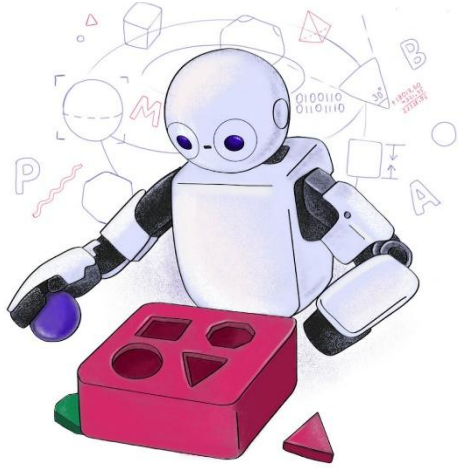
TP558 - Tópicos avançados em Machine Learning:

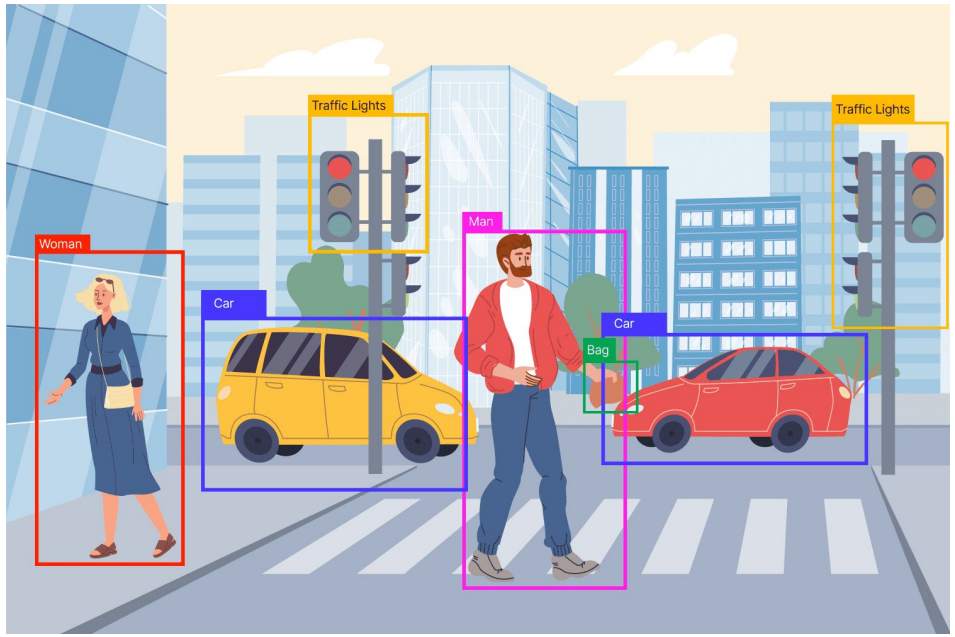
RetinaNet



Inatel

Pedro Márcio Raposo Pereira
pedro.marcio@inatel.br

Introdução



A detecção de objetos é essencial em diversas áreas, como direção autônoma, vigilância e robótica.

- Em direção autônoma, é crucial para identificar e rastrear veículos, pedestres e obstáculos.
- Em sistemas de segurança, detecta objetos anômalos, como intrusos ou dispositivos explosivos.

Introdução

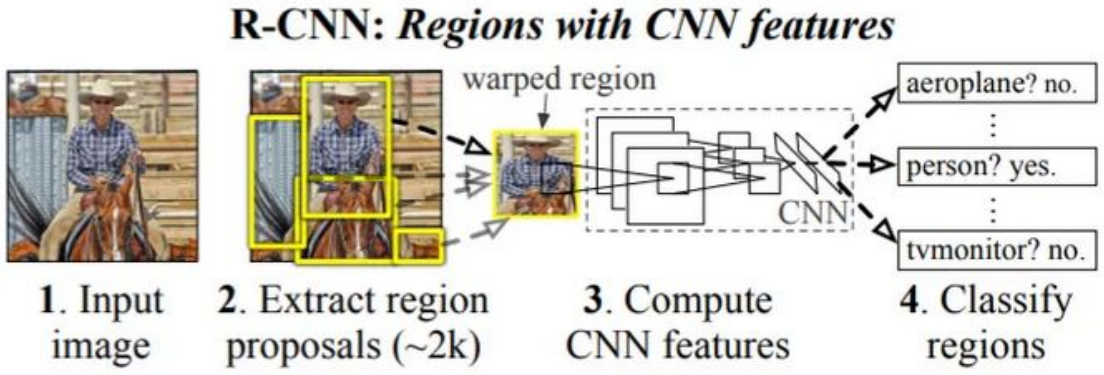
Uma abordagem comum é o uso de caixas delimitadoras para localizar objetos na imagem.

- As caixas delimitadoras fornecem uma descrição precisa da localização dos objetos na imagem.
- Podem ser definidas por coordenadas dos cantos ou do centro, largura e altura.
- Essa representação é amplamente adotada devido à sua simplicidade e eficácia na detecção de objetos.

Detectores de Estágio Duplo

- Os Detectores de Estágio Duplo são uma evolução da arquitetura R-CNN, dividindo o modelo em duas etapas distintas.
- Na primeira etapa, são extraídas as regiões que podem conter objetos de interesse.
- Em seguida, um segundo modelo é usado para classificar essas regiões e refinar sua localização.
- Embora essa abordagem resulte em maior latência, geralmente oferece maior precisão.

Pipeline do Detector de Estágio Duplo



- O pipeline do algoritmo de dois estágios é composto por três etapas distintas.
- Geração de Propostas de Região: O modelo gera candidatos a objetos na imagem, independentemente de categorias.
- Extração de Características: Uma rede neural convolucional calcula características para cada região candidata identificada no estágio anterior.
- Classificação e Refinamento: Uma camada totalmente conectada, classifica e refina as regiões candidatas com base nas características extraídas.

Método de Geração de Propostas de Região

- As propostas de região podem ser geradas por diversos métodos, sendo a busca seletiva utilizada no artigo original.
- Um algoritmo de segmentação é aplicado à imagem original para gerar um mapa de segmentação.
- Com base nesse mapa, são desenhadas propostas de região, ou caixas delimitadoras, que representam regiões candidatas onde objetos podem estar presentes.
- O mapa de segmentação é mesclado iterativamente para gerar propostas de região cada vez mais refinadas, contribuindo para uma melhor cobertura dos objetos na imagem.

Método de Geração de Propostas de Região

Detectores de Estágio Único

- Detectores de Estágio Único são uma classe de arquiteturas de detecção de objetos que operam em um único estágio.
- Diferentemente dos Detectores de Estágio Duplo, esses modelos abordam a detecção de objetos como um problema de regressão simples.
- A imagem de entrada é alimentada diretamente na rede neural, que produz as probabilidades de classe e as coordenadas das caixas delimitadoras, sem a necessidade de uma etapa intermediária de proposição de região.

Eliminação da Proposição de Região

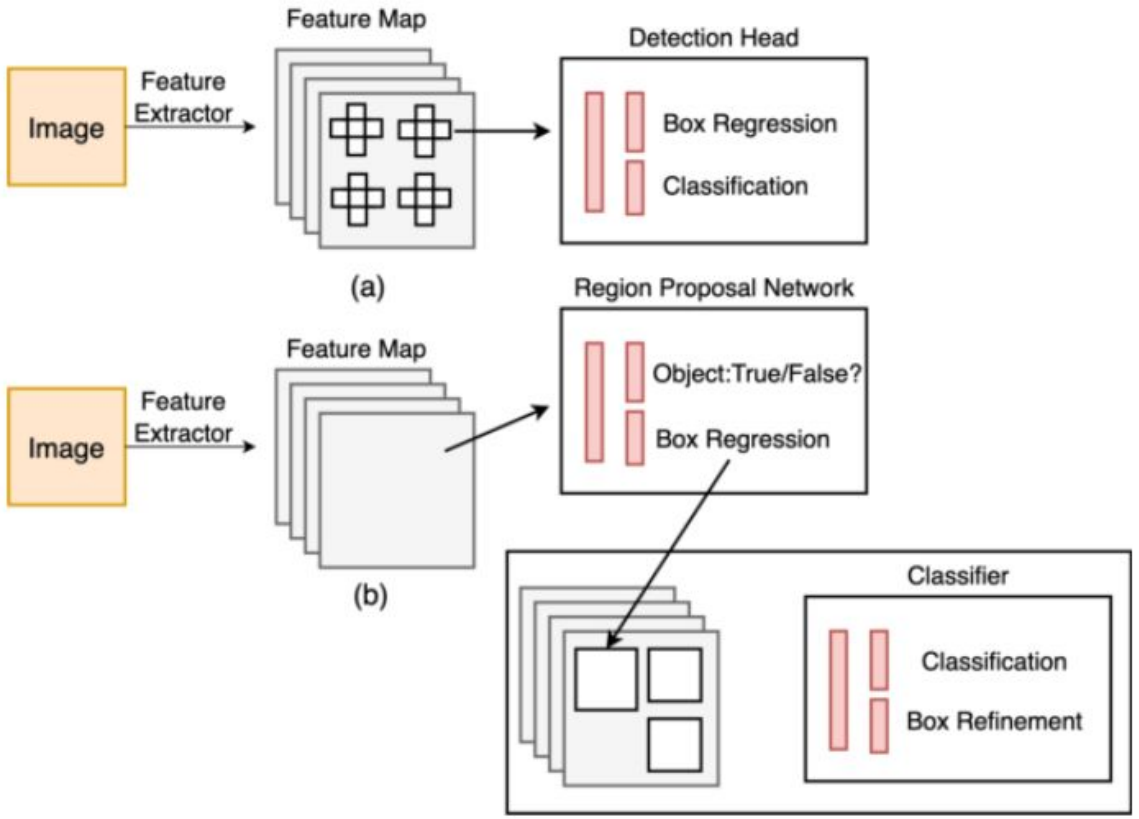
- Detectores de Estágio Único dispensam a etapa de proposição de região, comumente encontrada nos Detectores de Estágio Duplo.
- A proposição de região é responsável por identificar áreas na imagem que possivelmente contêm objetos de interesse.
- Ao eliminar essa etapa, os Detectores de Estágio Único simplificam o processo de detecção de objetos, oferecendo uma abordagem mais direta e eficiente.

Vantagens dos Detectores de Estágio Único

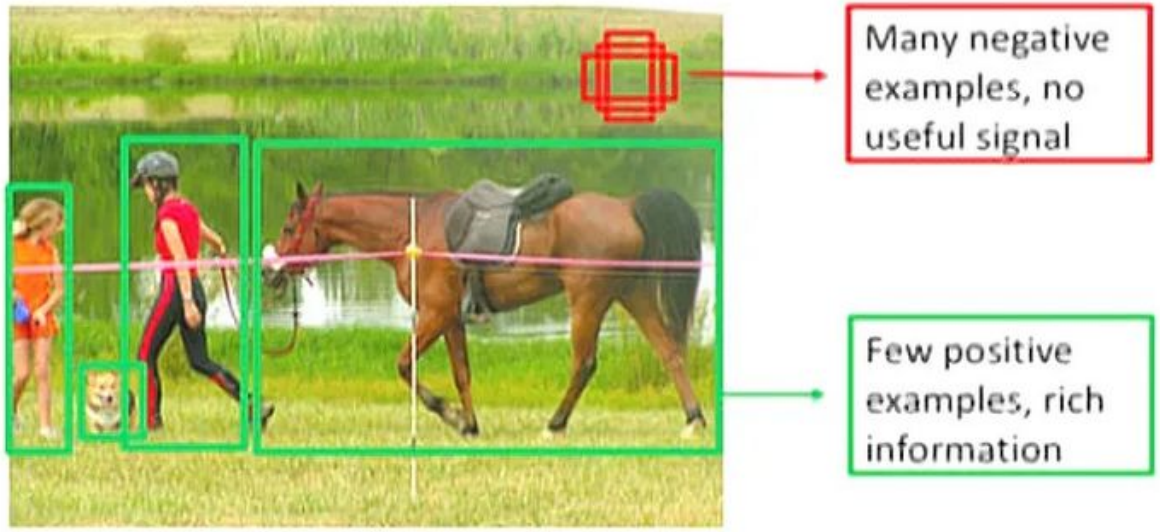
Simplificação do processo: Ao dispensar a proposição de região, os Detectores de Estágio Único simplificam o fluxo de detecção de objetos, resultando assim em menor latência (tempo de inferência).

Eficiência: Esses modelos geralmente oferecem uma abordagem mais eficiente para a detecção de objetos, uma vez que não precisam passar por uma etapa intermediária de geração de regiões candidatas.

Estágio Único vs. Estágio Duplo



Problema do Desbalanceamento de Classes



- Detectores de objetos baseados em uma abordagem de dois estágios têm sido os mais precisos até o momento.
- Detectores de um estágio, embora mais simples e rápidos, geralmente ficam atrás em termos de precisão.
- O desequilíbrio extremo de classes entre objetos e fundo é a principal causa dessa disparidade.

Problema do Desbalanceamento de Classes

- O desbalanceamento de conjuntos de dados é um problema comum em tarefas de classificação de aprendizado de máquina.
- Aplicar diretamente conjuntos de treinamento desbalanceados pode resultar em boa precisão geral, mas a precisão para classes minoritárias pode ser ruim devido à negligência durante o treinamento.

Problema do Desbalanceamento de Classes

- O desbalanceamento é ainda mais significativo em tarefas de detecção de objetos, onde algoritmos precisam propor regiões onde objetos podem estar presentes.
- Nos algoritmos R-CNN e Fast R-CNN, o número de regiões propostas é limitado intencionalmente, mas em modelos como Faster R-CNN, o número pode ser muito maior.

Problema do Desbalanceamento de Classes

- A maioria das regiões propostas são exemplos negativos, onde não há objeto presente.
- Esse desbalanceamento de classe deve ser abordado na detecção de objetos para garantir a eficácia do modelo.

O desbalanceamento de classes causa dois principais problemas durante o treinamento:

- Treinamento ineficiente devido à maioria dos locais serem negativos fáceis que não contribuem com aprendizado útil.
- Sobrecarga do treinamento e possíveis modelos degenerados devido à presença de negativos fáceis.

Como Resolver esse Problema?

Imagine o seguinte exemplo:

Em algumas competições na televisão, os jurados profissionais e toda a audiência votam para decidir quem é o vencedor. Como o número de espectadores é muito maior do que o número de jurados, devemos atribuir mais "peso" a estes votos. Pode ser que um voto de um jurado profissional seja equivalente a mil votos da audiência comum.

Como Resolver esse Problema?

Primeiramente, vamos discutir a perda de entropia cruzada e como ela se comporta em dados com classes desbalanceadas.

- Normalmente, usamos a função sigmoid para classificação binária e a função Softmax para classificação multiclasse para calcular a probabilidade da amostra ser de uma determinada classe.
- A função de perda usada, independentemente se é uma classificação binária ou multiclasse, é geralmente a perda de entropia cruzada.

Como Resolver esse Problema?

A expressão matemática para a versão discreta da entropia cruzada é:

$$H\left(p_i,\ q_i\right) = -\sum_{i=1}^{n} p_i \log\left(q_i\right)$$

em que p deve ser a probabilidade verdadeira da amostra pertencer à classe i e q deve ser a probabilidade inferida da amostra pertencer à classe i a partir da rede neural.

Para uma classificação binária a perda será:

$$L = -\left[y\log(p) + (1-y)\log(1-p)\right]$$

Perda Focal

- A Perda Focal foi desenvolvida para lidar com desequilíbrios extremos entre classes de primeiro plano e de fundo em cenários de detecção de objetos de um estágio (por exemplo, 1:1000).
- Funciona como uma alternativa eficaz às abordagens tradicionais, ajustando dinamicamente a contribuição das amostras durante o treinamento.

Perda Focal

- A perda focal é uma perda de entropia cruzada escalonada, onde o fator de escalonamento diminui conforme a confiança na classe correta aumenta.
- Intuitivamente, esse fator de escalonamento pode automaticamente reduzir a contribuição de exemplos fáceis durante o treinamento e focar rapidamente o modelo em exemplos difíceis.
- Experimentos mostram que a Perda Focal nos permite treinar um detector de um estágio de alta precisão que supera significativamente as alternativas de treinamento com as heurísticas de amostragem ou mineração de exemplos difíceis (técnicas anteriores de última para treinar detectores de um estágio).

Perda Focal

A perda focal é definida como:

$$FL(p) = -\alpha_i\left(1-p_i\right)^{\gamma}\log\left(p_i\right)$$

Onde, o parâmetro de focalização, γ, ajusta a taxa na qual os exemplos fáceis são reduzidos. Quanto maior o valor de γ, maior o efeito de modulação e

$$\alpha_i = \begin{cases} \alpha \ if \ \ y=1 \\ 1-\alpha \ caso \ contrário \end{cases}$$

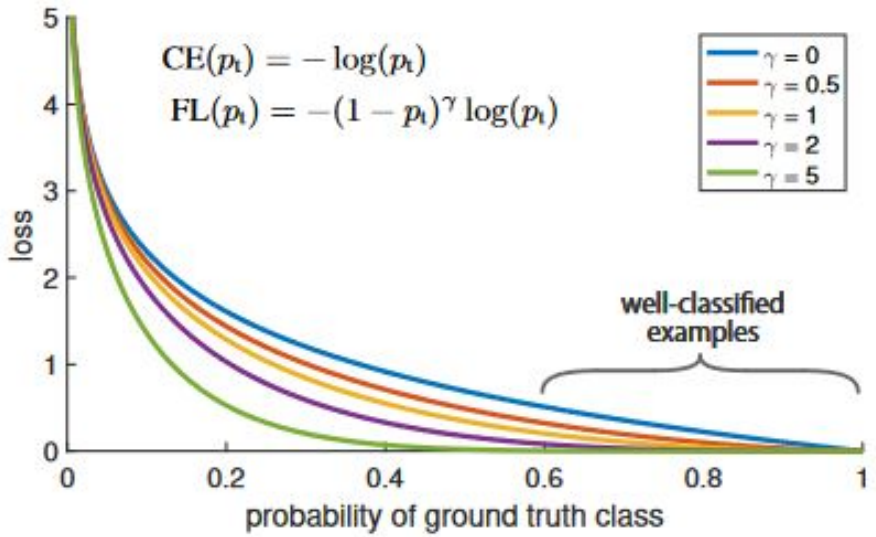
onde o parâmetro α é o fator de balanceamento entre 0 e 1.

Perda Focal

As duas propriedades da perda focal podem ser observadas como:

- Quando um exemplo é classificado incorretamente e p é pequeno, o fator de modulação está próximo de 1 e a perda não é afetada. Conforme $p \rightarrow 1$, o fator vai para 0 e a perda para exemplos bem classificados é reduzida.
- O parâmetro de focalização γ ajusta suavemente a taxa na qual os exemplos fáceis são reduzidos.Quando $\gamma = 0$, FL é equivalente a CE, e conforme γ é aumentado, o efeito do fator de modulação também é aumentado.

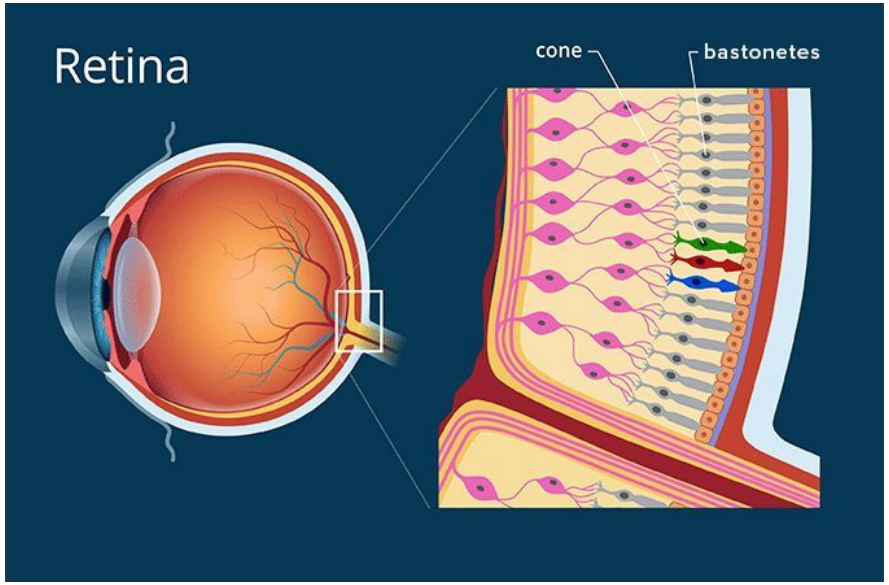
Perda Focal



Analisando as Propriedades Graficamente.

Intuitivamente, a Perda Focal atua reduzindo a contribuição de exemplos fáceis e aumentando a importância da correção de exemplos classificados incorretamente, resultando em modelos mais robustos e precisos.

RetinaNET



A primeira etapa no processo de visão ocorre na retina.

Nesse tecido sensível à luz na parte posterior do olho, a energia da luz é transformada em impulsos que podem ser interpretados pelo cérebro.

As células fotorreceptoras captam a luz **focada** pelo córnea.

Funcionamento do RPN (Region Proposal Network)

Cabeça de Regressão:

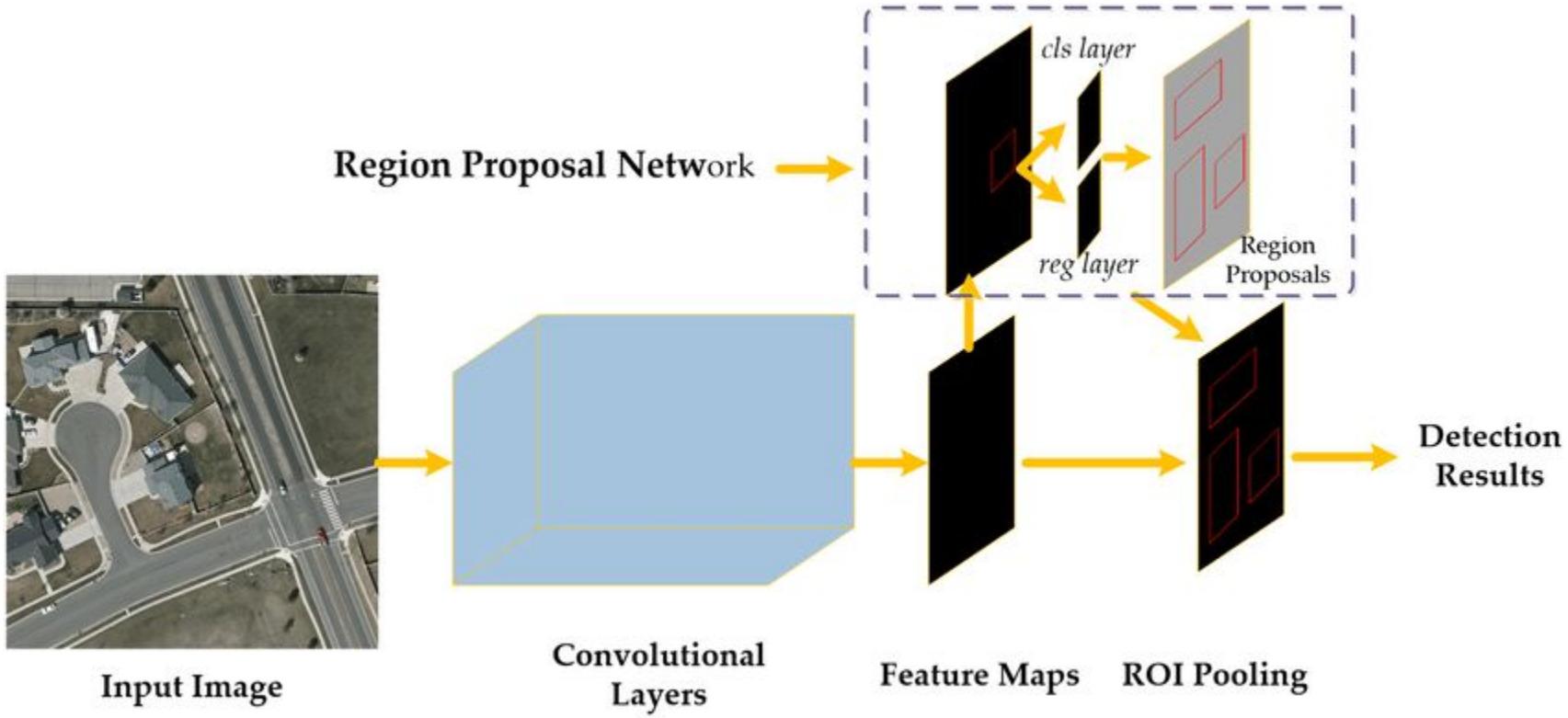
- A saída do RPN é um mapa de características, geralmente de 50x50.
- Uma camada convolucional percorre essa imagem, prevendo os valores de posição para cada caixa de ancoragem em cada localização.

Cabeça de Classificação:

- Similar à cabeça de regressão, prevê a probabilidade de um objeto estar presente ou não em cada localização para cada caixa de ancoragem.

Funcionamento do RPN

Arquitetura FASTER R-CNN



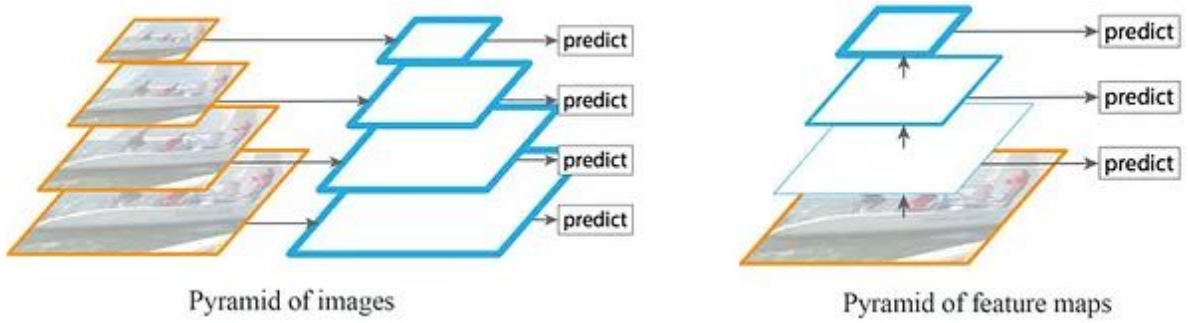
Problemas com o RPN:

- Perda de Informação Semântica: O mapa de características após subamostragem perde informação semântica em níveis baixos, dificultando a detecção de objetos pequenos. (Solução: Redes de Pirâmide de Características)
- Desbalanceamento de Classes: O uso de todas as amostras prejudica o treinamento devido ao desbalanceamento, onde haverá muitos exemplos facilmente classificados. (Solução: Perda Focal)

Redes de Pirâmide de Características

- No RPN, as caixas de ancoragem são construídas usando apenas o mapa de características de nível superior.
- ConvNets são robustas à variação de escala, porém, para melhorar a detecção em diferentes escalas, é necessário usar pirâmides de imagens.
- No processo de detecção de objetos em uma imagem de 800x800, por exemplo, é necessário considerar imagens em diferentes tamanhos, como 256x256, 300x300, 500x500 e 800x800.
- Para cada uma dessas imagens, calcula-se mapas de características correspondentes.
- No entanto, essa operação é custosa em termos computacionais.

Redes de Pirâmide de Características



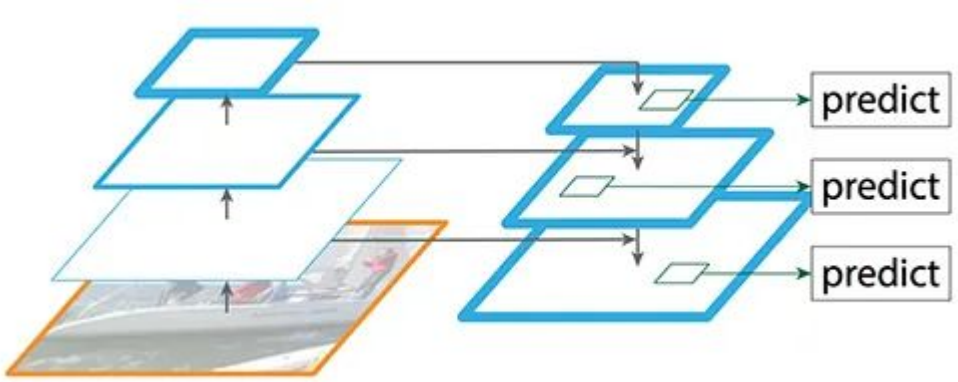
Alternativamente, pode-se usar uma pirâmide de características.

No entanto, os mapas de características mais próximos da camada de imagem são compostos por estruturas de baixo nível que não são eficazes para uma detecção precisa de objetos.

FPN é um extrator de características projetado para o conceito de pirâmide, levando em consideração a precisão e a velocidade.

FPN não é um detector de objetos por si só

Redes de Pirâmide de Características



Alternativamente, pode-se usar uma pirâmide de características.

No entanto, os mapas de características mais próximos da camada de imagem são compostos por estruturas de baixo nível que não são eficazes para uma detecção precisa de objetos.

FPN é um extrator de características projetado para o conceito de pirâmide, levando em consideração a precisão e a velocidade.

FPN não é um detector de objetos por si só

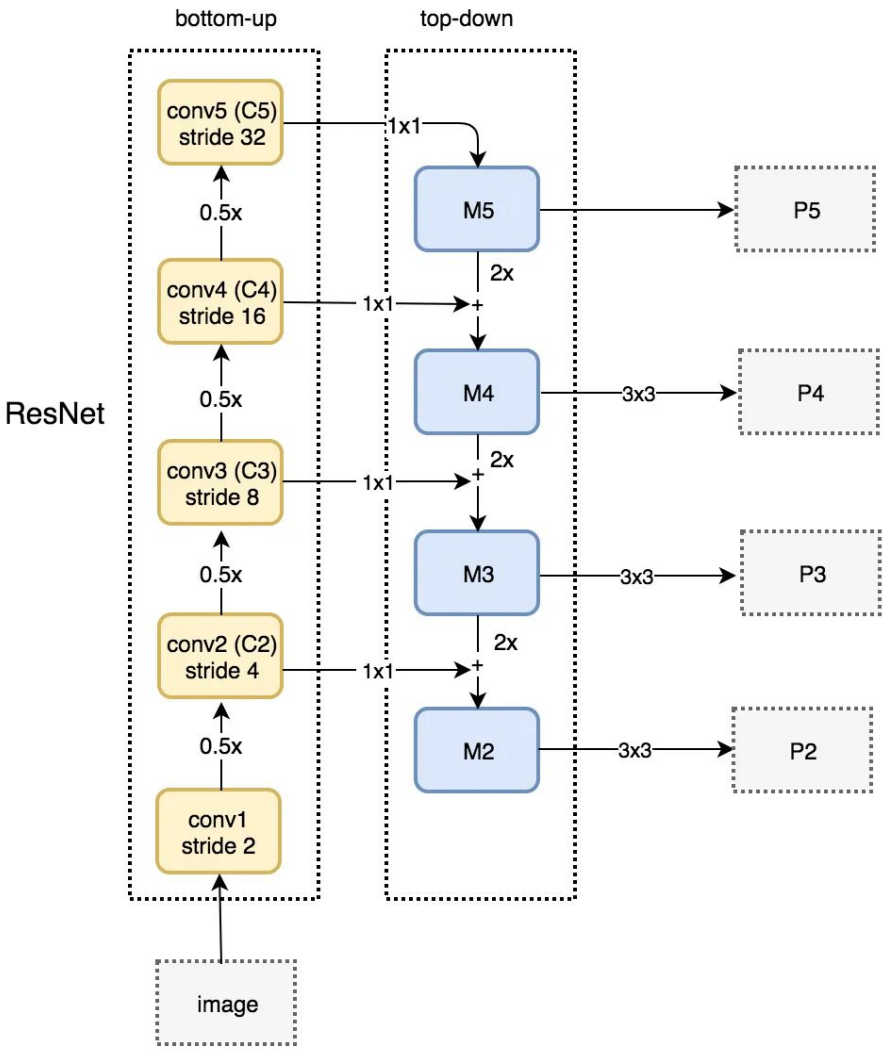
Fluxo de Dados

- O FPN é composto por dois caminhos:
  - Bottom-up (Baixo para cima)
  - Top-down (Cima para baixo)
- Caminho Bottom-up:
  - Utiliza a rede convolucional convencional para extração de características.
  - À medida que avançamos, a resolução espacial diminui.
  - Detecta estruturas de alto nível, aumentando o valor semântico para cada camada.

Fluxo de Dados

- Caminho Top-down:
  - Reconstrói a resolução espacial a partir de uma camada semanticamente rica.
  - Fornece um caminho de cima para baixo para construir camadas de resolução mais alta.
- Adição de Conexões Laterais:
  - As localizações dos objetos podem não ser precisas após downsampling e upsampling.
  - Conexões laterais entre as camadas reconstruídas e os mapas de características correspondentes ajudam a prever localizações com mais precisão.
  - Também funcionam como conexões residuais (similar ao ResNET).

Caminho de Baixo para Cima

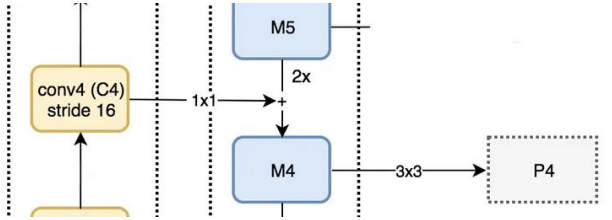


O caminho de baixo para cima emprega o ResNet para sua construção.

É composto por vários módulos de convolução.

Conforme avançamos, a dimensão espacial é reduzida pela metade.

Caminho de Cima para Baixo



Aplica-se um filtro de convolução 1 × 1 para reduzir a profundidade do canal C5 criando M5.

M5 é a primeira camada de mapa de características usada para a predição de objetos.

Ao descer o caminho de cima para baixo, aumentamos a camada anterior em 2 usando interpolação dos vizinhos mais próximos (nearest neighbors upsampling).

Aplica-se uma convolução 1 × 1 aos mapas de características correspondentes no caminho de baixo para cima.

Em seguida, é feita adição elemento a elemento (conexão residual).

Aplica-se uma convolução 3 × 3 a todos os as camadas adicionadas para reduzir o efeito de *aliasing*.

O processo para em P2 devido à grande dimensão espacial de C1, o que tornaria o processo muito lento.

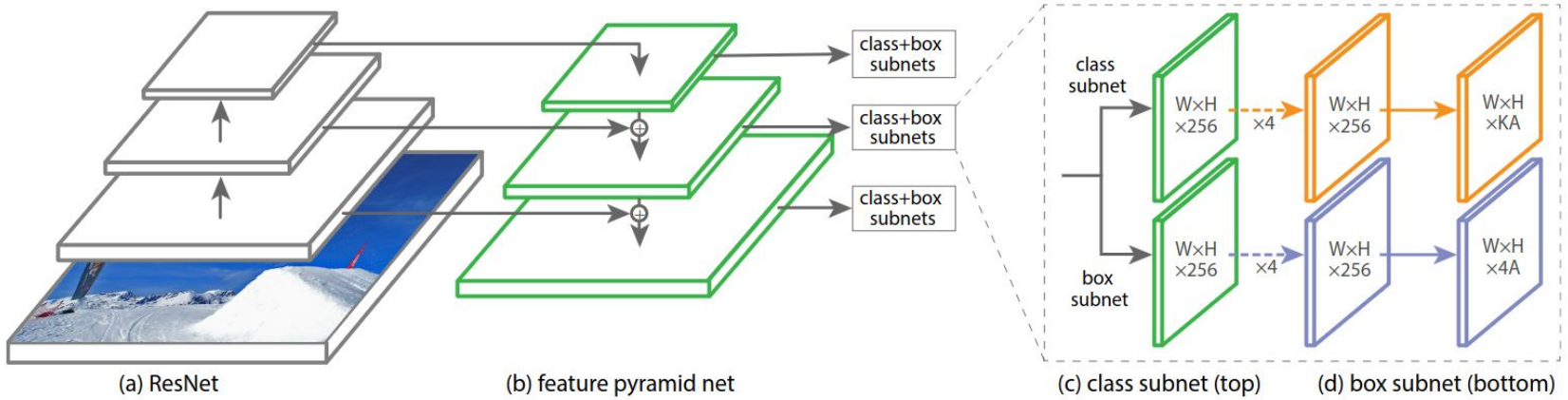
Todos os mapas de características da pirâmide (P5, P4, P3 e P2) têm canais de saída de 256 dimensões, compartilhando o mesmo classificador e regressor.

Há a possibilidade da inclusão de um nível P6 que pode ser obtido por *downsampling* de P5, porém não é utilizado na arquitetura Fast R-CNN.

RetinaNET

A arquitetura da rede RetinaNet de um estágio utiliza um *backbone* de Rede de Pirâmide de Características (b) sobre uma arquitetura ResNet (a).

Ao *backbone*, o RetinaNet anexa duas sub-redes, uma para classificar caixas de âncora (c) e outra para regressão das caixas de âncora para as caixas de objetos verdadeiras, *ground-truth* (d).



RetinaNET: *Backbone*

- A Rede de Pirâmide de Características (FPN) foi adotada como a espinha dorsal da rede RetinaNet.
- Constrói eficientemente uma pirâmide de características rica e multiescala a partir de uma única imagem de entrada com resolução.
- Cada nível da pirâmide pode ser usado para detectar objetos em uma escala diferente.
- Pirâmide construída com níveis P3 até P7.
- Todos os níveis da pirâmide têm C = 256 canais, seguindo a configuração de [2].

RetinaNET: *Backbone*

- O RetinaNet utiliza os níveis da pirâmide de características P3 a P7.
- P3 a P5 são computados a partir da saída do estágio residual ResNet correspondente (C3 a C5) usando conexões de cima para baixo e laterais.
- P6 é obtido através de uma convolução 3×3 com passo 2 em C5.
- P7 é calculado aplicando ReLU seguido de uma convolução 3×3 com passo 2 em P6.
- Diferenças em relação a [2]:
  - Não utilização do nível da pirâmide de alta resolução P2 por razões computacionais.
  - P6 é calculado por convolução com passo em vez de downsampling.
  - Inclusão de P7 para melhorar a detecção de objetos grandes.
- Essas modificações melhoram a velocidade mantendo a precisão.

RetinaNET: Sub-rede de Classificação

- A sub-rede de classificação é uma rede totalmente convolucional (FCN) conectada a cada nível da FPN.
- Consiste em quatro camadas convolucionais 3x3 com 256 filtros, seguidas por ativações ReLU.
- Em seguida, aplica-se outra camada convolucional 3x3 com KxA filtros, seguida por ativação sigmóide.
- A sub-rede tem parâmetros compartilhados em todos os níveis.
- A forma do mapa de características de saída seria (W, H, KA), onde W e H são proporcionais à largura e altura do mapa de características de entrada, K e A são os números de classes de objetos e caixas âncora.

RetinaNET: Sub-rede de Regressão

- A sub-rede de regressão está ligada a cada mapa de características da FPN em paralelo com a sub-rede de classificação.
- O design da sub-rede de regressão é idêntico ao da sub-rede de classificação, exceto que a última camada convolucional é 3x3 com 4A filtros.
- Portanto, a forma do mapa de características de saída seria (W, H, 4A).

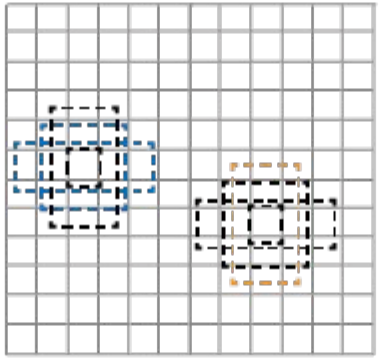
Caixas de Ancoragem.

Tanto a sub-rede de classificação quanto à sub-rede de regressão têm mapas de características de saída com largura W e altura H.

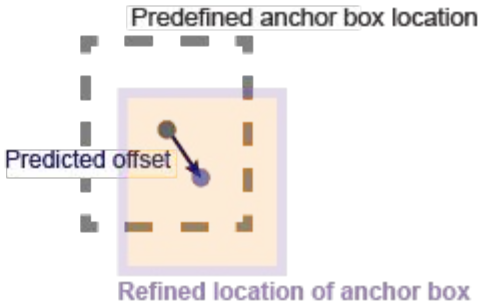
cada um dos elementos de WxH corresponde a uma região na imagem de entrada

Por que a sub-rede de classificação produz KA canais enquanto a sub-rede de regressão produz 4A canais, e a que correspondem esses canais, respectivamente?

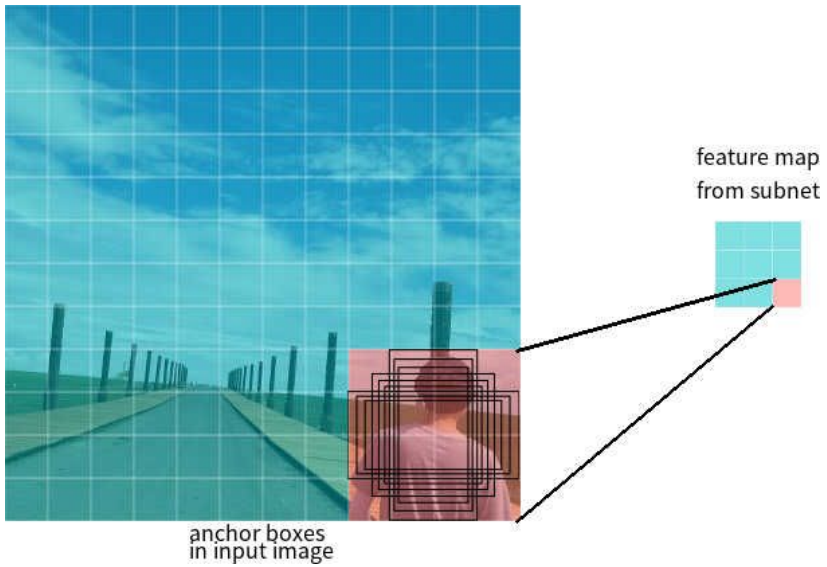
Ground truth image and bounding boxes



Anchor boxes at each predefined location in each feature map

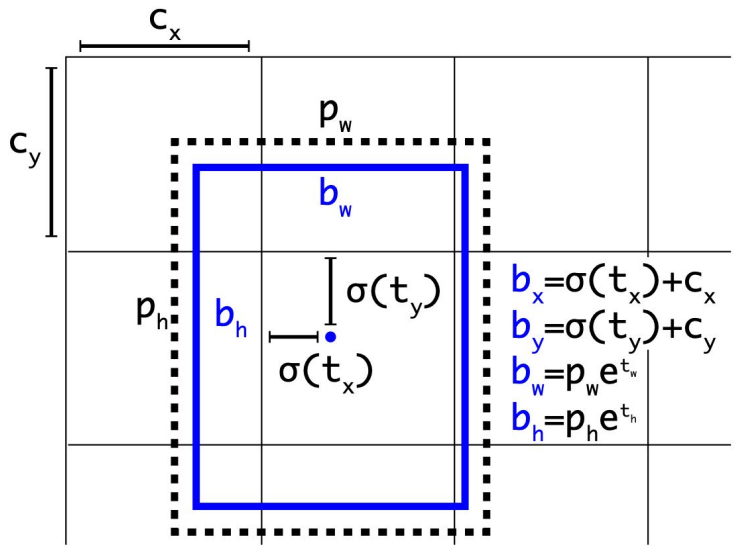


Caixas de Ancoragem



- RetinaNet define A=9 caixas chamadas caixas âncora, cada uma com tamanhos e proporções diferentes.
- Cada caixa âncora é responsável por detectar a existência de objetos de K classes na área que ela cobre.
- E como há A caixas delimitadoras, o mapa de características de saída da sub-rede de classificação têm KA canais.

Caixas de Ancoragem



- Além de detectar a existência/classe de objetos, cada caixa âncora também é responsável por detectar o tamanho/forma dos objetos.
- Isso é feito por meio da sub-rede de regressão, que produz 4 números para cada caixa âncora que preveem o deslocamento relativo (em termos de coordenadas do centro, largura e altura) entre a caixa âncora e a caixa de referência.
- Portanto, o mapa de características de saída da sub-rede de regressão têm 4A canais.

Caixas de Ancoragem

Assim, a saída de cada uma das sub-redes será um tensor com um total de $\left(\sum_{l=3}^{7} W_l \times H_l \times A\right) \times 4$ elementos para regressão e $\left(\sum_{l=3}^{7} W_l \times H_l \times A\right) \times 4K$ para a classificação, em que $l = 3, 4, \ldots, 7$ denota o nível da pirâmide.

A partir dessas saídas serão feitos os refinamentos das caixas de ancoragem e a predição das classes.

Caixas de Ancoragem

Três proporções foram selecionadas - {1:2, 1:1 e 2:1}.

Em cada nível, as proporções originais são escaladas de {1, 2⅓, 2⅔}, totalizando A=9 âncoras por level.

A área dos âncoras varia de 32 a 512 nos níveis de pirâmide P3 a P7.

Cada âncora é associada a um vetor one-hot de comprimento K para classificação e um vetor de comprimento 4 para as caixas delimitadoras.

Caixas de Ancoragem

Regras de Associação:

Computa-se a intersecção sobre a união (IoU) entre cada âncora e o os caixas alvos (ground-truth object boxes).

- Uma âncora é associada ao alvo se IoU >= 0.5
- Uma âncora é associada como objeto de fundo de 0 < IoU < 0.4
- Uma âncora é ignorada se 0.4 <= IoU < 0.5

Quando uma âncora é associada à um alvo, também será associada à um vetor one-hot correspondente. Se associada com objeto de fundo, acompanhará um vetor nulo.

Função de Custo

A função de custo do RetinaNet é multitarefa, que contém dois termos: um para localização (L_{loc}) e outro para classificação (L_{cls}). A perda multi-tarefa pode ser escrita como:

$$L = L_{loc} + L_{cls}$$

Custo Para Regressão

- Cada caixa de ancoragem pode estar ligada a uma caixa alvo (*ground-truth box*).
- Seja $(A^i, G^i)_{i=1,\ldots,N}$ em que A representa âncoras, G representa os alvos e N é o número de pareamentos.
- Para cada âncora pareada, a rede de regressão prediz 4 números, na forma, $P^i = (P^i_x, P^i_y, P^i_w, P^i_h)$ O primeiros números representam a diferença entre os centros das âncoras e dos alvos e os dois últimos a diferença entre altura e largura.

Custo Para Regressão

Para cada uma das predições, existe um valor para regressão alvo , que é calculado por $T = (T_x, T_y, T_w, T_h)$:

$$T_x = (G_x - A_x)/A_w$$

$$T_y = (G_y - A_y)/A_h$$

$$T_w = log(G_w/A_w)$$

$$T_h = log(G_h/A_h)$$

Custo Para Regressão

Por fim, o custo para regressão será definido como:

$$L_{loc} = \sum_{j \in x,y,z,w} smooth_{L1}(P_j^i - T_j^i),$$

em que

$$smooth_{L1} = \begin{cases} 0.5x^2 & |x| < 1 \\ |x| - 0.5 & |x| \geq 1 \end{cases}$$

Custo Para Classificação

Como discutido no início, o custo para a classificação será a perda focal, definida como:

$$L_{cls} = \sum_{i=1}^{K} y_i log(p_i)^{\gamma} \alpha_i + (1 - y_i) log(1 - p_i) p_i^{\gamma} (1 - \alpha_i)$$

em que K denota o número de classes, y será 1 se a amostra pertencer à classe verdadeira e 0 caso contrário, p é a probabilidade predita para cada classe, γ é o parâmetro de foco e α é o parâmetro de balanceamento.

No artigo original, foi encontrado que os valores de γ=2 e α=0.25 resultam no melhor desempenho.

Predição

- RetinaNet seleciona no máximo 1000 caixas âncora que possuem a maior probabilidade prevista para cada classe de cada nível FPN.
- Um objeto na imagem pode ser previsto por múltiplas caixas de âncora. Para remover a redundância, a supressão não máxima (NMS) é aplicada a cada classe de forma independente. Assim, irá escolher uma caixa de âncora com a maior confiança e remove quaisquer caixas de âncora sobrepostas com um IoU superior a 0,5.
- No último estágio, para cada âncora restante, a sub-rede de regressão fornece previsões de deslocamento que podem ser usadas para refinar a âncora para obter uma previsão de caixa delimitadora.

Inicialização

- Os modelos ResNet-50 e ResNet-101 pré-treinados no ImageNet1k.
- Todas as novas camadas convolucionais, exceto a última nas sub-redes RetinaNet, são inicializadas com viés b = 0 e um preenchimento de peso gaussiano com $\sigma = 0.01$.
- Para a última camada convolucional da sub-rede de classificação, definimos a inicialização do viés como $b = -\log((1 - \pi)/\pi)$, onde π especifica que no início do No artigo original, utiliza-se $\pi = .01$ em todos os experimentos.
- Essa inicialização impede que o grande número de âncoras de fundo gere um valor de perda grande e desestabilizante na primeira iteração do treinamento.

Treinamento e otimização

- O RetinaNet é treinado com gradiente estocástico descendente (SGD).
- Os modelos são treinados por 90.000 iterações com uma taxa de aprendizado inicial de 0,01, que é então dividida por 10 em 60.000 e novamente em 80.000 iterações.
- Utiliza-se a inversão horizontal de imagens como a única forma de aumento de dados.
- Utiliza-se uma decaimento de peso de 0.0001 e um momento de 0.9.

Comparação com outros algoritmos

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
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**
One-stage methods							
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

Table 2. **Object detection** *single-model* results (bounding box AP), *vs.* state-of-the-art on COCO `test-dev`. We show results for our RetinaNet-101-800 model, trained with scale jitter and for 1.5× longer than the same model from Table 1e. Our model achieves top results, outperforming both one-stage and two-stage models. For a detailed breakdown of speed versus accuracy see Table 1e and Figure 2.

Vantagens e desvantagens

Quando foi proposto o RetinaNET conseguiu alcançar resultados superiores a outros detectores de estágio único.

Muito se evoluiu desde então, como veremos a seguir, o RetinaNET é um modelo grande e possui alto tempo de inferência.

Mais recentemente, uma melhoria do RetinaNET foi apresentada, chamada de AX-RetinaNet.

Exemplo de aplicação

Table 3 Evaluation of deep learning models

Algorithm	Precision (%)	Recall (%)	F1 (%)	MAP (%)
RetinaNet	64.98	83.86	73.26	82.89
SSD	63.69	88.89	74.21	82.71
YOLO v3	69.65	80.67	74.77	80.69

Table 4 Indicators of models in identifying hard samples

Algorithm	MAP (%)	FPS	Model size
RetinaNet	79.61	22	157M
SSD	79.03	41	149M
YOLO v3	79.02	69	89M

O trabalho de [3] fez um estudo comparativo entre detectores de comprimidos médicos:

- A precisão média (AP) do RetinaNet atingiu 82,89%, porém os quadros por segundo (FPS) são apenas um terço do YOLO v3, o que torna difícil alcançar desempenho em tempo real.
- O SSD não apresenta um desempenho tão bom nos indicadores de MAP e FPS.
- Embora a MAP do YOLO v3 seja ligeiramente inferior às outras (80,69%), ele tem uma vantagem significativa em termos de velocidade de detecção.
- O YOLO v3 também teve um desempenho melhor ao lidar com a detecção de amostras difíceis, tornando-o mais adequado para implantação em equipamentos hospitalares.

Exemplo de aplicação

Uma versao melhorada do RetinaNET foi proposto em [4] para detecção de doenças em folhas.

AX-RetinaNet utiliza um módulo de fusão de características multinível melhorado, denominado X-module.

AX-RetinaNet adiciona um módulo de Atenção de Canal, que adiciona pesos otimizados adaptativamente a cada canal de mapa de características para garantir que a rede preste atenção às informações de características úteis e reduza a interferência de informações redundantes.

Perguntas?

Referências

Artigos Base:

[1] LIN, Tsung-Yi et al. Focal loss for dense object detection. In: **Proceedings of the IEEE international conference on computer vision**. 2017. p. 2980-2988.
[2] LIN, Tsung-Yi et al. Feature pyramid networks for object detection. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. 2017. p. 2117-2125.
[3] Tan, L., Huangfu, T., Wu, L. *et al.* Comparison of RetinaNet, SSD, and YOLO v3 for real-time pill identification. *BMC Med Inform Decis Mak* **21**, 324 (2021). https://doi.org/10.1186/s12911-021-01691-8
[4] Bao, W., Fan, T., Hu, G. *et al.* Detection and identification of tea leaf diseases based on AX-RetinaNet. *Sci Rep* **12**, 2183 (2022). https://doi.org/10.1038/s41598-022-06181-z

Textos Complementares:

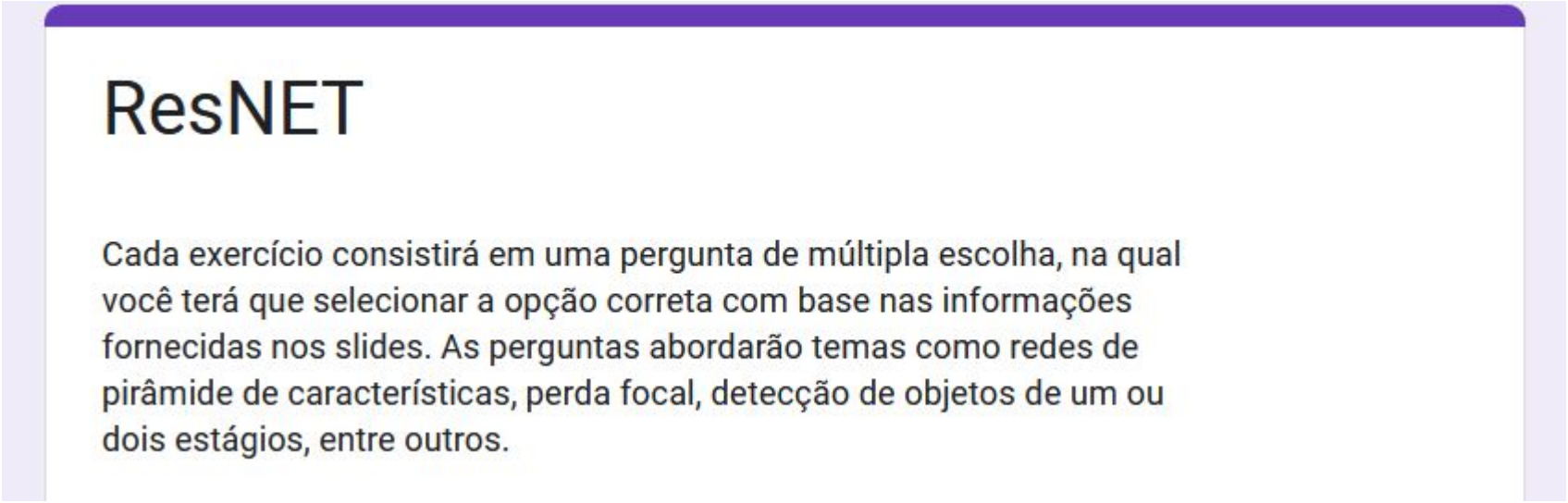
“visao computacional” - https://pt.d2l.ai/chapter_computer-vision/bounding-box.html
“rcnn” - https://medium.com/codex/a-guide-to-two-stage-object-detection-r-cnn-fpn-mask-r-cnn-and-more-54c2e168438c
“yolov1”-https://pyimagesearch.com/2022/04/11/understanding-a-real-time-object-detection-network-you-only-look-once-yolov1/
“desbalanceamento de classes”-https://medium.com/analytics-vidhya/how-focal-loss-fixes-the-class-imbalance-problem-in-object-detection-3d2e1c4da8d7
“perda focal”-https://leimao.github.io/blog/Focal-Loss-Explained/
“retinaNet”-https://medium.com/@14prakash/the-intuition-behind-retinanet-eb636755607d
“rpn”-https://medium.com/@codeplumber/region-proposal-network-rpn-backbone-of-faster-r-cnn-4a744a38d7f9
“fpn”-https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c
“retinaNet”-https://blog.zenggyu.com/posts/en/2018-12-05-retinanet-explained-and-demystified/index.html

Exemplo de Implementação:

“RetinaNET tensorflow”-https://keras.io/examples/vision/retinanet/

Exercícios

https://forms.gle/63J8hRgvUivD5w5W8



Obrigado!