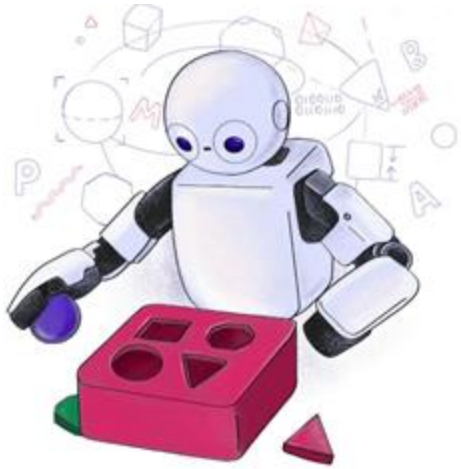
TP558 - Tópicos avançados em Machine Learning:

Knowledge Distillation



Inatel

Paulo Otavio Luczensky de Souza
paulo.souza@mtel.inatel.br

Introdução

- Proposto pelos estudiosos Bucilă, Caruana e Niculescu-Mizil (2006).
- Objetivo: **transferir o conhecimento** de um modelo ou um conjunto de modelos grandes para um modelo menor.
- Considerado a **primeira forma de compressão de modelos** em IA.
- Popularizado por Hinton (2015) que cunhou o termo *Knowledge Distillation*.

Inatel

Introdução

- O *Knowledge Distillation* pode ser aplicado em diversos cenários:
- **Visão Computacional**
  - Compressão de redes profundas como **YOLO, ResNet, EfficientNet**.
  - Deploy em **dispositivos móveis ou drones** com recursos limitados.
- **Processamento de Linguagem Natural (NLP)**
  - Modelos grandes como **BERT, GPT, T5** podem ser destilados.
  - Redução de custo computacional em **chatbots, tradução automática e sumarização**.
- **Edge AI e IoT**
  - Permite rodar modelos de IA em **sensores, câmeras inteligentes e dispositivos IoT**.
  - Mantém desempenho mesmo com **hardware restrito e baixo consumo de energia**.

Inatel

Fundamentação teórica

O que é destilação de conhecimento?

- Processo de **transferência de conhecimento** de um modelo ou conjunto de modelos grande e difícil de manejar para um único modelo menor.

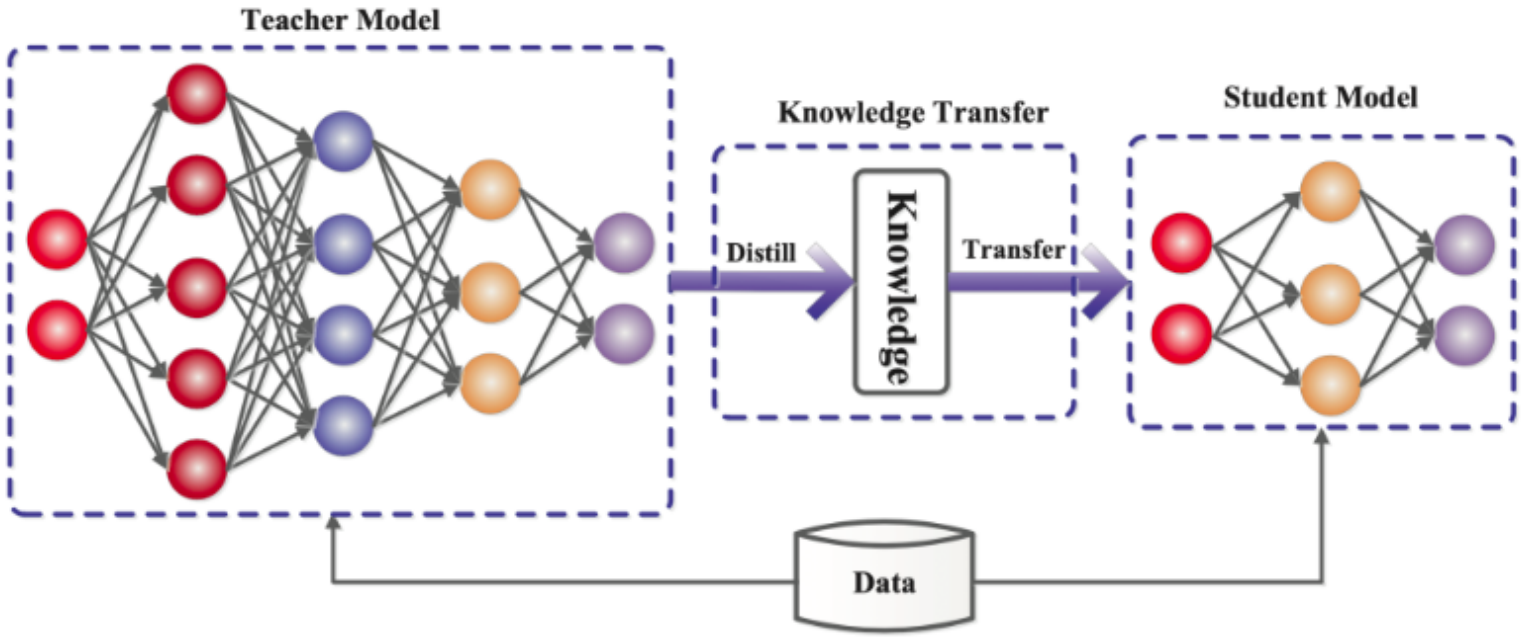
- Usado em modelos de redes neurais associados a arquiteturas complexas, incluindo diversas camadas e parâmetros de modelo.

- Forma de compressão de modelos, permitindo economia de recursos computacionais e aplicação prática em cenários com restrições do mundo real.

Inatel

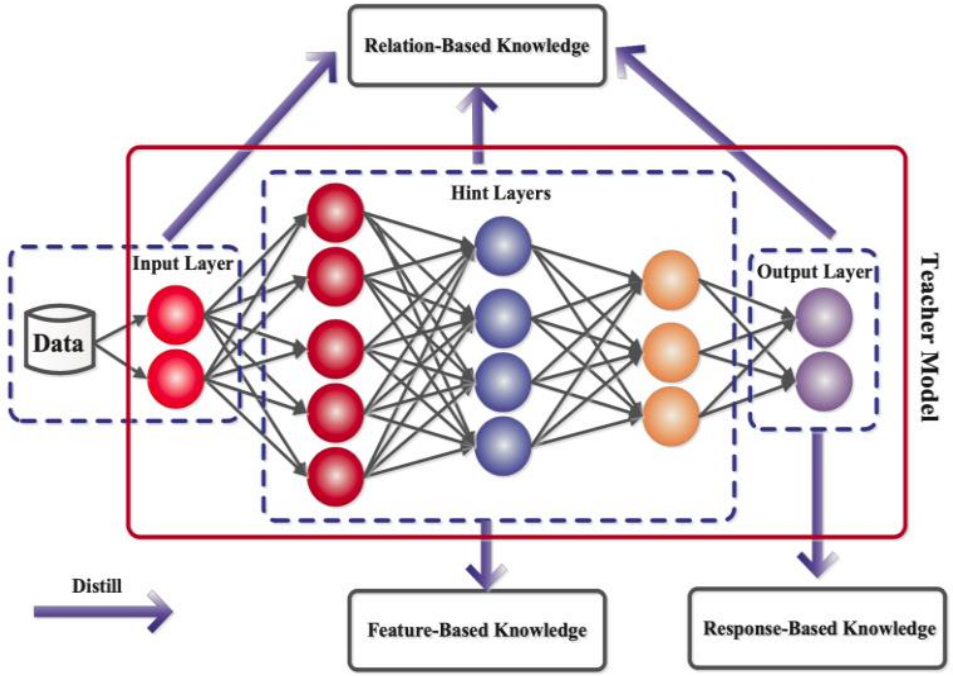
Fundamentação teórica

- Um **modelo menor(*Student*)** é treinado para **reproduzir o comportamento** de um **modelo maior(*Teacher*)**, aproveitando o conhecimento já adquirido.



Inatel

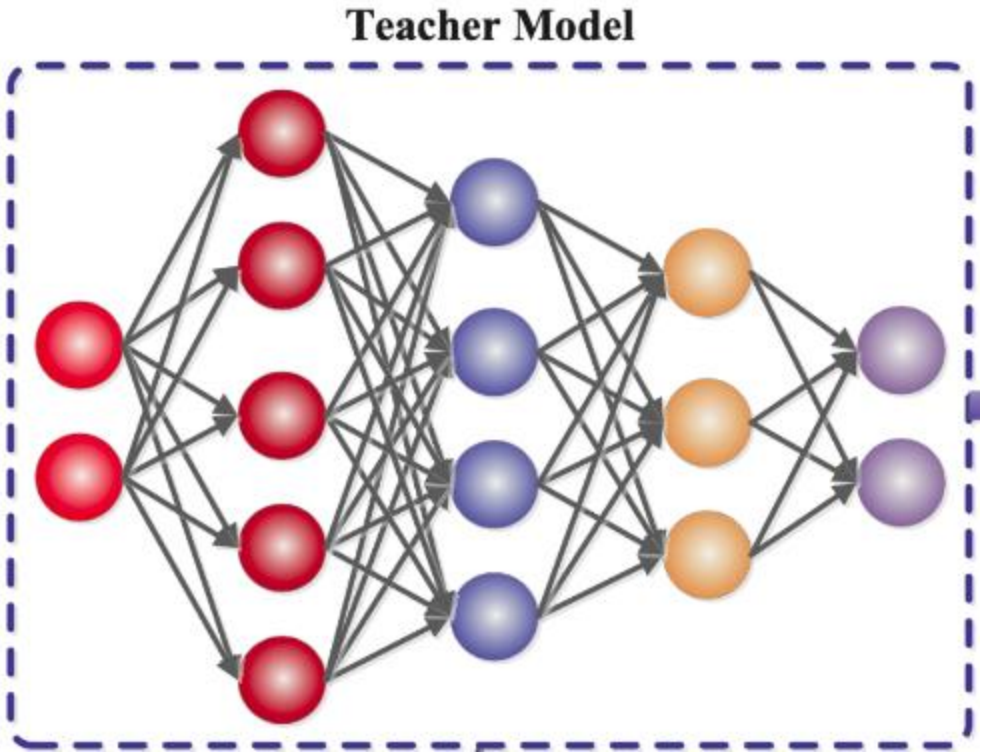
Fundamentação teórica - Redes Neurais



- **Conhecimento como parâmetros aprendidos**
  - Em redes neurais, o conhecimento é representado pelos **pesos e vieses ajustados durante o treinamento**, que capturam os padrões extraídos dos dados.
- **Diversidade de fontes em redes profundas**
  - Modelos profundos armazenam conhecimento em **diferentes níveis**, desde os ***logits* finais (valores antes da aplicação da função *softmax*)** até as **atividades das camadas intermediárias** (representações de alto nível).
- **Outras formas relevantes de conhecimento**
  - Além das saídas e ativações, também são valiosas as **relações entre neurônios e ativações** e os próprios **parâmetros estruturais do modelo professor**.

Inatel

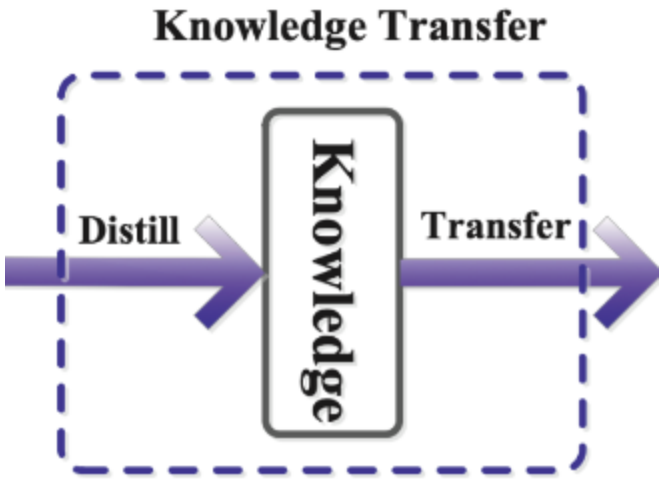
Fundamentação teórica - Professor(*Teacher*)



- **Modelo de referência:** *Teacher* é o modelo maior e mais preciso que fornece conhecimento para o *Student*.
- **Objetivo:** Guiar o *Student* a aprender padrões complexos sem precisar da mesma capacidade computacional.
- **Treinamento:** Geralmente treinado previamente em dados rotulados com alto desempenho.
- **Flexibilidade:** Pode transmitir diferentes tipos de conhecimento, como relações entre classes ou características internas do modelo.

Inatel

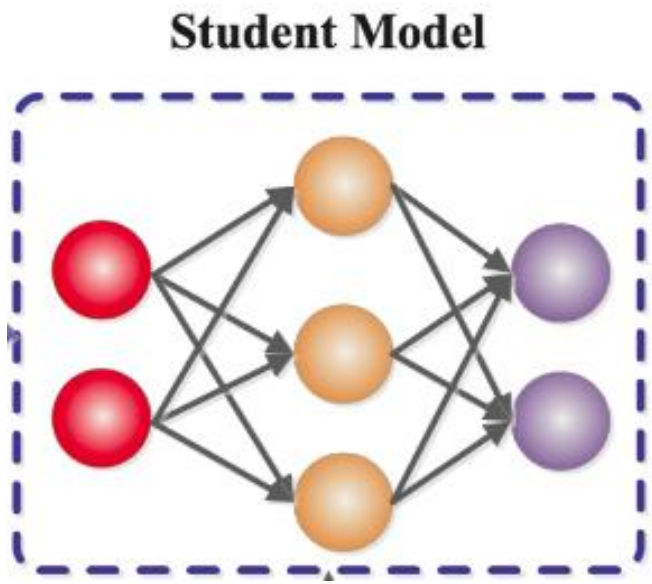
Fundamentação teórica - Transferência de Conhecimento



- **Fonte de conhecimento:** Fornece ***logits* finais** (valores brutos gerados pelo modelo antes da função *softmax*).
  - ***Logits*:** representam a "confiança crua" do modelo em cada classe, sem estarem normalizados. Não são probabilidades ainda, apenas indicam o quanto o modelo acredita em cada classe.
  - ***Softmax*:** transforma os *logits* em probabilidades que somam 1, facilitando interpretação e comparação.
  - **Importância na distilação:** usar os *logits* permite que o *Student* capture relações sutis entre classes (ex.: o modelo acha "gato" quase tão provável quanto "leão"), não somente rótulos discretos.
- **Tipos de conhecimento transferido:**
  - **Ativação de camadas intermediárias (*feature distillation*):**
    - Ex.: Se o *Teacher* aprende a detectar bordas e formas em camadas iniciais, o *Student* pode ser treinado para replicar essas ativações, mesmo sendo uma rede menor.
  - **Relações entre classes:**
    - Ex.: *Teacher* pode indicar que imagens de "gato" e "leão" têm semelhanças sutis (alta probabilidade relativa), mesmo que o rótulo real seja apenas "gato".
    - *Student* aprende essas relações, melhorando **generalização** e evitando confusões.

Inatel

Fundamentação teórica - Estudante(*Student*)



- **Modelo aprendiz:** *Student* é o modelo menor e eficiente, que aprende com o *Teacher*.
- **Objetivo:** Replicar o desempenho do *Teacher* usando menos parâmetros e menor custo computacional.
- **Treinamento:** Aprendizado baseado em **rótulos reais** combinados com ***soft targets*** (*logits* do *Teacher*) e, às vezes, ativações intermediárias.
- **Flexibilidade:** Pode ser projetado para diferentes restrições, como velocidade de inferência, memória ou energia.
- **Benefício prático:** Permite implementar modelos rápidos e leves mantendo boa precisão, ideal para dispositivos com recursos limitados.

Inatel

Fundamentação teórica

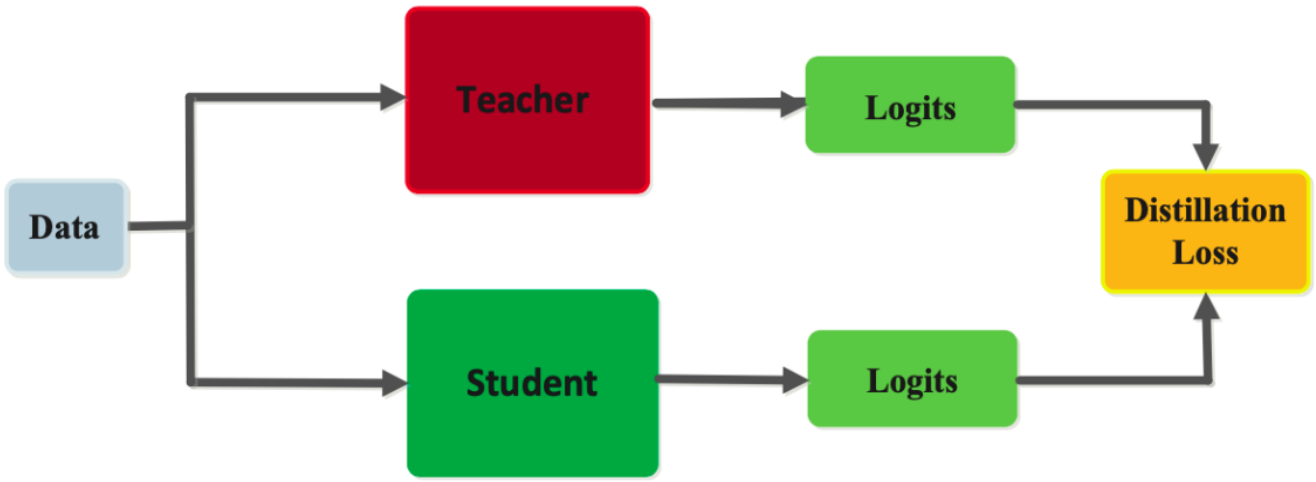
Ano	Cientista(s)	Contribuição/Evolução
2006	Bucilă, Caruana, Niculescu-Mizil	**Compressão de modelo** • Transferência de conhecimento de modelos grandes para menores. • Redução de custo computacional e memória. • Primeiro passo em *Knowledge Distillation*.
2015	Hinton, Vinyals, Dean	**Formalização da KD** • Uso de ***soft targets*** do *Teacher*. • *Student* aprende distribuições de probabilidade, não só rótulos reais. • **Antes disso, usavam-se apenas rótulos reais (*hard labels*)**. • Mantém desempenho próximo ao modelo grande.
2017	Romero	**FitNets** • Transfere também **representações intermediárias** do *Teacher*. • Melhora a capacidade de generalização do *Student*. • Permite treinamento mais eficiente de modelos menores.
2019	Touvron et al. / GShard (Google)	**GShard** • Modelo de linguagem com **100 bilhões de parâmetros**. • Distilação usada para **otimizar desempenho em larga escala**. • Facilita *deploy* de modelos massivos.
2023	Zhang et al.	**Data-Free KD** • Não exige **dados rotulados** para distilação. • Usa **síntese de características** e consistência espacial. • Permite compressão de modelos quando os dados originais não estão disponíveis.

Inatel

Tipos de Conhecimentos

Inatel

Arquitetura e funcionamento – **Conhecimento baseado em resposta**



- **Camada de saída final:** O foco está nas previsões do *Teacher*, representadas pelos ***logits*** antes da função *softmax*.
- **Imitação pelo *Student*:** O *Student* é treinado para replicar os logits do *Teacher*, aproximando-se do seu comportamento preditivo.
- **Função de perda de destilação:** Mede a diferença entre os *logits* do *Teacher* e do *Student*; ao ser minimizada, o *Student* melhora suas previsões.
- **Alvos suaves (*soft targets*):** Representam a distribuição de probabilidades sobre as classes, obtida via *softmax*.
- **Aplicação prática:** Usada em classificação de imagens, dentro do contexto de aprendizado supervisionado.
- **Benefício:** *Student* aprende padrões mais ricos do que aprenderia apenas com rótulos discretos, melhorando generalização.

Inatel

Função de Perda – **Conhecimento baseado em resposta**

A perda total combina a perda tradicional de classificação com a perda de distilação:

$$L_{total} = \alpha \cdot L_{CE}(y_{true}, y_{student}) + (1 - \alpha) \cdot T^2 \cdot L_{KL}\left(\text{softmax}(z_{teacher}/T), \text{softmax}(z_{student}/T)\right)$$

Onde:

- *LCE* : *Cross-Entropy Loss* entre os rótulos reais e as previsões do *Student*;
- *LKL*: *Kullback–Leibler Divergence* entre as distribuições do *Teacher* e do *Student*;
- *T*: temperatura que suaviza as probabilidades;
- α: peso que equilibra a importância entre a perda supervisionada e a distilação.

Inatel

Exemplo - Função de Perda – **Conhecimento baseado em resposta**

Imagine que o modelo **Teacher** e o **Student** estão classificando uma imagem entre 3 classes:
[Gato, Cachorro, Pássaro]

Modelo	Gato	Cachorro	Pássaro
Teacher (após softmax)	0.7	0.2	0.1
Student (após softmax)	0.5	0.4	0.1

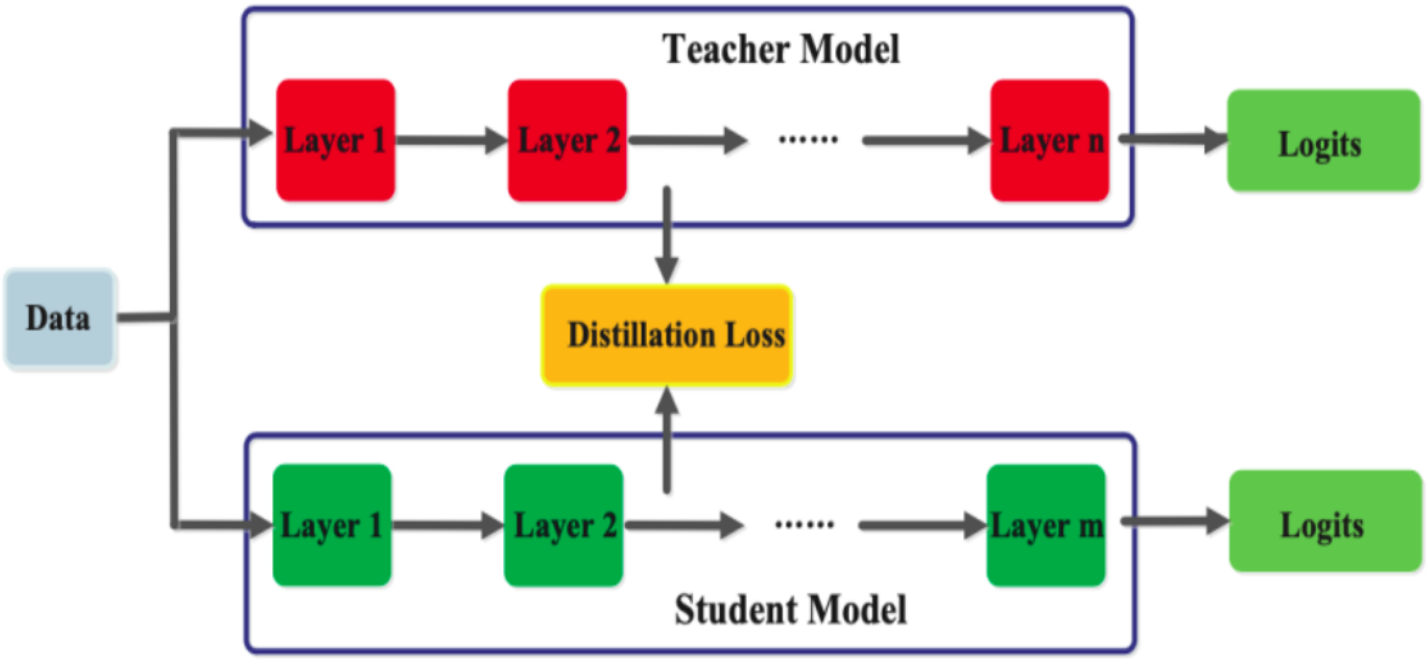
A **função de perda de distilação** mede o quanto as previsões do *Student* diferem das do *Teacher*.
Por exemplo, usando ***Kullback–Leibler Divergence*** **(KL)**:

$$L_{distill} = 0.7 \log \frac{0.7}{0.5} + 0.2 \log \frac{0.2}{0.4} + 0.1 \log \frac{0.1}{0.1} \approx 0.13$$

- **Interpretação:**
  O *Student* ainda não imita perfeitamente o *Teacher* (perda > 0).
  Ao treinar e reduzir essa perda, o *Student* aprende a se comportar mais como o *Teacher* mesmo sem ter acesso direto aos rótulos reais.

Inatel

Arquitetura e funcionamento – **Conhecimento baseado em recursos**



- **Camadas intermediárias:** O *Teacher*, além da saída final, captura conhecimento nas suas camadas internas, especialmente em redes profundas.
- **Extração de características:** Essas camadas aprendem a discriminar padrões específicos, como bordas, formas e texturas (em visão computacional).
- **Objetivo:** Treinar o *Student* para reproduzir as mesmas ativações de características aprendidas pelo *Teacher*.
- **Mecanismo:** Uma função de perda de destilação minimiza a diferença entre as ativações de características do *Teacher* e do *Student*.
- **Importância:** O *Student* aproveita representações ricas e estruturadas, acelerando o aprendizado e melhorando a generalização.

Inatel

Função de Perda – **Conhecimento baseado em recursos**

A **perda de distilação de recursos** mede a diferença entre esses vetores, por exemplo com o **erro quadrático médio (MSE):**

$$L_{distill} = \frac{1}{n}\sum_{i=1}^{n}(f_i^{Teacher} - f_i^{Student})^2$$

Onde:

- *Ldistill* representa a **função de perda de distilação**, que mede o quanto as **representações internas** do *Student* diferem das do *Teacher*.

- Cada *fiTeacher* e *fiStudent* corresponde às **ativaçãoes (*features*)** de uma mesma camada ou amostra.

- O valor é obtido pela **média do erro quadrático** entre elas, quanto menor *Ldistill* , mais o *Student* consegue **imitar o comportamento interno** do *Teacher*.

Inatel

Exemplo - Função de Perda – **Conhecimento baseado em recursos**

Durante o treinamento, queremos que o *Student* gere **ativação de camadas internas** parecidas com as do *Teacher*.
Por exemplo, comparando os mapas de ativação da camada 2:

Modelo	Ativações (simplificadas)
Teacher (camada 2)	[0.8, 0.1, 0.6, 0.3]
Student (camada 2)	[0.7, 0.2, 0.5, 0.4]

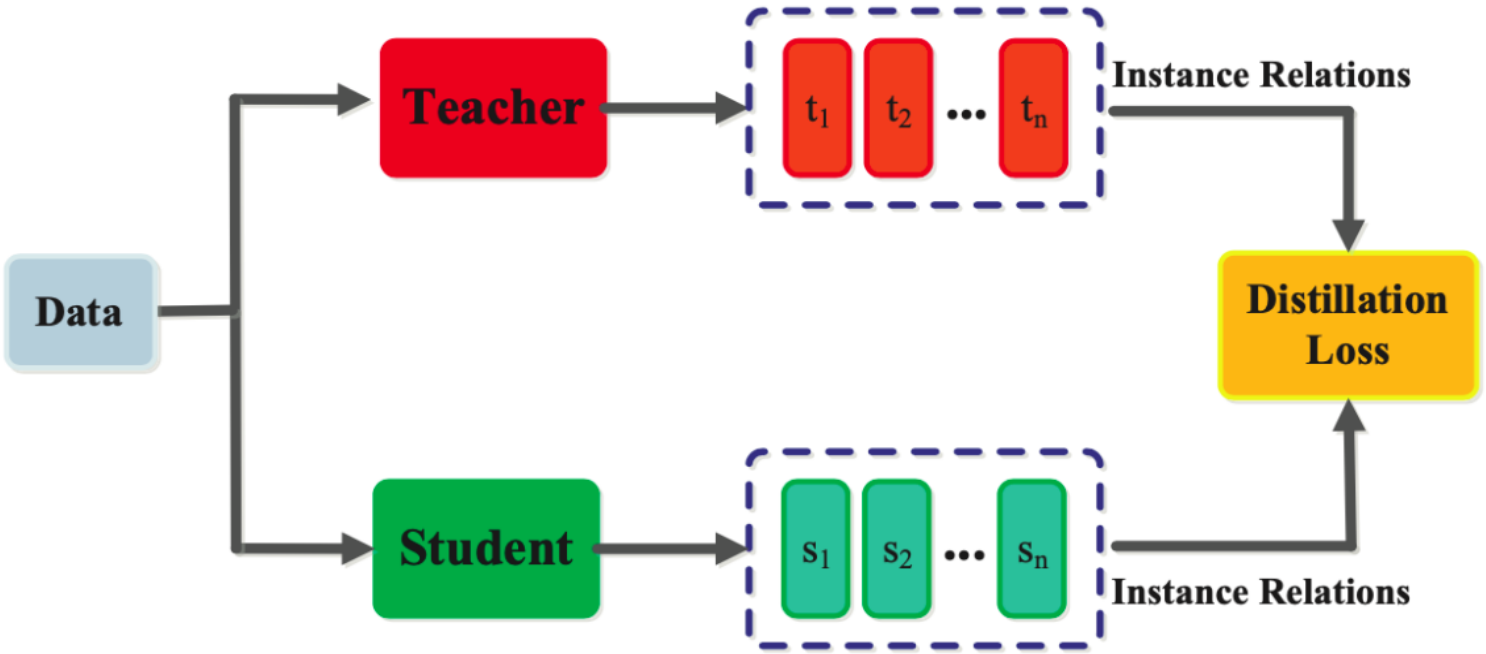
A **perda de distilação de recursos** mede a diferença entre esses vetores, por exemplo com o **erro quadrático médio (MSE):**

$$L_{distill} = \frac{1}{n}\sum_{i=1}^{n}(f_i^{Teacher} - f_i^{Student})^2 = 0.01$$

- **Interpretação:**
  O *Student* está aprendendo a **extrair padrões internos semelhantes** (bordas, formas, texturas), o que o ajuda a generalizar melhor — mesmo que ele tenha uma arquitetura menor.

Inatel

Arquitetura e funcionamento – **Conhecimento baseado em relações**



• **Foco:** Em vez de olhar só para as respostas finais ou ativações isoladas, o *Student* aprende **como diferentes partes da rede do *Teacher* se relacionam** entre si.

• **Objetivo:** Ensinar o *Student* a capturar **como diferentes representações se conectam** dentro da rede, e não apenas os valores individuais.

• **Formas de modelagem:**
  - **Correlação** entre mapas de características (o quanto "andam juntos").
  - **Grafos** que representam dependências entre neurônios/camadas.
  - **Matriz de similaridade** para medir proximidade entre representações.

• **Mecanismo:** Funções de perda são usadas para alinhar as relações do *Student* com as do *Teacher*.

• **Benefício:** O *Student* não copia só valores, mas entende a **estrutura oculta** do conhecimento, o que melhora a generalização e a captura de padrões complexos.

Inatel

Exemplo - Função de Perda – **Conhecimento baseado em relações**

Suponha que o *Teacher* gere duas representações internas para duas imagens:

$$t_1 = [0.9, 0.2], t_2 = [0.6, 0.7]$$

e o *Student*:

$$s_1 = [0.8, 0.3], s_2 = [0.5, 0.6]$$

Primeiro, medimos **como cada par de instâncias se relaciona** (ex.: usando similaridade de cosseno):

$$R_T = \cos(t_1, t_2) = 0.66 \quad \text{e} \quad R_S = \cos(s_1, s_2) = 0.64$$

A **função de perda relacional** mede a diferença entre essas relações:

$$L_{relational} = (R_T - R_S)^2 = (0.66 - 0.64)^2 = 0.0004$$

- **Interpretação:**
  O *Student* aprende **não apenas valores**, mas **como as representações se conectam**, ou seja, a **estrutura do conhecimento** do *Teacher*.
  Isso melhora a capacidade de generalização em tarefas mais complexas.

Inatel

Tipos de Conhecimentos - Resumo

Tipo de Conhecimento	Foco Principal	O que o *Student* aprende	Exemplos / Aplicação	Benefício
Baseado em Resposta	Camada de saída final (*logits*)	Reproduzir as previsões do *Teacher* (soft targets via *softmax*)	Classificação de imagens, aprendizado supervisionado.	Captura relações sutis entre classes, melhora generalização.
Baseado em Recurso	Camadas intermediárias	Ativações internas (padrões como bordas, formas, texturas)	Visão computacional, extração de características.	*Student* herda representações ricas sem reaprender do zero.
Baseado em Relação	Relações entre mapas de características	Estruturas e dependências entre representações	Correlação, grafos, matrizes de similaridade, embeddings.	*Student* entende conexões complexas, melhora robustez e generalização.

Inatel

Tipos de Destilação

Inatel

Arquitetura e funcionamento – **Destilação Offline**

- **Definição:** Método mais comum de *Knowledge Distillation*.
- **Processo:** *Teacher* é **pré-treinado** em um conjunto de dados e depois usado para guiar o *Student*.
- **Disponibilidade:** Grande variedade de **modelos pré-treinados** está disponível publicamente e pode ser usada como *Teacher*.
- **Vantagem:** Técnica **consolidada e mais simples de implementar** em comparação com outras abordagens.
- **Aplicação:** Útil em diferentes cenários, adaptando o *Teacher* ao caso de uso.

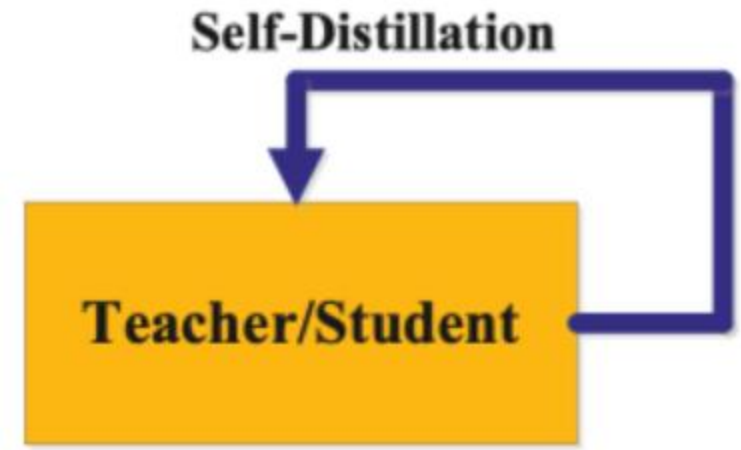
Inatel

Arquitetura e funcionamento – **Destilação Online**

- **Definição:** *Teacher* e *Student* são **treinados simultaneamente**.
- **Motivação:** Usada quando **não há modelo pré-treinado** disponível para atuar como *Teacher*.
- **Processo:** Ambos os modelos se atualizam ao mesmo tempo, compartilhando conhecimento durante o treinamento.
- **Execução:** Pode ser implementada com **computação paralela**, aumentando a eficiência.
- **Vantagem:** Método **altamente eficiente** e aplicável em cenários onde o pré-treinamento é inviável.

Inatel

Arquitetura e funcionamento – **Auto Destilação**



- **Definição:** O mesmo modelo atua como *Teacher* e *Student*, ou seja, ele ensina a si próprio durante o treinamento.

- **Processo interno:** As **camadas mais profundas** (que capturam representações mais complexas) fornecem sinais de aprendizado para treinar as **camadas mais superficiais**, fortalecendo a base da rede.

- **Variações de Aprendizado:**
  - Transferência entre **épocas**: o conhecimento obtido em épocas anteriores guia épocas posteriores.
  - Transferência entre **camadas**: informações de partes mais avançadas da rede ajudam as iniciais.

- **Categoria:** Considerada um **caso especial de destilação online**.

- **Vantagem:** Dispensa modelo externo, aproveitando melhor a própria rede para melhorar desempenho.

Inatel

Tipos de Destilação - Resumo

Tipo de Destilação	Definição	Processo	Vantagens	Limitações
Offline	*Teacher* pré-treinado guia o *Student*	*Teacher* é treinado antes e depois usado para destilar conhecimento	Simples de implementar; aproveita modelos pré-treinados disponíveis.	Depende da existência de um modelo *Teacher* robusto.
Online	*Teacher* e *Student* treinados simultaneamente	Ambos aprendem juntos em um processo de ponta a ponta	Altamente eficiente com computação paralela; não precisa de *Teacher* pré-treinado.	Mais complexo de implementar; exige maior poder computacional.
Autodestilação	O próprio modelo atua como *Teacher* e *Student*	Camadas profundas → camadas superficiais; épocas anteriores → épocas posteriores	Dispensa *Teacher* externo; melhora a própria rede.	Pode ter ganhos menores que distilação com Teacher maior e mais poderoso.

Inatel

Qual seria a melhor Destilação de Conhecimento?

- A **destilação offline** é o método tradicional, em que um modelo professor já treinado gera previsões fixas usadas para treinar o aluno. É eficiente e estável, mas não permite adaptação contínua.

- A **destilação online** realiza o treinamento conjunto do professor e do aluno, com o professor sendo atualizado durante o processo. Isso cria um aprendizado colaborativo e dinâmico, geralmente resultando em **melhor desempenho**, já que o aluno aprende de um professor em constante evolução.

- Na **self-distillation**, o próprio modelo atua como seu professor, refinando-se em iterações sucessivas. Apesar de não exigir um modelo externo, pode ter ganhos mais limitados, mas ainda melhora a generalização e reduz overfitting.

- Em síntese, **a destilação online tende a produzir os melhores resultados**, seguida pela self-distillation, enquanto a offline é a mais simples e estável.

Inatel

Exemplo de Aplicação

Inatel

YOLO

- **YOLO (*You Only Look Once*)**
- **O que é:** Algoritmo de **detecção de objetos em tempo real**.
- **Como funciona:** Analisa a imagem inteira de uma vez, prevendo **caixas delimitadoras (*bounding boxes*)** e **classes** simultaneamente.
- **Vantagem:** Muito rápido e eficiente, adequado para aplicações em **tempo real**.
- **Aplicações:** Vigilância, veículos autônomos, drones, análise de imagens médicas, entre outros.

Inatel

Exemplo de Aplicação

- Utilizou-se o **YOLOv5x** como ***Teacher*** e o **YOLOv5n** como ***Student*** para demonstrar a aplicação de ***Knowledge Distillation*** em detecção de objetos. Aplicando a **destilação *offline*.**

Modelo	Descrição	Papel na KD	Custo Computacional	Local de Uso
YOLOv5x	Versão **grande e robusta** do YOLOv5, com mais camadas e capacidade de aprendizado.	***Teacher***, fornece conhecimento detalhado e previsões precisas.	Alto, exige GPU potente para treinar e inferir.	Servidores, data centers, treinamentos offline.
YOLOv5n	Versão **compacta e eficiente** do YOLOv5, com menos parâmetros e menor custo.	***Student***, aprende com o *Teacher* para manter bom desempenho.	Baixo, rápido em CPU e dispositivos embarcados.	Dispositivos móveis, drones, edge computing, aplicações em tempo real.

- O exemplo de aplicação constitui um problema de "Classificação", mas o *Knowledge Distillation* pode ser usado em regressão e outros problemas.

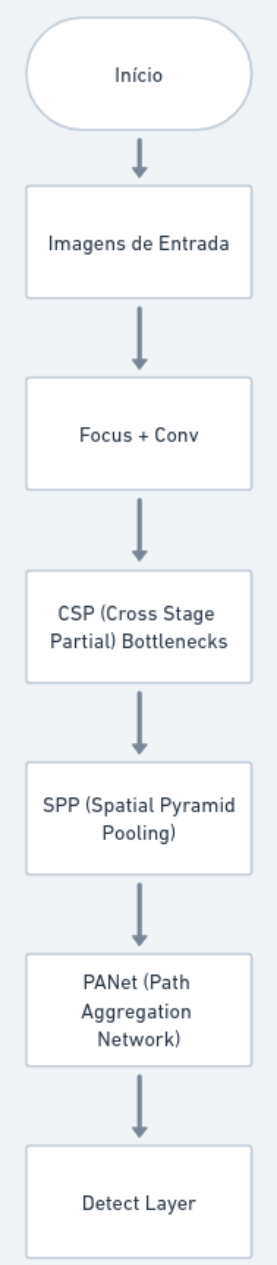
Inatel

YOLOv5x Vs YOLOv5n

Característica	YOLOv5x	YOLOv5n
Número de parâmetros	~97M (muito pesado)	~2,5M (muito leve)
Profundidade e largura	Mais camadas e filtros por camada → maior capacidade de representação.	Menos camadas e filtros → execução mais rápida.
Custo computacional (GFLOPs)	~246,9 GFLOPs (bilhões de operações por imagem).	~7,2 GFLOPs (≈34x mais leve)
Velocidade de inferência	Mais lento, indicado para GPUs potentes.	Muito rápido, ideal para dispositivos com recursos limitados.
Aplicação típica	Servidores e ambientes de alto desempenho, quando precisão máxima é prioridade.	Edge computing: smartphones, drones, IoT, onde velocidade é crítica.

Inatel

Arquitetura e funcionamento – **YOLOv5x e YOLOv5n**



- **Input Image**: Imagem de entrada do conjunto de folhas de tomate

- **Focus + Conv**: A camada *Focus* divide a imagem em partes menores e as concatena, aumentando a riqueza de informação. Extrai *features* iniciais. A convolução atua como um "filtro" que desliza sobre a imagem de entrada, buscando por características específicas

- **CSP (*Cross Stage Partial*) *Bottlenecks***: CSP divide os mapas de *feature* em blocos, processando apenas uma parte em cada etapa e depois combinando. Captura padrões complexos.

- **SPP (*Spatial Pyramid Pooling*)**: Permite que o modelo capture informações de diferentes escalas. Contexto multi-escala.

- **PANet (*Path Aggregation Network*)**: Integra informações de camadas profundas e superficiais. Fusiona features de diferentes níveis.

- **Detect Layer**: Gera as predições finais; bounding boxes que delimitam as regiões afetadas e a classe da doença. Previsão de bounding boxes e classes.

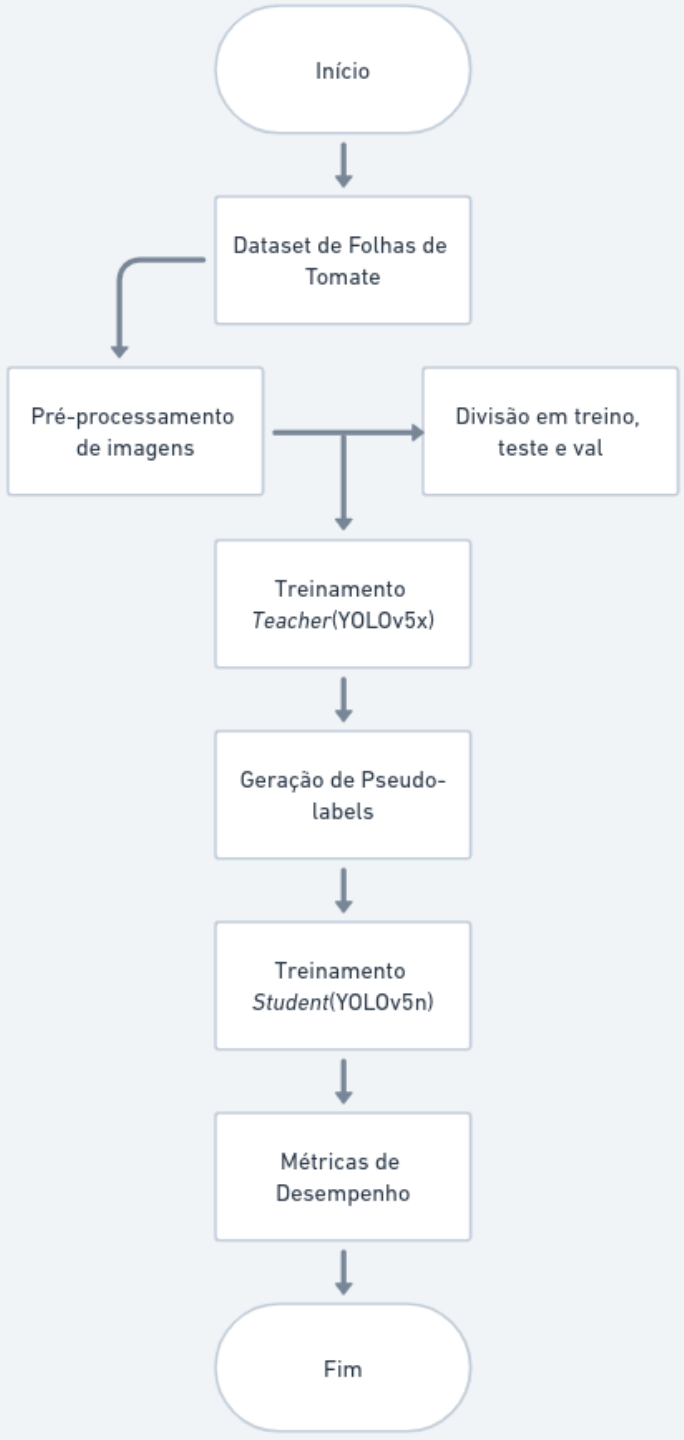
Exemplo de Aplicação

Dataset Utilizado

- **Base de dados:** *Tomato Dataset*
- **Características:**
- Total de imagens: **233**
- Divisão dos dados:
  - **111** para treinamento
  - **32** para validação
  - **90** para teste
- Tipo de dado: imagens de tomates para classificação/detecção

Configurações de Treinamento e Avaliação

Configuração	Valor	Descrição
Treino do *Teacher* e do *Student*	`imgsz=640`, `batch=5`, `epochs=10`	Resolução 640x640, mini-batch igual a 5, 10 épocas.
Avaliação	Métricas: Precision, Recall, mAP	Comparação *Teacher* vs *Student*
Comparação	Tamanho do modelo e FPS (Frames per Second)	Verifica a eficiência entre o *Teacher* e o *Student*.



- **Dataset de Folhas de Tomate**
  - Conjunto de imagens de folhas de tomate, categorizadas em diferentes doenças e folhas saudáveis.

- **Pré-processamento de Imagens**
  - Normalização, redimensionamento e possíveis *augmentations* (rotação, etc.).
  - Garantem que os modelos recebam dados consistentes e robustos.

- ***Split* em *train*, *val* e *test***
  - **Train:** conjunto usado para treinar o modelo.
  - **Val:** conjunto de validação, criado para monitorar performance durante o treinamento e evitar *overfitting*.
  - **Test:** conjunto final para avaliar o desempenho real do modelo.

- **Treinamento Teacher (YOLOv5x)**
  - Modelo robusto e pesado, com alta capacidade de aprendizado.

- **Geração de Pseudo-labels**
  - O *Teacher* é usado para gerar **pseudo-labels**, que são **labels preditas automaticamente para imagens do *train set***.
  - Essencial para transferir conhecimento do modelo grande para o modelo leve.

- **Treinamento Student (YOLOv5n)**
  - Aplicando a **destilação *offline*.**
  - Treinado usando pseudo-labels do *Teacher*.

- **Avaliação e Comparação**
  - Medição de métricas como: *Precision*, *Recall,* velocidade de inferência (FPS), etc.

Resultados do *Teacher* (YOLOv5x)

Época	Tempo (s)	Box Loss	Cls Loss	DFL Loss	Precisão	Recall	mAP@0.5	mAP@0.5:0.95	Val Box Loss	Val Cls Loss	Val DFL Loss
1	481.60	0.867	2.533	1.596	0.0566	0.969	0.0756	0.0373	1.031	259.564	61.572
5	2675.35	0.954	1.388	2.049	0.0184	1.000	0.0207	0.0063	2.591	4140.030	19.693
10	6427.91	0.418	0.566	1.322	0.9931	0.969	0.9941	0.9011	0.485	1.062	0.726

- **Box Loss** → erro na predição das caixas delimitadoras (quanto menor, melhor).
- **Cls Loss** → erro na classificação das classes dentro das caixas.
- **DFL Loss (Distribution Focal Loss)** → perda usada para melhorar a precisão das bordas das caixas.
- **Precisão (Precision)** → proporção de predições corretas entre todas as predições positivas feitas.
- **Recall** → proporção de verdadeiros positivos capturados em relação ao total existente.
- **mAP@0.5** → *mean Average Precision* com limiar de IoU = 0.5 (quanto maior, melhor).
- **mAP@0.5:0.95** → média do mAP em diferentes limiares de IoU (0.5 a 0.95, mais rigoroso).
- **Val Box Loss** → perda de localização (caixas) medida no conjunto de validação.
- **Val Cls Loss** → perda de classificação medida no conjunto de validação.
- **Val DFL Loss** → perda de distribuição focal medida no conjunto de validação.

Diferença entre mAP@0.5 e mAP@0.5:0.95

- **mAP@0.5**
  - Mede a qualidade das predições considerando **IoU ≥ 50%**.
  - Exemplo: se a caixa prevista cobre pelo menos metade da caixa real, conta como acerto.
  - É **mais permissivo**, por isso costuma ter valores mais altos.
- **mAP@0.5:0.95**
  - Calcula a média do mAP em vários níveis: **IoU de 0.5 até 0.95 (de 0.05 em 0.05)**.
  - Exige que o modelo seja bom tanto em sobreposições "fáceis" (50%) quanto em **muito precisas** (95%).
  - É **mais rigoroso** e mostra se o modelo realmente localiza os objetos com alta exatidão.
- Link para a compreensão do cálculo do IoU: https://learnopencv.com/intersection-over-union-iou-in-object-detection-and-segmentation/

Inatel

Resultados do *Student* (YOLOv5n)

Época	Tempo (s)	Box Loss	Cls Loss	DFL Loss	Precisão	Recall	mAP@0.5	mAP@0.5:0.95	Val Box Loss	Val Cls Loss	Val DFL Loss
1	34.37	0.826	2.788	1.538	0.0033	1.000	0.995	0.995	0.195	2.572	0.304
5	153.52	0.262	1.147	1.104	0.998	1.000	0.995	0.918	0.380	1.129	0.589
10	302.72	0.140	0.756	1.007	0.998	1.000	0.995	0.995	0.285	0.911	0.436

- **Box Loss** → erro na predição das caixas delimitadoras (quanto menor, melhor).
- **Cls Loss** → erro na classificação das classes dentro das caixas.
- **DFL Loss (*Distribution Focal Loss*)** → perda usada para melhorar a precisão das bordas das caixas.
- **Precisão (*Precision*)** → proporção de predições corretas entre todas as predições positivas feitas.
- **Recall** → proporção de verdadeiros positivos capturados em relação ao total existente.
- **mAP@0.5** → *mean Average Precision* com limiar de IoU = 0.5 (quanto maior, melhor).
- **mAP@0.5:0.95** → média do mAP em diferentes limiares de IoU (0.5 a 0.95, mais rigoroso).
- **Val Box Loss** → perda de localização (caixas) medida no conjunto de validação.
- **Val Cls Loss** → perda de classificação medida no conjunto de validação.
- **Val DFL Loss** → perda de distribuição focal medida no conjunto de validação.

Comparação Final (Época 10)

Modelo	Tempo (s)	Box Loss	Cls Loss	DFL Loss	Precisão	Recall	mAP@0.5	mAP@0.5:0.95	Val Box Loss	Val Cls Loss	Val DFL Loss
YOLOv5n	302.72	0.140	0.756	1.007	0.998	1.000	0.995	0.995	0.285	0.911	0.436
YOLOv5x	6427.91	0.418	0.566	1.322	0.993	0.969	0.994	0.901	0.485	1.062	0.726

- **Evolução das perdas**
  - As métricas *Box Loss*, *Cls Lloss* e *DFL Loss* diminuem ao longo das épocas, mostrando que o modelo está aprendendo a localizar e classificar melhor.
- **Precisão (*Precision*)**
  - Mede quantos dos positivos detectados estavam corretos.
  - No ***Student***, ficou muito alto logo desde as primeiras épocas (~0.99).
  - No ***Teacher***, precisou de mais épocas para estabilizar.
- **Recall**
  - Mede quantos objetos reais o modelo conseguiu detectar.
  - Ambos atingem valores próximos de 1, o que indica boa cobertura das classes.
- **mAP@0.5 e mAP@0.5:0.95**
  - ***Teacher*** demorou mais para estabilizar, mas atingiu bons valores (0.994 e 0.901).
  - ***Student*** atingiu valores estáveis rapidamente, mostrando eficiência mesmo com menos parâmetros.
- **Tempo de treinamento**
  - O ***Teacher*** é muito mais pesado, cada época levou centenas de segundos.

Resultados do YOLOv5n sem Destilação

Época	Tempo (s)	Box Loss	Cls Loss	DFL Loss	Precisão	Recall	mAP@0.5	mAP@0.5:0.95	Val Box Loss	Val Cls Loss	Val DFL Loss
1	34.0	0.7228	2.693	1.467	0.00608	1.0	0.995	0.978	0.00608	1.0	0.978
5	26.0	0.2283	0.99	1.08	0.998	1.0	0.995	0.992	0.998	1.0	0.992
10	26.0	0.1486	0.7276	0.9997	0.998	1.0	0.995	0.992	0.998	1.0	0.992

- **Box Loss** → erro na predição das caixas delimitadoras (quanto menor, melhor).
- **Cls Loss** → erro na classificação das classes dentro das caixas.
- **DFL Loss (*Distribution Focal Loss*)** → perda usada para melhorar a precisão das bordas das caixas.
- **Precisão (*Precision*)** → proporção de predições corretas entre todas as predições positivas feitas.
- **Recall** → proporção de verdadeiros positivos capturados em relação ao total existente.
- **mAP@0.5** → *mean Average Precision* com limiar de IoU = 0.5 (quanto maior, melhor).
- **mAP@0.5:0.95** → média do mAP em diferentes limiares de IoU (0.5 a 0.95, mais rigoroso).
- **Val Box Loss** → perda de localização (caixas) medida no conjunto de validação.
- **Val Cls Loss** → perda de classificação medida no conjunto de validação.
- **Val DFL Loss** → perda de distribuição focal medida no conjunto de validação.

Comparação Final Yolov5n Com e Sem Destilação

Modelo	Tempo (s)	Box Loss	Cls Loss	DFL Loss	Precisão	Recall	mAP@0.5	mAP@0.5:0.95	Val Box Loss	Val Cls Loss	Val DFL Loss
Yolov5n Com Destilação	302.72	0.140	0.756	1.007	0.998	1.0	0.995	0.995	0.285	0.911	0.436
Yolov5n Sem Destilação	327.6	0.1486	0.7276	0.9997	0.998	1.0	0.995	0.992	0.998	1.0	0.992

- Evolução das Perdas
  - YOLOv5n Sem Destilação: Apresenta uma queda clara e constante em todas as perdas (Box Loss, Cls Loss, DFL Loss) ao longo das 10 épocas, indicando um aprendizado rápido.
  - YOLOv5n Com Destilação: A Box Loss final (0.140) é marginalmente menor que a versão Sem Destilação, sugerindo um ganho sutil na precisão de localização.

- Precisão (Precision - P) e Recall (R)
  - Precisão (P): Ambas as versões atingiram um valor final de 0.998. O modelo Sem Destilação atingiu e estabilizou neste valor muito rapidamente, mostrando excelente capacidade de evitar Falsos Positivos.
  - Recall (R): Ambas as versões alcançaram o valor perfeito de 1.000.

- mAP@0.5 e mAP@0.5:0.95
  - mAP@0.5: O valor final é idêntico em ambas as versões: 0.995. O modelo Sem Destilação atingiu esse valor já na primeira época, demonstrando a facilidade do problema ou a robustez do setup.
  - mAP@0.5:0.95: O modelo Com Destilação obteve um desempenho marginalmente superior (0.995) em comparação com o modelo Sem Destilação (0.992), indicando uma melhoria na localização das caixas de delimitação em limiares de IoU mais estritos.

- Tempo (Time)
  - **YOLOv5n Com Destilação: 302.72 segundos**.
  - **YOLOv5n Sem Destilação: 327.6 segundos.**

Vantagens e Desvantagens: *Knowledge Distillation*

Ponto de Comparação	Vantagens	Desvantagens
Tamanho do modelo	Student pode ser muito menor que o Teacher, economizando memória.	Redução excessiva pode perder desempenho.
Velocidade de inferência	Modelos destilados rodam mais rápido, ideais para dispositivos edge/mobile.	Aceleração depende da arquitetura escolhida; nem sempre é significativa.
Generalização	Captura relações sutis entre classes (via logits/soft labels).	Pode herdar vieses e erros do Teacher.
Treinamento	Aproveita conhecimento de modelos já treinados, acelerando aprendizado.	Processo de destilação adiciona complexidade (precisa treinar Teacher + Student).
Aplicação prática	Permite deploy de IA em ambientes com restrição de recursos.	Requer acesso a um Teacher adequado e bem treinado.

Perguntas?

Referências

[1] **HINTON, Geoffrey; VINYALS, Oriol; DEAN, Jeff.** Distilling the knowledge in a neural network. *arXiv*, 9 mar. 2015. Disponível em: https://arxiv.org/pdf/1503.02531

[2] **AMRANI, Abderraouf et al.** Insect detection from imagery using YOLOv3-based adaptive feature fusion convolution network. *Crop and Pasture Science*, v. 74, n. 6, p. 615–627, 2022. Disponível em: https://www.publish.csiro.au/cp/pdf/CP21710

[3] **TEKI, Sundeep.** Knowledge distillation: princípios, algoritmos e aplicações. *Neptune.ai*, 29 set. 2023. Disponível em: https://neptune.ai/blog/knowledge-distillation

[4] **GANESH, Prakhar.** Knowledge distillation simplified. *Towards Data Science*, 27 mar. 2020. Disponível em: https://towardsdatascience.com/knowledge-distillation-simplified-dd4973dbc764

Referências

[5] **BUCILUǍ, Cristian; CARUANA, Rich; NICULESCU-MIZIL, Alexandru.** Model compression. In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '06)*. New York: ACM, 2006. p. 535–541. DOI: 10.1145/1150402.1150464.

[6] **KWABENA, Patrick Mensah et al.** Dataset for crop pest and disease detection. *Mendeley Data*, v. 1, 26 abr. 2023. DOI: 10.17632/bwh3zbpkpv.1. Disponível em: https://data.mendeley.com/datasets/bwh3zbpkpv/1

[7] **JOCHER, Glenn.** YOLOv5. GitHub, 2020. Disponível em: https://github.com/ultralytics/yolov5

Inatel

Repositório GitHub

- Link de acesso ao repositório:

- https://github.com/PauloLuczensky/Knowledge-Distillation

Inatel

GitHub

Quizz

- Link de acesso ao Quizz:

https://docs.google.com/forms/d/e/1FAIpQLScffH2pj0BP74CGNaK9KljXnDXpirXIWqp88cDxi1jlNQvZNQ/viewform?usp=sharing&ouid=106472963272670763809

Inatel

Obrigado!