Aula 9
Detecção de Objetos

Eduardo L. L. Cabral

Objetivos

- Definir o que é detecção de objetos.
- Apresentar método de localização de objetos.
- Apresentar detecção de pontos de referência.
- Apresentar método de detecção de objetos.
- Apresentar a RNA YOLO.

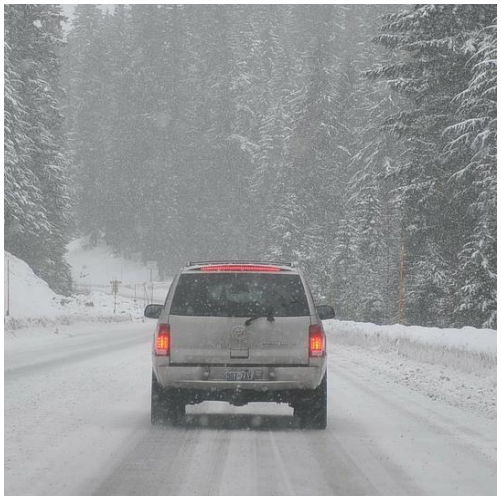
Detecção de objetos

- Detecção de objetos talvez seja a área de redes neurais aplicadas à visão computacional que obteve o maior sucesso nos últimos anos.
- Definições:
  - Classificação de objetos $\Rightarrow$ consiste em determinar se uma imagem mostra ou não um determinado objeto.
  - Classificação com localização de objeto $\Rightarrow$ consiste em determinar se uma imagem mostra ou não determinado objeto e determinar onde se encontra esse objeto na imagem.
  - Detecção de objetos $\Rightarrow$ consiste em classificar e localizar múltiplos objetos em uma imagem, ou seja, deseja-se classificar e localizar todos os objetos (dentro de determinadas classes) mostrados em uma imagem.

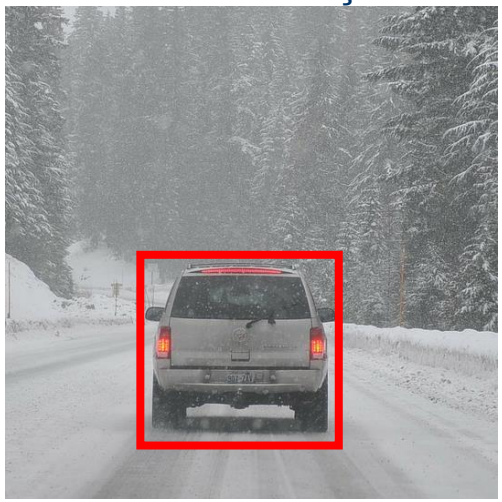
Detecção de objetos

- Na classificação e localização tem-se um único objeto na imagem.
- Na detecção tem-se múltiplos objetos, iguais e diferentes, na imagem.

Classificação de objeto



Classificação com localização



Detecção

Um único objeto na imagem

Vários objetos

Classificação de objetos

- Classificar um objeto mostrado em uma imagem consiste de um problema de classificação de múltiplas classes $\Rightarrow$ que foi visto nas aulas anteriores.
- Para resolver um problema de classificação de objetos em uma imagem é utilizada uma RNA convolucional com camada de saída do tipo softmax.
- O número de neurônios na camada de saída softmax é igual ao número de classes que se deseja classificar.

Localização de objetos

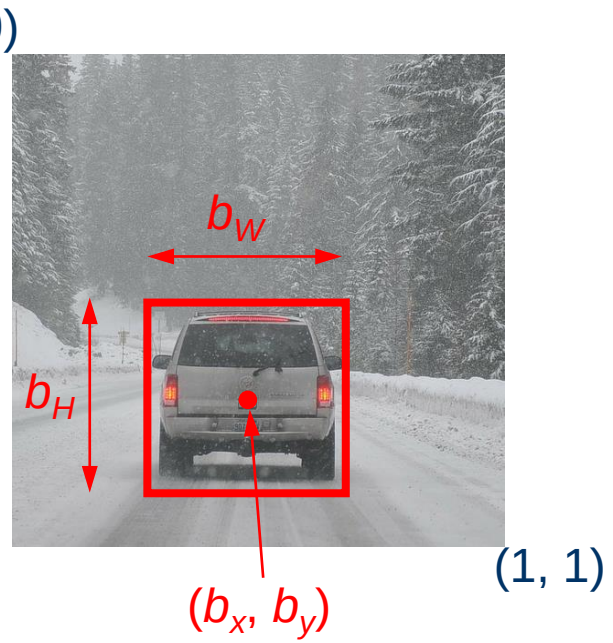
- Para localizar um objeto em uma imagem usa-se uma caixa delimitadora (CB), ou “bounding box” (BB) em inglês.
- A caixa delimitadora contém completamente o objeto e define onde ele se encontra na imagem.
- A caixa delimitadora possui quatro parâmetros:

  b_x = posição horizontal central da caixa;

  b_y = posição vertical central da caixa;

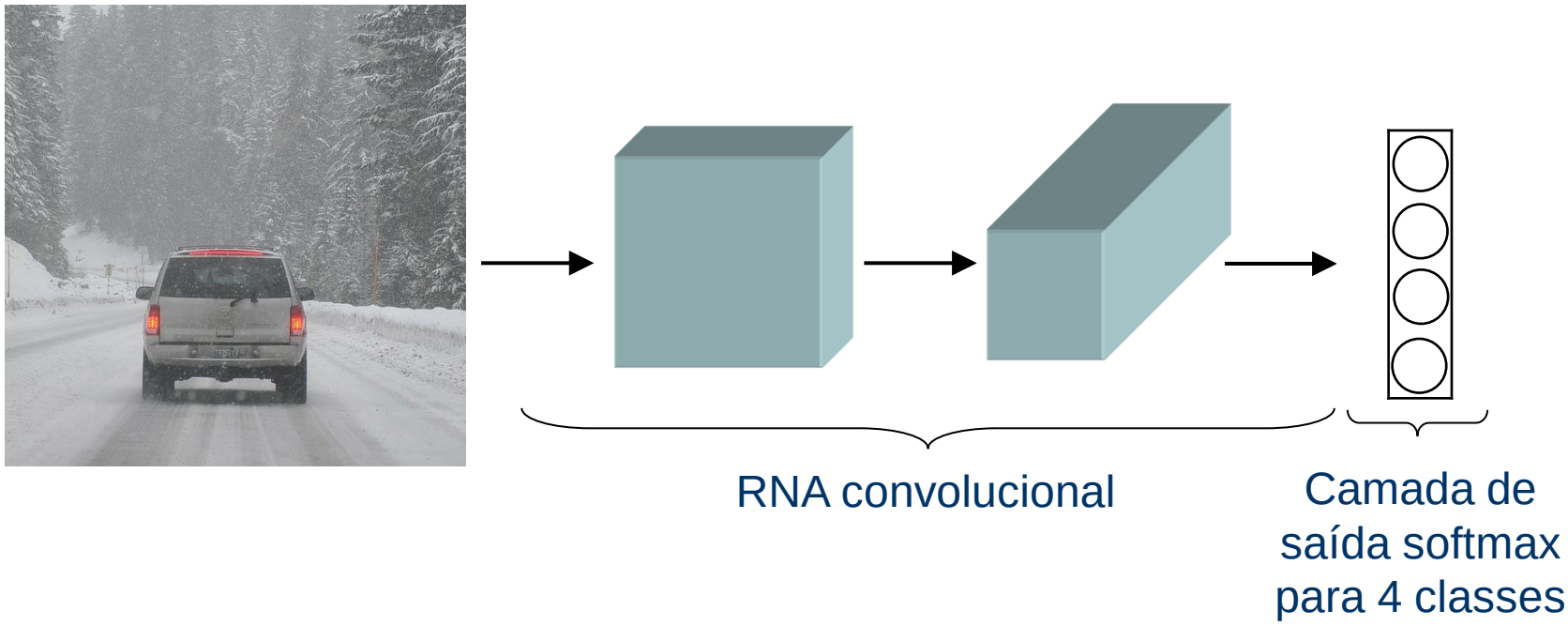
  b_H = altura da caixa;

  b_W = largura da caixa.
- No exemplo: $b_x = 0{,}5$; $b_y = 0{,}7$; $b_H = 0{,}3$ e $b_W = 0{,}4$.



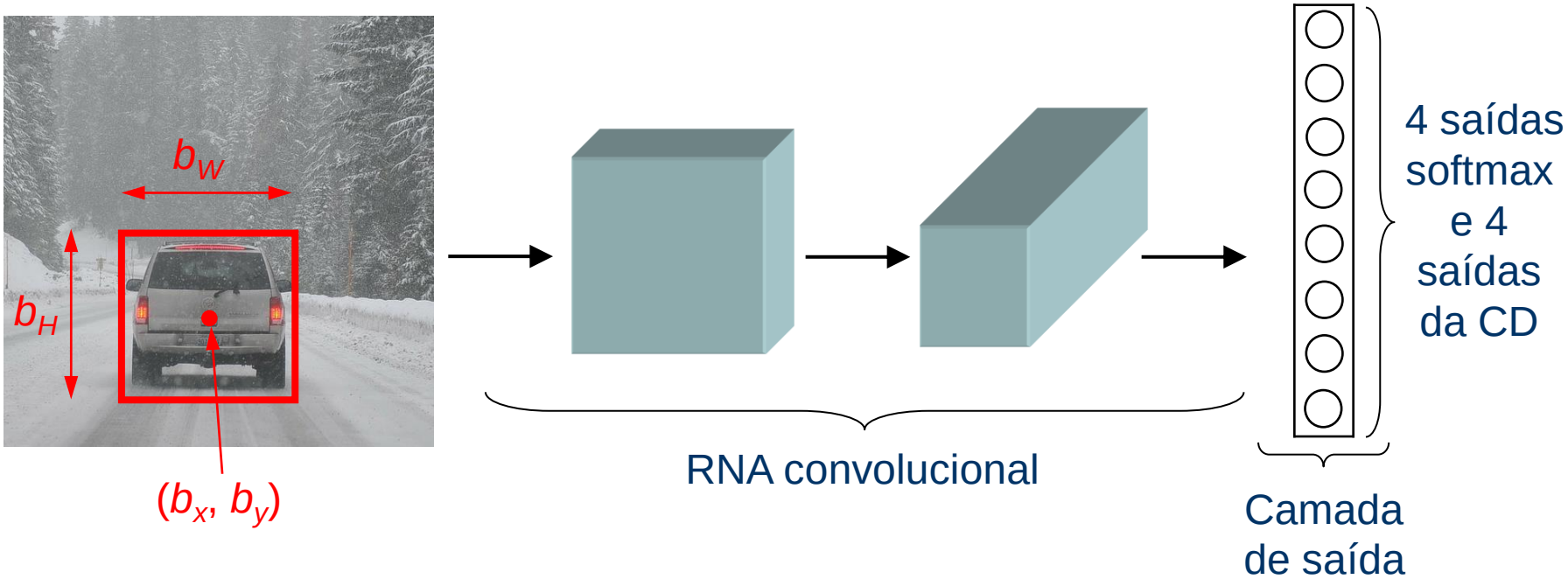
Localização de objetos

- Exemplo de RNA para classificação de objeto com 4 classes:
  1 - pedestre; 2 - carro; 3 - moto; 4 - fundo.



Localização de objetos

- Para classificar e localizar objetos em uma imagem deve-se alterar a camada de saída softmax para incluir a posição do objeto na imagem, representada pelos 4 parâmetros da caixa de delimitadora (CD).



Localização de objetos

- No exemplo com 4 classes $\Rightarrow$ a saída da RNA precisa possuir os 4 parâmetros da caixa delimitadora e o rótulo das 4 classes.
- Saída da RNA para classificação e localização:

$$\mathbf{y} = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_H \\ b_W \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

Onde:

p_c = probabilidade de existir um objeto das 3 classes;

$\begin{cases} p_c = 0 \Rightarrow \text{não tem nenhum objeto das 3 classes;} \\ p_c = 1 \Rightarrow \text{mostra algum dos três tipos de objeto;} \end{cases}$

b_x, b_y, b_H, b_W = parâmetros da caixa delimitadora;

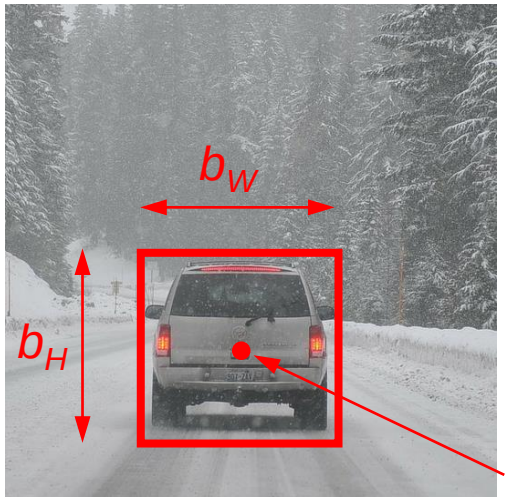
c_1 = probabilidade de mostrar pedestre;

c_2 = probabilidade de mostrar carro;

c_3 = probabilidade de mostrar moto.

Localização de objetos

- Exemplos de saídas da RNA:
  - Quando $p_c = 0 \Rightarrow$ significa que a imagem não mostra nenhum dos objetos que se deseja classificar e, portanto, os valores das outras saídas não importam.



$$\Rightarrow \mathbf{y} = \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_H \\ b_W \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Imagem com carro

$$\Rightarrow \mathbf{y} = \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

Imagem de fundo

Localização de objetos

- Função de custo utilizada para treinamento:
  - Pode usar a função do erro quadrático para todos os elementos da saída:

$$E(\hat{\mathbf{y}}, \mathbf{y}) = \begin{cases} (\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2 + \ldots + (\hat{y}_8 - y_8)^2, \text{se } p_c = 1 \; (y_1 = 1) \\ (\hat{y}_1 - y_1)^2, \text{se } p_c = 0 \; (y_1 = 0) \end{cases}$$

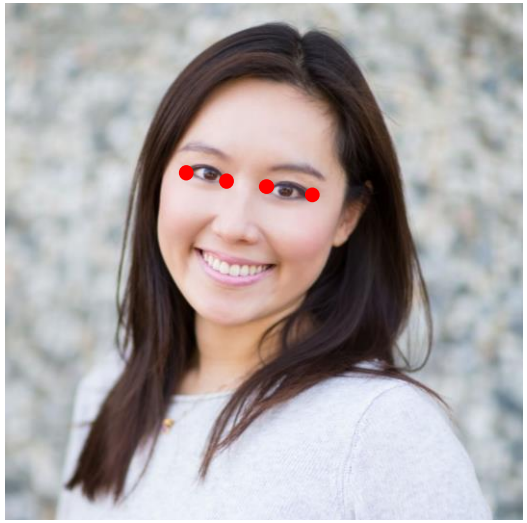
Note que a função de custo é calculada de forma diferente dependendo se $p_c = 0$ ou 1.

  - Pode usar funções de custo diferentes para cada tipo de saída $\Rightarrow$ melhor:

$$\mathbf{y} = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_H \\ b_W \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \Rightarrow \begin{cases} p_c \rightarrow \text{regressão logística (classificação binária);} \\ b_x, b_y, b_H, b_W \rightarrow \text{erro quadrático médio (regressão);} \\ c_1, c_2, c_3 \rightarrow \text{entropia cruzada de múltiplas classes (classificação multi-classe).} \end{cases}$$

Detecção de pontos de referência

- O método usado para localizar objetos em uma imagem pode ser utilizada para detectar e localizar pontos de referência em uma imagem.
- Por exemplo ⇒ determinar a localização dos cantos dos olhos em uma face.

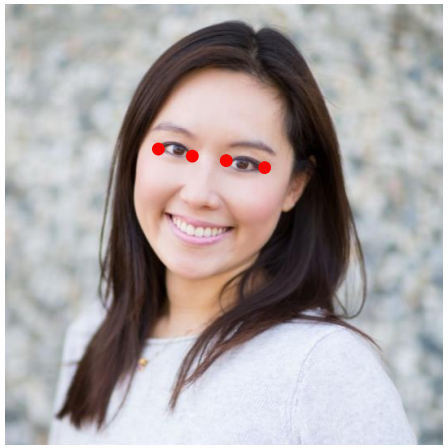


- Saída da RNA precisa fornecer a posição de cada ponto de referência, ou seja, as duas coordenadas de posição dos cantos dos dois olhos (4 cantos).

Detecção de pontos de referência

- Saída da RNA para detectar os 4 cantos dos olhos:

$$\mathbf{y} = \begin{bmatrix} p_c \\ l_{1x} \\ l_{1y} \\ l_{2x} \\ l_{2y} \\ l_{3x} \\ l_{3y} \\ l_{4x} \\ l_{4y} \end{bmatrix} \Rightarrow \begin{cases} p_c = \text{probabilidade da imagem apresentar ou não face;} \\ l_{1x}, l_{1y} = \text{coordenadas do primeiro canto de olho;} \\ l_{2x}, l_{2y} = \text{coordenadas do segundo canto de olho;} \\ l_{3x}, l_{3y} = \text{coordenadas do terceiro canto de olho;} \\ l_{4x}, l_{4y} = \text{coordenadas do quarto canto de olho.} \end{cases}$$



Detecção de pontos de referência

- Pode escolher detectar quanto pontos de referência desejar ⇒ por exemplo, determinar 64 pontos da face:
  - Nesse caso a saída é um vetor de 129 elementos;
  - A primeira saída é a probabilidade da imagem mostrar ou não uma face;
  - As outras 128 saídas são as coordenadas dos pontos que deseja detectar.
- Reconhecer pontos de referência de uma face (ou de algum outro objeto) é uma operação básica para diversos algoritmos, tais como:
  - Reconhecimento de emoções;
  - Realidade aumentada ⇒ incluir objetos na face (chapéu, óculos, barba, bigode etc).
- Para treinar uma RNA para detectar pontos de referência precisa de imagens de treino com todos os pontos de referência definidos.

Detecção de pontos de referência

- Existem muitas outras aplicações de detecção de pontos de referência.
- Uma aplicação, que é muito comum, é a determinação da pose de pessoas (ou objetos):
  - Escolhe pontos chaves do corpo para a RNA detectar;
  - Após detectar pontos chaves usa outra RNA que recebe esses pontos e determina a pose.

Detecção de caixas delimitadoras

- Existem vários algoritmos para detectar caixas delimitadoras:
  - Janela deslizante;
  - Caixas âncoras (YOLO);
  - Proposta de região (R-CNN).
- O melhor algoritmo atualmente é o usado pela YOLO.
- Referência YOLO $\Rightarrow$ Redmon et al., 2015, You Only Look Once: Unified real-time object detection.

Detecção de caixas delimitadoras

- Para detectar caixas delimitadoras o primeiro passo é dividir a imagem em células usando uma malha.
- O algoritmo YOLO usa originalmente uma malha 19x19.
- A detecção das caixas é realizada por uma RNA convolucional.
- Por exemplo, queremos detectar 3 tipos de objetos:

  1 - pedestre;

  2 - carro;

  3 - moto.

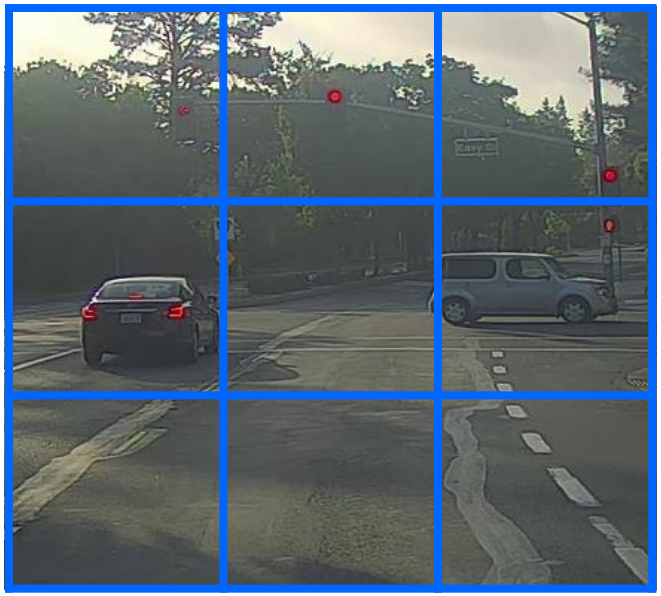


Imagem com malha 3x3

Detecção de caixas delimitadoras

- Para cada célula da malha a saída da RNA é uma caixa delimitadora $\Rightarrow$ no caso de detecção de 3 classe de objetos diferentes a caixa tem 8 elementos.
- Saída para cada célula da malha:

$$\mathbf{y} = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_H \\ b_W \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

p_c = probabilidade de existir um objeto das 3 classes;

b_x, b_y, b_H, b_W = parâmetros da caixa delimitadora;

c_1 = probabilidade de mostrar pedestre;

c_2 = probabilidade de mostrar carro;

c_3 = probabilidade de mostrar moto.

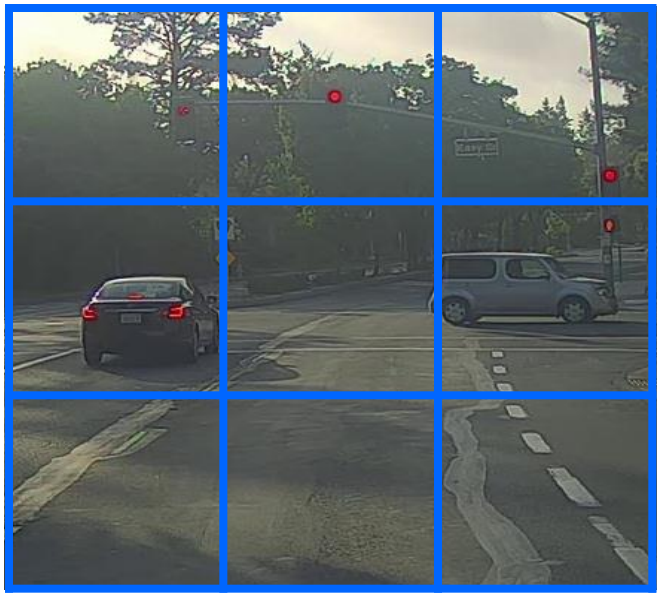


Imagem com malha 3x3

Detecção de caixas delimitadoras

- Na figura foram detectadas duas caixas delimitadoras, ou seja 2 objetos (carros).
- Considera-se que uma célula possui um objeto se o centro do objeto está dentro da célula ⇒ um objeto somente é detectado em uma única célula.
- Na figura somente as células (2, 1) e (2, 3) possuem objetos.
  - Caixa da célula (2,1) ⇒ cor laranja;
  - Caixa da célula (2, 3) ⇒ cor vermelha.

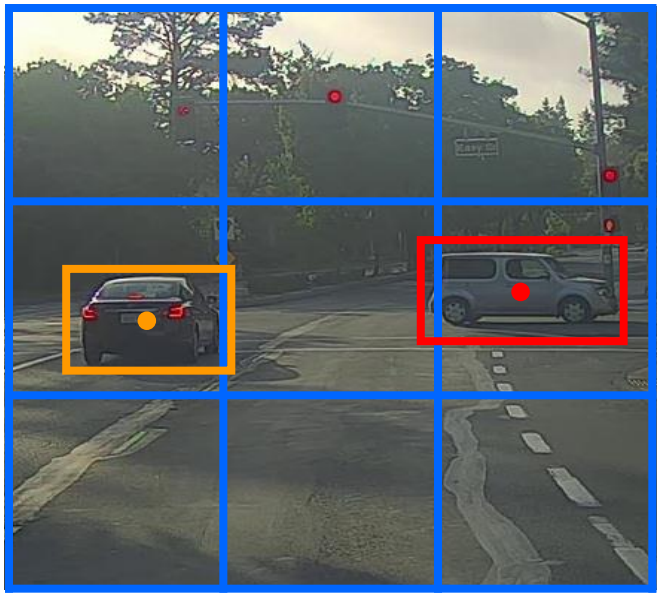


Imagem com malha 3x3

Detecção de caixas delimitadoras

- Saída para células sem objetos:

$$\mathbf{y} = \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix} \rightarrow p_c = 0$$

Outros parâmetros não importam

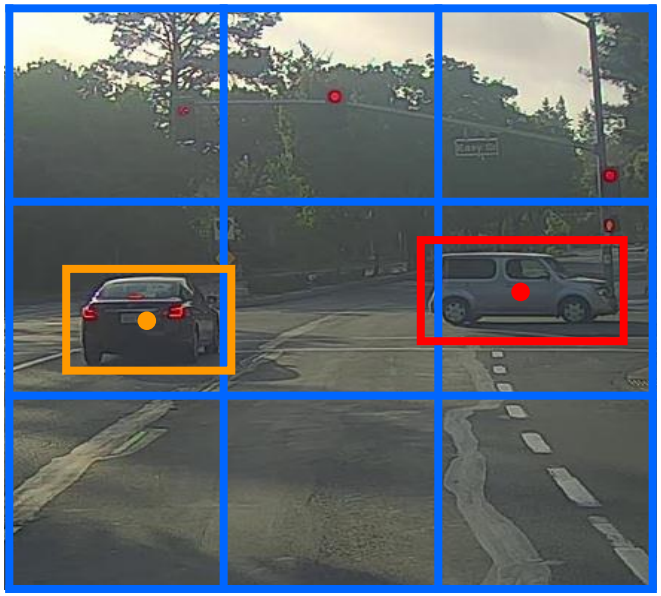


Imagem com malha 3x3

Detecção de caixas delimitadoras

- Saídas para células com objeto:

$$\mathbf{y} = \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_H \\ b_W \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$1 \rightarrow p_c = 1$

b_x, b_y, b_H, b_W: Posição do centro e tamanho da caixa

$1 \rightarrow$ Classe carro

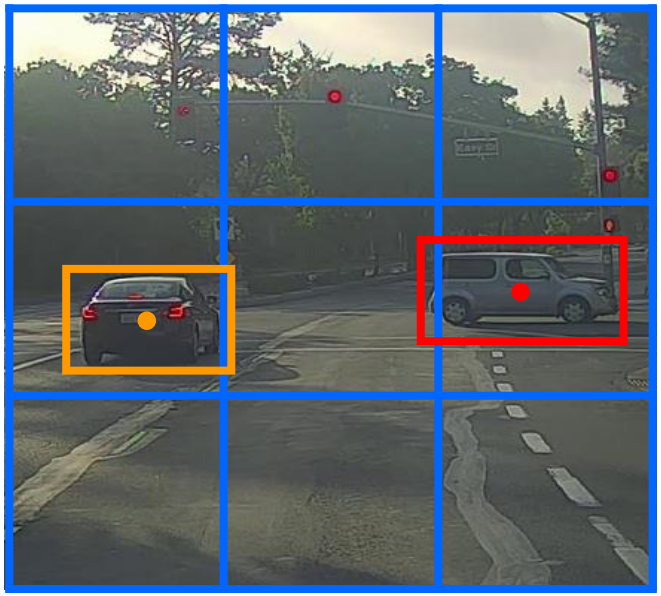


Imagem com malha 3x3

Detecção de caixas delimitadoras

- Parâmetros b_x, b_y, b_H, $b_W \Rightarrow$ valores são referentes à célula.
- Parâmetros das caixas da figura:
  - Caixa laranja:

    $b_x = 0,7$; $b_y = 0,6$; $b_H = 0,6$; $b_W = 0,8$.
  - Caixa vermelha:

    $b_x = 0,3$; $b_y = 0,5$; $b_H = 0,5$; $b_W = 0,9$.
- Posição do centro: $0 < b_x$ e $b_y < 1$;
- Tamanho do objeto:
  - b_H e $b_W > 0$;
  - b_H e $b_W > 0$ e podem ser > 1 (objeto pode ser maior que célula).

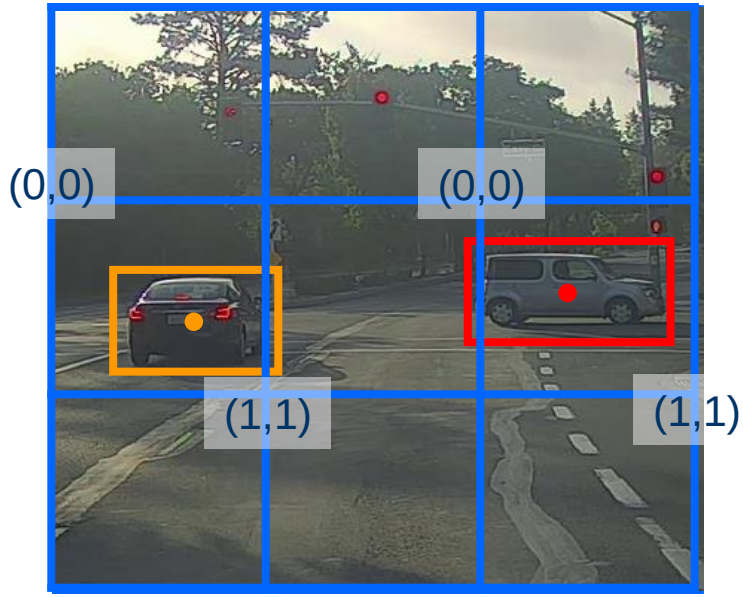
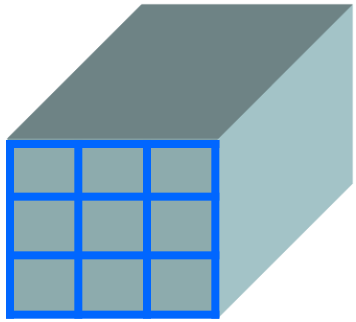


Imagem com malha 3x3

Detecção de caixas delimitadoras

- Para cada célula a saída é um vetor de 8 elementos para caso de 3 classes.
- Saída da RNA é um tensor 3x3x8 (malha 3x3 e 8 elementos para cada célula).



Saída $\Rightarrow$ tensor 3x3x8

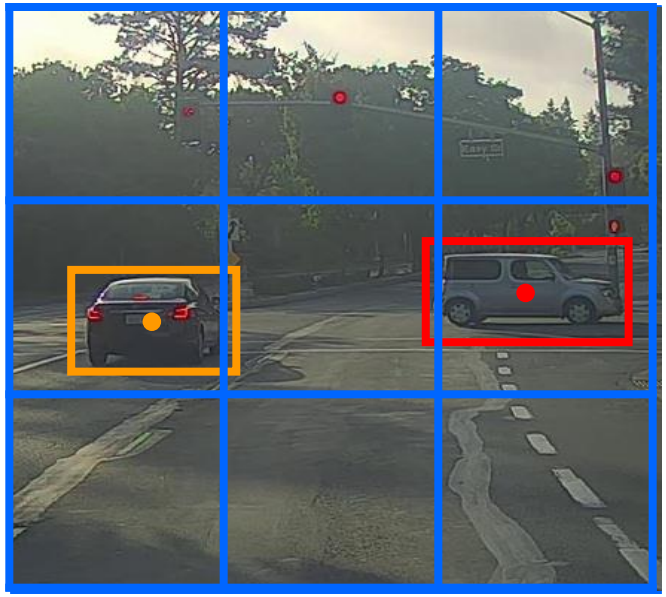


Imagem com malha 3x3

Detecção de caixas delimitadoras

- Detecção das caixas delimitadoras é realizada por uma RNA totalmente convolutional.
- ➢ RNA não possui nenhuma camada densa.
- Entrada da RNA ⇒ imagem de dimensão (n_H, n_W, 3);
  - n_H = altura da imagem;
  - n_W = largura da imagem.
- Saída da RNA ⇒ tensor de dimensão (n_{GH}, n_{GW}, 5+N_C);
  - n_{GH} x n_{GW} = tamanho da malha (no exemplo 3x3), YOLO original usa malha de 19x19;
  - N_C = número de classes do problema (no exemplo 3 classes).
- RNA convolucional deve ser configurada para receber uma imagem de entrada de dimensão (n_H, n_W, 3) e gerar na sua saída um tensor de dimensão (n_{GH}, n_{GW}, 5+N_C).

Detecção de caixas delimitadoras

- RNA convolucional para detecção de caixas delimitadoras:

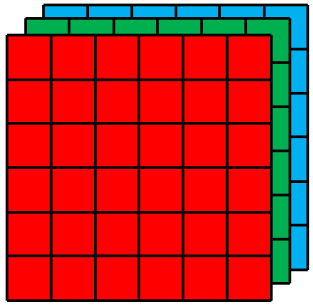
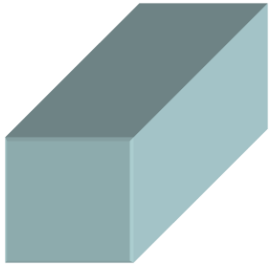


Imagem
$n_H x n_W x 3$

Camadas convolucionais e de pooling



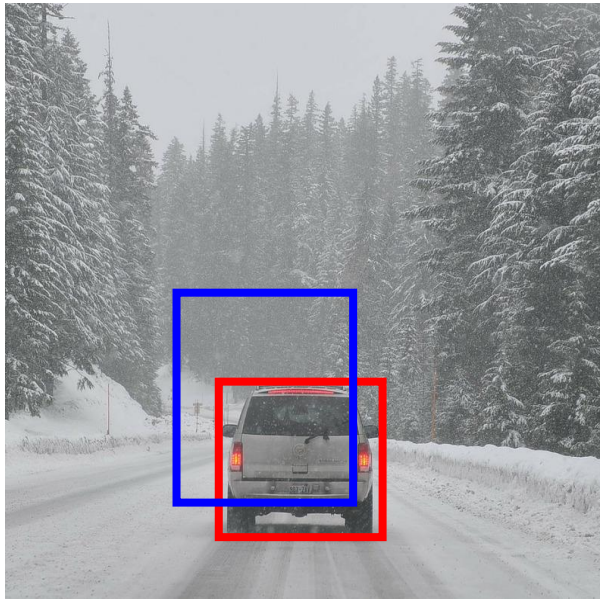
Saída
$n_{GH} x n_{GW} x (5+N_C)$

Detecção de caixas delimitadoras

- Problema desse método ⇒ não é possível detectar mais do que um único objeto em cada célula da malha.
- Usar malha mais fina minimiza o problema de existir mais de um objeto na mesma célula ⇒ quanto menor a célula menor a probabilidade de se ter mais de um objeto na célula.
- Mesmo com células pequenas não é possível garantir que sempre vai ter um único objeto em cada célula.
- YOLO modifica o método de detecção de caixas delimitadoras para permitir mais do que um único objeto em cada célula.

Intersecção sobre união

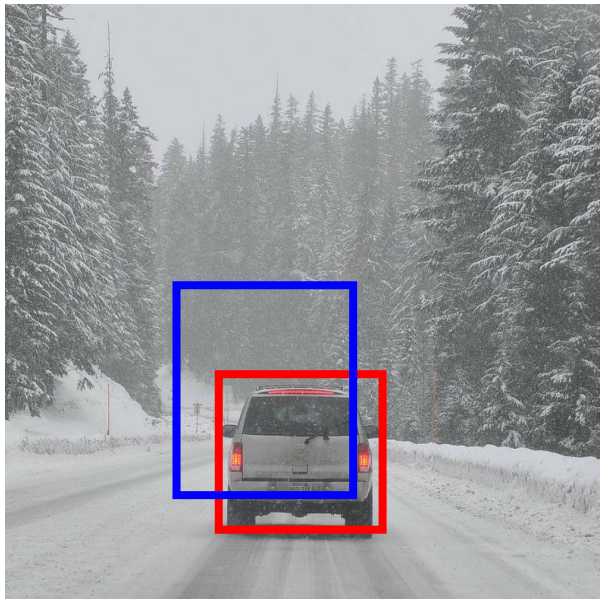
- Intersecção sobre união (*IoU*) é um parâmetro de medida utilizado para avaliar e aprimorar algoritmos de detecção de objetos.
- *IoU* serve para verificar se uma caixa delimitadora de um objeto foi detectada corretamente.

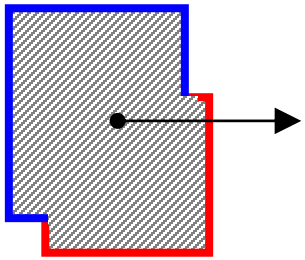


- Um objeto (carro) é detectado pela caixa azul, sendo que o correto seria a caixa vermelha.
- Como medir esse erro de detecção?

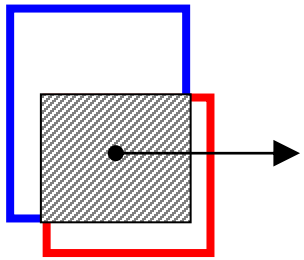
Intersecção sobre união

- Tem-se duas caixas:
  - Caixa vermelha ⇒ real;
  - Caixa azul ⇒ detectada.
- Define-se dois parâmetros ⇒ área da união e área da intersecção entre as duas caixas.





Área da união entre as duas caixas (*U*)



Área da intersecção entre as duas caixas (*I*)

Intersecção sobre união

- Intersecção sobre União (*IoU*):

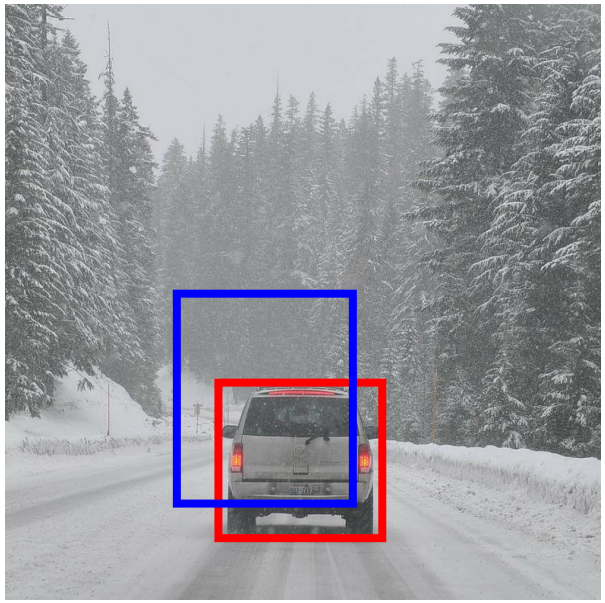
$$IoU = \frac{I}{U}$$

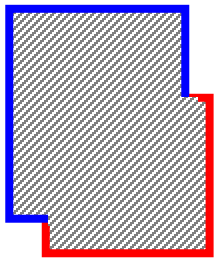
Se $IoU = 1 \Rightarrow$ as duas caixas são idênticas;

- *IoU* é uma medida de quão similar são as duas caixas.
- Quanto maior *IoU* melhor o resultado da detecção.
- Em geral adota-se como convenção que a detecção está correta se:

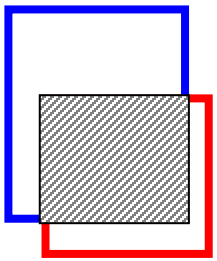
$IoU \geq 0{,}5$.

- Pode-se mudar esse limiar.





(*U*)



(*I*)

Supressão não máxima

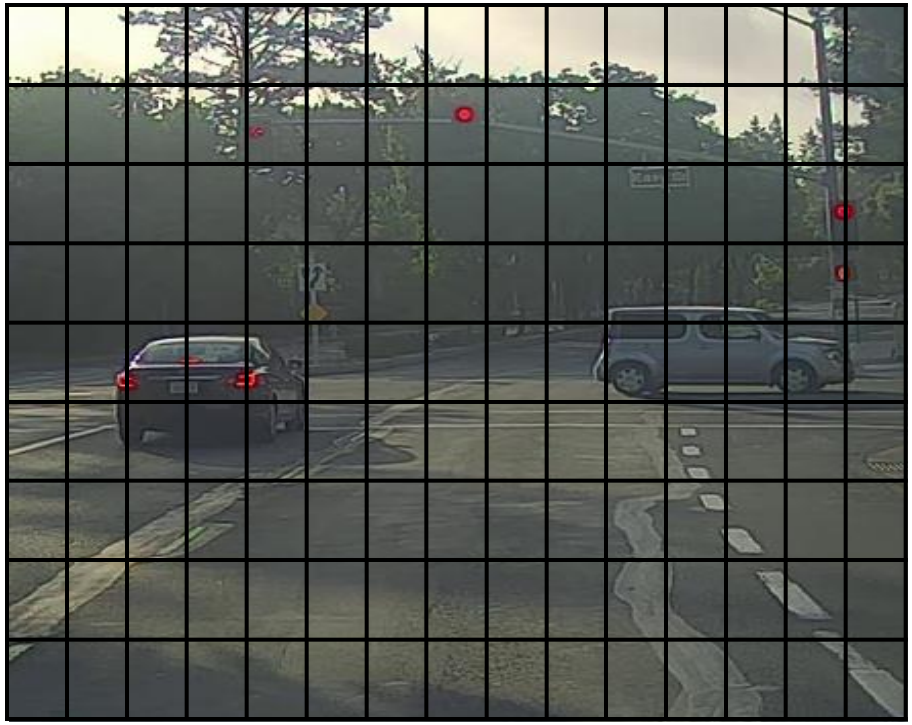
- Um dos problemas do algoritmo de detecção de caixas delimitadoras é que ele acha múltiplas caixas referentes ao mesmo objeto, ou seja, o mesmo objeto pode ser detectado inúmeras vezes em células diferentes.
- A supressão não máxima (SNM) é uma forma de eliminar caixas adicionais do mesmo objeto e, assim, fazer com que um objeto seja detectado por somente uma única caixa delimitadora.
- Supressão não máxima significa que são eliminadas as caixas delimitadoras que possuem probabilidade de existir um objeto de determinada classe menor do que a máxima.

Supressão não máxima

- Exemplo do algoritmo de supressão não máxima:
  - Dada uma imagem e as classes de objetos que se deseja detectar.

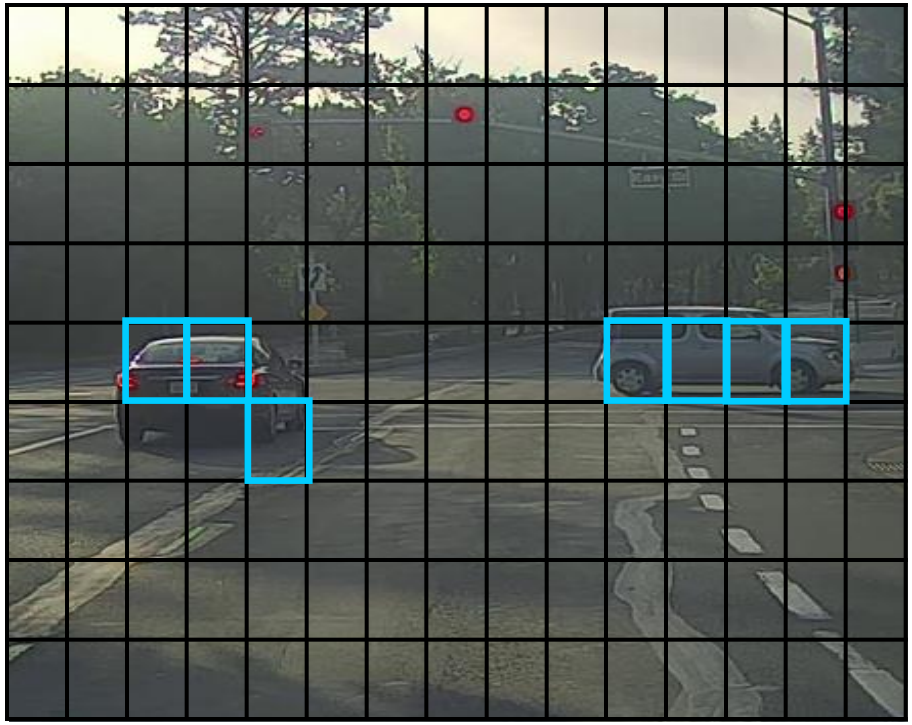
Supressão não máxima

- Exemplo do algoritmo de supressão não máxima:
  - Divide a imagem com uma malha, no caso da figura uma malha 19x19.



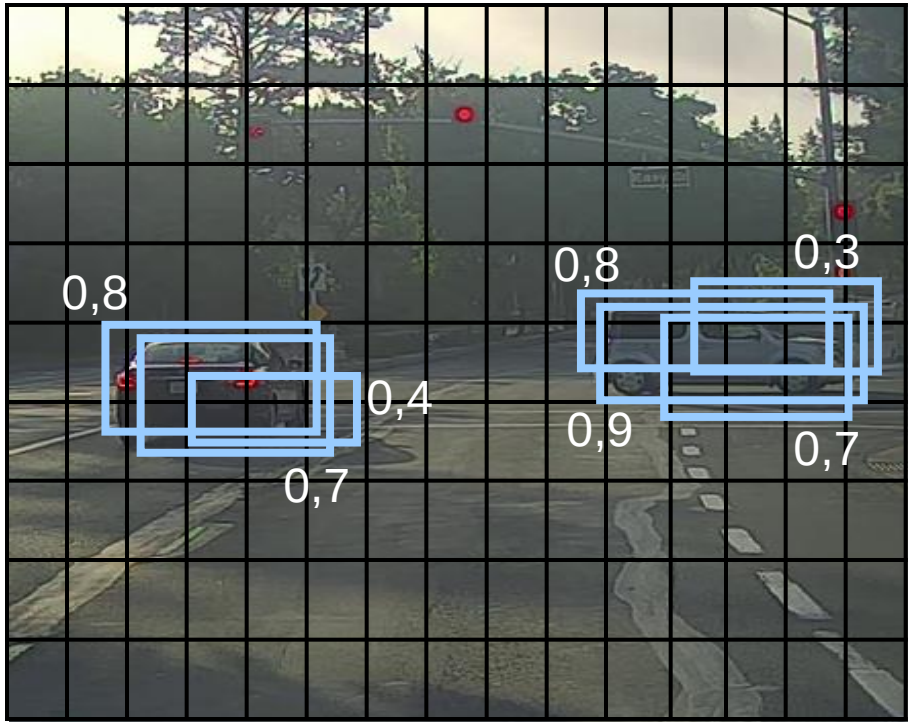
Supressão não máxima

- Exemplo do algoritmo de supressão não máxima:
  - Algoritmo de detecção de caixas detecta que existem carros em várias células (células marcadas em azul).



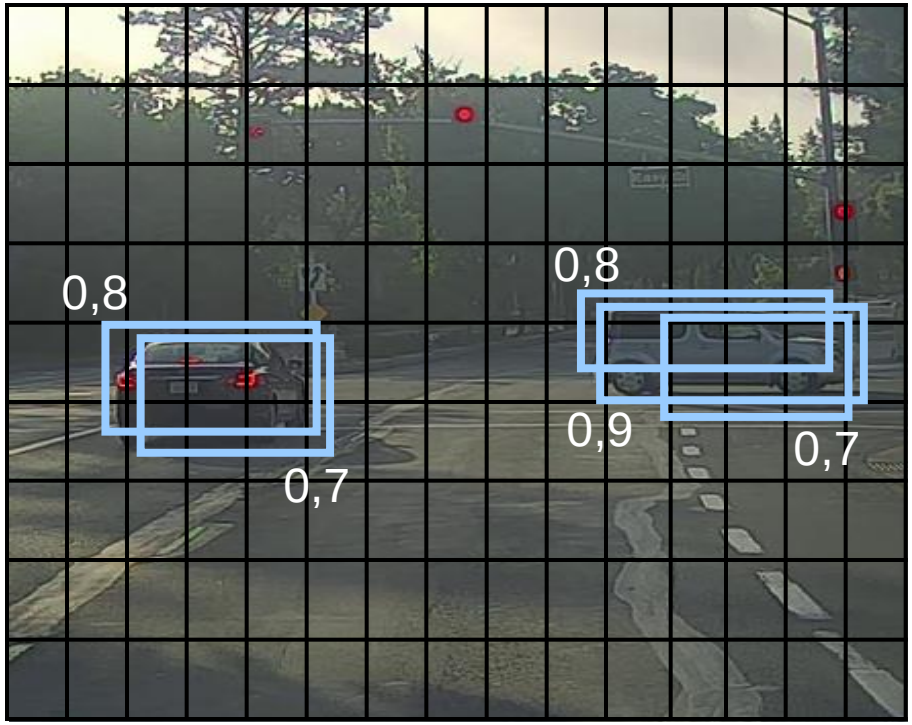
Supressão não máxima

- Exemplo do algoritmo de supressão não máxima:
  - Algumas caixas têm p_c alto e outras p_c baixo.



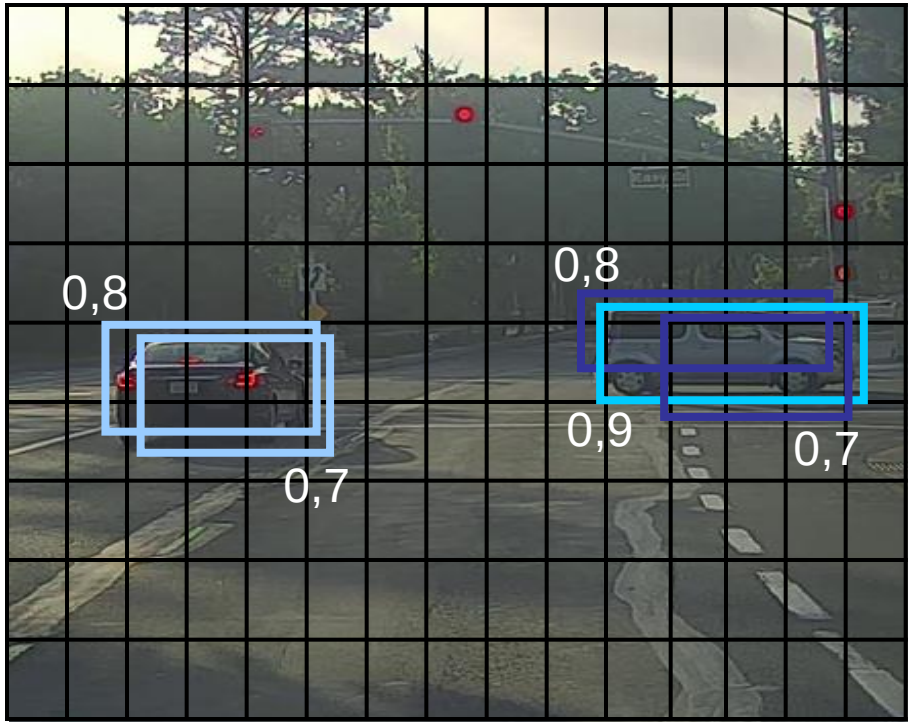
Supressão não máxima

- Exemplo do algoritmo de supressão não máxima:
  - Primeiramente eliminam-se todas as caixas com $p_c < 0,5$.



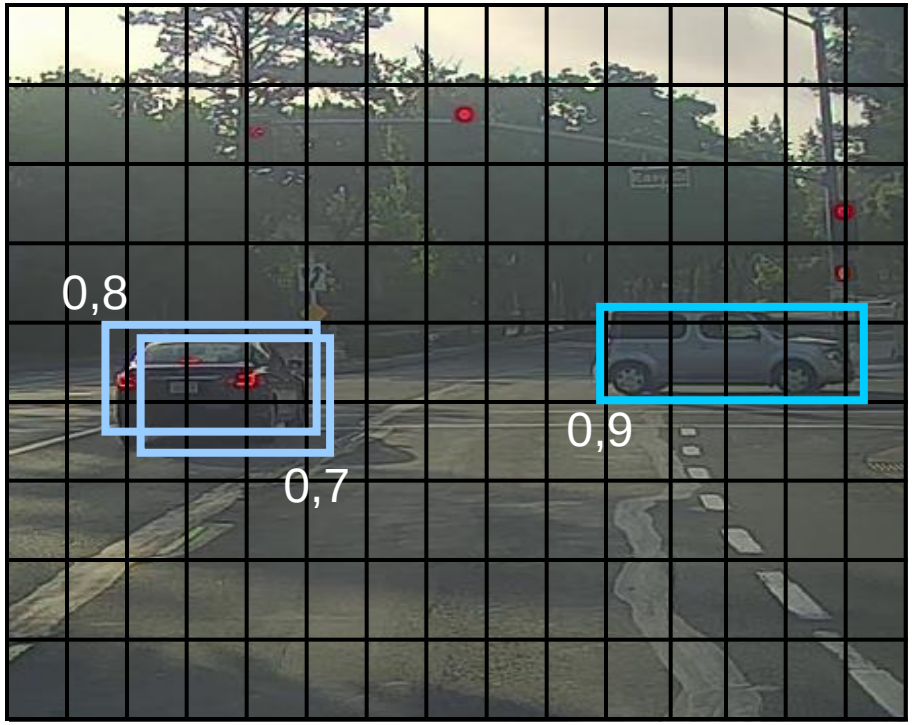
Supressão não máxima

- Exemplo do algoritmo de supressão não máxima:
  - Escolhe a caixa com maior p_c na imagem;
  - Calcula *IoV* de todas as caixas em relação à caixa de maior p_c.



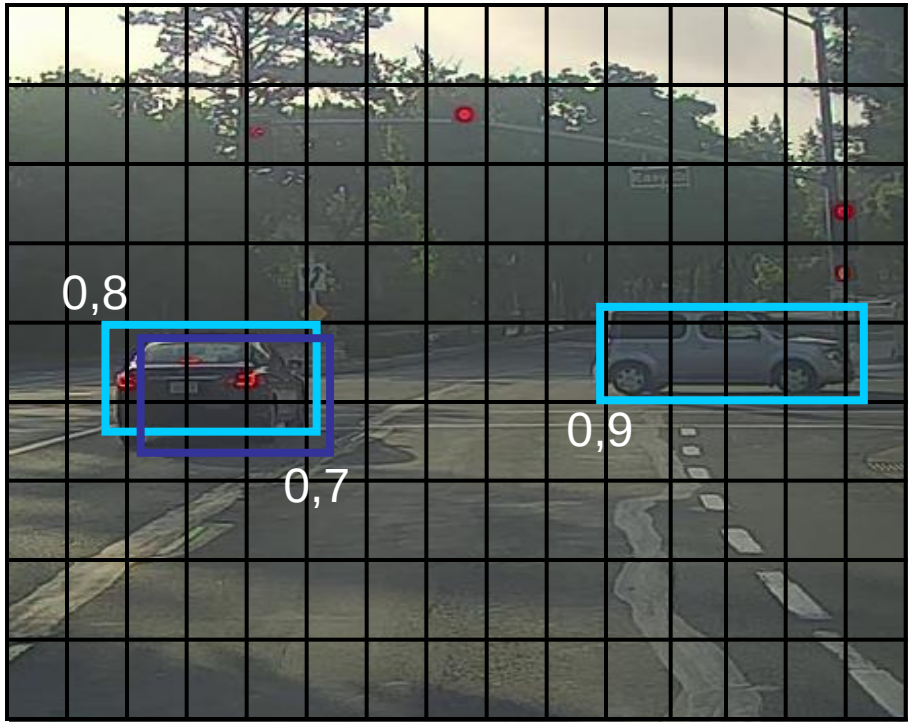
Supressão não máxima

- Exemplo do algoritmo de supressão não máxima:
  - Escolhe a caixa com maior p_c na imagem;
  - Calcula *IoV* de todas as caixas em relação à caixa de maior p_c;
  - As caixas que possuem *IoU* alto são descartadas.



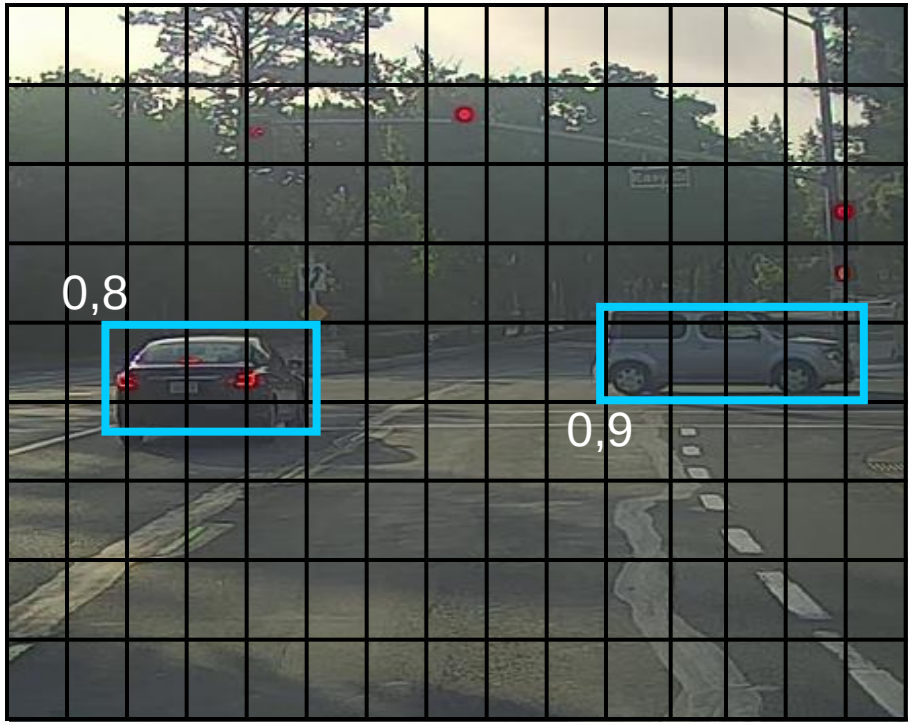
Supressão não máxima

- Exemplo do algoritmo de supressão não máxima:
  - Escolhe a caixa com maior p_c na imagem;
  - Calcula *IoV* de todas as caixas em relação à caixa de maior p_c;
  - As caixas que possuem *IoU* alto são descartadas;
  - Repete o processo para as caixas restantes, até não ter mais caixas para verificar.



Supressão não máxima

- Exemplo do algoritmo de supressão não máxima:
  - Escolhe a caixa com maior p_c na imagem;
  - Calcula *IoV* de todas as caixas em relação à caixa de maior p_c;
  - As caixas que possuem *IoU* alto são descartadas;
  - Repete o processo para as caixas restantes, até não ter caixas para verificar.



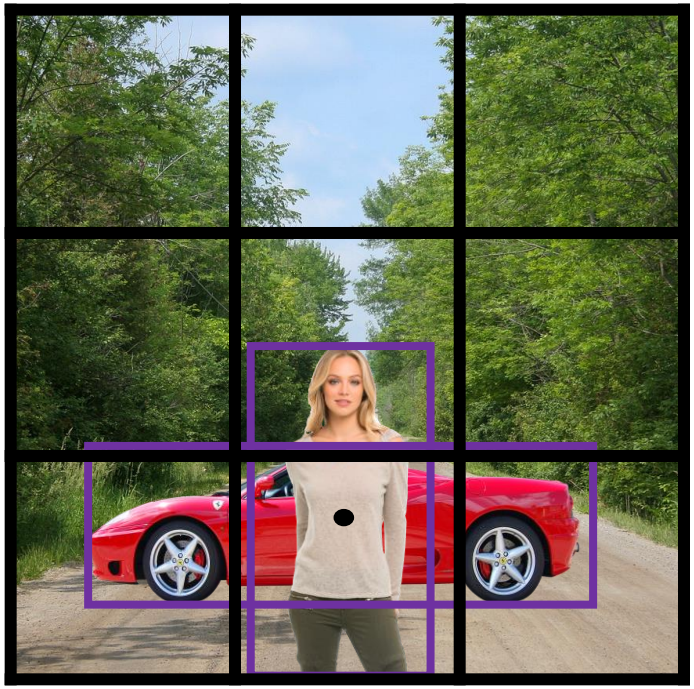
Supressão não máxima

- Algoritmo de supressão não máxima $\Rightarrow$ para cada classe de objeto realizar o seguinte procedimento:

  1. Descartar caixas delimitadoras com $p_c \leq 0{,}5$;
  2. Selecionar na imagem a caixa com maior p_c;
  3. Descartar qualquer caixa que tenha $IoU \geq 0{,}5$ em relação à caixa de maior p_c selecionada no passo 2;
  4. Enquanto existir caixas da mesma classe com $p_c > 0{,}5$ e $IoU < 0{,}5$ em relação às caixas já analisadas, repetir os passos 2 e 3.

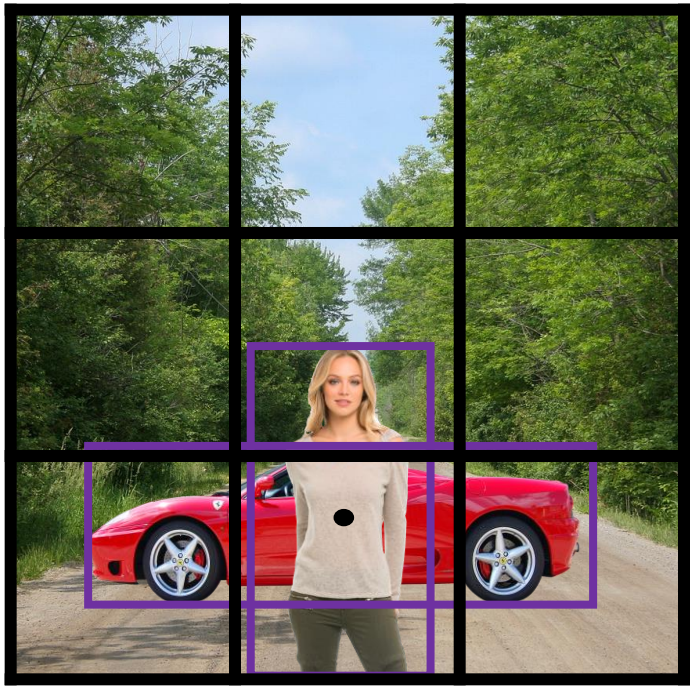
Caixas âncora

- Outro problema do algoritmo de detecção de caixas delimitadoras é que em cada célula da malha somente é possível detectar um único objeto.
- Na figura ao lado o ponto central da pessoa e do carro estão na mesma célula e quase na mesma posição.
- No algoritmo anterior de detecção de caixas delimitadores, como se tem somente uma caixa delimitadora, devemos escolher um dos objetos e ignorar o outro.



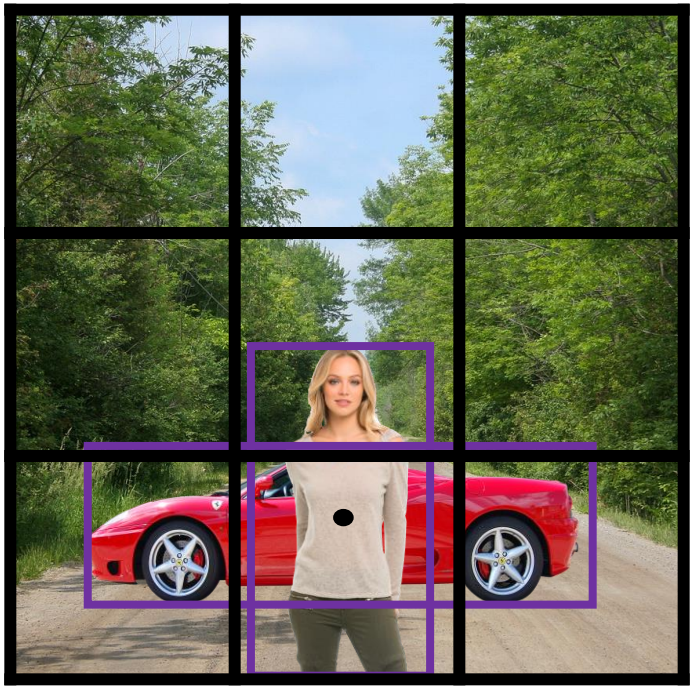
Caixas âncora

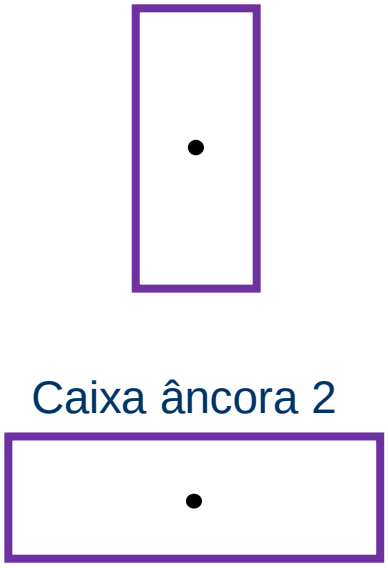
- Para detectar múltiplos objetos em uma célula ⇒ usam-se caixas âncoras (CA).
- A idéia é definir mais de uma caixa delimitadora para cada célula.
- Com mais de uma caixa pode-se detectar mais de um objeto em uma célula ⇒ um para cada caixa delimitadora.
- Cada caixa âncora em uma célula pode detectar um objeto diferente.



Caixas âncora

- Exemplo de detecção de objeto usando duas caixas âncoras em cada célula.
  - Saída da RNA para cada célula em um problema com 3 classes:





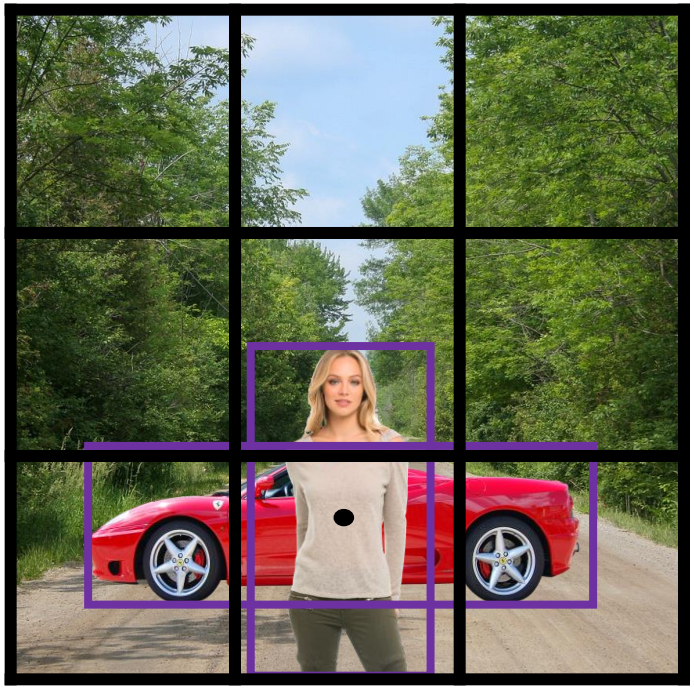
$$\mathbf{y} = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_H \\ b_W \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_H \\ b_W \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

Caixa âncora 1

Caixa âncora 2

Caixas âncora

- Exemplo de detecção de objeto, com 3 classes, usando 2 caixas âncoras em cada célula, com uma malha 3x3:
  - A saída da RNA é definida de forma que cada objeto na imagem é posicionado na célula onde fica o seu centro;
  - Cada objeto é associado com uma caixa âncora na célula onde está posicionado;
  - Cada caixa âncora em cada célula tem 8 elementos;
  - A saída de cada célula é um vetor de dimensão 16x1 (2 caixas em cada célula);
  - Saída da RNA $\Rightarrow$ tensor 3x3x16 (malha 3x3).



Caixas âncora

- As caixas âncoras são detectadas por uma RNA totalmente convolutional.
- ➢ RNA não possui nenhuma camada densa.
- Entrada da RNA $\Rightarrow$ imagem de dimensão (n_H, n_W, 3);
  - n_H = altura da imagem;
  - n_W = largura da imagem.
- Saída da RNA $\Rightarrow$ tensor de dimensão (n_{GH}, n_{GW}, N_A*(5+N_C));
  - n_{GH} x n_{GW} = tamanho da malha;
  - N_C = número de classes do problema;
  - N_A = número de caixas âncoras em cada célula.
- RNA convolucional deve ser configurada para receber uma imagem de entrada de dimensão (n_H, n_W, 3) e gerar na sua saída um tensor de dimensão (n_{GH}, n_{GW}, N_A*(5+N_C)).

Caixas âncora

- RNA convolucional para detecção de caixas âncoras:

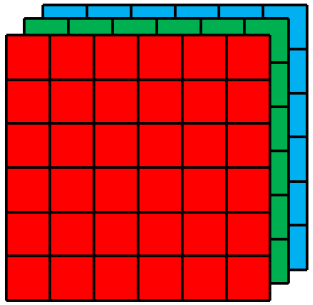
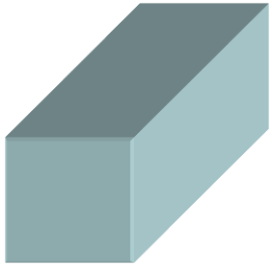


Imagem
$n_H x n_W x 3$

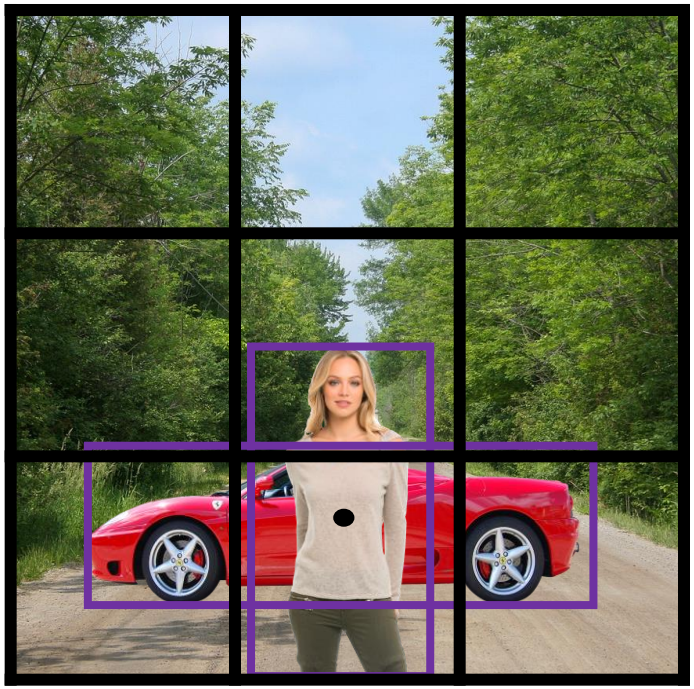
Camadas
convolucionais,
de pooling e
de normalização de
batelada



Saída
$n_{GH} x n_{GW} x N_A * (5+N_C)$

Caixas âncora

- Exemplo detecção de objeto, com 3 classes, usando 2 caixas âncoras em cada célula, com uma malha 3x3.



Saída para célula sem objeto ⇨

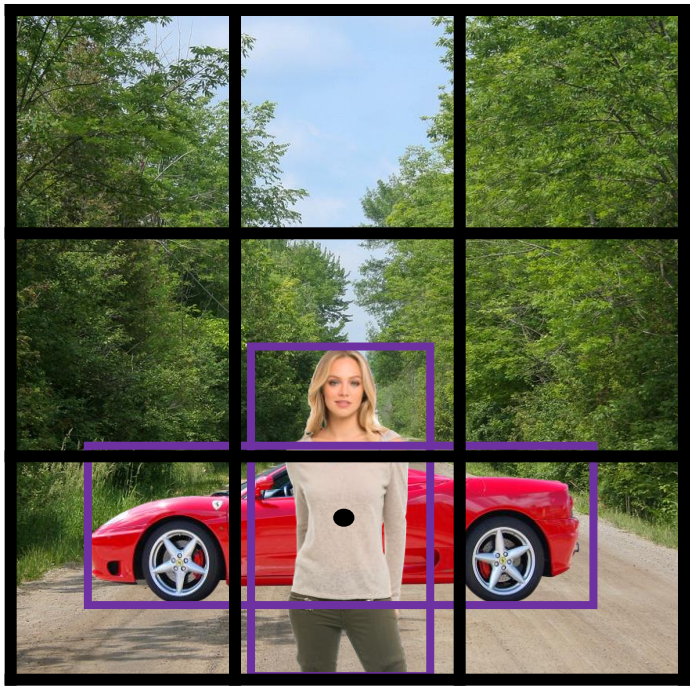
$$\mathbf{y} = \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

Caixa âncora 1

Caixa âncora 2

Caixas âncora

- Exemplo detecção de objeto, com 3 classes, usando 2 caixas âncoras em cada célula, com uma malha 3x3.



Saída para célula com pedestre e carro

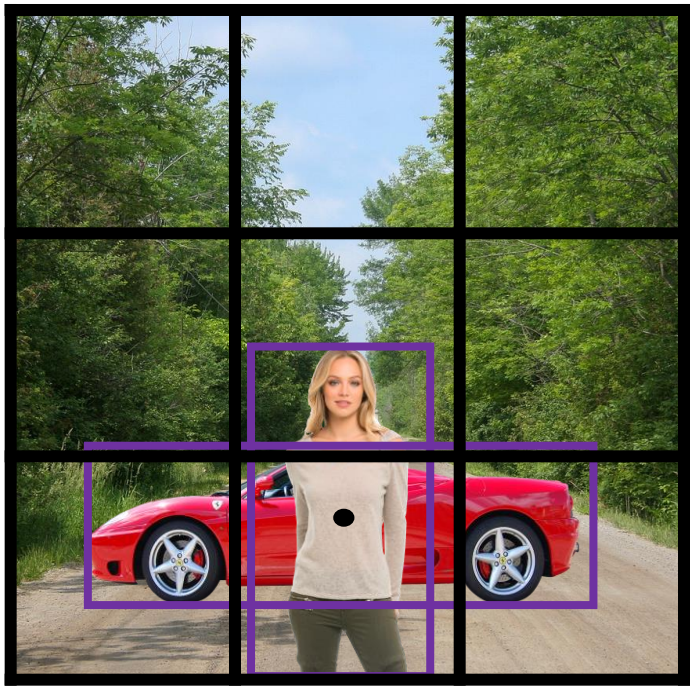
$$\mathbf{y} = \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_H \\ b_W \\ 1 \\ 0 \\ 0 \\ 1 \\ b_x \\ b_y \\ b_H \\ b_W \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Caixa âncora 1

Caixa âncora 2

Caixas âncora

- Exemplo detecção de objeto, com 3 classes, usando 2 caixas âncoras em cada célula, com uma malha 3x3.



Saída para célula somente com pedestre

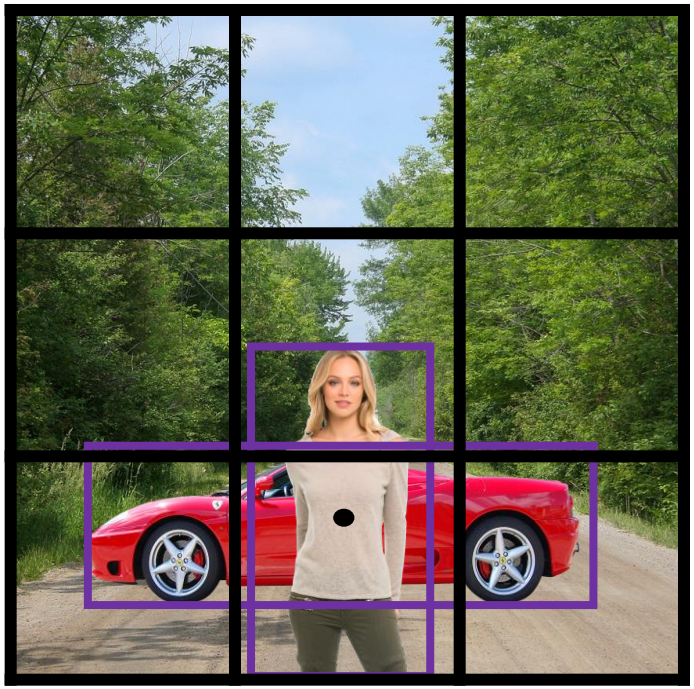
$$\mathbf{y} = \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_H \\ b_W \\ 1 \\ 0 \\ 0 \\ 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

Caixa âncora 1

Caixa âncora 2

Caixas âncora

- Exemplo detecção de objeto, com 3 classes, usando 2 caixas âncoras em cada célula, com uma malha 3x3.



Saída para célula somente com carro

$$\mathbf{y} = \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ 1 \\ b_x \\ b_y \\ b_H \\ b_W \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Caixa âncora 1

Caixa âncora 2

Caixas âncora

- Observações:
  - Se existirem mais objetos do que caixas âncoras numa mesma célula o algoritmo não funciona direito;
  - Pode-se ter mais do que 2 caixas âncoras em cada célula $\Rightarrow$ comum usar 5 caixas.
  - Se existirem 2 objetos da mesma classe com caixas âncoras de mesmo formato, o algoritmo não funciona direito;
  - Aumentar número de células diminui a probabilidade de dois objetos da mesma classe estarem na mesma célula;
  - Pode-se usar caixas âncoras especializadas com formato (relação altura/largura) pré-definidos, por exemplo, alto e magro, largo e baixo, etc.

YOLO

- O melhor método para detectar objetos que existe atualmente é a YOLO ⇒ You Only Look Once.
- Referência ⇒ Redmon et al., 2015, You Only Look Once: Unified real-time object detection.
- A YOLO é capaz de ser executado em tempo real em hardware embarcado, tal como a raspberry pi.
- Existem diversas versões da YOLO:
  - A Tiny YOLO é uma versão mais leve específica para aplicações de tempo real com hardware embarcado.
- A YOLO integra os diversos componentes de detecção de objetos.

YOLO

- Algoritmo da YOLO:
  1. Detecção de caixas âncoras usando uma RNA convolucional:
     - Nessa etapa muitas caixas são detectadas na imagem, sendo que a maioria não representa de fato um objeto e muitas são repetições do mesmo objeto;
     - Se forem utilizadas, por exemplo, 3 caixas âncoras por célula da malha, então em princípio são detectados 3 objetos por célula, mesmo que eles de fato não existam.
  2. Execução do algoritmo de supressão não máxima para eliminar caixas âncoras que não representam objetos e caixas que correspondem ao mesmo objeto $\Rightarrow$ essa etapa é realizada após a execução da RNA convolucional (não faz parte da RNA).

YOLO

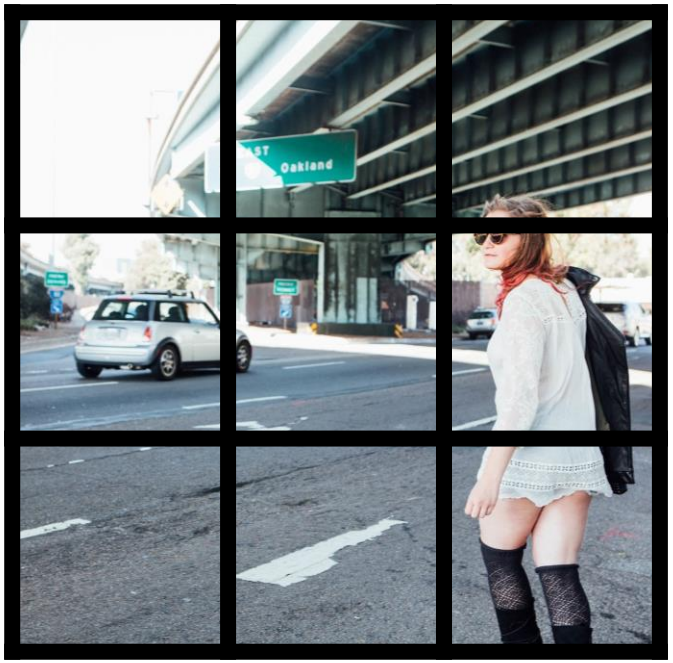
- Exemplo do algoritmo da YOLO:

YOLO

- Exemplo do algoritmo da YOLO:
  - Parâmetros: malha 3x3, com 2 caixas âncoras por célula;

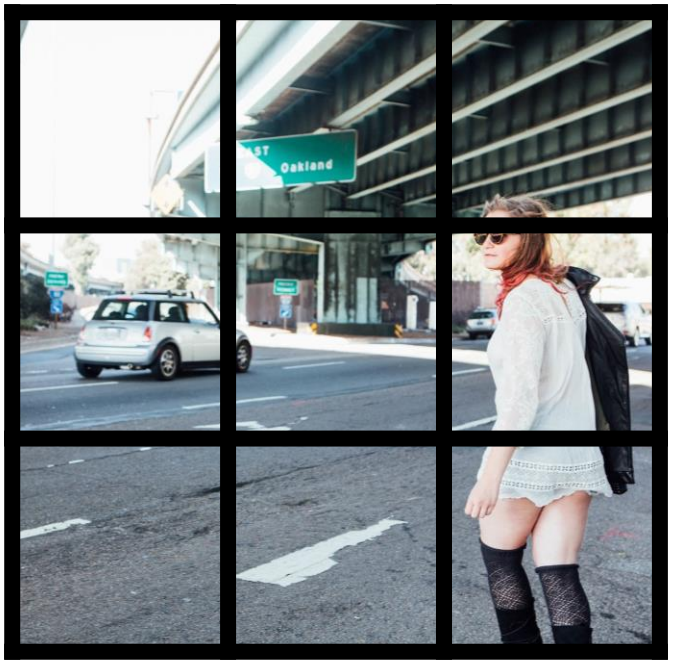
YOLO

- Exemplo do algoritmo da YOLO:
  - Parâmetros: malha 3x3, com 2 caixas âncoras por célula;



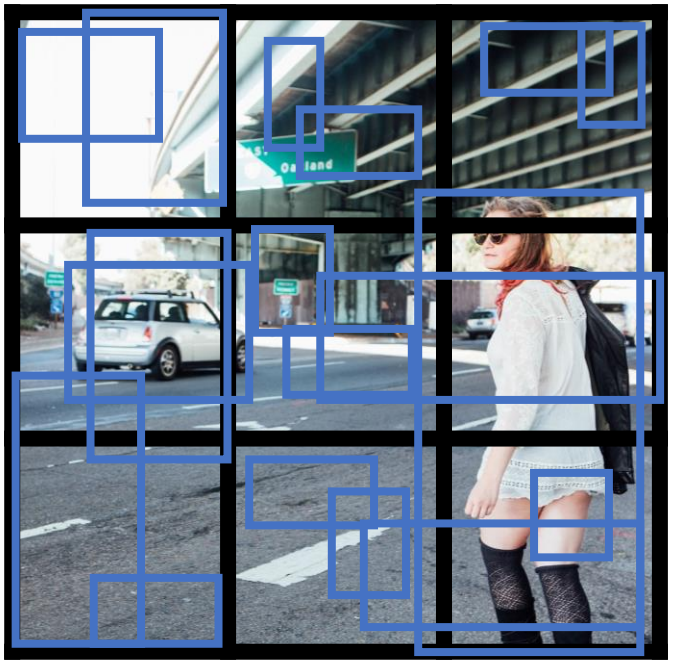
YOLO

- Exemplo do algoritmo da YOLO:
  - Parâmetros: malha 3x3, com 2 caixas âncoras por célula;
  - Executar a RNA convolucional para detectar as caixas âncoras ⇒ são detectadas 2 caixas por célula mesmo que não exista objeto na célula;



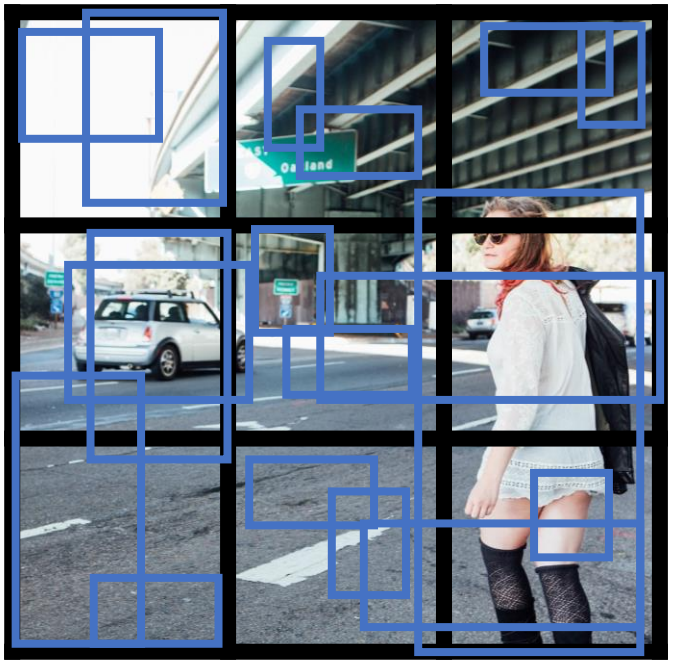
YOLO

- Exemplo do algoritmo da YOLO:
  - Parâmetros: malha 3x3, com 2 caixas âncoras por célula;
  - Executar a RNA convolucional para detectar as caixas âncoras ⇒ são detectadas 2 caixas por célula mesmo que não exista objeto na célula;



YOLO

- Exemplo do algoritmo da YOLO:
  - Parâmetros: malha 3x3, com 2 caixas âncoras por célula;
  - Executar a RNA convolucional para detectar as caixas âncoras $\Rightarrow$ são detectadas 2 caixas por célula mesmo que não exista objeto na célula;
  - Eliminar as caixas com baixa probabilidade de possuir objeto ($p_c < 0{,}5$);



YOLO

- Exemplo do algoritmo da YOLO:
  - Parâmetros: malha 3x3, com 2 caixas âncoras por célula;
  - Executar a RNA convolucional para detectar as caixas âncoras $\Rightarrow$ são detectadas 2 caixas por célula mesmo que não exista objeto na célula;
  - Eliminar as caixas com baixa probabilidade de possuir objeto ($p_c < 0{,}5$);

YOLO

- Exemplo do algoritmo da YOLO:
  - Parâmetros: malha 3x3, com 2 caixas âncoras por célula;
  - Executar a RNA convolucional para detectar as caixas âncoras $\Rightarrow$ são detectadas 2 caixas por célula mesmo que não exista objeto na célula;
  - Eliminar as caixas com baixa probabilidade de possuir objeto ($p_c < 0{,}5$);
  - Para cada classe que se quer detectar usar o algoritmo de supressão não máxima para eliminar caixas repetidas.

YOLO

- Exemplo do algoritmo da YOLO:
  - Parâmetros: malha 3x3, com 2 caixas âncoras por célula;
  - Executar a RNA convolucional para detectar as caixas âncoras $\Rightarrow$ são detectadas 2 caixas por célula mesmo que não exista objeto na célula;
  - Eliminar as caixas com baixa probabilidade de possuir objeto ($p_c < 0{,}5$);
  - Para cada classe que se quer detectar usar o algoritmo de supressão não máxima para eliminar caixas repetidas.