# **11 - BEYOND IMAGE CLASSIFICATION: OBJECT DETENCTION**

## From classification to detection

Capire cosa c’è in una certa immagine con una certa percentuale di accuratezza

A picture containing outdoor, horse, holding, young

Description automatically generated

**Detection**: localizzare oggetti, in una stessa immagine possiamo avere più oggetti dello stesso tipo e più oggetti diversi.

Graphical user interface, application

Description automatically generated

A close up of a road

Description automatically generatedIndustry 4.0 : esempio -> detection dei difetti

**Image classification**: un unico oggetto

**Classificazione con localizzazione**: oltre a sapere se c’è o meno la macchina, devo anche identificare dove questa si trovi nell’immagine

**Detection**: possiamo avere più oggetti dello stesso tipo all’interno della scena

## Classification with localization

Rispetto alla classificazione che si ottiene come output della softmax, mi interessa trovare il bounding box. Si può immaginare di portare in output anche le informazioni della **posizione e grandezza del bounding box.**

A picture containing diagram

Description automatically generated

Text

Description automatically generatedAdatterei l’allenamento supervisionato in modo tale da apprendere non solo le categorie ma anche le bounding box.

**Il vettore di uscita** sarà adattato nella seguente maniera

*Nota: pc è una probabilità discretizzata (0 o 1)*

### Esempio:

A picture containing diagram

Description automatically generatedA picture containing diagram

Description automatically generated

inserisco don’t care in quanto  
  
non mi interessa

Ora senza pensare al tipo di uscite, categoriche e continue, enumero tutte le uscite allo stesso modo y1, … y8

A picture containing diagram

Description automatically generated

Posso scrivere la **loss function** che dice: per il singolo example se l’oggetto è presente -> i quadrati delle differenze, se l’oggetto non c’è mi interessa solo avere il primo termine.   
*Questo fa intuire la natura binaria del primo termine e la natura contigua degli altri.*

Però questa Loss Function è troppo semplice perché non tiene conto della diversa natura dei dati di uscita;   
posso **distinguere i tre tipi di uscite** trattandoli come problemi diversi:

* Y1 è una logistic regression (0,1) -> probabilità discretizzata che ci sia
* Y2…Y5 è una linear regression -> posizione
* Y6…Y8 è un softmax

Posso aggregare la prima è l’ultima classe in un’unica softmax con 4 classi.

## Landmark detection

A picture containing text

Description automatically generated

Landmark: punti salienti che possono essere usati per face-recognition, action-recognition, etc

## Object detection

Chart

Description automatically generated

usando le reti profonde si è aumentata la precisione

## Sliding window detection

Nel training set abbiamo dei crop molto piccoli. Abbiamo una convnet a cui passando i crop mi dice car e not car.   
Per il momento ignoro i bounding box perché non sapendo quante volte può esserci una car aggiungere tale informazione può essere un problema.

Graphical user interface, application

Description automatically generated

-> do in pasto alla rete la finestra piccola fino a quando non mi dirà che c’è l’immagine: mi dirà 50% quando avrò metà auto, continuo a spostare e mi darà 100%. Dunque, facendo una sliding window riesco a farlo. Molto importante la **dimensione della finestra** e il **passo**.

Parto prendendo una porzione dell’immagine da dare in pasto e continuo spostando questa finestra, scegliendo un passo, fino a dare in pasto tutta l’immagine al classificatore convnet.

A close up of a road

Description automatically generated

* Non sto sfruttando la similarità tra il crop è quello successivo

Bisogna anche decidere la dimensione della window.

A sign on the side of a road

Description automatically generated

La dimensione della finestra e il passo si possono scegliere a mano oppure brute force aumentando il carico computazionale.

**È un approccio forza bruta e costa tantissimo -> cerchiamo altri approcci**

## **Trasformare una rete FullyConnected in un layer Convolutivo**

L’idea è di implementare l’approccio sliding window in **maniera convolutiva**.

Prendiamo una rete che ad un certo punto ha dei livelli fully connected e infine viene fatto l’unrolling ottenendo il softmax.

A picture containing clock

Description automatically generated

Proviamo sostituire i livelli fully connected con dei livelli convoluzionali:

* 400 filtri 5x5 -> mi permettono di passare dal 5x5x16 a 1x1x400
* Convoluzione 1x1x400
* Convoluzione 1x1 x4 filtri

Non c’è la stessa informazione all’interno in quanto ci sono operazioni differenti

* Sopra: Fully connected
* Sotto: copertura dei filtri

Shape

Description automatically generated

## Convolutional sliding window

A picture containing diagram

Description automatically generated

Immaginando l’immagine fosse 16x16

Nel caso 16x16 si può pensare di passare tutta l’immagine alla rete sfruttando la capacità convolutiva della rete perché applicando la sliding window, ad esempio 14x14, si avrebbe ridondanza inutile di computazione.

In questo caso l’output che non è più 1x1 ma 2x2 ci dice che nel primo parallelepipedo blu ci sono le informazioni della parte blu dell’immagine 16x16. Il blocco in basso a dx corrisponde alla 13x14x3 partendo da in basso a dx.

In pratica si fanno tutte le predizioni in un colpo solo evitando la sequenzialità.

Questo è scalabile ad un’immagine di dimensioni più grandi come ad esempio 28x28.

A picture containing chart

Description automatically generated

Ci sono 3 problemi:

* *Per immagini molto grandi ottengo degli output molto grandi e anche molta complessità computazionale*
* *La sliding window ha un fattore di forma limitato che potrebbe presentare problemi con oggetti ad esempio allungati etc -> aspect ratio*
* *Accuratezza dovuta allo stride*

## **Region proposal: R-CNN**

Può succedere che il nostro target non abbia dimensioni compatibili con la finestra o che non lo centro.

* Difficile che l’intera persona sia in un quadrato precisamente

A picture containing road, person, walking, young

Description automatically generatedA person standing in a parking lot

Description automatically generated

L’algoritmo propone delle regioni in cui andare a cercare l’oggetto.

TWO STAGE:

* Dove andare a cercare: genero delle regioni proposte
* Poi vado a cercare: classifico ogni regione proposta

**L’algoritmo prende il nome di R-CNN:**

* Sull’immagine di input viene fatto girare un algoritmo **di selective search** spesso non convolutiva che tira fuori delle ROIs (Region Of Interest) -> regione dove è probabile trovare l’oggetto
* Sulle ROIs viene fatto un **warping**, ovvero vengono adattate le dimensioni per andare in pasto alla convnet
* Le ROIs warpate vengono passate alla convnet che fanno la classificazione ottenendo se ci sono gli oggetti cercati e il bounding box in relazione alla ROI. ( si può fare in parallelo)

Diagram

Description automatically generatedImmagine che contiene testo, Carattere, schermata, bianco

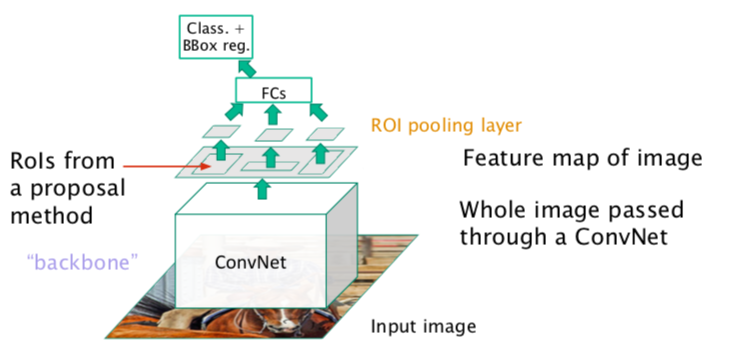
Descrizione generata automaticamente

* Rete convolutiva con 2 teste (classificazione + parte regressiva per bounding boc)

Il problema è che l’addestramento è molto lento (84h) e richiede molta archiviazione.   
È lento anche nel fare le predizioni *(47 s/image con VGG 16).* Influisce anche la scelta della rete convolutiva da utilizzare con questo meccanismo.

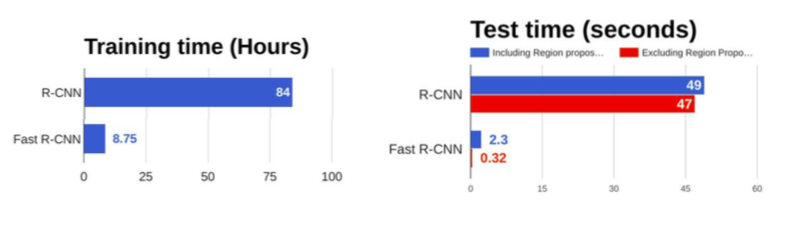
## **Region proposal: Fast R-CNN**

Nel 2015 è stata proposta una versione evoluta chiamata Fast R-CNN che usa lo stesso principio ma utilizza **un’implementazione convolutiva** delle sliding windows per classificare tutte le regioni in una sola volta.

Per prima cosa fa la convoluzione che:

* Traduce il dominio dell’immagine nello spazio delle feature
* Cerca delle ROI sulla feature map dell’immagine
* Prende queste regioni (della feature map) e le ridimensiona (pooling)
* Esegue classificazione su queste regioni con dei livelli fullly-connected

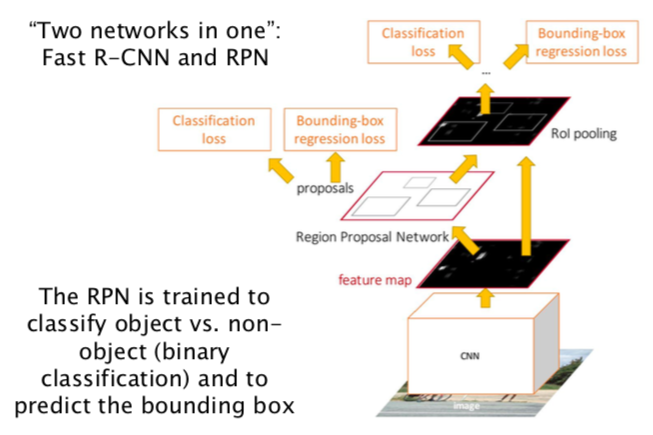
Confrontiamo i tempi delle due reti:



Nella Fast R-CNN il collo di bottiglia è dato dalla generazione delle ROI (2,3s).

Per andare ancora più veloci Faster R-CNN -> traformare in convolutivo anche il region-proposal.

## Region proposal: **Faster R-CNN**



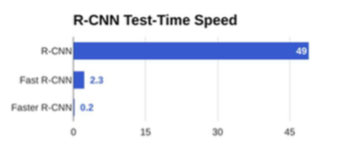
* Rete convolutiva trasforma immagine in dominio delle feature -> feature map
* Rete RPN (Region Proppsal Network) che tira fuori dalla feature map, delle ROI dello spazio delle feature
* Viene fatto il pooling sulle ROI -> trasformate in regioni quadrate
* Si fa la classificazione

Continuiamo ad avere una CNN per estrarre la feature map dell’immagine; su questa non uso un algoritmo tipo selective search ma vado ad utilizzare un’altra rete per tirare fuori le ROIs. Quindi la feature map viene passato ad un pooling di ROI che le porta tutte alla stessa dimensione e a questo punto si vanno a tirare fuori le informazioni volute.

In questo caso è come se si avesse un algoritmo con **due reti l’una nell’altra**: una Fast R-CNN e una RPN (Region Proposal Network).

L’idea è quella di avere loss sia per la parte di classificazione e bounding box e sia per la proposal delle ROIs.

Vedendo i tempi di tutte e tre le reti:



Tempo di inferenza si è abbassato sa 49 a 0.2s.

* Convoluzioni più possibile per risparmiare calcoli
* Convoluzioni il prima possibile perché andando nello spazio delle feature si riduce la rappresentazione
* Andiamo a prendere in partid iverse della rete quello che ci serve e trattiamo in maniera convolutiva

Successivamente si è tentato di fare tutto in un colpo solo -> Single Shot

## **YOLO**

### BB predicitons (Basis of YOLO)

Prima implementazione si basa sulla SSD (Single-Shot Detection) che non usa la region proposal. Si cerca di fare localizzazione e classificazione contemporaneamente sfruttando la divisione in griglia per diminuire la complessità. (Divide and Impera)

A picture containing diagram

Description automatically generated

Si crea **per ogni unità della griglia un vettore y.**

Per descrivere l’appartenenza di un oggetto si considerano i midpoint, ovvero il centro di dove immaginiamo verrà posizionato il bounding box e in base alla cella a cui appartiene questo oggetto usiamo il midpoint.

La dimensione di uscita à 3x3(dimensione della griglia) x8(4 di bounding box + 1 si/no + 3 classi).

Il limite è che posso avere al più un oggetto per unità di griglia.

Diagram

Description automatically generated

In realtà **YOLO** *(You Only Look Once)* fa qualcosa di più sofisticato:

* Prende l’immagine 100x100x3
* Fa una serie di operazioni convoluzionali
* Ottengo l’uscita 3x3x8 dove posso avere al più un oggetto per vettore

Nota:

* Non usa sliding window
* Divide problmea in sotto-problemi
* Funziona con più classi diverse
* Funziona con più oggetti diversi o anche di stessa classe in celle diverse
* Non funziona se oggetti in stessa cella

A picture containing stove

Description automatically generated

Nella realtà viene utilizzato con griglie con celle più piccole per far scendere la probabilità e la criticità di avere più midpoint nella stessa griglia

### Evaluating object localization

Per dire se la predizione di un algoritmo è buona ci sono diversi approcci ma quello base è la “Intersection over Union”: il rapporto tra l’area sovrapposta (gialla) e l’unione della predizione con il ground truth (fucsia).

Graphical user interface, text, application

Description automatically generated

Si imposta quindi una soglia (spesso di 0.5) per dire se la predizione è corretta. (1 solo se coincidono perfettamente)

C’è il problema che questa dimensione cresce sia se sottostimiamo sia se sovrastimiamo.

**Non-max suppression**

Siccome si suddivide l’immagine in una griglia, ci possono essere per ogni oggetto diverse celle della griglia in cui si trova e quindi l’output dell’algoritmo restituirà diverse bounding box.

A picture containing text, sky, road, outdoor

Description automatically generated

Per ognuno di essi la rete da in output le probabilità ovvero i livelli di confidenza.   
L’algoritmo di non-max suppression prende il bounding box con la probabilità più alta, e calcola la IoU con tutti gli altri presenti nella scena. Tutti quelli che si sovrappongono oltre una certa soglia, li butta via. A questo punto prende gli altri bouynding box, non sovrapposti, prende il massimo e ripete la stessa cosa.

Otteniamo:

**Non-max suppression algorithm**

Prendiamo l’immagine, applico una griglia grande per evitare che ci siano due midpoint nella stessa cella.

Chart, histogram

Description automatically generated

Ottengo dei vettori di output, **scarto tutti i box che hanno pc <= 0.6** e **scarto qualsiasi box rimanente con IoU <= 0.5**.

Se ci sono più classi si ripete il procedimento per ogni classe.

*Nota: IoU alta -> maggior precisione nell’identificare le region. Soglia più bassa -> recall più alto, esser sicuri di non mancare nessuna automobile.*

## Overlapping objects

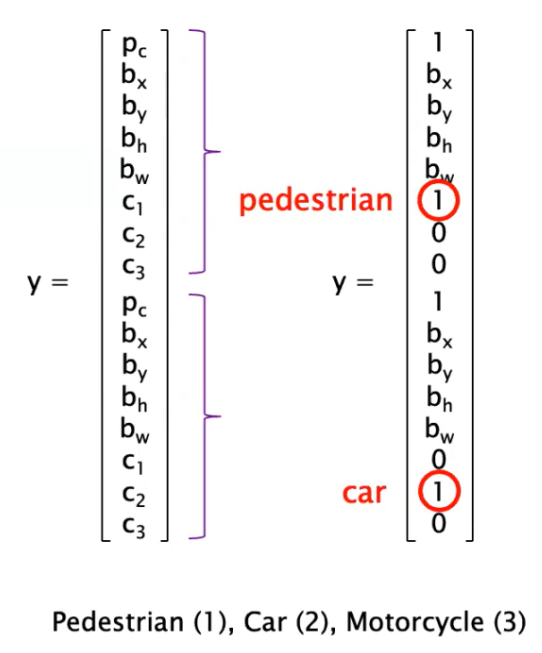
Cosa succede se ci sono più oggetti in una cella della griglia? Dobbiamo essere in grado di fare più predizioni per cella

A person standing in front of a red car

Description automatically generated with medium confidence

L’idea alla base è quella di avere **anchor boxes**, ovvero si definiscono a priori delle ancore di una dimensione prefissata con dei propri fattori di forma.

Prima di usare questa idea avremmo avuto un output di otto valori per cella; invece, quello che si è fatto è stato avere nello stesso output l’unione delle predizioni per le singole ancore.



Graphical user interface, diagram

Description automatically generated

Graphical user interface, text

Description automatically generated

* Hand-enginnering

**YOLO algorithm**

A picture containing diagram

Description automatically generated

**Training YOLO**

Volume 3x3x16 con 2 ancore -> ci devo mettere una rete e lo addestro -> mi dira che non c’è nulla per l’anchor box verticale mentre per l’anchor box orrizzontale c’è l’automobile

Diagram

Description automatically generated

**Non-max suppression with YOLO**

A picture containing text, person

Description automatically generated

**Datasets for object detection**

A picture containing graphical user interface

Description automatically generated

* PASCAL VOC: triaining set da 11k immagini con in media 2.7 oggetti ciascuna
* IMAGENET large scale: tantissime immagini, circa il 60% con un solo oggetto all’interno
* MS COCO: solo il10% dell’immagini contiene una singola categoria di oggetti.

### Evaluating detection performance

La valutazione viene fatta innanzi tutto sul classificatore usando la precision e la recall.

Diagram

Description automatically generatedSi combinano queste metriche con la IoU

Graphical user interface, text, application

Description automatically generated

Si disegnano delle **curve chiamate precision-recall** e guardandole si può scegliere dove posizionarsi se andare più sulla precision o sulla recall.

Chart, line chart

Description automatically generated

Il problema è che l’andamento delle curve non è necessariamente monotonico perché, se abbassiamo la soglia facciamo crescere sia i falsi positivi che i falsi negativi e la precision può andare un po’ dove vuole.

La curva dipende anche dalla proporzione tra true positive e true negative (data imbalance).

**Una buona misura di bontà di un algoritmo è data dall’area sotto la curva -> F1-score.**

#### mAP (mean Average Precision)

Si prende l’area sotto tutte le curve (una curva per classe) e si calcola la media.

Text, letter

Description automatically generated

**Altre metriche**

Table

Description automatically generated with low confidence

**COCO leaderboard**

Si vede quali sono gli algoritmi che hanno performato meglio col tempo

Graphical user interface, text, application

Description automatically generated

Il grafico non tiene conto dei tempi di esecuzione.

**Considerazioni**

Object detectors non sono architetture end-to-end, cioè dove do l’immagine in input e ottengo l’output ma in mezzo ci sono una serie di meccanismi. -> Serve lavoro ulteriore, hand-crafted post-processing.

Tutti utilizzano un’architettura di base che poi viene variata e quindi queste si possono considerare come hyper-parametri o delle procedure da seguire.

La rappresentazione di bounding box non è ottimale in tutti i casi (shapes non regolari, classi sovrapposte, oggetti in posizioni non canoniche), così come le anchors (anche se convenienti se ci sono molte infomrazioni a priori).

La loss standard della regressione non necessariamente garantisce un buon IoU.