.0	9.6	31.9	43.1
(c) FPN	RPN, $\{P_k\}$	$\{P_k\}$	2fc	✓	✓	56.9	33.9	17.8	37.7	45.8

Detectia obiectelor – modele în 2 etape

FPN – concluzii

- Îmbunătățește capacitatea de a selecta un descriptor adecvat dimensiunii obiectului;
- Se poate aplica în orice rețea convezională stratificată (ce oferă o piramidă de trăsături);
- Cost de implementare insignifiant;
- Un nou state-of-the-art.

Detectia obiectelor – modele în 2 etape

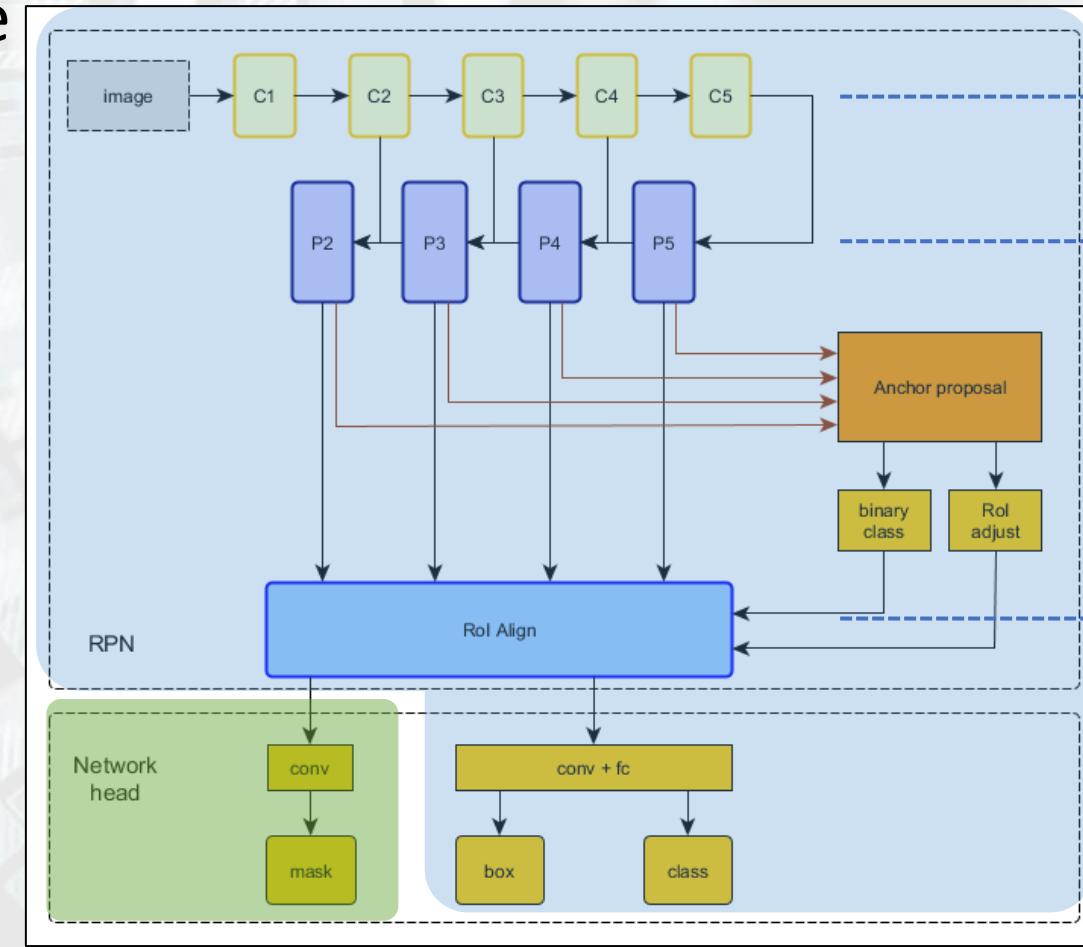
Mask R-CNN [35]

- Extinde Faster R-CNN cu o ramură care generează măști pentru obiecte => instance segmentation;
- Mask R-CNN = Faster R-CNN + segmentare măști;
- Introduce o variantă îmbunătățită a propagării Rol-urilor;
- Testează noi arhitecturi backbone (inclusiv FPN) și head.

Detectia obiectelor – modele în 2 etape

Mask R-CNN - pipeline

Mască de segmentare



Extragere trăsături

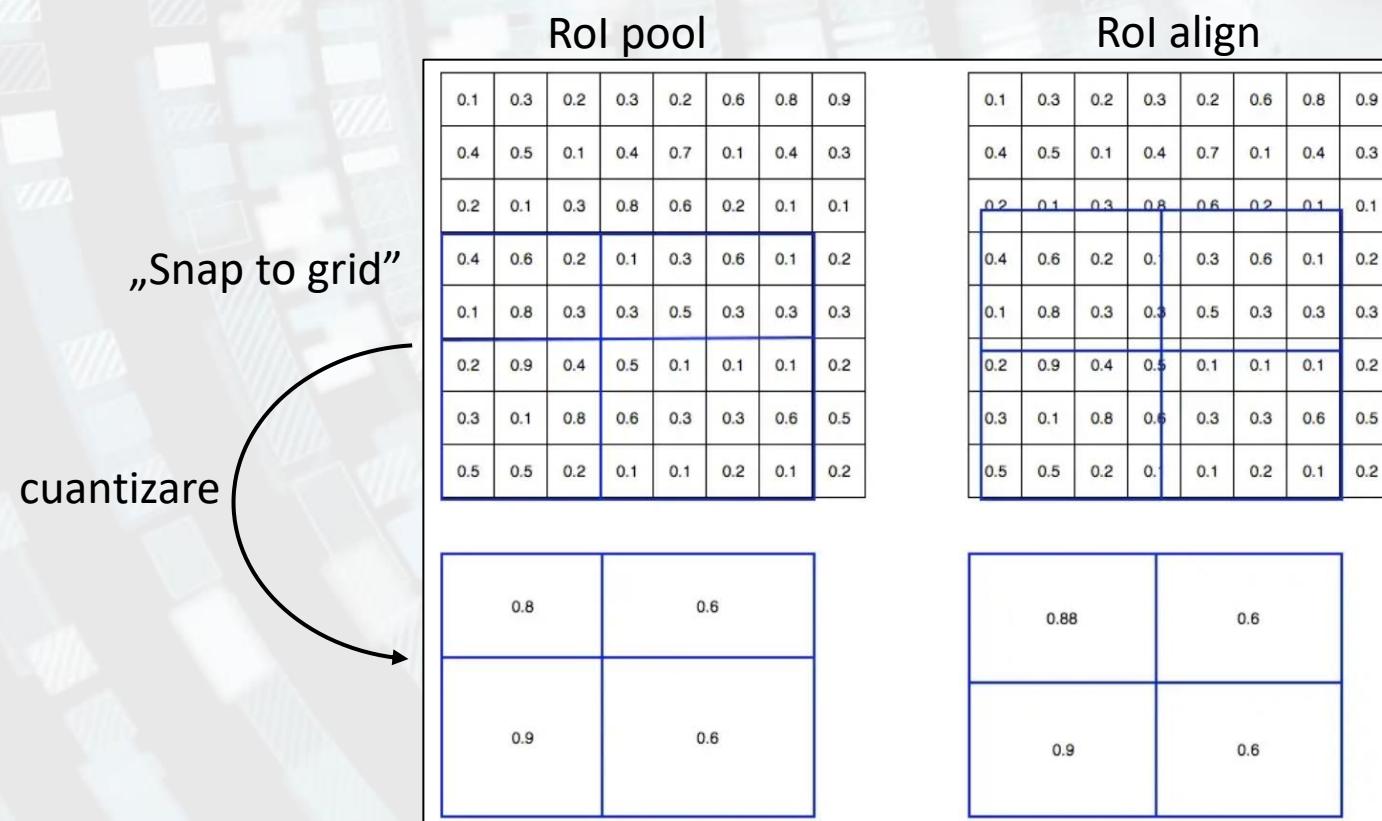
FPN – reprezentarea multi-scală a trăsăturilor

RoI Align – alegerea celui mai bun RoI

Faster R-CNN

Detectia obiectelor – modele în 2 etape

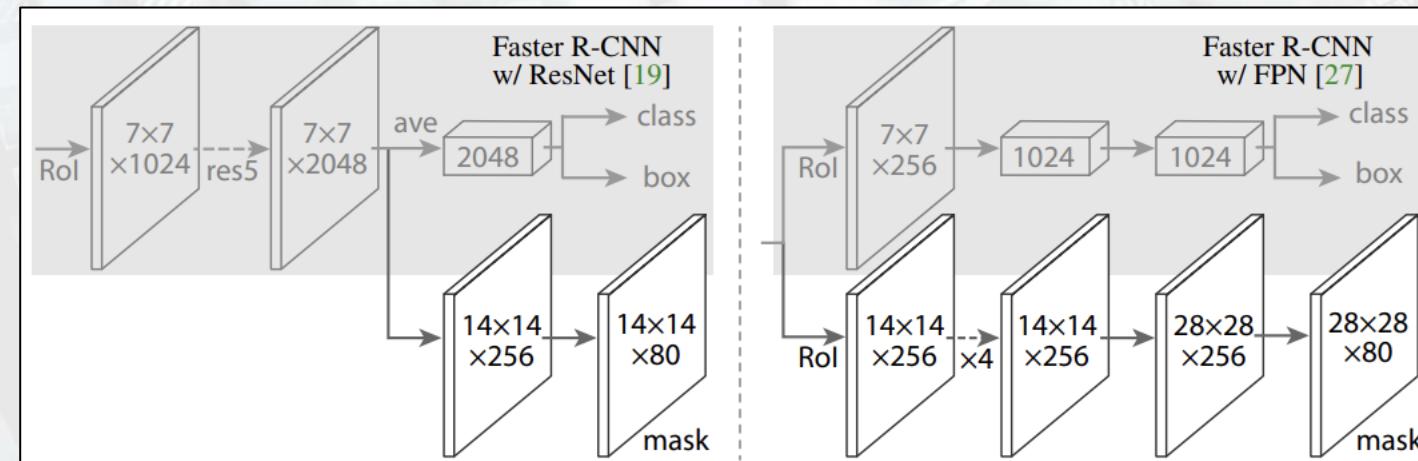
Mask R-CNN – ROI Align (ROI pooling++)



Nicio aproximare sau cuantizare
Se folosește interpolarea biliniară

Detectia obiectelor – modele în 2 etape

Mask R-CNN – network head(s)



Detectia obiectelor – modele în 2 etape

Mask R-CNN – generarea măștii

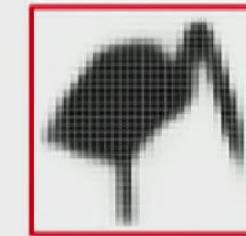
- Pixel-to-pixel aligned



RoI



28x28 FCN prediction



resized soft prediction



final mask



Detectia obiectelor – modele în 2 etape

Mask R-CNN – rezultate (testare pe COCO mini-val)

	backbone	AP ^{bb}	AP ^{bb} ₅₀	AP ^{bb} ₇₅	AP ^{bb} _S	AP ^{bb} _M	AP ^{bb} _L
Faster R-CNN+++ [19]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [27]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [21]	Inception-ResNet-v2 [41]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [39]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	52.1
Faster R-CNN, RoIAlign	ResNet-101-FPN	37.3	59.6	40.3	19.8	40.2	48.8
Mask R-CNN	ResNet-101-FPN	38.2	60.3	41.7	20.1	41.1	50.2
Mask R-CNN	ResNeXt-101-FPN	39.8	62.3	43.4	22.1	43.2	51.2

Average Precision (AP):

AP % AP at IoU=.50:.05:.95 (primary challenge metric)
AP^{IoU=.50} % AP at IoU=.50 (PASCAL VOC metric)
AP^{IoU=.75} % AP at IoU=.75 (strict metric)

AP Across Scales:

Ap_{small} % AP for small objects: area < 32²
Ap_{medium} % AP for medium objects: 32² < area < 96²
Ap_{large} % AP for large objects: area > 96²

Detectia obiectelor – modele în 2 etape

Mask R-CNN – rezultate



Detectia obiectelor – modele în 2 etape

Mask R-CNN – concluzii

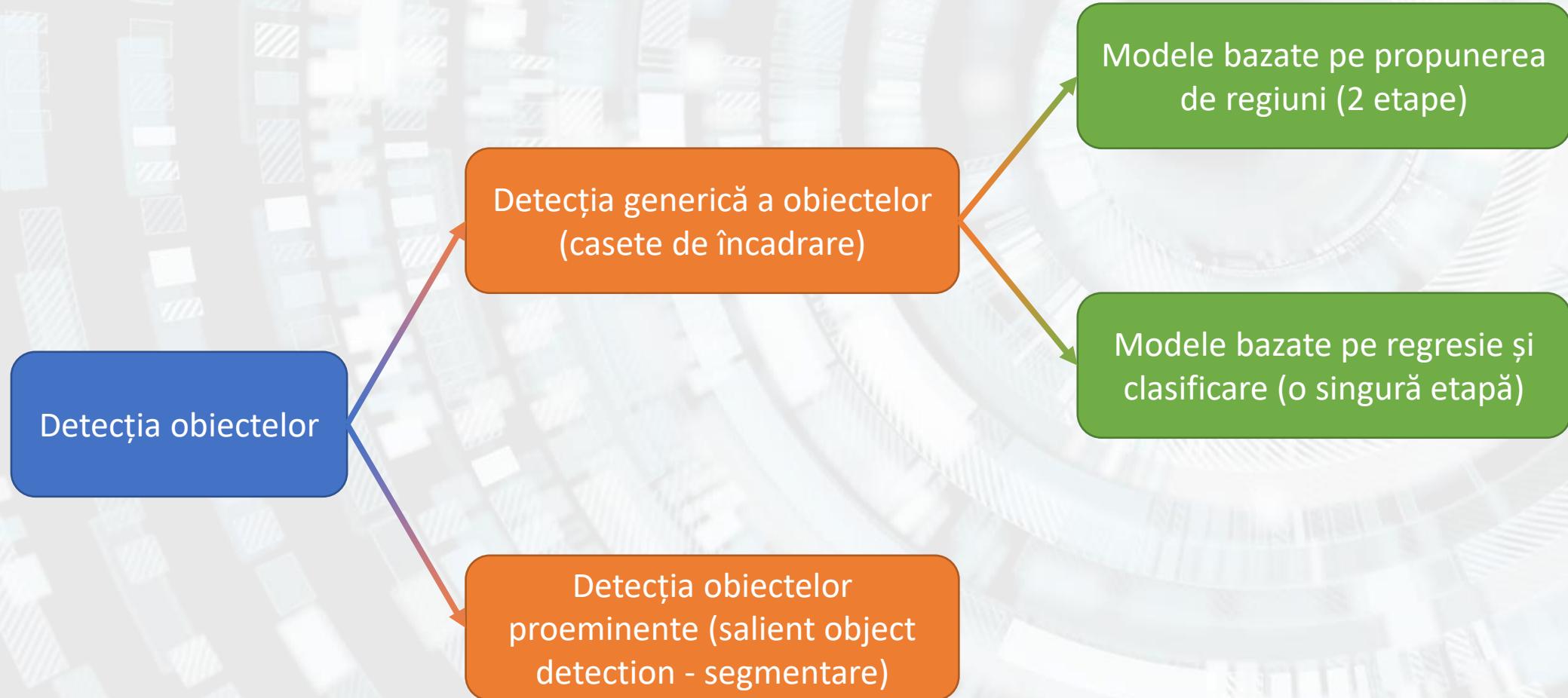
- Introduce segmentarea instanțelor cu o simplă rețea convezională în continuarea RPN;
- Rezolvă problema nelinierii Rol-urilor prin mecanismul Rol align;
- Obține rezultate state-of-the-art pe detectia obiectelor;
- Modelul cu backbone ResNet-101-FPN obține inferență în 0.2 s/img.

Detectia obiectelor – modele în 2 etape (concluzii generale)

- Se face în 2 etape principale: propunerea de regiuni (network backbone) și clasificarea (network head);
- Utilizarea **în comun** a **hărților de trăsături** din backbone reduce semnificativ durata procesării.
- Utilizarea unei **rețele pentru RPN** reduce semnificativ durata procesării;
- Arhitecturi mai recente: Cascade Mask R-CNN, DetectoRS, Swin Transformer, etc.

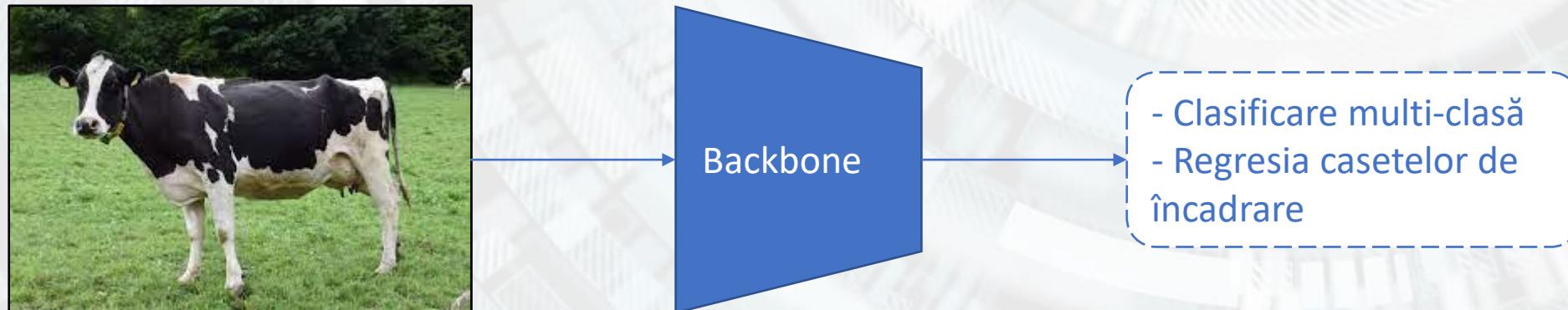


Detectia obiectelor – modele



Detectia obiectelor – modele într-o etapă

- Modelele într-o singură etapă presupun procesarea imaginilor cu o rețea densă de eșantionare și regresia casetelor de încadrare cu o singură parcurgere a rețelei. Spre deosebire de detectoarele în 2 etape, nu mai este nevoie de o componentă de propunere de regiuni.



Detectia obiectelor – modele într-o etapă

You Only Look Once (YOLO) [36] = Faster R-CNN simplificat

- O nouă abordare a detectiei de obiecte – nu se mai folosește propunerea de regiuni;
- A fost propus (cronologic) după Faster R-CNN;
- Reprezintă punctul de plecare pentru detectoarele într-o singură etapă.

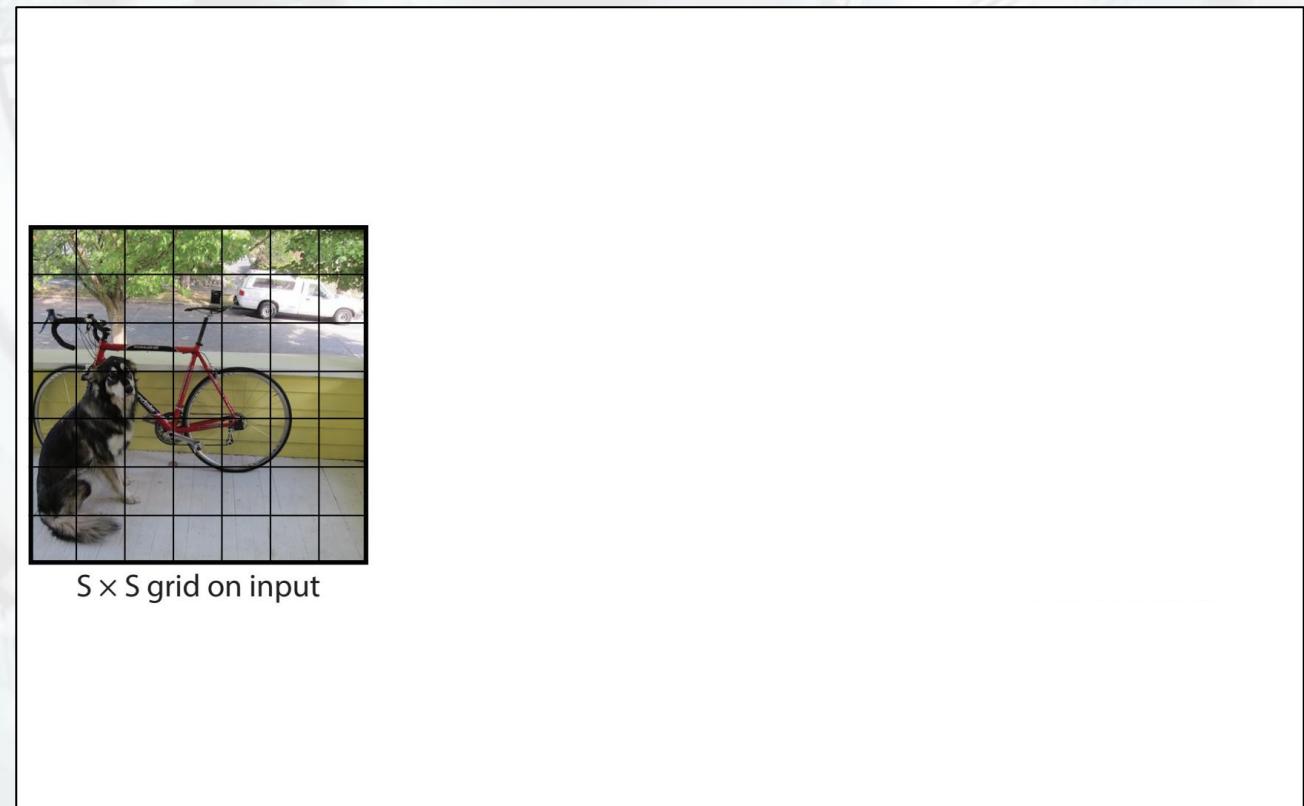
Detectia obiectelor – modele într-o etapă

YOLO

Pași:

1. Se împarte imaginea într-o rețea de $S \times S$ celule;
2. Fiecare celulă prezice B casete de încadrare (ancore) și scorurile lor de detectie a obiectului. În același timp, se prezic și probabilitățile de clasă pentru fiecare celulă;
3. Se aplică un prag de detectie și Non-Maximal Suppression.

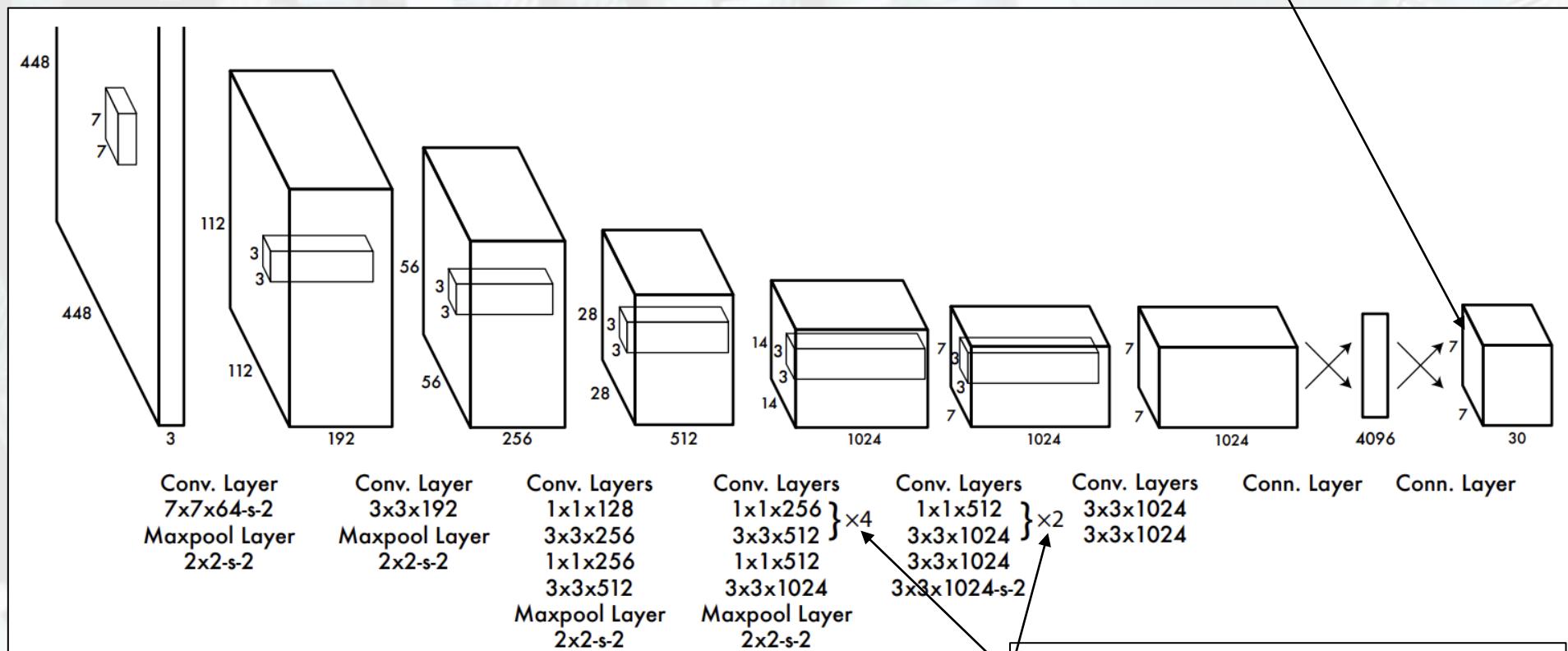
$$\begin{aligned}Scor &= \Pr(Obiect) * IoU_{pred}^{gt} * \Pr(Clasa_i | Obiect) \\&= \Pr(Clasa_i) * IoU_{pred}^{gt}\end{aligned}$$



Detectia obiectelor – modele într-o etapă

YOLO

$$S \times S \times (B * 5 + C) = \\ 7 \times 7 \times (2 * 5 + 20)$$



Detectia obiectelor – modele într-o etapă

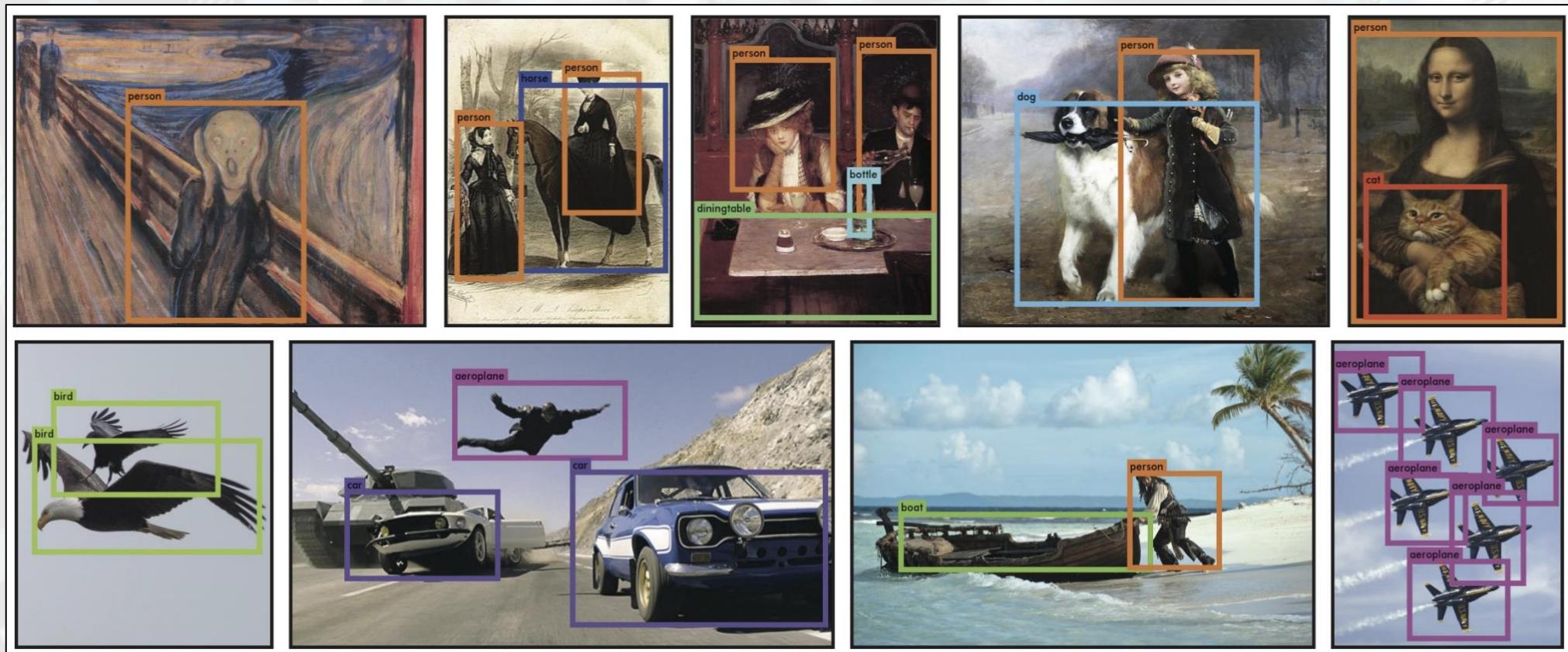
YOLO – rezultate pe Pascal VOC 07

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	155
YOLO	2007+2012	63.4	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

	mAP	Combined	Gain
Fast R-CNN	71.8	-	-
Fast R-CNN (2007 data)	66.9	72.4	.6
Fast R-CNN (VGG-M)	59.2	72.4	.6
Fast R-CNN (CaffeNet)	57.1	72.1	.3
YOLO	63.4	75.0	3.2

Detectia obiectelor – modele într-o etapă

YOLO – exemple



Detectia obiectelor – modele într-o etapă

YOLO – concluzii

- Reprezintă un sistem de referință printre modelele într-o singură etapă;
- Este deosebit de rapid;
- Este comparabil cu sistemele în 2 etape ca performanță;
- Nu funcționează bine pentru obiecte mici sau aglomerate;
- A fost îmbunătățit în multe versiuni: YOLOv2, ..., YOLOv7, YOLOX etc.

Detectia obiectelor – modele într-o etapă

YOLO – bonus

Joseph Redmon: <https://pjreddie.com/>

home darknet coq tactics publications projects r  sum  

Joseph Chet Redmon

Welcome to my website!

I am a graduate student advised by Ali Farhadi. I work on computer vision.

I maintain the Darknet Neural Network Framework, a primer on tactics in Coq, occasionally work on research, and try to stay off twitter.

Outside of computer science, I enjoy skiing, hiking, rock climbing, and playing with my Alaskan malamute puppy, Kelp.

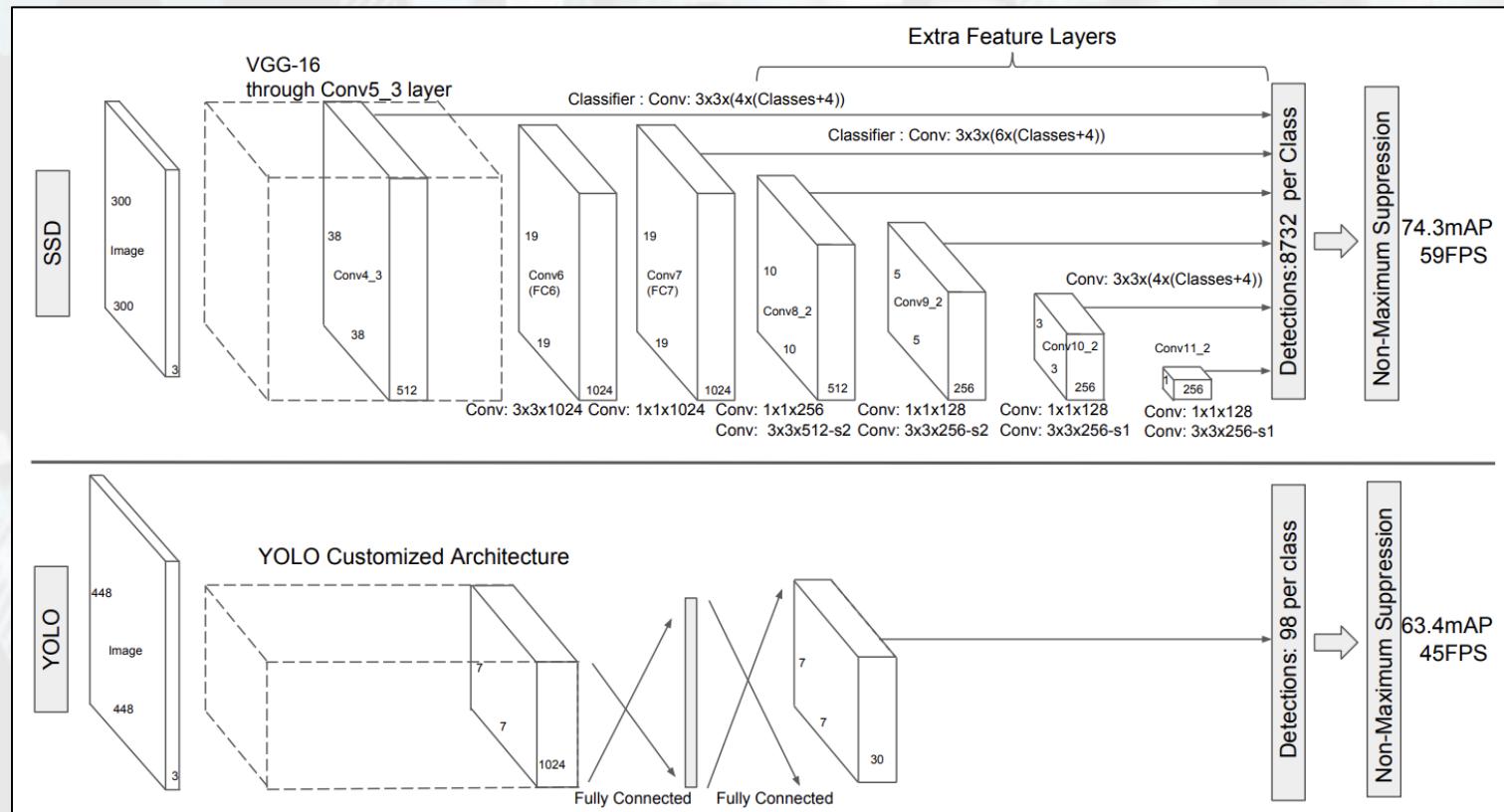
Talks and Teaching

The Ancient Secrets of Computer Vision - University of Washington, Spring 2018



Detectia obiectelor – modele într-o etapă

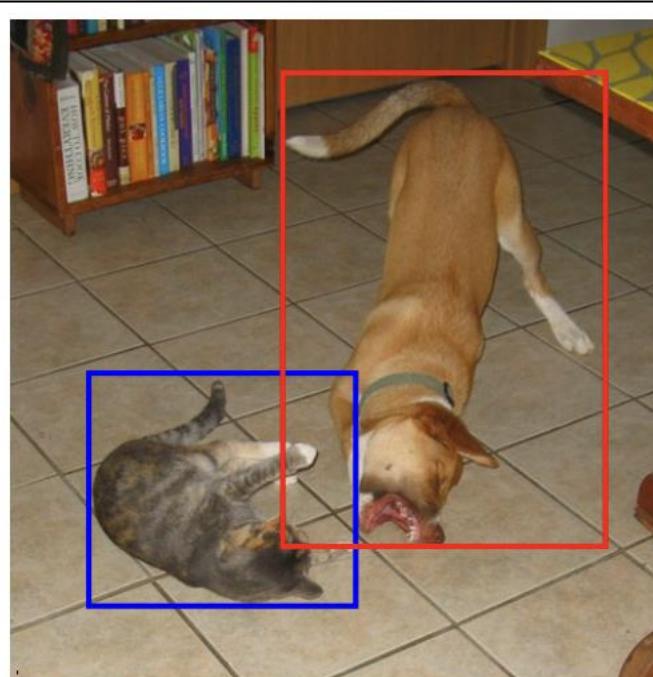
Single Shot MultiBox Detector (SSD) [37]



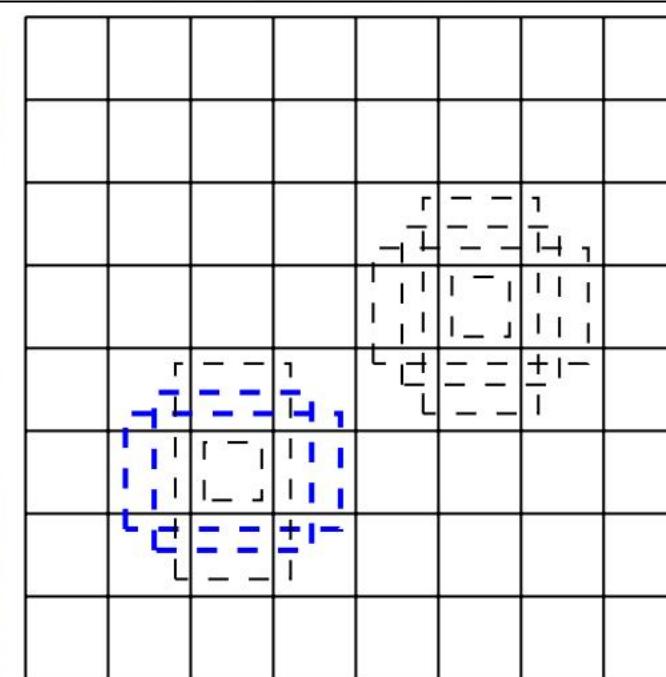
- Procesare asemănătoare cu YOLO, dar pe diferite nivele ale hărții de trăsături;
- Folosește 6 ancore/nivel cu raporturi de aspect diferențiate;
- Dimensiunea ancorei depinde de nivelul hărții de trăsături;
- Reușește să detecteze obiecte de dimensiuni mici;
- Poate utiliza diferite arhitecturi backbone.

Detectia obiectelor – modele într-o etapă

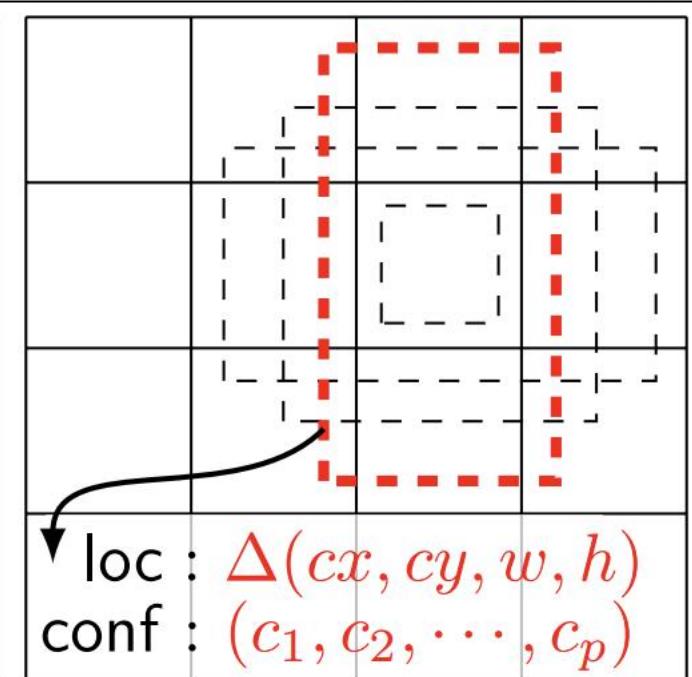
SSD – ancore și output



(a) Image with GT boxes



(b) 8×8 feature map



loc : $\Delta(cx, cy, w, h)$
conf : (c_1, c_2, \dots, c_p)

(c) 4×4 feature map

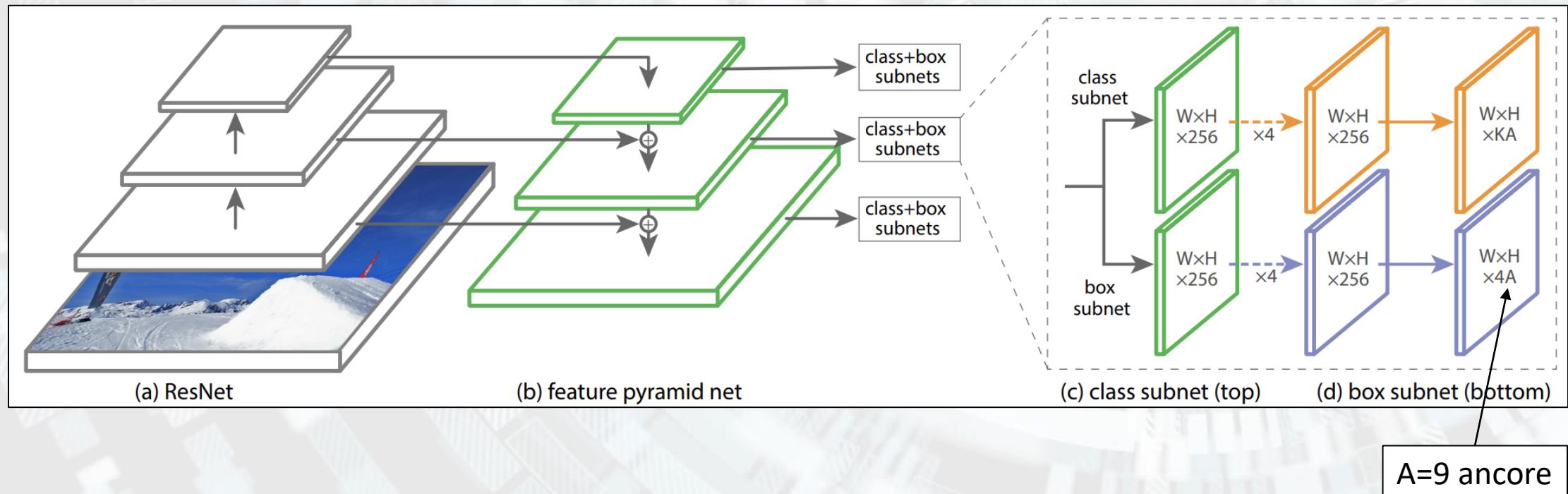
Detectia obiectelor – modele într-o etapă

SSD – rezultate pe MS-COCO

Method	data	Avg. Precision, IoU:			Avg. Precision, Area:			Avg. Recall, #Dets:			Avg. Recall, Area:		
		0.5:0.95	0.5	0.75	S	M	L	1	10	100	S	M	L
Fast [6]	train	19.7	35.9	-	-	-	-	-	-	-	-	-	-
Fast [24]	train	20.5	39.9	19.4	4.1	20.0	35.8	21.3	29.5	30.1	7.3	32.1	52.0
Faster [2]	trainval	21.9	42.7	-	-	-	-	-	-	-	-	-	-
ION [24]	train	23.6	43.2	23.6	6.4	24.1	38.3	23.2	32.7	33.5	10.1	37.7	53.6
Faster [25]	trainval	24.2	45.3	23.5	7.7	26.4	37.1	23.8	34.0	34.6	12.0	38.5	54.4
SSD300	trainval35k	23.2	41.2	23.4	5.3	23.2	39.6	22.5	33.2	35.3	9.6	37.6	56.5
SSD512	trainval35k	26.8	46.5	27.8	9.0	28.9	41.9	24.8	37.5	39.8	14.0	43.5	59.0

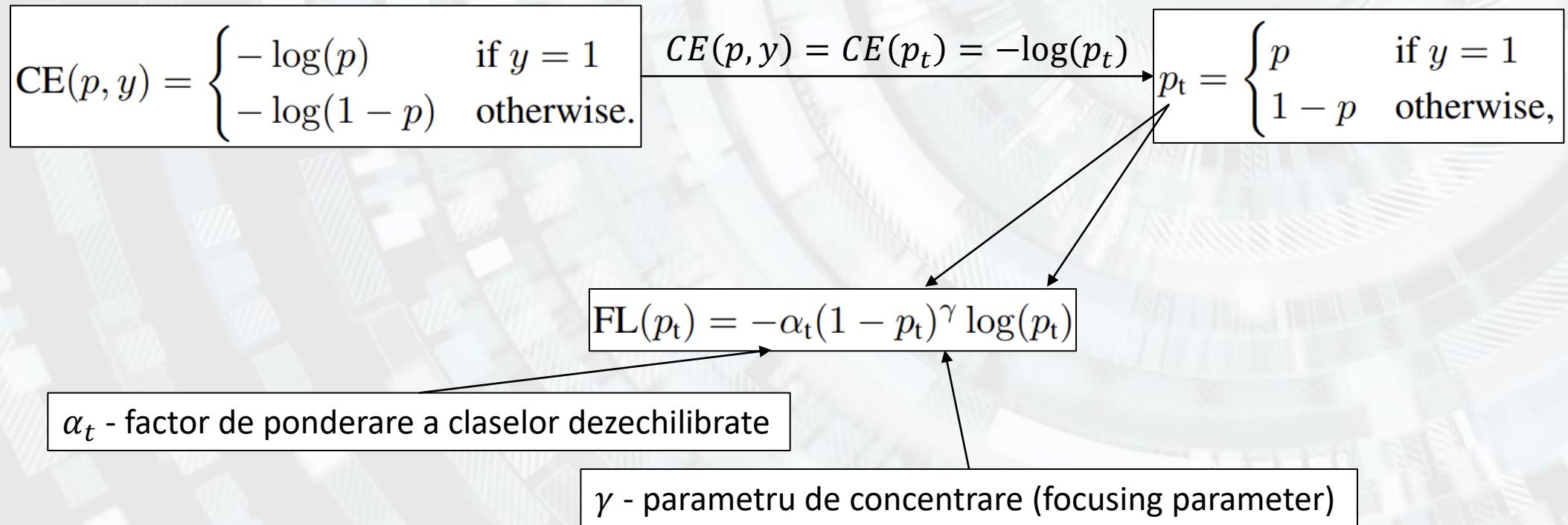
Detectia obiectelor – modele într-o etapă

RetinaNet [38] = (Faster R-CNN + FPN) simplificat



Detectia obiectelor – modele într-o etapă

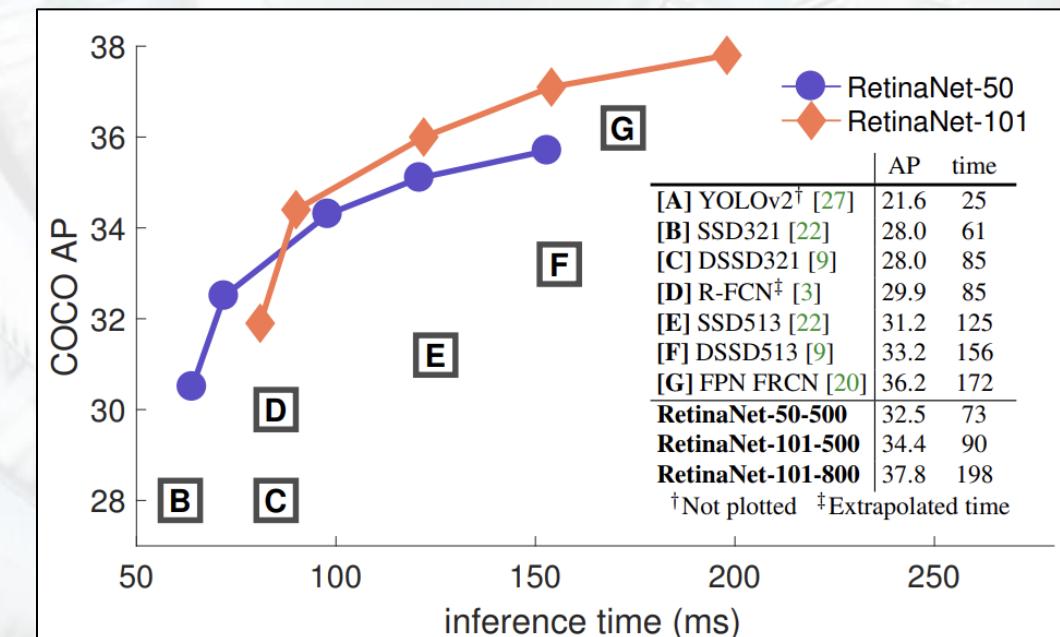
RetinaNet – pierderea focală (focal loss)



Detectia obiectelor – modele într-o etapă

RetinaNet – rezultate pe MSCOCO

depth	scale	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L	time
50	400	30.5	47.8	32.7	11.2	33.8	46.1	64
50	500	32.5	50.9	34.8	13.9	35.8	46.7	72
50	600	34.3	53.2	36.9	16.2	37.4	47.4	98
50	700	35.1	54.2	37.7	18.0	39.3	46.4	121
50	800	35.7	55.0	38.5	18.9	38.9	46.3	153
101	400	31.9	49.5	34.1	11.6	35.8	48.5	81
101	500	34.4	53.1	36.8	14.7	38.5	49.1	90
101	600	36.0	55.2	38.7	17.4	39.6	49.7	122
101	700	37.1	56.6	39.8	19.1	40.6	49.4	154
101	800	37.8	57.5	40.8	20.2	41.1	49.2	198



Detectia obiectelor – modele într-o etapă

RetinaNet – rezultate pe MSCOCO

	backbone	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
<i>Two-stage methods</i>							
Faster R-CNN+++ [16]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [20]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [17]	Inception-ResNet-v2 [34]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [32]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	52.1
<i>One-stage methods</i>							
YOLOv2 [27]	DarkNet-19 [27]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [22, 9]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [9]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet (ours)	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet (ours)	ResNeXt-101-FPN	40.8	61.1	44.1	24.1	44.2	51.2

Detectia obiectelor – modele în 2 etape (concluzii generale)

- Integrează în aceeași etapă propunerea de regiuni și clasificarea lor;
- Utilizează, în general, conceptul ancorelor;
- Sunt concepute pe baza detectoarelor în 2 etape;
- Sunt mai rapide decât detectoarele în 2 etape, dar mai puțin precise;
- Arhitecturi mai recente: CenterNet, EfficientDet, YOLOX, Swin Transformer, etc.

Detectia obiectelor

R-CNN: Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).

SPP-Net: He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9), 1904-1916.

Fast R-CNN: Girshick, Ross. "Fast r-cnn." In Proceedings of the IEEE international conference on computer vision, pp. 1440-1448. 2015.

Faster R-CNN: Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

FPN: Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2117-2125).

Mask R-CNN: He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

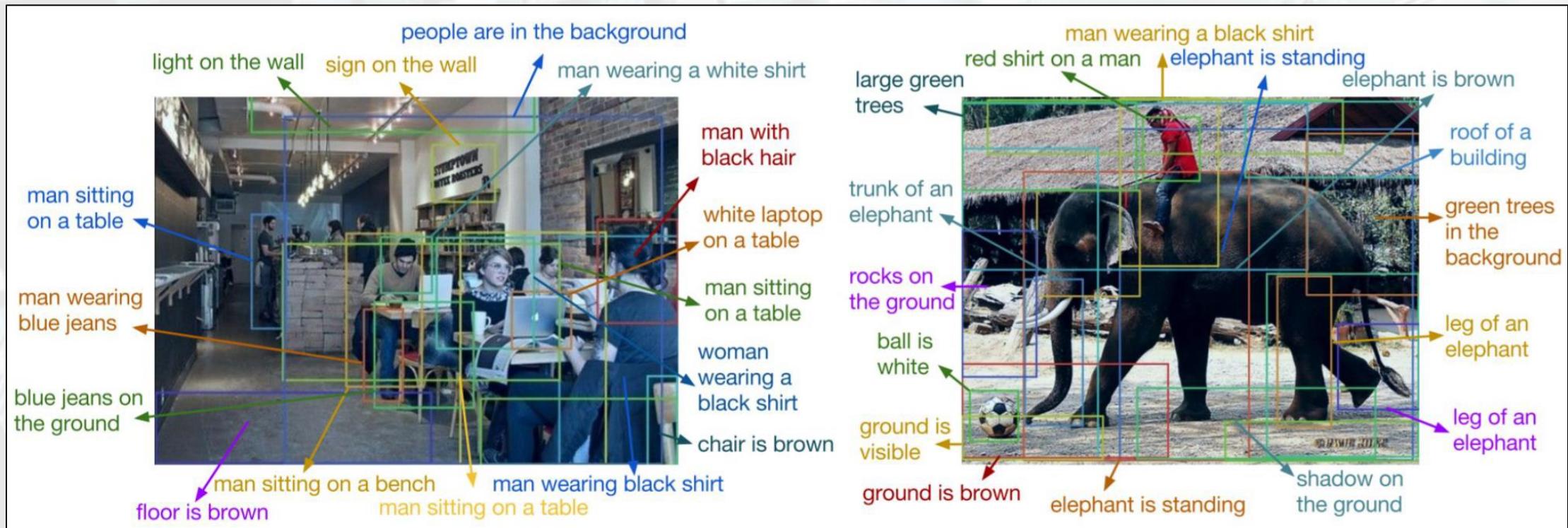
YOLO: Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

SSD: Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In European conference on computer vision (pp. 21-37). Springer, Cham.

RetinaNet: Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision (pp. 2980-2988).

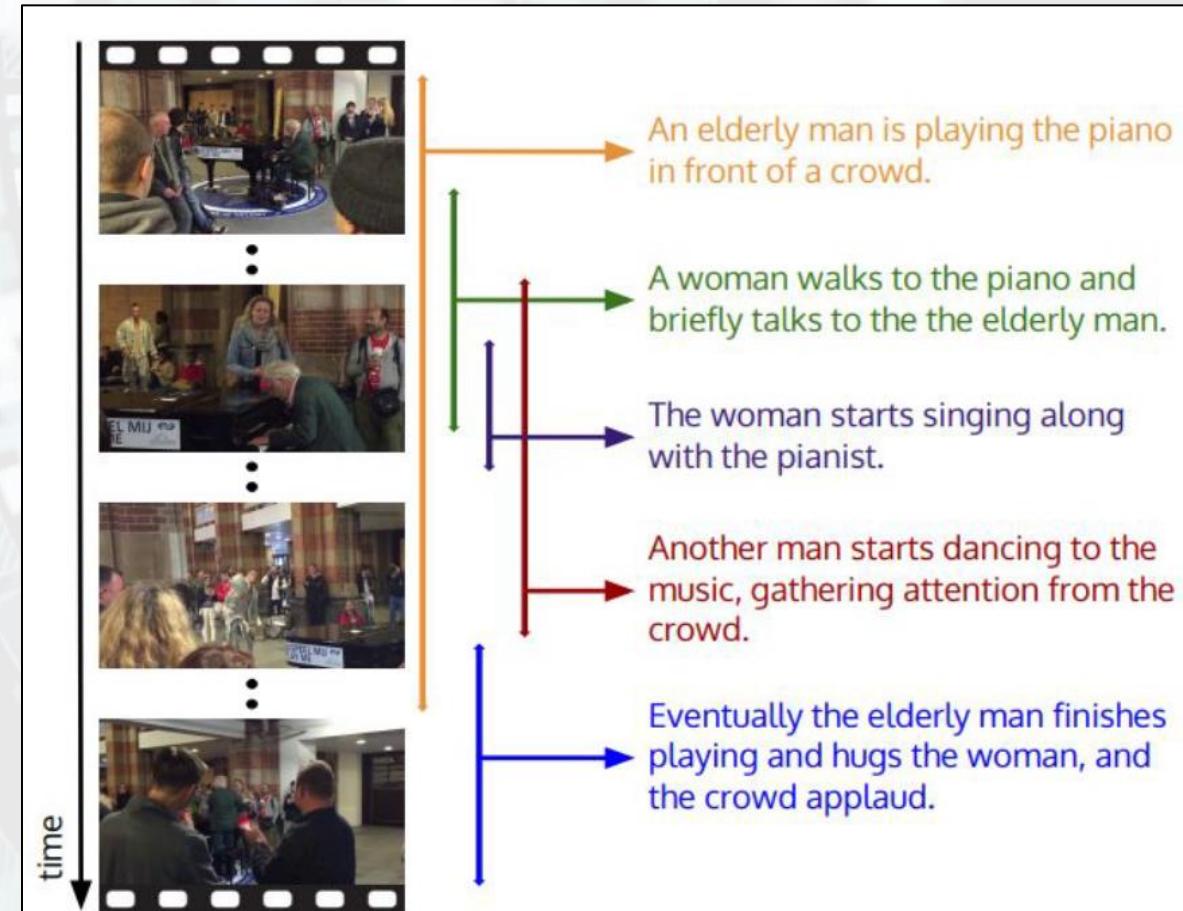
Aplicații ale detecției de obiecte

Object Detection + Captioning = Dense Captioning



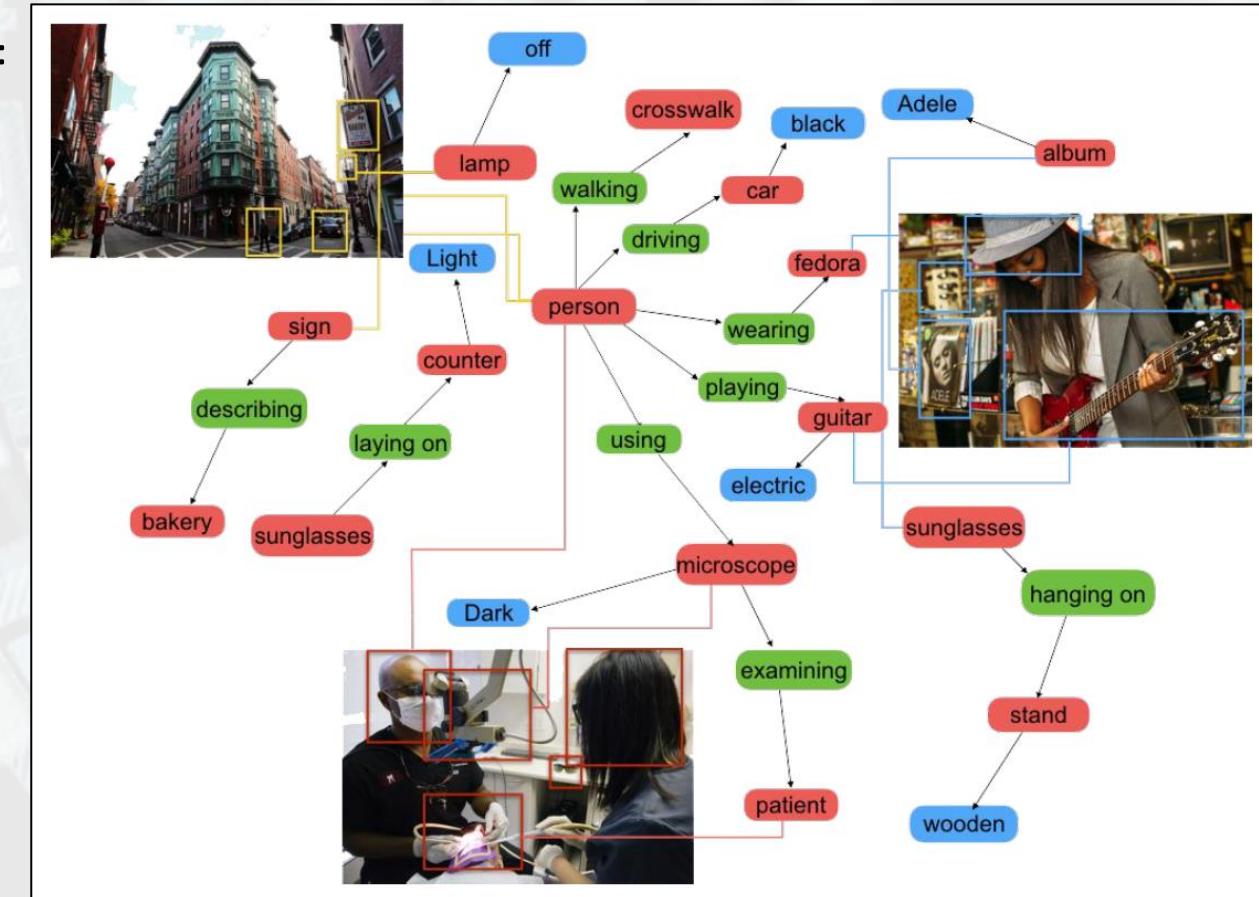
Aplicații ale detecției de obiecte

Dense Video Captioning



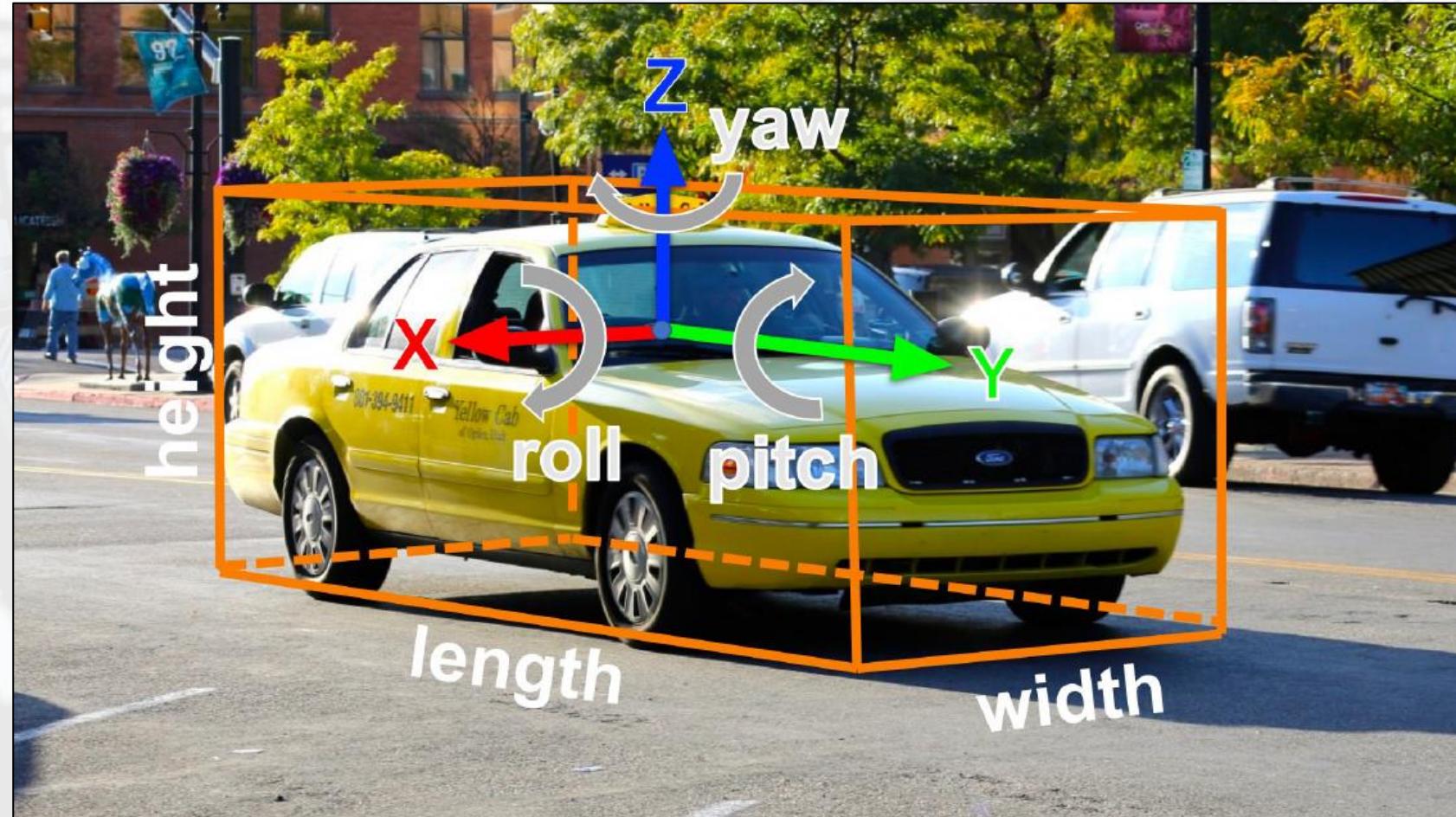
Aplicații ale detecției de obiecte

Obiecte + Relații =
Grafuri Scenice



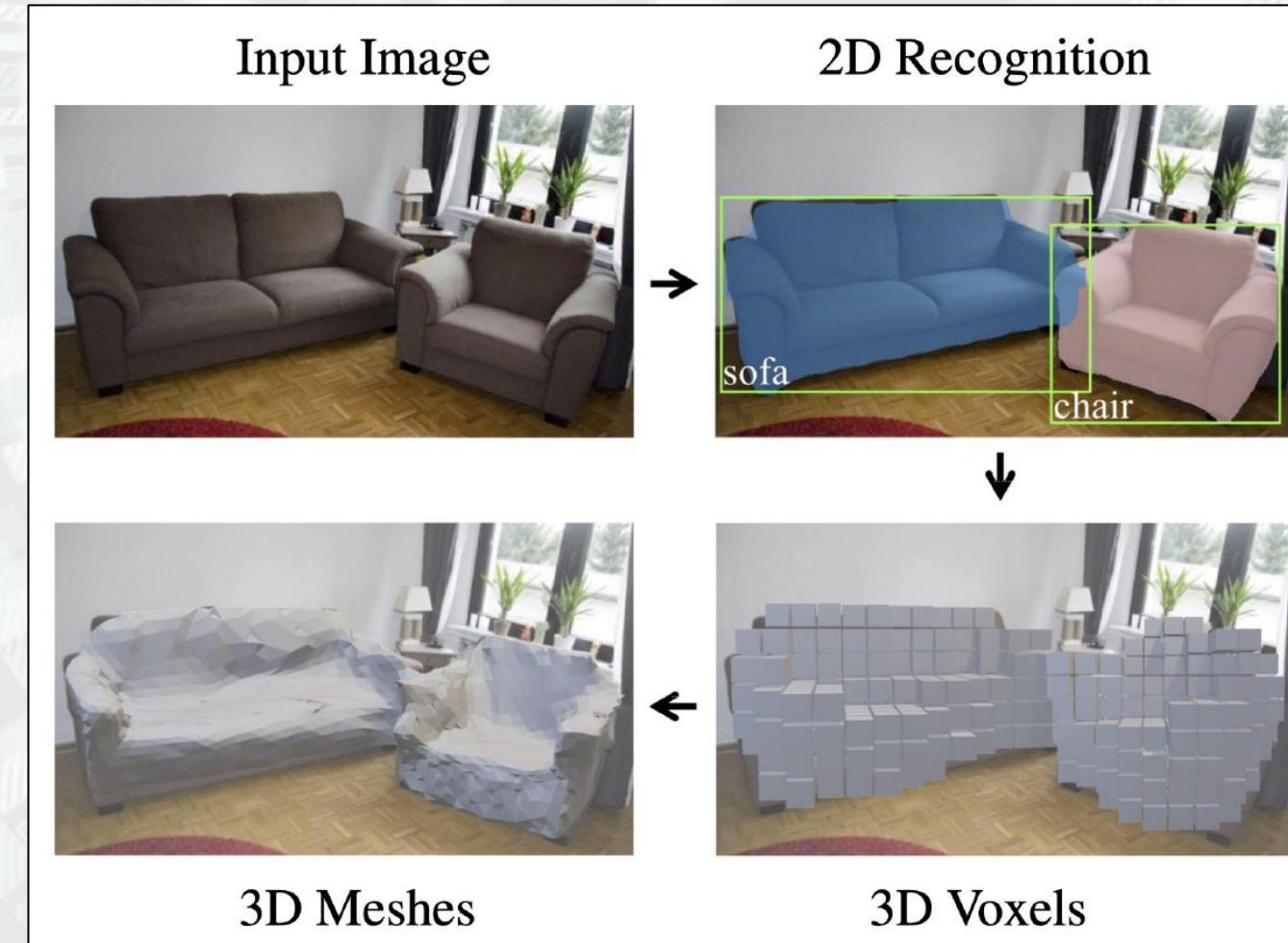
Aplicații ale detecției de obiecte

Detectie de obiecte 3D



Aplicații ale detecției de obiecte

Predictia Formelor 3D



Sfârșit M3

Bibliografie

- [1] <http://yann.lecun.com/exdb/mnist/>
- [2] <http://ufldl.stanford.edu/housenumbers/>
- [3] <https://www.cs.toronto.edu/~kriz/cifar.html>
- [4] <http://places2.csail.mit.edu/>
- [5] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.
- [6] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), 84-90.
- [7] Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In European conference on computer vision (pp. 818-833). Springer, Cham.
- [8] Lin, M., Chen, Q., & Yan, S. (2014). Network in network. 2nd International Conference on Learning Representations (ICLR 2014).
- [9] Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. 3rd International Conference on Learning Representations (ICLR 2015), 1–14.
- [10] Szegedy, C. et al. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

Bibliografie

- [11] [He, K., Zhang, X., Ren, S., & Sun, J. \(2016\). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition \(pp. 770-778\).](#)
- [12] [Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. \(2017\). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition \(pp. 4700-4708\).](#)
- [13] Chen, L., Li, S., Bai, Q., Yang, J., Jiang, S., & Miao, Y. (2021). Review of image classification algorithms based on convolutional neural networks. *Remote Sensing*, 13(22), 4712.
- [14] [Hu, J., Shen, L., & Sun, G. \(2018\). Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition \(pp. 7132-7141\).](#)
- [15] [Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. \(2020\). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.](#)
- [16] Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015) Explaining and harnessing adversarial examples. International Conference on Learning Representations (poster).
- [17] Nguyen, A., Yosinski, J., & Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 427-436).
- [18] [Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. \(2016\). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. arXiv preprint arXiv:1602.07360.](#)

Bibliografie

- [19] Han, S., Mao, H., & Dally, W. J. (2016). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. ICLR 2016.
- [20] [Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. \(2017\). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.](#)
- [21] [Zhang, X., Zhou, X., Lin, M., & Sun, J. \(2018\). Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the IEEE conference on computer vision and pattern recognition \(pp. 6848-6856\).](#)
- [22] Singh, S., & Batra, S. (2020). An efficient bi-layer content based image retrieval system. Multimedia Tools and Applications, 79(25), 17731-17759.
- [23] <http://host.robots.ox.ac.uk/pascal/VOC/>
- [24] <https://cocodataset.org/#home>
- [25] <https://storage.googleapis.com/openimages/web/index.html>
- [26] Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2017). Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1492-1500).

Bibliografie

- [27] Wang, C. Y., Liao, H. Y. M., Wu, Y. H., Chen, P. Y., Hsieh, J. W., & Yeh, I. H. (2020). CSPNet: A new backbone that can enhance learning capability of CNN. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops (pp. 390-391).
- [28] Tan, M., & Le, Q. (2019, May). Efficientnet: Rethinking model scaling for convolutional neural networks. In International conference on machine learning (pp. 6105-6114). PMLR.
- [29] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).
- [30] Felzenszwalb, P. F., & Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. International journal of computer vision, 59(2), 167-181.
- [31] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence, 37(9), 1904-1916.
- [32] Girshick, Ross. "Fast r-cnn." In Proceedings of the IEEE international conference on computer vision, pp. 1440-1448. 2015.
- [33] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28.

Bibliografie

- [34] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2117-2125).
- [35] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).
- [36] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).
- [37] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In European conference on computer vision (pp. 21-37). Springer, Cham.
- [38] Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision (pp. 2980-2988).