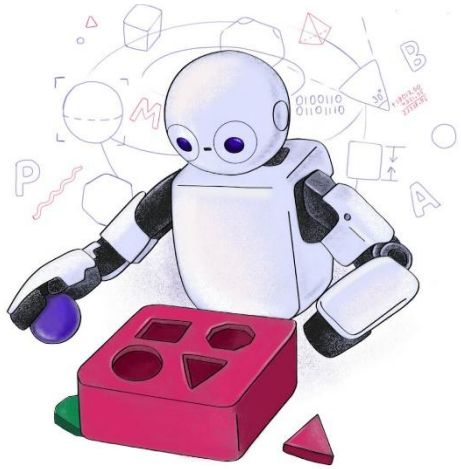
TP558 - Tópicos avançados em Machine Learning

SAHI - SLICING AIDED HYPER INFERENCE AND FINE-TUNING FOR SMALL OBJECT DETECTION



Inatel

Jonas Vilasboas Moreira
jonasmoreira@dtel.inatel.br

Introdução

“Imagine uma câmera de vigilância em uma rua: um carro passa, um corredor segue seu percurso e, ocasionalmente, cruza com um ciclista. Em determinado momento, esse ciclista faz um movimento e saca uma arma, que ocupa apenas uma fração mínima da imagem. Será que os sistemas de visão computacional conseguiriam detectar esse objeto pequeno e distante?”

Situação:

- Os objetos são representados por poucos pixels, com poucos detalhes. Para um segurança olhando a tela, é fácil perceber os carros e as pessoas logo de cara. Mas e aquele detalhe — uma pessoa segurando um objeto pequeno, como uma arma? Muitas vezes o olho humano pode não notar de imediato, principalmente em meio a tanta coisa acontecendo.

Como resolver essa situação?

Introdução

Detecção de Objetos

Detecção de objetos é uma técnica de visão computacional que **permite a um modelo identificar automaticamente o que está presente em uma cena e onde cada coisa está**.

Em uma rua movimentada, por exemplo, o sistema pode reconhecer carros, grupos de pessoas e até objetos pequenos e distantes, como uma arma sendo carregada por alguém.

Introdução

Introdução

O sistema de monitoramento inteligente por imagens de um laboratório evitou, na manhã do dia 26 de julho (2022), uma tentativa de assalto a mão armada na área central de Campo Mourão.

Por volta das 06h00 da manhã a base de monitoramento da Viptech Smart Solutions recebeu um alerta de uma das câmeras que detectou um homem armado indo em direção a uma mulher que realizava uma corrida em via pública. Isso só foi possível graças à um sistema de inteligência artificial de detecção de pessoas e objetos chamado **BRAIN**, uma solução proprietária desenvolvida pela Fábrica de Software da empresa mourãoense **Viptech Smart Solutions**.

O alerta de arma de fogo gerado imediatamente pelo sistema permitiu que o operador da Viptech,que estava de plantão, acionasse a sirene do laboratório em que estava instalada a câmera coibindo assim o assalto (vídeo).

Segundo o Diretor Executivo da Viptech Smart Solutions André Cardeal Santana, o BRAIN é uma solução tecnológica inovadora para segurança de residências e demais segmentos de negócios, que se baseia em computação em nuvem e algoritmos estatísticos que simulam redes neurais humanas e garantem precisão de até 95% na identificação de pessoas e objetos.

Dessa forma, a solução busca prevenir situações de risco em ambientes internos e externos, uma vez que permite a detecção de atitudes suspeitas e objetos como armas de fogo de maneira sistematizada. Essa tecnologia aliada ao monitoramento fornece informações qualificadas que garantem efetividade no serviço de segurança.

Para saber mais sobre o BRAIN e demais soluções oferecidas pela Viptech Smart Solutions acesse: https://www.viptech.com.br/BRAIN/

Introdução

Pipeline de Detecção

1. Pré-processamento da Imagem
    - Redimensionamento e normalização da imagem (para caber no modelo).
    - Data augmentation (ex.: flips, crops, brilho, ruído) para aumentar robustez.

2. Extração de Características
    - Uso de uma rede convolucional backbone (ex.: ResNet, VGG, MobileNet, Swin Transformer) para extrair mapas de características.
    - Esses mapas codificam informação espacial e semântica da imagem.

Introdução

Pipeline de Detecção

3. Geração de Propostas / Localização Inicial
    - Existem dois paradigmas principais:
        - Two-stage (ex.: Faster R-CNN):
            - Um Region Proposal Network (RPN) gera regiões candidatas onde pode haver objetos.
        - One-stage (ex.: YOLO, SSD, RetinaNet):
            - O modelo prevê diretamente caixas delimitadoras e classes em um único passo, a partir dos mapas de características.

4. Refinamento e Classificação
    - Para cada região candidata ou célula do mapa de features:
    - Regressão de bounding box: ajusta a posição, altura e largura da caixa.
    - Classificação: atribui uma classe (ou "fundo" se não houver objeto).

Introdução

Pipeline de Detecção

5. Pós-processamento
    - Non-Maximum Suppression (NMS): remove caixas muito sobrepostas, mantendo apenas a de maior confiança.
    - Filtragem por limiar de confiança: descarta previsões de baixa probabilidade.
    - Conversão para coordenadas da imagem original (se houve redimensionamento).

6. Saída Final
    - Lista de objetos detectados com:
        - Bounding box (x, y, largura, altura),
        - Classe prevista,
        - Confiança/probabilidade associada.

Imagem → Pré-processamento → Extração de features → Geração de propostas/previsões → Regressão + Classificação → NMS + Filtragem → Objetos detectados

Introdução

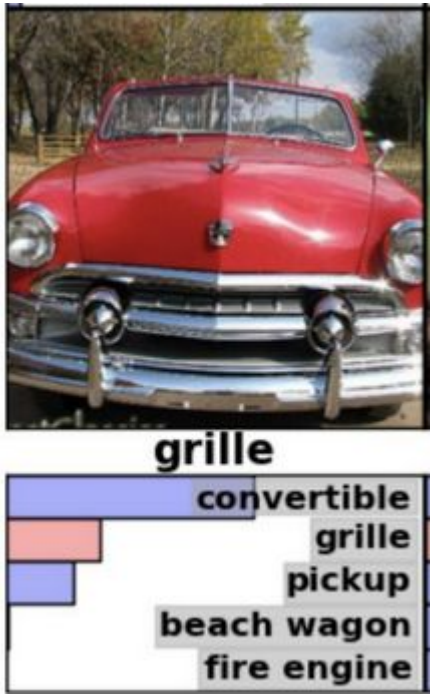
- Nos últimos anos, a detecção de objetos avançou muito. Hoje temos modelos como **Faster R-CNN [1]**, **RetinaNet [2]**, **Cascade R-CNN [3], Varifocal Net [4]** e **YOLO [10]** (entre outros), que conseguem identificar carros, pessoas e rostos com alta precisão.

- Esses modelos são treinados em bases de imagens com milhares de amostras, como **ImageNet [5]**, **Pascal VOC [6]** e **COCO [7]**.

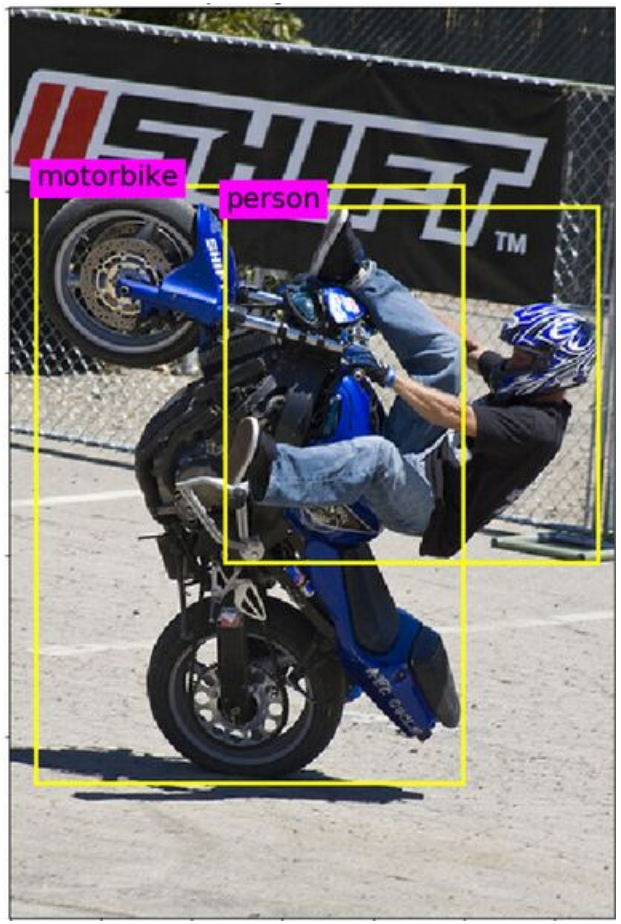
- Mas há um detalhe: esses bancos de dados usam imagens de **baixa resolução (640x480)**, com **objetos grandes** — cobrindo em média 60% da altura da imagem.

Introdução

Exemplos de Imagens dos Datasets:



ImageNet



Pascal VOC

MS Coco

Introdução

- Quando aplicamos os mesmos modelos em drones ou câmeras modernas de vigilância, com **imagens em alta resolução (4k)**, e **objetos pequenos** na cena, o **desempenho cai drasticamente**.

- Com a popularização dos drones e câmeras 4K, surge a necessidade de detecção em imagens de alta resolução.

Dota Dataset

Introdução

- Existe um padrão chamado **DORI [8]** (InfinityOptics) que define: para apenas detectar um objeto, ele precisa ter pelo menos 10% da altura da imagem; para reconhecer, 20% (algo como 108 pixels em um vídeo Full HD).
  - Detection (Detecção): perceber que há algo presente na cena (ex.: “há uma pessoa ali”).
  - Observation (Observação): ver o objeto com mais clareza, suficiente para acompanhar sua movimentação (ex.: “uma pessoa andando pela calçada”).
  - Recognition (Reconhecimento): distinguir a categoria do objeto (ex.: “essa pessoa está uniformizada como segurança”).
  - Identification (Identificação): nível máximo de detalhe, capaz de reconhecer quem ou exatamente o que é (ex.: “essa pessoa é o João, segurança da empresa”).

- Portanto, existe a demanda de se localizar esses pequenos objetos nas imagens, onde essa necessidade ainda não foi satisfeita através dos métodos atuais, pois exigem muito mais processamento e memória.

Como lidar com isso?

Técnicas de Detecção de Objetos

- **Single-stage detector (Detector de estágio único)**
  - Modelo de detecção de objetos que faz tudo em um único passo: ele prevê diretamente as caixas delimitadoras (bounding boxes) e as classes dos objetos ao mesmo tempo.
  - Vantagem: muito rápido.
  - Desvantagem: geralmente menos preciso que os de dois estágios em casos complexos.

Técnicas de Detecção de Objetos

- **Two-stage detector (Detector de dois estágios)**
  - Modelo que divide a detecção em duas etapas:
    - Primeiro propõe regiões da imagem que podem conter objetos (Region Proposal).
    - Depois analisa cada região para classificar e refinar a caixa delimitadora.
  - Vantagem: mais preciso.
  - Desvantagem: mais lento que os de estágio único.

Técnicas de Detecção de Objetos

- **Anchor-free detector (Detector sem âncoras)**
  - Modelo que não depende de “caixas pré-definidas” (anchors) para prever objetos. Em vez disso, ele detecta objetos diretamente a partir de pontos-chave, como o centro ou os cantos do objeto.
  - Vantagem: simplifica o processo, reduz hiperparâmetros e pode lidar melhor com objetos de diferentes tamanhos.
  - Desvantagem: ainda pode ter desafios em precisão em alguns cenários (ex.: objetos muito próximos uns dos outros, objetos muito pequenos, parcialmente cobertos, etc).

Técnicas de Detecção de Objetos

- **Single-stage detectors**:
  - Single Shot MultiBox Detector (SSD) [9]
  - You Only Look Once (YOLO) [10]
  - RetinaNet [2]

- **Two-stage detectors**:
  - Fast R-CNN [11]
  - Faster R-CNN [1]
  - Cascade R-CNN [3]

- **Detectores anchor-free** (sem caixas pré-definidas)
  - Fully Convolutional One-Stage Object Detection (FCOS) [13]
  - VarifocalNet (VFNet) [4]
  - Task-Aligned One-Stage Object Detection (TOOD) [14]

Técnicas de Detecção de Pequenos Objetos

Aplicações Específicas

- **Particle Swarm Optimization + Bacterial Foraging Optimization (PBLS) [15]**
  - Otimiza classificador e função de perda.
  - Problema: exige treino do zero, não aproveita pesos pré-treinados; difícil adaptação.

- **Augmentation for Small Object Detection [16]**
  - Cria cópias de pequenos objetos para aumentar o dataset.
  - Problema: requer anotações de segmentação → incompatível com datasets de detecção comuns.

- **SSD-MSN: Improved Multi-Scale Object Detection [17]**
  - Seleciona áreas com pequenos objetos, recorta e amplia.
  - Problema: aumenta custo computacional; seleção das áreas é trabalhosa.

- **Small Target Detection Network (STDnet) [18]**
  - Rede convolucional com mecanismo de atenção precoce.
  - Identifica regiões promissoras com pequenos objetos + contexto.

Técnicas de Detecção de Pequenos Objetos

Aplicações Específicas

- **Multiscale Rapid Detection in Satellite Imagery [19]**
  - Divide a imagem em pedaços menores para melhorar detecção.
  - Problema: implementação não genérica → só funciona em detectores específicos.

- **Joint Classification and Super-Resolution Network (JCS-Net) [20]**
  - Combina detecção com super-resolução para pedestres em pequena escala.
  - Problema: exige pré-treinamento do zero em grandes datasets.

- **Finding Tiny Faces in the Wild (GAN) [21]**
  - Usa redes adversariais para gerar faces nítidas a partir de rostos pequenos/borrados.
  - Problema: também requer pré-treinamento pesado com bases grandes.

Redes adversariais são duas redes treinadas em competição, onde uma cria dados falsos e a outra tenta identificar se são reais ou falsos, fazendo com que o gerador aprenda a criar dados muito realistas.

SAHI - Arquitetura e funcionamento

Ideia Geral

O SAHI é um pipeline genérico de fine-tuning e inferência auxiliado por slicing que pode ser utilizado sobre qualquer detector de objetos existente.

- Problema: pequenos objetos têm poucas informações (poucos pixels).
- Solução proposta: dividir imagens em patches sobrepostos → objetos pequenos ficam maiores proporcionalmente dentro de cada patch.

Essa estratégia é usada em duas fases

- Slicing Aided Fine-tuning (SF) – durante o treinamento.
- Slicing Aided Hyper Inference (SAHI) – durante a inferência.

https://github.com/obss/sahi

SAHI - Arquitetura e funcionamento

Slicing Aided Fine-tuning (SF)

Base inicial

- Usar frameworks de detecção populares (Detectron2 [22], MMDetection [23], YOLOv5 [24]).
- Aproveitar pesos pré-treinados em datasets grandes (ImageNet [5], MS COCO [7]).
- Vantagem: não precisa treinar nenhum modelo do zero (economia de tempo e dados).

Problema dos pré-treinamentos

- Esses datasets contêm imagens de baixa resolução (640 × 480).
- Os objetos são relativamente grandes (≈60% da altura da imagem).
- Resultado: bom desempenho para objetos grandes, mas fraco para pequenos objetos em imagens de alta resolução (ex: drones, câmeras modernas).

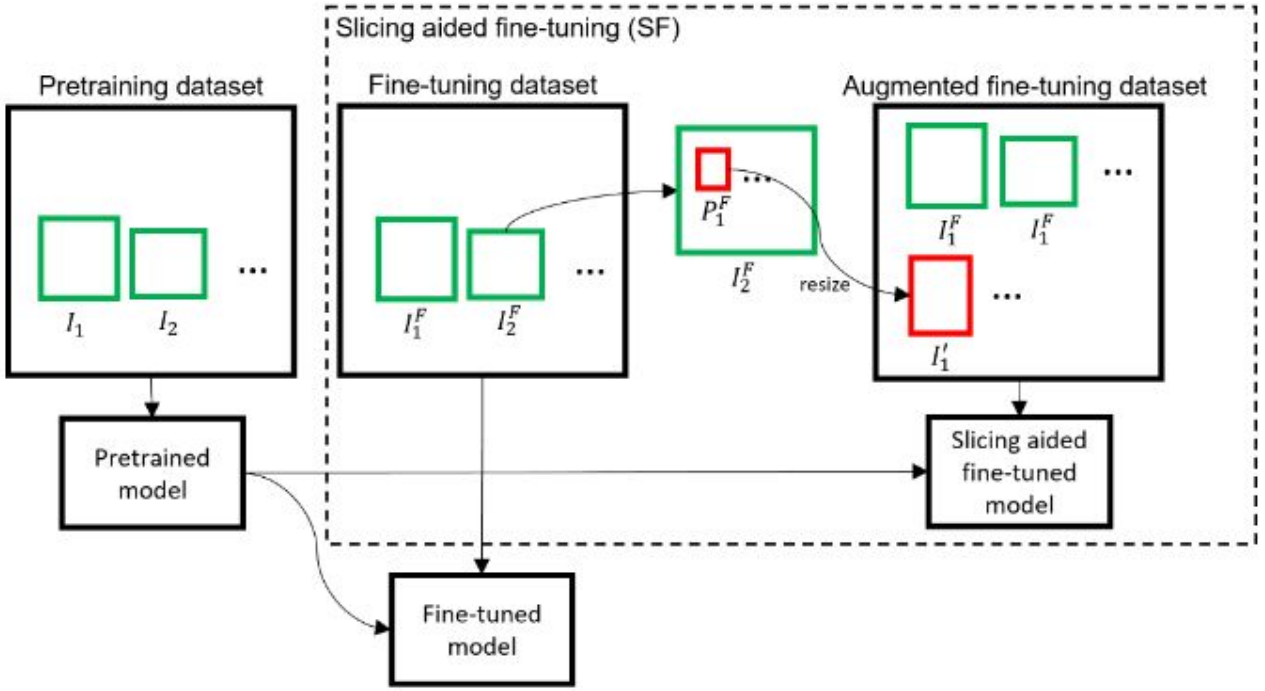
SAHI - Arquitetura e funcionamento

Estratégia de slicing para treinamento

- Cada imagem do conjunto de fine-tuning (IF1, IF2, … IFj) é dividida em patches sobrepostos (PF1, PF2, … PFk).
- O tamanho dos patches é definido por dois hiperparâmetros:
- M (largura) e N (altura).
- Valores escolhidos dentro de intervalos definidos: [Mmin, Mmax] e [Nmin, Nmax].
- Cada patch é redimensionado preservando a proporção.
- O novo tamanho é ajustado para largura entre 800 e 1333 pixels.
- Isso faz com que os objetos dentro do patch ocupem mais pixels (ficam “maiores”).

SAHI - Arquitetura e funcionamento

Slicing Aided Fine-tuning (SF)



Um **fine-tuning dataset** é um conjunto de dados usado para ajustar (refinar) um modelo previamente treinado em um conjunto de dados maior e mais genérico. Um **fine-tuning model** (ou modelo ajustado/fine-tuned) é um modelo de aprendizado de máquina que, após ter sido treinado em um grande conjunto de dados, é refinado para uma tarefa específica usando um conjunto de dados menor e mais específico.

SAHI - Arquitetura e funcionamento

Dataset aumentado

- O conjunto de treinamento final inclui:
    - Os patches redimensionados (I′1, I′2, … I′k).
    - As imagens originais (IF1, IF2, … IFj) → isso garante que objetos das imagens com os tamanhos originais também sejam aprendidos.

Vantagem: a rede passa a ver pequenos objetos ampliados, mas não perde a noção dos pequenos.

Limitação

- Se os patches forem muito pequenos, objetos grandes podem não caber totalmente em um patch.
- Isso prejudica a detecção de objetos maiores → há um trade-off na escolha do tamanho dos patches.

SAHI - Arquitetura e funcionamento

Slicing Aided Hyper Inference (SAHI)

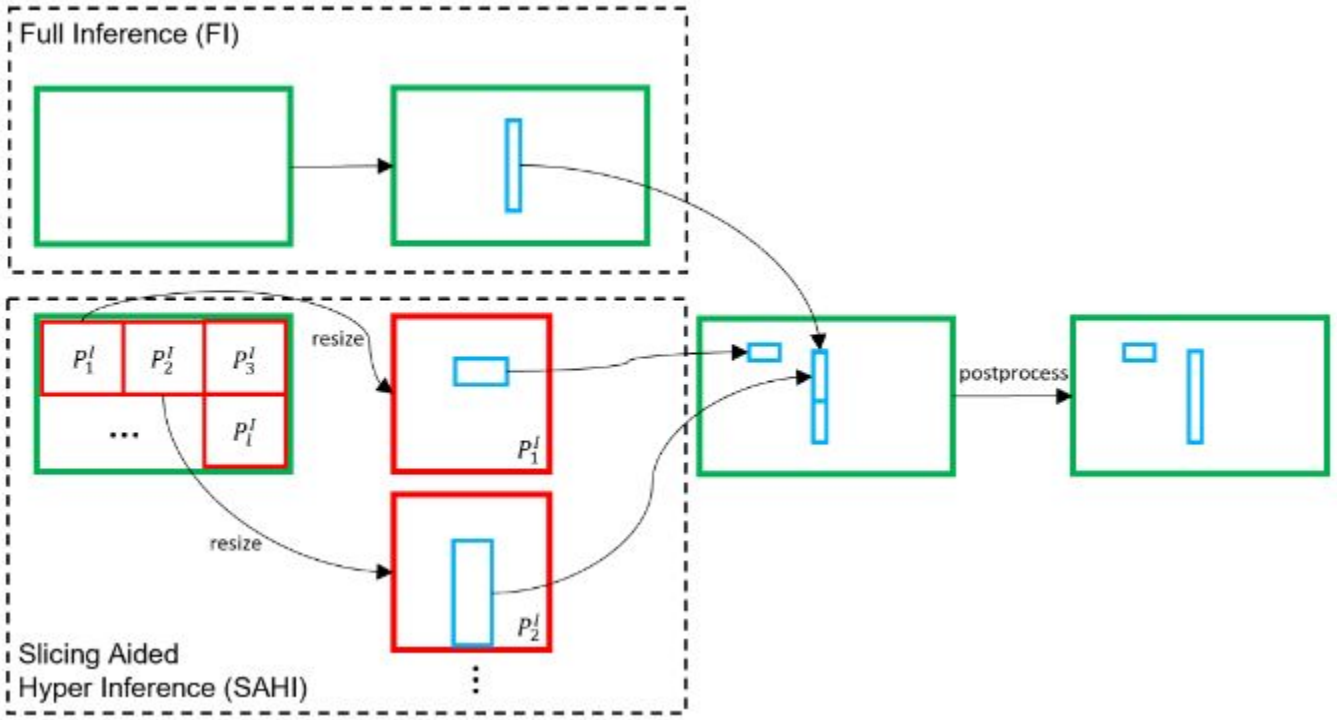
Pipeline da inferência:

1. A imagem de entrada (I) é dividida em l patches sobrepostos (patch overlap) de tamanho M × N.
2. Cada patch é redimensionado (mantendo proporção).
3. O detector é aplicado em cada patch individualmente.
   a. Opcional: realizar também uma inferência na imagem inteira (Full Inference – FI).
4. Útil para capturar objetos grandes que poderiam ser cortados em patches.
5. Combinar todos os resultados dos patches + FI (se usado).

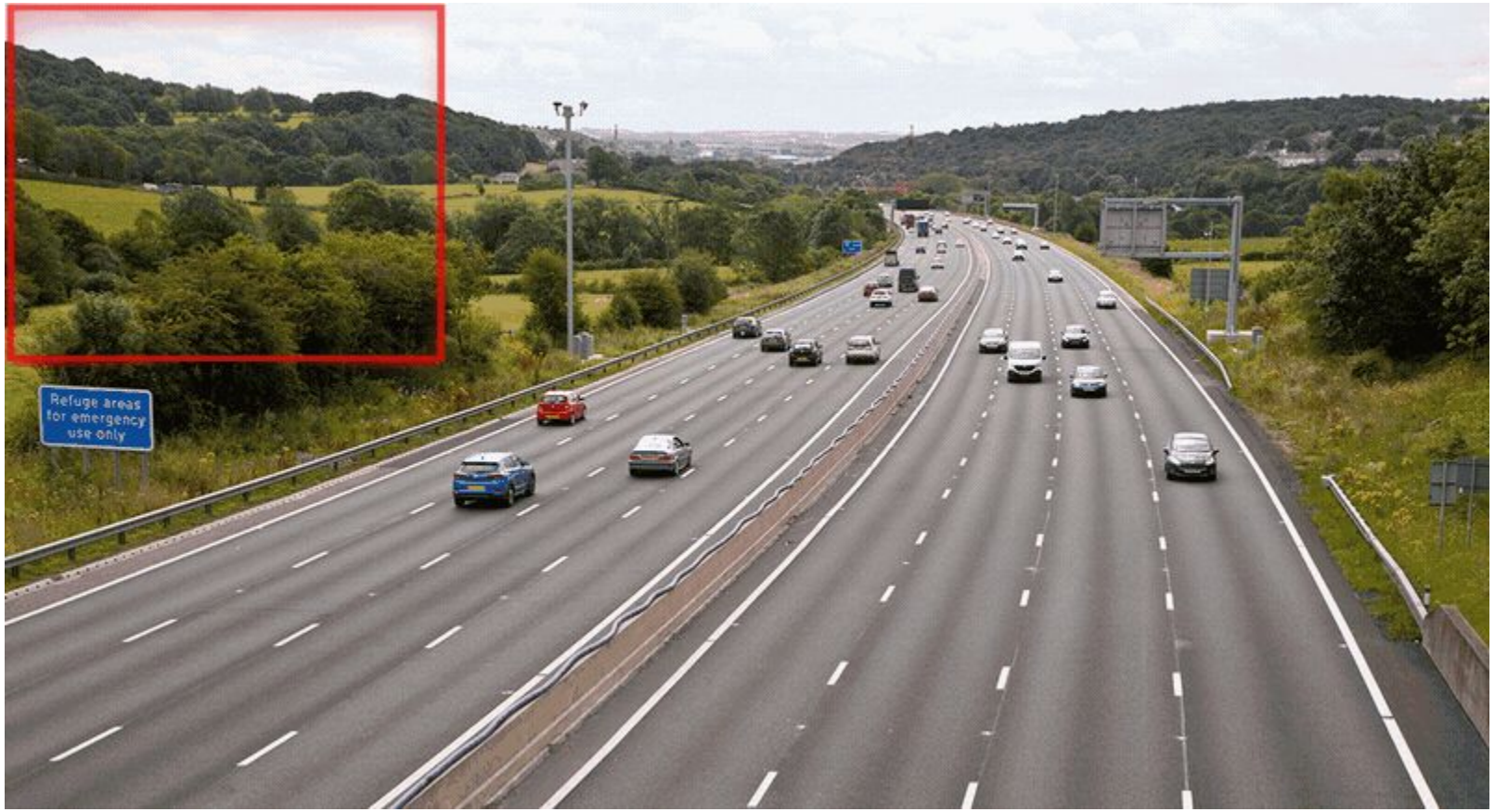
Patch overlap é a quantidade de sobreposição entre regiões cortadas (patches) de uma imagem durante o processamento, usada para melhorar a detecção de objetos próximos às bordas das fatias.

SAHI - Arquitetura e funcionamento

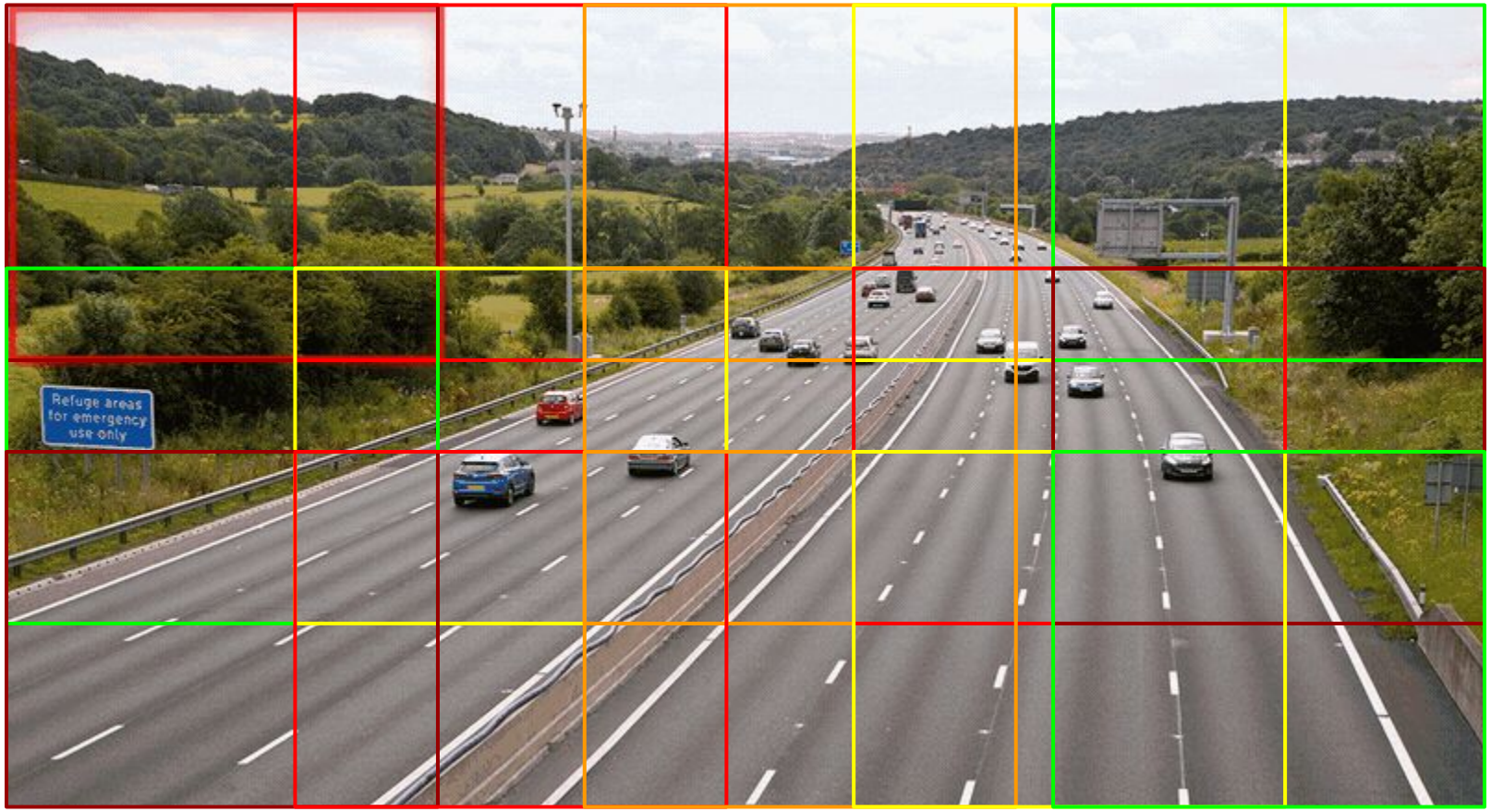
Full Inference (FI)



SAHI - Arquitetura e funcionamento



SAHI - Arquitetura e funcionamento



SAHI - Arquitetura e funcionamento

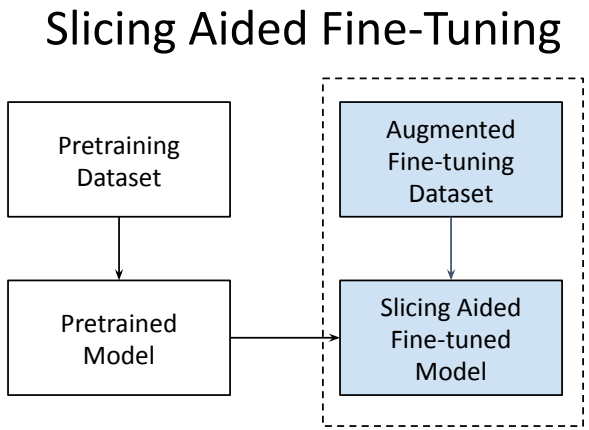
Pós-processamento com NMS (Non-Maximum Suppression)

- As detecções sobrepostas são comparadas pelo IoU (Intersection over Union).
- Se o IoU ≥ Tm (threshold de matching), elas são consideradas a mesma detecção.
- Entre as detecções sobrepostas, mantém-se a de maior confiança.
- Detecções com probabilidade < Td (threshold de detecção) são removidas.

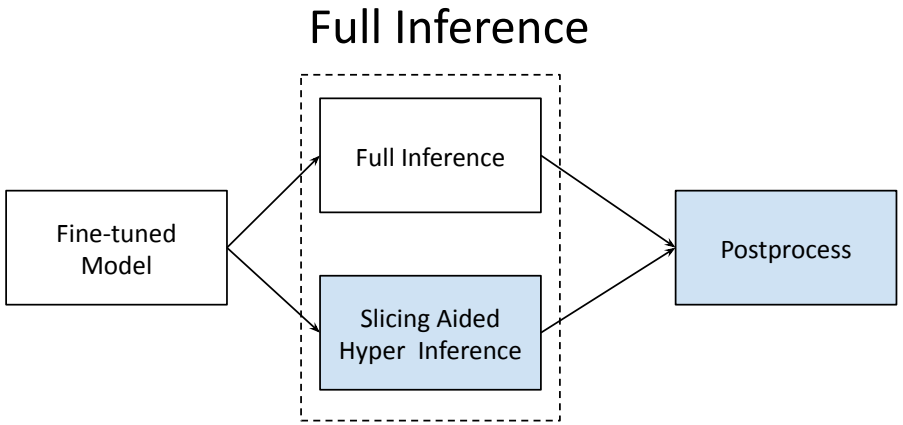
Resumo da vantagem

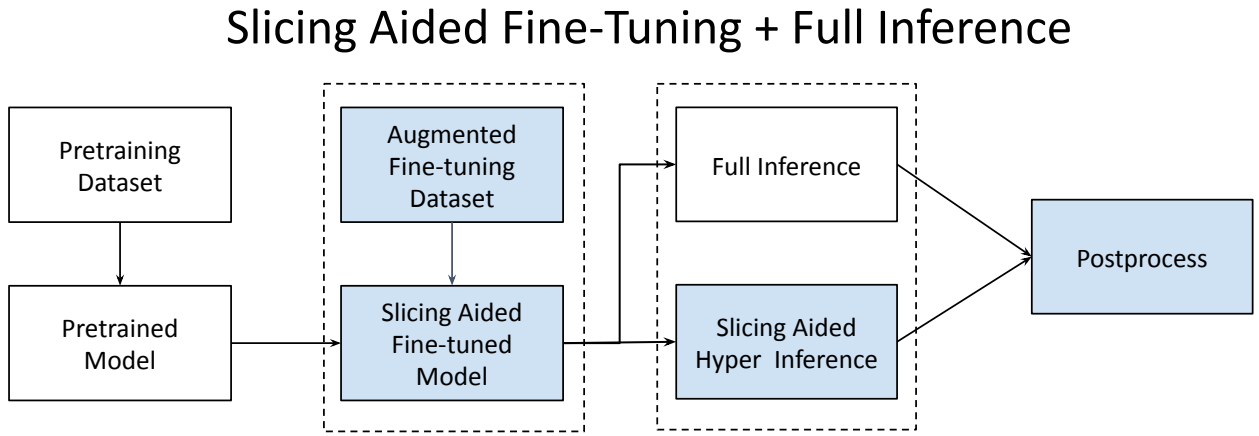
- Pequenos objetos ficam mais visíveis nos patches redimensionados → melhora a taxa de detecção.
- FI garante que objetos grandes também sejam detectados.
- O método é genérico: funciona em qualquer detector já existente.

SAHI - Arquitetura e funcionamento



Resumo
Como usar o SAHI





Resultados Experimentais

Integração da Proposta

- Método foi integrado em 3 detectores: FCOS [13], VarifocalNet (VFNet) [4], TOOD [14].
- Implementado usando o framework MMDetection [23].
- Foram disponibilizados publicamente:
  - Arquivos de configuração, scripts de conversão e avaliação.
  - Resultados dos experimentos.
  - Operações de slicing como módulo reutilizável em outros frameworks.

https://github.com/fcakyon/small-object-detection-benchmark

Resultados Experimentais

Bases de Dados

- VisDrone2019-Detection [25]
  - 8.599 imagens capturadas por drones em diferentes locais e alturas.
  - Características:
    - Objetos pequenos, densos e parcialmente ocluídos.
    - Variações de iluminação e perspectiva.
- Anotações: mais de 540k bounding boxes em 10 categorias (pedestrian, person, bicycle, car, van, truck, tricycle, awning-tricycle, bus, motor).
- Categorias agrupadas em supercategorias: pedestrian, motor, car, truck.
- Divisão: 6471 imagens treino / 548 imagens validação.

Resultados Experimentais

Bases de Dados

- xView [26]
  - Um dos maiores datasets de imagens de satélite para detecção.
  - Mais de 1 milhão de instâncias anotadas em 60 classes.
  - Conjunto usado: 75% treino, 25% validação.
  - Característica: objetos muito pequenos (largura < 1% da largura da imagem).

Resultados Experimentais

Configuração de Treinamento

- Otimizador: SGD.
- Hiperparâmetros:
  - Learning rate = 0.01.
  - Momentum = 0.9.
  - Weight decay = 0.0001.
  - Warmup linear = 500 iterações.
- Agendamento do learning rate: dec decaimento exponencial nas épocas 16 e 22.
- Slicing Aided Fine-tuning (SF):
  - Patches gerados a partir das imagens + anotações.
  - Tamanho dos patches:
    - VisDrone: largura/altura entre 480 e 640 px.
    - xView: largura/altura entre 300 e 500 px.
  - Imagens redimensionadas para largura entre 800 e 1333 px .
- Inferência - NMS com threshold Tm = 0.5.

Resultados Experimentais

Métrica de Avaliação

- Protocolo: MS COCO [7].
- Métrica principal: AP50 (IoU = 0.5).
- Limite: até 500 detecções por imagem.
- Resultados analisados por tamanho de objeto: small, medium, large.

Resultados Experimentais

Resultados – VisDrone

- Baseline: FI (Full Inference) → rodar detecção na imagem inteira.
- Ganho com SAHI: aumento de +6.8% (FCOS), +5.1% (VFNet), +5.3% (TOOD).
- Ganho adicional com SF (fine-tuning): até +12.7%, +13.4%, +14.5% de AP.
- Patch Overlap (25%):
  - Melhora AP em objetos pequenos/médios e AP geral.
  - Leve redução no AP de objetos grandes (causada por falsos positivos em regiões grandes).
- Melhor combinação para pequenos objetos: SF + SAHI.
- Melhor combinação para grandes objetos: SF + FI.

Resultados Experimentais

Resultados – xView

- Treinamento com imagens originais → desempenho muito baixo (objetos minúsculos).
- Com SF → melhora substancial na detecção.
- Integração com FI:
  - Aumenta AP (Average Precision) em objetos grandes (até +3.3%).
  - Pequena queda em objetos pequenos/médios (esperada, pois alguns grandes não aparecem nos slices).
- Patch Overlap (25%): aumento de até +1.7% no AP.
- Observação:
  - Apesar de ser um detector mais antigo, FCOS teve desempenho semelhante ao VFNet → confirmado pela eficácia do Focal Loss em lidar com classes desbalanceadas.
  - TOOD obteve os melhores resultados entre os três detectores.

Conclusão

Integração direta

- O método pode ser incorporado em qualquer pipeline de detecção de objetos.
- Não requer pré-treinamento → aproveita modelos já existentes.

Resultados principais

- Até +6.8% de AP apenas com SAHI (inferência fatiada).
- +14.5% de AP adicional com Slicing Aided Fine-tuning (SF) para pequenos objetos.
- +2.9% de AP adicional ao aplicar 25% de sobreposição entre slices.

Conclusão

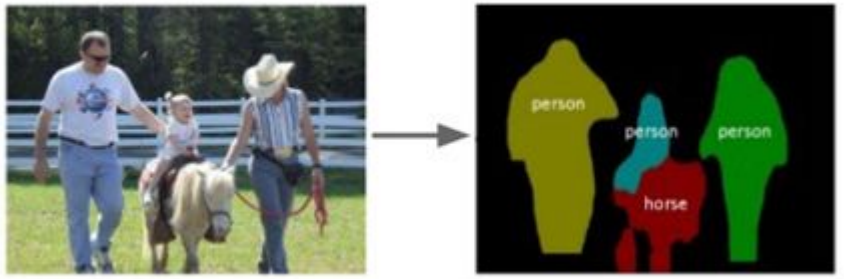
Custo computacional

- Treinar redes diretamente em imagens de alta resolução exige mais memória e processamento (feature maps maiores).
- A abordagem proposta:
  - Aumenta o tempo de computação de forma linear.
  - Mantém os requisitos de memória fixos.
- Permite ajustar o tamanho dos patches para equilibrar custo computacional e desempenho, dependendo da plataforma de uso.

Conclusão

Trabalhos futuros

- Extensão para modelos de segmentação por instância.
  - Modelos de segmentação por instância são técnicas de visão computacional que não apenas identificam quais objetos estão presentes em uma cena (classificação) e onde eles estão localizados (detecção), mas também delimitam com precisão os contornos de cada ocorrência individual desses objetos.



- Avaliação de diferentes técnicas de pós-processamento combinadas com slicing.

Demonstração & Quiz

- Slicing Aided Fine-tuning (SF)
  - https://github.com/jonasvm/seminario-sahi/blob/main/sahi-demo.ipynb
- Full Inference (FI)
  - https://github.com/jonasvm/seminario-sahi/blob/main/sahi-demo-2.ipynb
- Quiz
  - https://forms.gle/7oG8zwWzW39xBpuFA

Perguntas?

Referências

1. Ren et al. – *Faster R-CNN: Towards Real-Time Object Detection* (2015)
2. Lin et al. – *Focal Loss for Dense Object Detection* (2017)
3. Cai & Vasconcelos – *Cascade R-CNN: Delving into High Quality Object Detection* (2018)
4. Zhang et al. – *VarifocalNet: An IoU-Aware Dense Object Detector* (2021)
5. Deng et al. – *ImageNet: A Large-Scale Hierarchical Image Database* (2009)
6. Everingham et al. – *Pascal VOC Challenge* (2010)
7. Lin et al. – *Microsoft COCO: Common Objects in Context* (2014)
8. DORI – *https://www.infinitioptics.com/whitepapers/dori-detection-observation-recognition-identification*
9. Liu et al. – *SSD: Single Shot MultiBox Detector* (2016)
10. Bochkovskiy et al. – *YOLOv4: Optimal Speed and Accuracy of Object Detection* (2020)
11. Girshick – *Fast R-CNN* (2015)
12. Lin et al. – *Feature Pyramid Networks for Object Detection* (2017)
13. Tian et al. – *FCOS: Fully Convolutional One-Stage Object Detection* (2019)

Referências

14. Feng et al. – *TOOD: Task-Aligned One-Stage Object Detection* (2021)
15. Wang et al. – *Faster R-CNN para Condução Autônoma com PSO e BFO* (2019)
16. Kisantal et al. – *Augmentation for Small Object Detection* (2019)
17. Chen et al. – *SSD-MSN: Improved Multi-Scale Object Detection* (2019)
18. Bosquet et al. – *STDnet: Small Target Detection Network* (2018)
19. Van Etten – *Multiscale Rapid Detection in Satellite Imagery* (2019)
20. Pang et al. – *JCS-Net: Joint Classification and Super-Resolution Network* (2019)
21. Bai et al. – *Finding Tiny Faces in the Wild* (2018)
22. Wu et al. – *Detectron2* (2019)
23. Chen et al. – *MMDetection: OpenMMLab Toolbox* (2019)
24. Jocher et al. – *YOLOv5* (2021)
25. Du et al. – *VisDrone Object Detection Challenge* (2019)
26. Lam et al. – *xView: Objects in Context in Overhead Imagery* (2018)

Obrigado!