UNIVERSIDADE DO VALE DO ITAJAÍ
ESCOLA POLITÉCNICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

IDENTIFICAÇÃO DE SOLO EXPOSTO E CUPINZEIROS EM PASTAGENS

Ian Pinto de Almeida

Orientador (a): Anita Maria da Rocha Fernandes, Dra.

Itajaí, junho/2024

UNIVERSIDADE DO VALE DO ITAJAÍ
ESCOLA POLITÉCNICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

IDENTIFICAÇÃO DE SOLO EXPOSTO E CUPINZEIROS EM PASTAGENS

Ian Pinto de Almeida

Itajaí, 06/2024

Orientador (a): Anita Maria da Rocha Fernandes, Dra.
Área de Concentração: Inteligência Artificial.
Linha de Pesquisa: Inteligência Aplicada ao Meio Ambiente
Palavras-chave: Agricultura 5.0, Aprendizado de máquina, Reconhecimento de padrões, Degradação de Pastagens.
Número de páginas: 82

RESUMO

A degradação de pastagens é um desafio significativo na prática da pecuária no Brasil, afetando a sustentabilidade ambiental e econômica do setor. Este fenômeno ocorre devido ao manejo inadequado, sobrepastoreio, desmatamento e uso intensivo de terras para a criação de gado. Soluções que auxiliem na identificação de solo exposto e cupinzeiros forneçam subsídios para a melhoria das pastagens são extremamente relevantes. Tecnologia de ponta refere-se a inovações e avanços recentes que estão na vanguarda do desenvolvimento em um determinado campo. No contexto da Agricultura 5.0, tecnologias de ponta incluem avanços como inteligência artificial, automação, sensores avançados e outras soluções tecnológicas de última geração aplicadas ao setor agropecuário. Essas tecnologias visam otimizar processos, aumentar a eficiência e promover uma abordagem mais sustentável na agricultura, sendo essencial para contribuir com a melhora de processos como o manejo e desempenho das pastagens. Dessa forma é possível propor modelos para identificação de solo exposto, bem como indicadores de degradação da pastagem, tais como a quantidade de cupinzeiros em uma determinada área. Sendo assim, esse trabalho propõe a análise dos modelos You Only Look Once (YOLO), YOLOv4, YOLOv8n e YOLOv8x de aprendizado de máquina, para identificar pasto, solo exposto e cupinzeiros em pastagens. O apoio técnico da EMBRAPA Milho e Sorgo foi imprescindível para a classificação de imagens que compõem a base de dados. Essas imagens foram fornecidas pelo Laboratório de Processamento de Imagens e Geoprocessamento (LAPIG). Os resultados demonstram que a utilização de detecção de objetos para analisar níveis de degradação em pastagens é promissor apesar de uma precisão impraticável de 53,9%. Para que uma precisão de 80% a 90% possa ser atingida, é preciso superar a principal barreira encontrada nesse trabalho que é a quantidade limitada de imagens rotuladas na base de dados.

UNIVERSIDADE DO VALE DO ITAJAÍ
ESCOLA POLITÉCNICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

IDENTIFICAÇÃO DE SOLO EXPOSTO E CUPINZEIROS EM PASTAGENS

Ian Pinto de Almeida

Itajaí, 06/2024

Advisor: Anita Maria da Rocha Fernandes, Ph.D.
Concentration area: Artificial Inteligence.
Research line: Applied Intelligence to the Environment.
Key-words: Agriculture 5.0, Machine Learning, Pattern Recognition, Pasture Degradation.
Number of pages: 82

ABSTRACT

Pasture degradation is a significant challenge in livestock farming in Brazil, affecting the environmental and economic sustainability of the sector. This phenomenon occurs due to inadequate management, overgrazing, deforestation and intensive use of land for livestock farming. Solutions that help identify exposed soil and termite mounds and provide support for improving pastures are extremely relevant. Cutting-edge technology refers to recent innovations and advancements that are at the forefront of development in a given field. In the context of Agriculture 5.0, cutting-edge technologies include advances such as artificial intelligence, automation, advanced sensors and other cutting-edge technological solutions applied to the agricultural sector. These technologies aim to optimize processes, increase efficiency and promote a more sustainable approach to agriculture, being essential to contribute to the improvement of processes such as pasture management and performance. This way, it is possible to propose models for identifying exposed soil, as well as indicators of pasture degradation, such as the number of termite mounds in a given area. Therefore, this work proposes the analysis of the You Only Look Once (YOLO), YOLOv4, YOLOv8n and YOLOv8x machine learning models, to identify pasture, exposed soil and termite mounds in pastures. The technical support from EMBRAPA Milho e Sorgo was essential for classifying the images that make up the database. These images were captured from pastures in the regions of Goiás and Mato Grosso and are being provided by the Image Processing and Geoprocessing Laboratory (LAPIG). The results demonstrate that the use of object detection to analyze degradation levels in pastures is promising despite an impractical accuracy of 53.9%. In order for an accuracy of 80% to 90% to be achieved, it is necessary to overcome the main barrier found in this work, which is the limited number of labeled imagens in the database.

LISTA DE FIGURAS

LISTA DE TABELAS E QUADROS

LISTA DE ABREVIATURAS E SIGLAS

AP	Agricultura de precisão
AP	*Average Precision*
AM	Aprendizado de Máquina
BFLOP	*Billion Floating-point Operations*
CNN	*Convolutional Neural Network*
CSP	*Cross-Stage Partial*
IoT	*Internet of Things*
FN	Falso Negativo
FP	Falso Positivo
FPS	*Frames Per Second*
FPN	*Feature Pyramid Network*
GPU	*Graphics Processing Unit*
LAPIG	Laboratório de Processamento de Imagens e Geoprocessamento
LASSO	*Least Absolute Shrinkage and Selection Operator*
MLP	*Multi-layer Perceptron*
NDVI	*Nomalized Difference Vegetation Index*
PAN	*Path Aggregation Network*
PDI	*Pasture Degradation Index*
ReLU	*Rectified Linear Unit*
SAM	*Spatial Attention Module*
SIG	Sistema de Informação Geográfica
SPP	*Spatial Pyramid Pooling*
SVM	*Support Vector Machine*
SVR	*Support Vector Regression*
TIC	Tecnologia da Informação e Comunicação
VANT	Veículo Aéreo Não Tripulado
VN	Verdadeiro Negativo
VP	Verdadeiro Positivo
YOLO	*You Only Look Once*

FORMULÁRIO

SUMÁRIO

1 INTRODUÇÃO

Segundo MapBiomas (2021), o principal uso do solo brasileiro é a pastagem: ela ocupa 154 milhões de hectares de norte a sul do país que constitui aproximadamente 18% dos 850 milhões de hectares estimados para o tamanho do território brasileiro, com presença em todos os seis biomas (Amazônia, Caatinga, Cerrado, Pantanal, Mata Atlântica e Pampa). A análise das imagens de satélites entre 1985 e 2020 permitiu avaliar a qualidade das pastagens brasileiras e constatar uma queda nas áreas com sinais de degradação de 70% em 2000 para 53% em 2020. No caso das pastagens severamente degradadas houve uma redução ainda mais expressiva; representavam 29% das pastagens em 2000 (46,3 milhões de hectares) e agora representam 14% (22,1 milhões de hectares) (MAPBIOMAS, 2021).

A degradação das pastagens pode ocorrer por uma série de fatores, devida a má prática e negligência no manejo das pastagens que se refere às práticas adotadas para garantir a saúde e a produtividade das áreas de pastagens (CARVALHO et al., 2017; DIAS-FILHO, 2023). Portanto, áreas que passam por uma conversão inadequada para serem utilizadas como pastagens ou terras agrícolas, pastagens que passam por períodos de pastoreio intensivo, ou seja, manter o gado em uma mesma área de pasto sem garantir um período de descanso (DIAS FILHO, 2023), e, a implantação de espécies de forrageiras que são as plantas cultivadas para servir de alimento aos animais (KLUTHCOUSKI et al., 2013; DIAS-FILHO, 2023), quando, não compatíveis com a pastagens são alguns dos fatores que levam a pastagem a se enquadrar em algum nível de degradação (DIAS FILHO, 2023).

Outro indicador de degradação das pastagens são os cupins-de-montículo. Esses insetos são sensíveis às mudanças no ambiente e podem sinalizar má qualidade do solo e na forragem da pastagem. A presença desses montículos pode indicar áreas onde o solo está compactado, pobre de nutrientes ou com algum grau de erosão. Sendo assim, o cupim-de-montículo é um indicador valioso para verificar a existência de problemas e alteração no solo e na vegetação (VALÉRIO, 2006).

Apesar da queda na quantidade de pastagens degradadas ou em algum estado de degradação, o setor agropecuário tem como um de seus objetivos continuar reduzindo esses valores, e, a modernização do processo de manejo de pastagens através de tecnologias de ponta é um fator crucial para continuar reduzindo a quantidade de pastagens degradadas.

A introdução da Agricultura de Precisão (AP) marcou o início de uma era revolucionária na agricultura, ao fazer uso de tecnologias para obter benefícios máximos. A AP concentra-se nas variabilidades específicas do local dentro do campo, a fim de abordar todas as preocupações de forma precisa (BERNARDI et al., 2014). Tecnologias como SIG (Sistema de Informação Geográfica), GPS, satélites, sistemas de sensoriamento remoto, sensores, entre outros, têm sido amplamente utilizadas para a aplicação da AP. A produção, produtividade e lucratividade melhoraram em comparação com a agricultura tradicional a qual é fundamentada em métodos convencionais, como a aplicação de insumos em toda a área de cultivo, utilização de equipamentos agrícolas que não incorporam tecnologias avançadas, decisões de manejo baseado em observações visuais e experiência práticas invés de utilizar uma análise de dados detalhada, entre outros. Além disso, uma gestão melhor foi alcançada por meio da AP. Em resumo, a AP trata totalmente da otimização dos insumos para obter um resultado máximo, de acordo com as variações no campo (AHMAD; NABI, 2021).

Os avanços nas tecnologias de satélite, sensores remotos e drones estão crescendo rapidamente, proporcionando imagens de alta qualidade que requerem processamento eficiente para aplicações agrícolas inteligentes. As recentes tecnologias de aprendizado profundo permitem a integração da visão computacional e inteligência artificial na agricultura, lidando com grandes volumes de dados para auxiliar na tomada de decisões (AHMAD; NABI, 2021).

Tombe (2020) apresenta uma abordagem estruturada que utiliza visão computacional para caracterizar imagens de plantações e determinar o estado de saúde delas. Uma rede neural convolucional profunda é aplicada para extrair e representar as características das imagens, que são então alimentadas em um modelo de máquina de vetores de suporte (SVM – Support Vector Machine) para treinamento e interpretação das imagens. Os resultados experimentais mostram que essa técnica proposta supera outros métodos existentes na interpretação visual das imagens de cena.

A integração de soluções baseadas em Tecnologia da Informação e Comunicação (TIC) na agricultura permite alcançar a segurança de produção, redução dos impactos ambientais e a sustentabilidade agrícola. A utilização dessas soluções combinadas, possibilita melhores resultados na interpretação das imagens de cena, contribuindo para a agricultura de precisão.

1.1 PROBLEMA DE PESQUISA

A maior ocupação do território brasileiro se dá para as pastagens, que desde o início das práticas de formação das pastagens, apresentam algum tipo de nível de degradação. Sendo assim, a degradação é prejudicial à produtividade da pastagem, consequentemente, reduz a eficiência do uso da pastagem pelo gado. Diversos fatores como, o pastoreio intensivo, o cultivo de espécie forrageira inadequada, plantas daninhas, pragas e doenças degradam as pastagens.

Dessa forma, é essencial que práticas e ferramentas eficazes para o manejo de pastagens sejam utilizadas nas pastagens brasileiras. Para isso, ferramentas e tecnologias que são utilizadas em países mais desenvolvidos, ainda não estão muito difundidas na agropecuária brasileiras. Portanto, almejando a modernização do manejo das pastagens, este trabalho pretende desenvolver uma ferramenta moderna e eficaz para auxiliar o manejo das pastagens no Brasil.

1.1.1 Solução Proposta

Este trabalho propõe o desenvolvimento e análise de um modelo classificação baseado em aprendizado de máquina através da ferramenta *You Only Look Once* (YOLO) que possa ser embarcado para identificar áreas de pasto, áreas de solo exposto e cupinzeiros em pastagens, visando complementar a análise de nível de degradação das pastagens, através da identificação dos elementos propostos.

1.1.2 Delimitação do Escopo

Portanto, este trabalho de conclusão de curso visa desenvolver um modelo de classificação baseado em aprendizado de máquina supervisionado utilizando a ferramenta YOLO para a identificação de áreas de solo exposto e cupinzeiros em pastagens.

A base de dados contendo imagens e vídeos capturados por drones das regiões do Mato Grosso e Goiás foi fornecida via Google Drive pelo Laboratório de Processamento de Imagens e Geoprocessamento da Universidade Federal de Goiás (LAPIG).

Os itens a serem detectados pelo modelo de classificação são os cupinzeiros, áreas de solo exposto e áreas de pasto. Esses elementos serão identificados nas imagens da base de dados fornecido pelo LAPIG.

1.1.3 Justificativa

O projeto proposto visa abordar desafios do manejo das áreas de pastagens brasileiras, que desempenham um papel central na agropecuária do país. Apesar de uma redução na degradação ao longo dos anos, as pastagens ainda enfrentam problemas, destacando a necessidade de estratégias inovadoras. A identificação de áreas de solo exposto e cupinzeiros surge como indicadores cruciais de degradação, fornecendo insights sobre a qualidade do solo e a saúde da pastagem (MAPBIOMAS, 2021; VALÉRIO, 2006).

A integração de tecnologias avançadas, como o YOLO para visão computacional, representa uma abordagem promissora para monitorar e avaliar a saúde das pastagens. Inspirado pelos sucessos da Agricultura de Precisão (AP) em otimizar o uso de insumos agrícolas (BERNARDI et al., 2014), o projeto propõe a aplicação dessas técnicas no contexto específico das pastagens brasileiras. Avanços recentes em tecnologias de sensoriamento remoto e aprendizado profundo oferecem a capacidade de processar eficientemente grandes volumes de dados de imagens de satélite, drones e sensores para identificar áreas de interesse com alta precisão (AHMAD; NABI, 2021).

Experiências bem-sucedidas em outras regiões, como a classificação de ervas daninhas em pastagens australianas (OLSEN et al., 2019), fornecem um precedente otimista para a aplicação de métodos similares no contexto brasileiro. A integração de Tecnologia da Informação e Comunicação (TIC) na agricultura se mostra essencial, ampliando a interpretação de imagens de cena e contribuindo para uma gestão mais eficiente das pastagens, promovendo, assim, a sustentabilidade agrícola (TOMBE, 2020).

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Identificar áreas de pasto, solo exposto e cupinzeiros em pastagens por meio de aprendizado de máquina.

1.2.2 Objetivos Específicos

1. Criar uma base de dados rotuladas com imagens de áreas de solo exposto e cupinzeiros;
2. Selecionar os algoritmos de aprendizado de máquina;

3. Analisar o desempenho dos algoritmos;
4. Criar o modelo de classificação; e
5. Propor uma abordagem para a análise de área degradada X área produtiva.

1.3 METODOLOGIA

Nesta seção, será descrita a classificação deste trabalho, a metodologia adotada na pesquisa e serão apresentados os procedimentos metodológicos utilizados para o desenvolvimento deste trabalho de conclusão de curso.

1.3.1 Metodologia de Pesquisa

A abordagem metodológica adotada para este estudo é centrada na pesquisa aplicada, com o intuito de desenvolver um modelo de aprendizado de máquina voltado para a identificação de áreas de solo exposto e cupinzeiros em pastagens. Foi realizada uma abordagem qualitativa, dada a natureza do problema, que apresenta nuances não quantificadas e conceituais, como a escassez de informações sobre o nível de infestação em pastagens.

A condução do estudo seguiu uma abordagem experimental, com objetivo exploratório. Esta escolha se justifica pela lacuna existente na literatura em relação à investigação dessa temática em pastagens, tornando essencial uma exploração mais aprofundada. Enquanto há uma grande quantidade de trabalhos dedicados à identificação de níveis de degradação em pastagens através de imagens de satélite, observa-se uma notável ausência de pesquisas aplicadas nesse intuito que possuam uma base de dados diferente.

1.3.2 Procedimento Metodológicos

Os passos metodológicos foram iniciados por meio de uma revisão abrangente da literatura, focalizando trabalhos relacionados ao emprego de aprendizado de máquina na identificação de índices de degradação em pastagens. O propósito primordial era avaliar a viabilidade da aplicação de técnicas de aprendizado de máquina para abordar a problemática de pesquisa.

A etapa subsequente envolveu a pesquisa e análise de ferramentas, tecnologias, metodologias e modelos pertinentes, visando compreender a abordagem mais eficaz para manipular as imagens.

Na fase seguinte, delineou-se o critério para a seleção, classificação e separação das imagens. Esses procedimentos serão conduzidos em colaboração com um especialista engenheiro agrônomo para assegurar uma abordagem apropriada.

A implementação do modelo aprendizado de máquina utilizando o método de aprendizado supervisionado, o YOLO e as ferramentas escolhidas nas fases anteriores foram o foco da etapa subsequente.

Por fim, a última etapa contemplou a execução de testes e validações em conjunto com o especialista engenheiro agrônomo. O objetivo foi avaliar em que medida a abordagem proposta pode contribuir efetivamente para a resolução da problemática identificada.

1.4 ESTRUTURA DO TRABALHO

Este trabalho está dividido em quatro capítulos. No Capítulo 1, Introdução, são abordados a visão geral do estudo, as justificativas, objetivos e a metodologia. O Capítulo 2, Fundamentação Teórica e Capítulo 3, Trabalhos Relacionados, apresentam um referencial teórico para o desenvolvimento do trabalho proposto. Nesses capítulos é apresentada uma revisão bibliográfica sobre o manejo de pastagens dentro do movimento da Agricultura Digital, análise de degradação de pastagens, práticas de aprendizado de máquina na agricultura, técnicas de aprendizado de máquina, algoritmos e a ferramenta YOLO. O Capítulo 4 demonstra o desenvolvimento da pesquisa. O Capítulo 5 aborda e discute os resultados obtidos. Por fim, o Capítulo 6 apresenta as conclusões e os trabalhos futuros relativos a pesquisa.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são expostos os conceitos e conteúdos pertinentes para o desenvolvimento, implementação e validação da solução proposta neste trabalho. Inicialmente, é apresentado sobre o manejo de pastagens, pastagens degradadas e a Agricultura Digital. Em seguida, é discutida a aplicabilidade do aprendizado de máquina em reconhecimento de padrões em imagens. Por fim, é realizada uma análise sobre trabalhos correlatos, bem como, um comparativo com este trabalho.

2.1 MANEJO DE PASTAGENS

A formação de pastagens pode ser feita desde a retirada de árvores e queima de vegetação nativa até a reutilização de áreas que já foram utilizadas para o plantio (FEIGL; CARLOS, 1995; DIAS FILHO, 2023). A formação e manejo inadequado das pastagens podem causar limitações nutricionais no solo que se tornam evidentes quando a quantidade e qualidade da forragem disponível não satisfazem as exigências dos animais. Essas restrições podem ocorrer tanto por períodos breves quanto prolongados, de acordo com a extensão da estação de crescimento das plantas forrageiras (SERRÃO; FALESI; VEIGA, 1982; DIAS FILHO, 2023).

As pastagens raramente se encontram em equilíbrio. Com frequência, o consumo dos animais varia, e nem sempre está de acordo com a quantidade de produção das forragens (COSTA, MAGALHÃES; BENDAHAN, 2022). Isso resulta em situações indesejáveis, como o superpastejo, que diminui a produtividade e a qualidade da pastagem, afetando a produção animal e causando rápida degradação da área. Por outro lado, o subpastejo leva ao acúmulo de forragem, onde a produção exacerbada resulta na redução de qualidade e na necessidade da retirada do excesso para reduzir a degradação do solo (DIAS FILHO, 2022; DIAS FILHO, 2023). Considerando isso, é fundamental que o produtor realize uma avaliação cuidadosa a fim de ajustar a utilização das pastagens de acordo com a quantidade de forragem disponível, a fim de evitar a degradação do solo causada pelos fatores mencionados anteriormente (FLORES et al., 2008).

Existem várias opções disponíveis para melhorar a utilização das pastagens, como a adubação direta ou sua rotação com culturas, a introdução de espécies forrageiras de melhor qualidade como a *Brachiaria decumbens* e *Brachiaria brizatha* Kluthcouski (CORDEIRO & OLIVEIRA, 2013) e o controle da utilização das pastagens por meio do ajuste da taxa de lotação

ou da implementação de um sistema de manejo adequado (DIAS FILHO, 2022). Essas estratégias visam não apenas melhorar a produtividade da pastagem, mas também preservar a saúde do solo e promover uma produção animal sustentável a longo prazo.

2.1.1 Pastagens Degradadas

É possível observar casos de degradação de pastagens em todo o território brasileiro. Tal situação implica em resultados de produção abaixo do desejado. A falta de investimentos em tecnologia e insumos no manejo das pastagens tem sido uma das principais causas dessa situação (DIAS-FILHO, 2017).

Segundo a EMBRAPA (2017) e Minighin et al (2017), a degradação da pastagem é caracterizada pela queda constante de produtividade da pastagem. Uma maneira prática de identificar esse processo é observar a capacidade da pastagem em suportar o gado ao longo do tempo, ou seja, quantos animais ela pode sustentar sem causar prejuízos. Caso, a capacidade esteja diminuindo com o passar do tempo sem um motivo conhecido, é um grande indicador do processo de degradação. Existem dois tipos principais de degradação:

- **Degradação agrícola**: é causada por plantas indesejadas, também chamadas de ervas da ninha, competem por recursos do solo com a forragem, reduzindo a sua a produção. Esse fator impacta diretamente na alimentação dos animais. Esses tipos de plantas também podem ser vetores para pragas e doenças (GUIMARÃES; INOUE; IKEDA, 2018).
- **Degradação biológica**: é principalmente associada a deterioração do solo. Quando há um aumento na quantidade de áreas com solo exposto, facilitando a erosão do solo, a perda de matéria orgânica e nutrientes do solo (WADT et al., 2003). Cupins-de-montículo em altos índices de infestação também são indicadores de degradação biológica, pois estão ligados a recuperação do solo. A construção dos montículos auxilia na estrutura do solo, aumentando a sua porosidade e facilitando a absorção de água e nutrientes (VALÉRIO, 2006).

Dentro dos quatro níveis de degradação ilustrados na Figura 1, é possível diferenciar dois grandes grupos de pastagens. O primeiro grupo, denominado de “pastagens em degradação”, é constituído pelos níveis um e dois de degradação. O segundo grupo, formado pelos níveis três e quatro, é o grupo das “pastagens degradadas” propriamente ditas.

Figura 1 - Níveis de Degradação.

Fonte: Dias-Filho (2017).

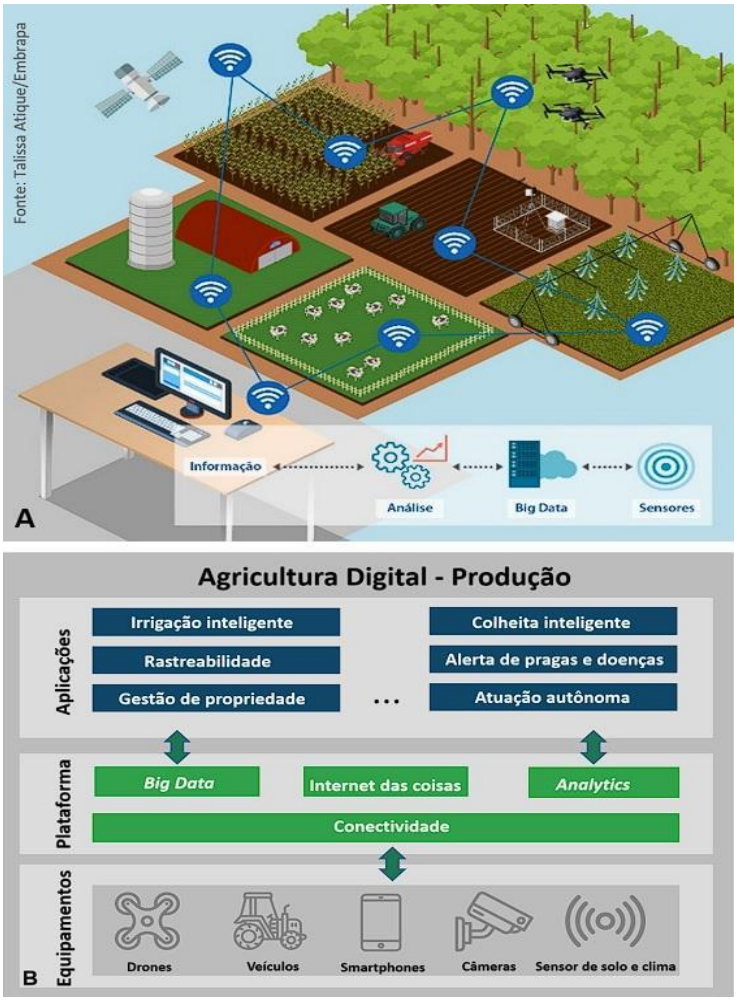
2.2 AGRICULTURA DIGITAL

A introdução da Agricultura de Precisão (AP) marcou o início de uma era revolucionária na agricultura, impulsionada pela tecnologia para obter benefícios máximos. A AP concentra-se nas variabilidades específicas do local dentro do campo, a fim de abordar todas as preocupações de forma precisa. Em suma, a AP trata totalmente da otimização dos insumos para obter um resultado máximo, de acordo com as variações no campo (AHMAD; NABI, 2021).

Todavia, à medida que a tecnologia digital avançou, nasceu um novo conceito conhecido como Agricultura 4.0. Este paradigma vai além da simples coleta de dados, incorporando tecnologias como a Internet das Coisas (IoT) para conectar dispositivos, Big Data para análise avançada e Inteligência Artificial para tomada de decisões automatizada (VILLAFUERTE et al., 2018; SAIZ-RUBIO; ROVIRA-MÁS, 2020).

Na Figura 2, é possível observar um exemplar de fazenda aplicando os conceitos de Agricultura Digital. Existem diversos sensores espalhados pela fazenda, acompanhando toda a cadeia produtiva, em conjunto com o uso de satélites, são essenciais para a aquisição de um grande volume de dados. Esses dados, por sua vez, são processados e analisados, fazendo que sejam úteis para aplicações finais.

Figura 2 - Ilustração de uma fazenda utilizando Agricultura 4.0.



Fonte: Massuhá et al., (2020).

Segundo MyFarm (2021), a agricultura 5.0 é marcada pelo uso da inteligência artificial, biotecnologia e análise de dados para todo o processo produtivo, permitindo assim, aumentar a qualidade da automação aplicada. Por fim, dentro do processo evolutivo da agricultura ilustrada na Figura 3, esse trabalho está situado como uma ferramenta da Agricultura 5.0 ao fazer uso de aprendizado de máquina para o reconhecimento de padrões em imagens. Esses tópicos serão apresentados a seguir.

Figura 3 - Evolução da Agricultura.

Fonte: Sementes (2020).

2.3 RECONHECIMENTO DE PADRÕES

Os padrões quando analisados nesse contexto vão além de objetos visíveis, englobando sistemas de dados complexos. Um padrão é uma descrição quantitativa ou estrutural de um objeto ou entidade, e classes de padrões consistem em grupos de padrões que compartilham propriedades em comum. O reconhecimento de padrões, em especial na inteligência artificial, é fundamental em diversas áreas do conhecimento, sendo utilizado em aplicações científicas, médicas, na agricultura, entre outros (BOW, 2002).

Segundo Bow (2002), existem dois tipos de itens para reconhecimento: itens concretos, incluindo imagens, símbolos e formas de onda, e itens abstratos, como ideias e estilos de escrita. A técnica de reconhecimento de padrões é aplicada de forma orientada para resolver problemas

específicos, e não há um modelo genérico aplicável a todos os casos. Isso destaca a necessidade de técnicas específicas e eficazes para diferentes contextos.

Portanto, a pesquisa em reconhecimento de padrões visa simular o mecanismo de reconhecimento humano, envolvendo aquisição de informações por órgãos sensoriais, processamento e tomada de decisões. Para isso, existem diversas técnicas de aprendizado de máquina, que podem ser mais ou menos adequadas de acordo com a necessidade da aplicação.

2.4 APRENDIZADO DE MÁQUINA

O Aprendizado de Máquina (AM) é amplamente utilizado para o reconhecimento de padrões. Dito isso, o AM tem como objetivo principal o desenvolvimento de programas que aprimoram seu desempenho com base em exemplos (MITCHELL, 1997). Para isso, é preciso de um grande volume de dados, que serão úteis para o treinamento do modelo de predição com base em experiências passadas.

Um dos tipos que o Aprendizado de Máquina pode assumir, é o aprendizado com ou sem a supervisão humana, podendo ser categorizados em não supervisionado, semi-supervisionado, aprendizado por reforço e supervisionado, que é a técnica de AM utilizada neste trabalho (FACELI et al., 2021).

- **Não Supervisionado**: no aprendizado não supervisionado, os pontos de dados não são rotulados – o algoritmo os rotula para você organizando os dados ou descrevendo a estrutura deles. Essa técnica é útil quando você não sabe como deve ser a aparência do resultado.
- **Semi-supervisionado**: conhecida também como aprendizado semi-supervisionado, assume que, juntamente com o conjunto de treinamento, há um segundo conjunto, de exemplos não rotulados, também disponível durante o treinamento.
- **Por reforço**: o aprendizado de reforço usa algoritmos que aprendem com os resultados e decidem qual ação será executada em seguida. Após cada ação, o algoritmo recebe comentários que o ajudam a determinar se a escolha feita foi correta, neutra ou incorreta. É uma boa técnica a ser usada para sistemas automatizados que precisam tomar muitas decisões simples sem diretrizes humanas.

- **Supervisionado**: o aprendizado supervisionado conta com uma base rotulada para seu treinamento. Esta técnica é a utilizada neste trabalho e será abordada com mais profundidade a seguir.

2.4.1 Aprendizado Supervisionado

A Aprendizagem de Máquina Supervisionada tem como modelo a análise de entradas e saídas a fim de estabelecer uma relação entre as classificações feitas pela entrada e o resultado esperado na saída. O propósito dessa abordagem, é treinar o programa para que possam ser classificadas novas entradas com base em experiências anteriores. Portanto, é imprescindível que as classes e rótulos alvo sejam conhecidos, pois, o programa não é capaz de estabelecer novas classificações a partir de novas entradas (NORVIG; RUSSEL, 2021).

O aprendizado supervisionado aborda principalmente dois tipos de problemas: problemas de classificação e problemas de regressão. Os problemas de classificação envolvem a atribuição de uma categoria específica a cada observação, como fazer a identificação de solo exposto e cupinzeiros em imagens de pastagens. Já em problemas de regressão, o objetivo é prever os resultados em uma saída contínua, para mapear variáveis de entrada para alguma função contínua (GÉRON, 2021).

2.4.1.1 Algoritmos de Aprendizado Supervisionado

Algoritmos amplamente utilizados para aprendizado de máquina voltada a visão computacional.

1. **k-Nearest Neighbors (k-NN)**: o método dos k-vizinhos mais próximos é uma técnica de classificação comum que se baseia na utilização de medidas de distância. Essa abordagem assume que o conjunto de amostras contém não apenas os dados, mas também a classe desejada para cada item. Para classificar um novo item, é necessário calcular a distância entre ele e todos os itens do conjunto de amostras. Em seguida, os K itens mais próximos no conjunto de amostras são selecionados para determinar a classe do novo item. O novo item é categorizado na classe que contém a maioria dos itens entre os K mais próximos. A distância entre duas amostras reflete sua similaridade, portanto, os atributos de uma instância são fundamentais para definir suas características (BISHOP, 2006).

2. **Support Vector Machines (SVM)**: um algoritmo de Máquina de Vetores de Suporte (SVM) é uma ferramenta de computação que, através de exemplos, aprende a atribuir rótulos a objetos. Em essência, uma SVM é uma entidade matemática, um conjunto de instruções para maximizar uma função matemática específica com base em um conjunto de dados fornecidos (BISHOP, 2006). Além disso, as SVMs têm demonstrado ser bem-sucedidas em uma ampla gama de aplicações na área da biologia (HUANG et al., 2018).
3. **Algoritmos de regressão**: preveem o valor de um novo ponto de dados com base em dados históricos. Um deles é a regressão linear que mostra ou prevê a relação entre duas variáveis ou dois fatores ajustando uma linha reta contínua aos dados. A linha geralmente é calculada com a função Custo de Erro Quadrado. A regressão linear é um dos tipos mais populares de análise de regressão (BISHOP, 2006).
4. **Redes Neurais Convolucionais (CNNs)**: é um algoritmo de aprendizado de máquina profundo especializado para o reconhecimento de imagens. Tem como motivação o cérebro humano, tentando imitar a maneira como os neurônios enviam sinais uns aos outros. Esse assunto será abordado com mais profundidade a seguir (BISHOP, 2006).

2.4.2 Redes Neurais Convolucionais

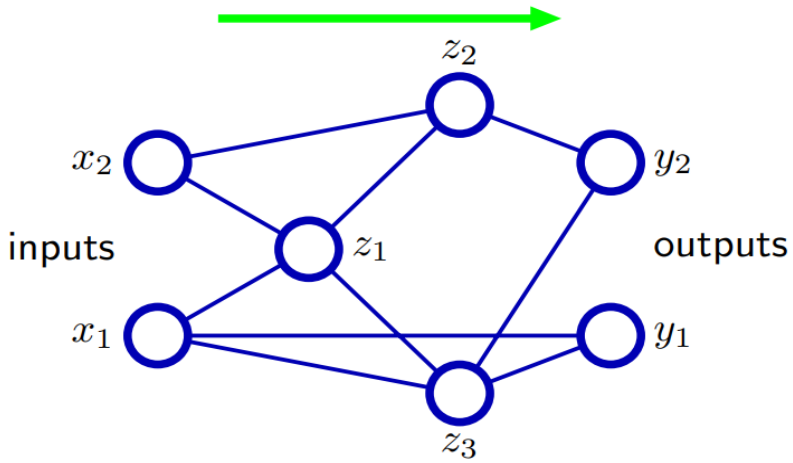
As Redes Neurais Convolucionais, conhecidas como CNNs ou *ConvNets*, foram propostas por LeCun et al. (1989), resultando na popular CNN conhecida como *LeNet* em 1998 (LECUN et al., 1998).

As CNNs representam um tipo de arquitetura de rede neural usada para o processamento de dados, com uma topologia que se assemelha a uma grade. Por exemplo, pode-se interpretar os dados de uma imagem como uma grade 2D de pixels. O termo "Rede Neural Convolucional" refere-se ao uso da operação matemática de convolução. A convolução é uma operação linear especializada, bastante utilizada para entender e modelar filtros digitais e análise de sinais em geral. As redes convolucionais são, essencialmente, redes neurais que fazem uso da operação de convolução em alguma de suas camadas, em vez da multiplicação de matrizes comumente utilizadas (BISHOP, 2006).

Como pode ser observado na Figura 4, as redes convolucionais funcionam a partir da convolução entre duas componentes, a entrada x e os pesos z, que são fatores multiplicativos atuando como filtros gerando uma saída y. Portanto, para a construção da arquitetura de uma

CNN, é preciso de alguns blocos, os comuns entre redes neurais como, camadas entre rede totalmente conectadas (*dense layers*), funções de ativação e as especiais para CNNs que são as camadas convolucionais e camadas de *pooling* (GÉRON, 2021). A ferramenta YOLO utilizada neste trabalho, é possibilitada através da implementação de uma CNN e será apresentada a sua arquitetura e funcionamento na seção seguinte.

Figura 4 - Topologia feed-foward.



Fonte: Bishop (2006).

2.4.2.1 Funções de Ativação

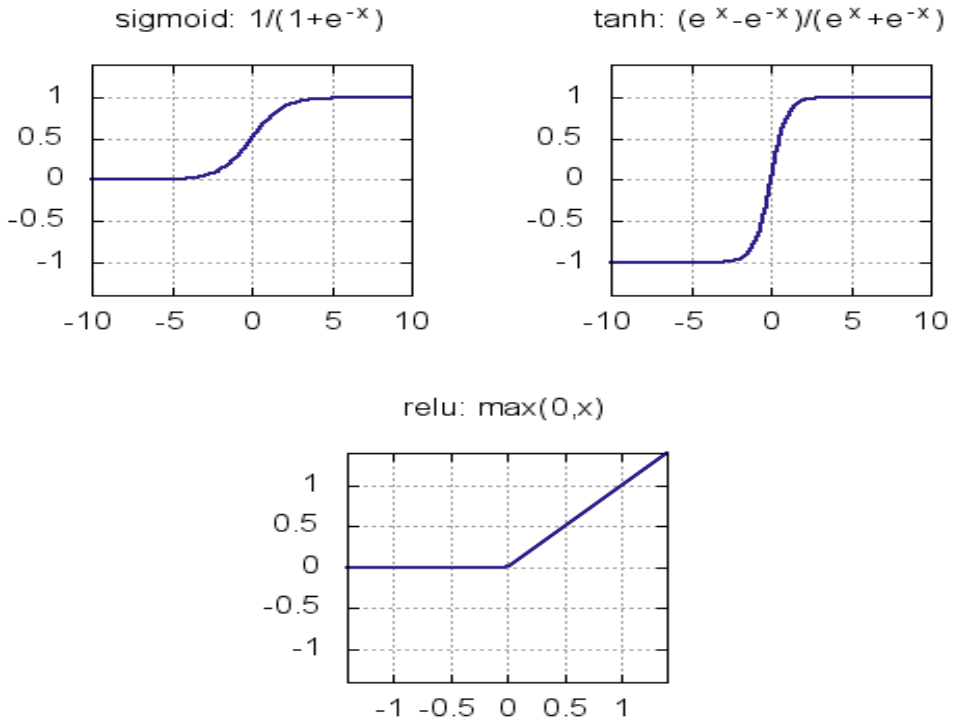
As funções de ativação são utilizadas em redes neurais para introduzir não-linearidades, permitindo que a rede aprenda padrões e relações complexas entre os dados. Elas são uma transformação não linear que é aplicada ao sinal de entrada, convertendo o resultado em um valor específico, geralmente dentro de um intervalo definido. A saída transformada é então enviada para a próxima camada de neurônios como entrada (BISHOP, 2006). A resposta as funções citadas a seguir, podem ser observadas na Figura 5.

1. **Funções Sigmóide:** A função sigmóide produz uma saída entre 0 e 1. É especialmente útil para modelos onde temos que prever a probabilidade como uma saída (BISHOP, 2006).

2. **Funções Tangente Hiperbólica (tanh):** A função tanh é semelhante à função sigmóide, mas sua saída varia de -1 a 1. Isso significa que a tanh preserva a informação de polaridade dos dados de entrada, podendo ser útil em diversos contextos (BISHOP, 2006).

3. **Funções Unidade Linear Retificada (ReLU):** A função ReLU é a função de ativação mais popular atualmente, introduz não linearidade a rede, característica crucial para o aprendizado utilizando dados complexos. Ela produz a entrada diretamente se for positiva; caso contrário, produz zero (BISHOP, 2006).

Figura 5 - Funções de ativação.

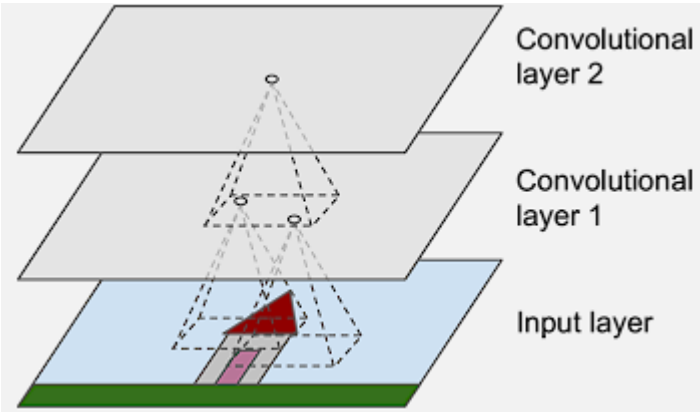


Fonte: Bishop(2006).

2.4.2.2 Camadas Convolucionais

O bloco mais crucial de uma CNN é a camada convolucional, responsável pela extração de características dos dados de entrada. Os neurônios na primeira camada convolucional não se conectam a cada pixel individual na imagem de entrada, mas apenas aos pixels em suas áreas receptivas específicas (conforme ilustrado na Figura 6). Em seguida, cada neurônio na segunda camada convolucional está ligado apenas aos neurônios localizados dentro de uma pequena região retangular na primeira camada. Essa arquitetura permite que a rede se concentre em pequenas características de baixo nível na primeira camada oculta e, gradualmente, as combine para formar características maiores e de nível mais alto nas camadas subsequentes. Essa estrutura hierárquica é comum em imagens do mundo real, o que explica em parte por que as CNNs são tão eficazes no reconhecimento de imagens (GÉRON, 2021).

Figura 6 - Camadas de CNN com campos receptivos locais retangulares.

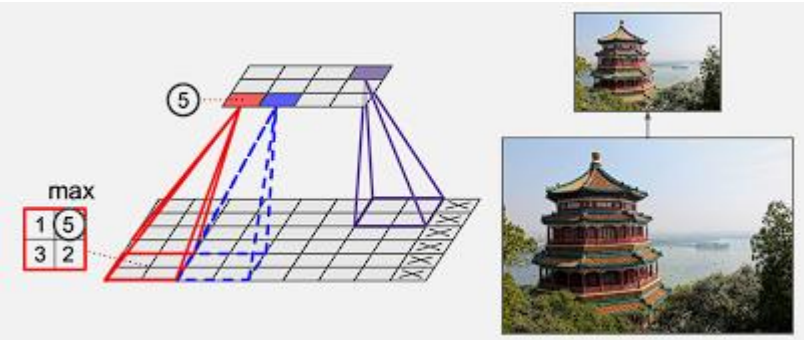


Fonte: Géron (2019).

2.4.2.3 Camadas de Pooling

Assim como nas camadas de convolução, cada neurônio em uma camada de *pooling* está conectado às saídas de um número limitado de neurônios na camada anterior, localizados dentro de um pequeno campo receptivo retangular. No entanto, um neurônio de *pooling* não possui pesos, tudo o que é feito é agregar as entradas usando uma função de agregação, como o máximo ou a média. A Figura 7 mostra uma camada de max *pooling*, que é o tipo mais comum de camada de *pooling*. Neste exemplo, é utilizado um kernel de *pooling* de 2 × 2, com um passo de 2 e sem preenchimento. Apenas o valor máximo de entrada em cada campo receptivo é transmitido para a próxima camada, enquanto as outras entradas são descartadas (GÉRON, 2021).

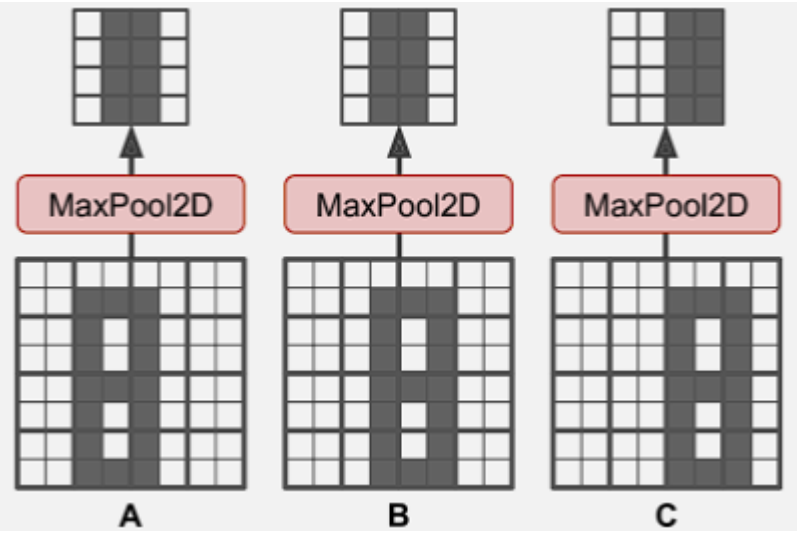
Figura 7 - Camadas de max pooling (kernel de pooling 2 x 2, passo 2, sem preenchimento).



Fonte: Géron (2021).

Além de otimizar cálculos e economizar memória, também introduz invariância a pequenas traduções. Essa invariância é crucial em tarefas de visão computacional, pois permite que a rede reconheça padrões relevantes independentemente de sua posição exata na entrada, tornando a arquitetura mais eficiente e robusta, como mostrado na Figura 8 (GÉRON, 2021).

Figura 8 - Invariância a pequenas translações.



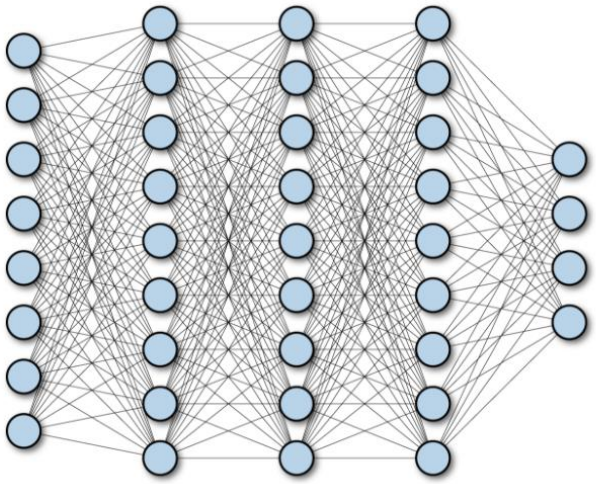
Fonte: Géron (2021).

2.4.2.4 Camadas Totalmente Conectadas (Fully Conected Layers)

Também chamadas de camadas densas (ou *fully connected layers*) são camadas de uma rede neural onde cada neurônio está conectado a todos os neurônios da camada anterior. Elas geralmente vêm após as camadas de convolução e *pooling* e são usadas para classificar as imagens com base nas características extraídas. Essas camadas aprendem os padrões que foram

extraídos pelas camadas convolucionais e, ao fazer isso, a rede se torna capaz de reconhecer objetos (GÉRON, 2021).

Figura 9 - Camadas Finais.



Fonte: Géron (2021).

2.4.2.5 Aprendizado da Rede

Etapas que compõem o processo de treinamento de uma CNN.

1. **Extração de Características:** durante a extração de características, a rede identifica características úteis, por exemplo, bordas, formas básicas, cores, etc. Cada neurônio da primeira camada oculta está conectado a um pequeno conjunto de neurônios de entrada, formando um campo receptivo (GÉRON, 2021).
2. **Função de Perda:** é uma medida que quantifica o quão bem o modelo está performando durante o treinamento. Ela compara a saída prevista da rede com a saída real e calcula a diferença (GÉRON, 2021).
3. **Otimização:** a otimização refere-se ao processo de ajuste dos pesos da rede para minimizar a função de perda. Isso é geralmente feito usando um algoritmo chamado "backpropagation". Durante esse processo, o algoritmo ajusta os pesos com base no gradiente da função de perda em relação a cada peso (GÉRON, 2021).
4. **Ajustes de hiperparâmetros:** parâmetros como, a taxa de aprendizado, o número de épocas (ciclos completos de passagem através do conjunto de dados de treinamento), o tamanho do lote, etc., são ajustados para melhorar o desempenho do modelo (GÉRON, 2021).

5. **Treinamento e validação:** o conjunto de dados é separado em dois, o conjunto de treinamento e o conjunto de validação. O modelo é treinado utilizando o conjunto de dados de treinamento e o desempenho do modelo é validado usando um conjunto de dados de validação (GÉRON, 2021).
6. **Predição:** no fim do processo a rede pode gerar um valor de probabilidade para a predição de um objeto na imagem (GÉRON, 2021).

2.4.3 Fine-tuning

O *fine-tuning*, também conhecido como ajuste fino, consiste em utilizar um modelo pré-treinado e fazer ajustes específicos em seus hiperparâmetros, visando melhorar a performance do modelo para um propósito específico. O *fine-tuning* foca na otimização das últimas camadas do modelo pré-treinado, enquanto as camadas iniciais, que já estão devidamente generalizadas, permanecem inalteradas. Essa abordagem permite que o modelo aprenda as nuances e detalhes específicos para o propósito desejado de forma mais eficiente, sem a necessidade de realizar o treinamento de um novo modelo para uma nova arquitetura (CAI, PENG, 2021) .

Cai e Peng (2021) aponta que a seleção de hiperparâmetros para CNNs é feita através do método de "tentativa e erro" e que não existe uma metodologia generalista o suficiente para servir de base para todos os tipos de aplicações. Porém, o autor também menciona que alguns aspectos de uma CNN podem ser alvos de *fine-tuning*:

- **Profundidade da rede (*network depth*):** Número de camadas convolucionais empilhadas;
- **Largura da rede *(network width*):** Número de filtros (neurônios) em cada camada convolucional;
- **Função de ativação não linear:** Função aplicada à saída de cada camada para introduzir não-linearidade na rede;
- **Método de *pooling*:** Técnica de redução de dimensionalidade da saída das camadas convolucionais;
- **Método de inicialização de parâmetros:** Estratégia para atribuir valores iniciais aos pesos e vieses da rede; e
- **Estratégia de ajuste da taxa de aprendizagem (*learning rate*):** Define como a taxa de aprendizagem é atualizada durante o treinamento.

2.4.3.1 Otimizadores

Os algoritmos de otimização, são ferramentas matemáticas que trabalham nos bastidores do aprendizado de máquina, ajustando os parâmetros de um modelo para minimizar uma função de erro. No contexto do *fine-tuning*, essa função de erro representa a diferença entre as previsões do modelo e os resultados reais. O objetivo do otimizador é encontrar a combinação ideal de parâmetros que minimize essa diferença, ou seja, que faça com que o modelo aprenda a realizar a tarefa com a maior precisão possível. Alguns otimizadores utilizados em CNNs são:

SGD (*Stochastic Gradient Descent*): Sutskever et al. (2013) apresenta que o SGD é um otimizador clássico que atualiza os parâmetros do modelo na direção do gradiente da função de perda. A atualização para cada parâmetro θ_i é dada por:

$$\theta\_i(t+1) = \theta\_i(t) - \alpha * \nabla\_\theta\_i\, L(\theta, t) \textbf{ (1)}$$

Onde:

- **θ_i(t)**: Valor do parâmetro θ_i no instante t;
- **α**: Taxa de aprendizado, controlando a magnitude da atualização;
- **∇_θ_i L(θ, t)**: Gradiente da função de perda L em relação ao parâmetro θ_i no instante t;

Adam (*Adaptive Moment Estimation*): Levine et al. (2016) apresenta que o Adam é um otimizador adaptativo que ajusta a taxa de aprendizado para cada parâmetro, levando em consideração o histórico de gradientes. As atualizações para cada parâmetro θ_i são dadas por:

$$m\_i(t+1) = \beta\_1 * m\_i(t) + (1 - \beta\_1) * \nabla\_\theta\_i\, L(\theta, t)$$

$$v\_i(t+1) = \beta\_2 * v\_i(t) + (1 - \beta\_2) * (\nabla\_\theta\_i\, L(\theta, t))\text{^}2$$

$$\theta\_i(t+1) = \theta\_i(t) - \alpha * (m\_i(t+1) / \sqrt{(v\_i(t+1) + \varepsilon)}) \textbf{ (2)}$$

Onde:

- **m_i(t)**: Momento médio do gradiente para o parâmetro θ_i no instante t;
- **v_i(t)**: Momento da variância do gradiente para o parâmetro θ_i no instante t;
- **β_1**: Parâmetro de ponderação para o momento médio, geralmente 0.9;
- **β_2**: Parâmetro de ponderação para o momento da variância, geralmente 0.999;
- **ε**: Pequeno valor para evitar divisão por zero;

AdamW (*Adam with Weight Decay*): Loshchilov & Hutter (2019) apresenta que o AdamW incorpora a regularização L2 no Adam, penalizando pesos grandes e combatendo o overfitting. As atualizações para cada parâmetro θ_i são dadas por:

$$m\_i(t+1) = \beta\_1 * m\_i(t) + (1 - \beta\_1) * (\nabla\_\theta\_i\, L(\theta, t) + \lambda * \theta\_i(t))$$

$$v\_i(t+1) = \beta\_2 * v\_i(t) + (1 - \beta\_2) * (\nabla\_\theta\_i\, L(\theta, t))\text{^}2$$

$$\theta\_i(t+1) = \theta\_i(t) - \alpha * (m\_i(t+1) / \sqrt{(v\_i(t+1) + \varepsilon)}) \textbf{ (3)}$$

Onde:

- **λ**: Parâmetro de regularização L2, controlando a intensidade da penalização;

RMSProp (*Root Mean Square Propagation*): Ruder (2017) apresenta que o RMSProp é um otimizador adaptativo que utiliza a média quadrática ponderada dos gradientes passados para ajustar a taxa de aprendizado. As atualizações para cada parâmetro θ_i são dadas por:

$$g\_i(t) = \nabla\_\theta\_i\, L(\theta, t)$$

$$E[g\_i\text{^}2](t+1) = \beta\_2 * E[g\_i\text{^}2](t) + (1 - \beta\_2) * (g\_i(t))\text{^}2$$

$$\theta\_i(t+1) = \theta\_i(t) - \alpha * g\_i(t) / \sqrt{(E[g\_i\text{^}2](t+1) + \varepsilon)} \textbf{ (4)}$$

Onde:

- **Eg_i²**: Média quadrática ponderada dos gradientes passados para o parâmetro θ_i no instante t

2.4.4 Métricas

Segundo Bishop (2006), a matriz de confusão é uma tabela que descreve o desempenho de um modelo de classificação. Ela compara as previsões do modelo com os rótulos reais e categoriza os resultados em quatro quadrantes: Verdadeiro Positivo (VP), Falso Positivo (FP), Falso Negativo (FN) e Verdadeiro Negativo (VN).

- **Verdadeiro Positivo (VP):** Refere-se aos casos em que o modelo previu corretamente a presença de um objeto (ou classe) e o rótulo real também indica a presença desse objeto (BISHOP, 2006);
- **Falso Positivo (FP):** Representa os casos em que o modelo previu erroneamente a presença de um objeto que não estava realmente presente de acordo com o rótulo real (BISHOP, 2006);
- **Falso Negativo (FN):** Indica os casos em que o modelo falhou em prever a presença de um objeto que realmente estava presente, conforme indicado pelo rótulo real (BISHOP, 2006);
- **Verdadeiro Negativo (VN):** Refere-se aos casos em que o modelo previu corretamente a ausência de um objeto, e o rótulo real também indica a ausência desse objeto (BISHOP, 2006).

2.4.4.1 Acurácia

A acurácia é uma métrica geral que mede a taxa de previsões corretas do modelo em relação ao total de previsões. Ela é calculada como a soma dos verdadeiros positivos e verdadeiros negativos dividida pelo número total de exemplos (GÉRON, 2021).

$$Acurácia = \frac{VP+VN}{VP+FP+FN+VN} \textbf{ (5)}$$

2.4.4.2 Precisão

A precisão é uma métrica que avalia a proporção de instâncias positivas previstas corretamente em relação ao total de instâncias positivas previstas pelo modelo (MÜLLER, 2016). A fórmula é:

$$Precisão = \frac{VP}{VP+FP} \textbf{ (6)}$$

2.4.4.3 Recall

O recall (ou sensibilidade) mede a proporção de instâncias positivas que foram previstas corretamente em relação ao total de instâncias positivas existentes (JAPKOWICZ, 2011). A fórmula é:

$$Recall = \frac{VP}{VP+FN} \textbf{ (7)}$$

2.4.5 YOLOv1 (You Only Look Once)

O YOLO é uma ferramenta desenvolvida para agir como o sistema visual do ser humano, é rápido e preciso, permitindo a execução de tarefas complexas como dirigir sem estar totalmente entregue a essa tarefa. Algoritmos de detecção de objetos que sejam rápidos e precisos, tornariam possíveis sistemas robóticos responsivos para propósitos gerais como a direção de carros de forma autônoma.

De acordo com Redmon et al. (2015) essa ferramenta visa ser uma alternativa mais otimizada em comparação com as abordagens mais comuns, como as redes neurais convolucionais baseadas em região (R-CNN). Para isso, o YOLO reestrutura a detecção de objetos para que seja um problema de regressão único, partindo do pixel das imagens para as coordenadas dos rótulos e classes das probabilidades.

Figura 10 - O sistema de detecção.

Fonte: Redmon et al. (2015).

Na Figura 10, o sistema (1) reajusta a imagem para o tamanho de 448 × 448, (2) passa uma única rede convolucional na imagem, e (3) limita as detecções resultantes pela confiança do modelo (REDMON et al., 2015).

Este modelo unificado apresenta diversas vantagens em relação aos métodos tradicionais de detecção de objetos. O YOLO é extremamente rápido. Como a detecção é reformulada como um problema de regressão, não precisamos de um *pipeline* complexo. O único passo é executar a rede neural do YOLO em uma nova imagem durante o teste para prever as detecções. A rede base roda a 45 quadros por segundo sem processamento em lote em uma GPU Titan X, e uma versão mais rápida roda a mais de 150 quadros por segundo (REDMON et al., 2015).

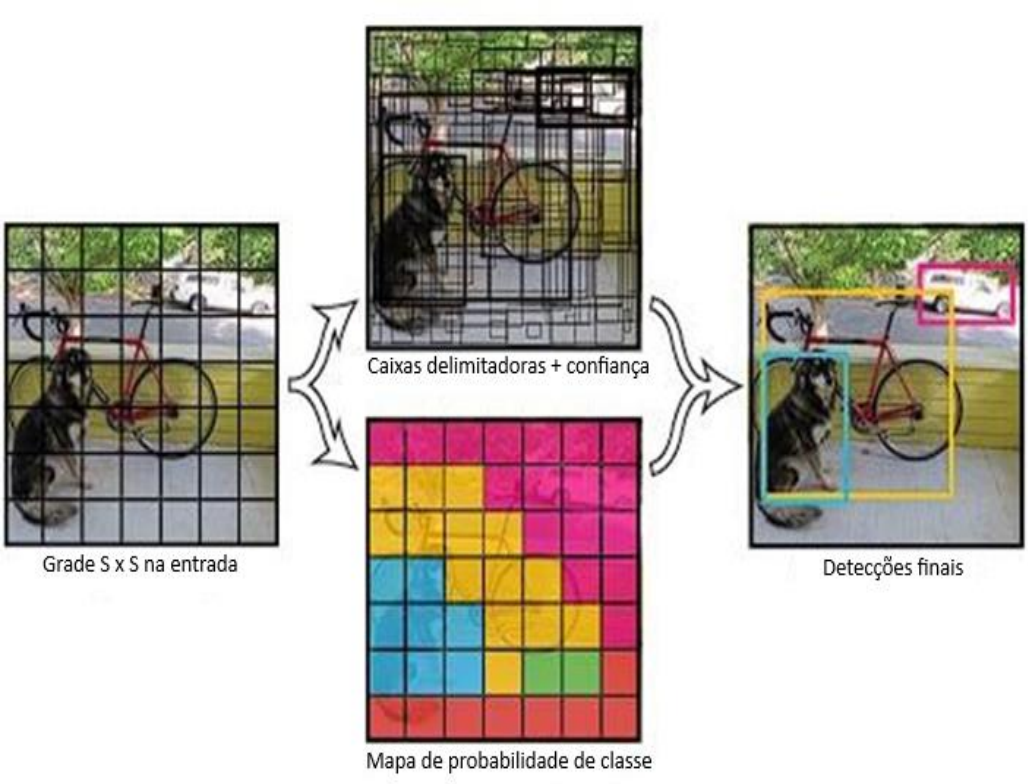
O YOLO aprende representações generalizáveis de objetos que se referem a características aprendidas pelo YOLO que são aplicáveis a uma variedade de objetos,

permitindo ao modelo reconhecer e compreender diferentes classes de objetos em diversas situações. Quando treinado em imagens naturais e testado em obras de arte, o YOLO supera amplamente os principais métodos de detecção, como DPM e R-CNN (REDMON et al., 2015).

No entanto, o YOLO ainda fica atrás dos sistemas de detecção de última geração em precisão. Embora ele possa identificar rapidamente objetos em imagens, ele tem dificuldade em localizar com precisão alguns objetos, especialmente os pequenos (REDMON et al., 2015).

Na Figura 11, o sistema modela a detecção como um problema de regressão. Ele divide a imagem em uma grade de S × S e, para cada célula da grade, prevê B caixas delimitadoras, confiança para essas caixas e probabilidades de classe C. (REDMON et al., 2015).

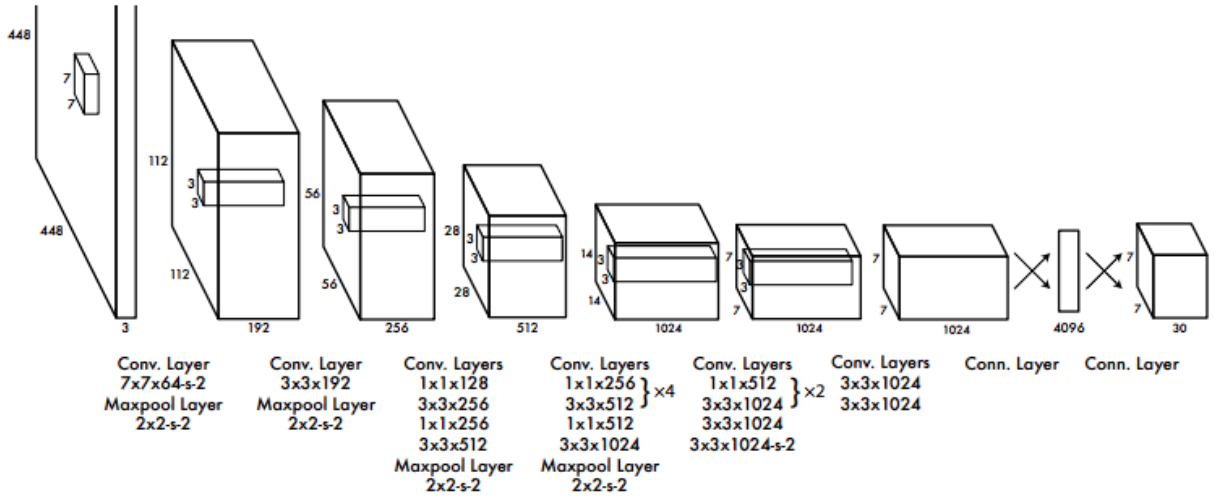
Figura 11 - O modelo.



Fonte: Redmon et al. (2015).

Na Figura 12 é possível analisar a arquitetura do YOLO onde, a rede de detecção possui 24 camadas convolucionais seguidas por 2 camadas totalmente conectadas. Camadas convolucionais de 1 × 1 alternadas reduzem o espaço de características das camadas anteriores (REDMON et al., 2015).

Figura 12 - A arquitetura.



Fonte: Redmon et al. (2015).

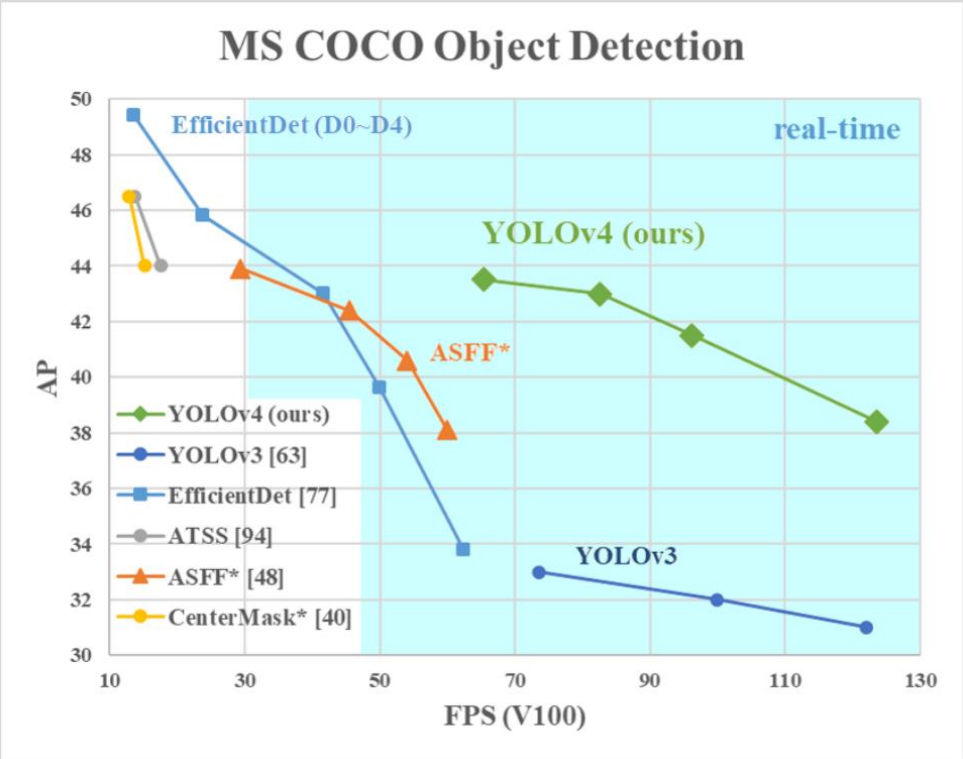
2.4.5.1 YOLOv4

A maioria dos detectores de objetos baseados em CNN são amplamente aplicáveis apenas para sistemas de recomendação. Por exemplo, a busca por vagas de estacionamento gratuitas por câmeras de vídeo urbanas é executada por modelos lentos e precisos, enquanto o aviso de colisão de veículos está relacionado a modelos rápidos e imprecisos. Melhorar a precisão do detector de objetos em tempo real permite usá-los não apenas para sistemas de recomendação de geração de dicas, mas também para gerenciamento de processos autônomos e reduzir a necessidade de interação humana. A operação do detector de objetos em tempo real em Unidades de Processamento Gráfico (GPUs) convencionais permite seu uso em massa a um preço acessível. As redes neurais modernas mais precisas não operam em tempo real e requerem um grande número de GPUs para treinamento com um tamanho de mini-lote grande. Esses problemas foram abordados criando uma CNN que opera em tempo real em uma GPU convencional e cujo treinamento requer apenas uma GPU convencional (BOCHKOVSKIY et al., 2020).

O YOLOv4 tem como objetivo propor uma velocidade de operação rápida de um detector de objetos em sistemas de produção e otimização para cálculos paralelos, em vez do indicador teórico de baixo volume de computação (BFLOP). A ideia é que o objeto projetado

possa ser facilmente treinado e usado. Por exemplo, qualquer pessoa que use uma GPU convencional para treinar e testar pode obter resultados de detecção de objetos em tempo real, de alta qualidade e convincentes, como os resultados mostrados na Figura 13 onde o YOLOv4 é duas vezes mais rápido que o EfficientDet com desempenho comparável. Melhora o AP e FPS do YOLOv3 em 10% e 12%, respectivamente (BOCHKOVSKIY et al., 2020).

Figura 13 - Comparação do YOLOv4 proposto e outros detectores de objetos de última geração.



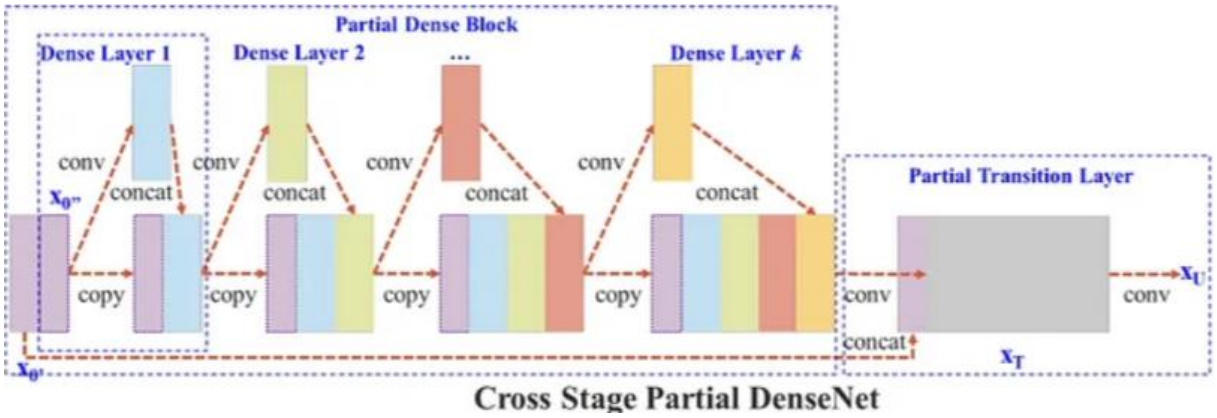
Fonte: Bochkovskiy et al. (2020).

2.4.5.2 YOLOv4 - Arquitetura

2.4.5.2.1 Backbone

No que diz respeito às conexões parciais entre estágios (CSP), o objetivo principal é superar os desafios associados ao fluxo de informações e propagação de gradiente na rede. A abordagem envolve dividir o mapa de recursos em duas partes, processando cada parte independentemente e, em seguida, mesclando-as novamente. Essa estratégia pode ser observada na Figura 14 e aborda efetivamente o problema do desaparecimento do gradiente, permitindo uma troca de informações mais eficiente entre diferentes estágios da rede (BOCHKOVSKIY et al., 2020).

Figura 14 - Ilustração da Cross Stage Partitial DenseNet (CSPDenseNet).



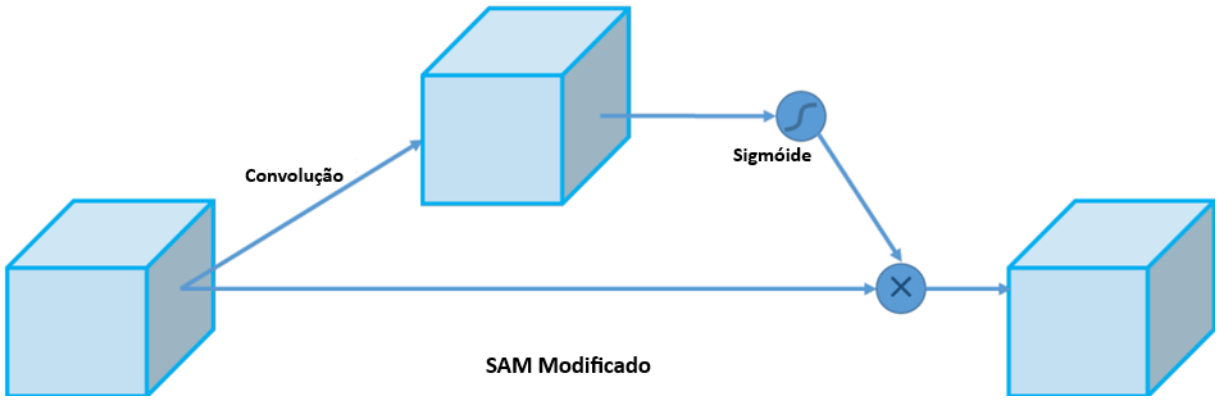
Fonte: Bochkovskiy et al. (2020).

A incorporação de conexões parciais entre estágios no YOLOv4 aprimora significativamente a capacidade do modelo de compreender padrões complexos e recursos hierárquicos, levando a uma melhor precisão. O design da arquitetura visa encontrar um bom equilíbrio, garantindo o fluxo eficiente de informações e facilitando a propagação do gradiente. Isso ajuda a melhorar o desempenho geral da rede neural (BOCHKOVSKIY et al., 2020).

2.4.5.2.2 Attention

Após a execução do *backbone*, a saída passa por um processamento adicional por meio de um mecanismo de atenção (Figura 15), que é uma versão modificada do Módulo de Atenção Espacial (SAM). É importante notar que, embora esse mecanismo seja implementado no YOLOv4, ele pode não ser utilizado em todas as variações do modelo. O código base original inclui várias versões, cada uma com sua arquitetura exclusiva (BOCHKOVSKIY et al., 2020).

Figura 15 - SAM modificado.

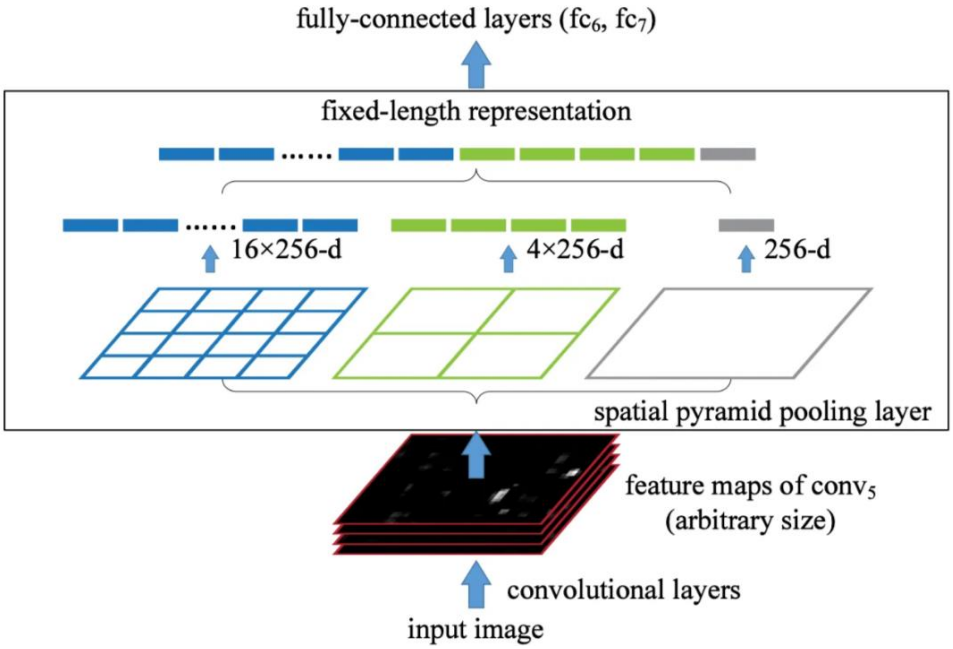


Fonte: Bochkovskiy et al. (2020).

2.4.5.2.3 Neck

Após a aplicação do Módulo de Atenção Espacial (SAM) modificado aos mapas de recursos, a saída é direcionada para o *Pooling* Piramidal Espacial (SPP). O SPP permite a coleta de informações em várias escalas sem a necessidade de um tamanho de entrada específico. Esse processo envolve dividir o mapa de recursos de entrada em uma grade com vários níveis, aplicar operações de *pooling* em cada nível e, posteriormente, concatenar os resultados, o funcionamento pode ser compreendido melhor com a Figura 16. A vantagem do SPP reside na sua capacidade de tornar a rede invariante ao tamanho de entrada, permitindo a captura de informações em multi-escala. Além disso, ele produz uma representação de tamanho fixo para camadas subsequentes, tornando-se particularmente útil em cenários onde objetos podem aparecer em várias escalas dentro de uma imagem (BOCHKOVSKIY et al., 2020).

Figura 16 - Uma rede com uma camada de Pooling Piramidal Espacial.



Fonte: Bochkovskiy et al. (2020).

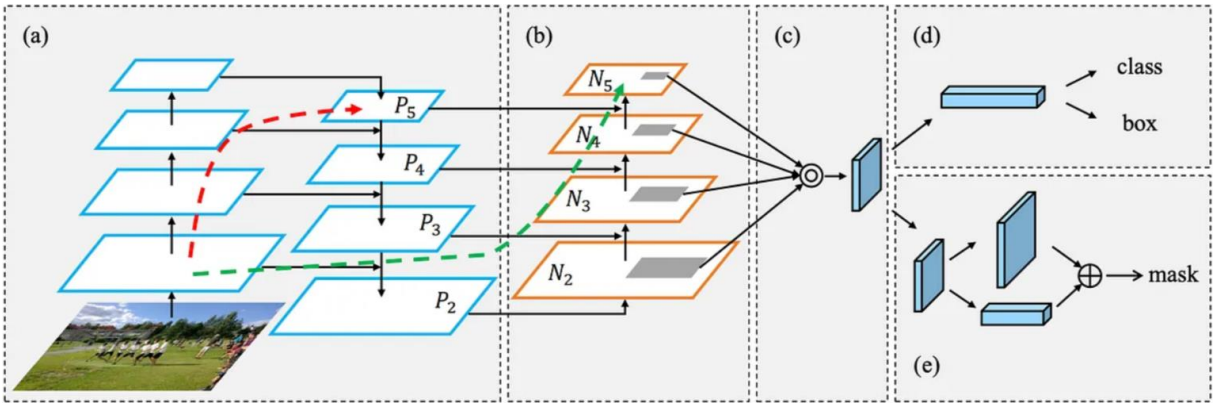
Após o *Pooling* Piramidal Espacial (SPP), é utilizada uma versão modificada da Rede de Agregação de Caminhos (PAN). O PAN é empregado para aprimorar ainda mais a capacidade da rede de agregar informações de diferentes caminhos, contribuindo para uma melhor representação de recursos e desempenho de detecção de objetos (BOCHKOVSKIY et al., 2020).

Na Figura 17 é possível observar a estrutura do *framework*, onde, (a) é o *Backbone* FPN, (b) o Aumento do caminho de baixo para cima, (c) a camada de *Pooling* adaptativo de features, (d) a Branch de caixas e (e) é a Fusão totalmente conectada. O PAN pega os *features maps* de diferentes estágios da rede *backbone*, normalmente em 3 escalas diferentes P3, P4 e P5 no YOLOv4. Ele:

- Aumenta a amostragem de recursos de baixa resolução (P3, P4) para corresponder à resolução mais alta (P5). Isso preserva detalhes locais importantes para a detecção de objetos pequenos.
- Reduz a amostragem do recurso de alta resolução (P5) para corresponder às resoluções mais baixas. Isso incorpora contexto global útil para a detecção de objetos maiores.

No PAN original, os caminhos de baixo para cima e de cima para baixo são somados elemento a elemento com os recursos originais em cada nível. No entanto, o YOLOv4 modificado usa a concatenação elemento a elemento em vez da adição, o que ajuda a preservar mais informações e melhorar a precisão (BOCHKOVSKIY et al., 2020).

Figura 17- Ilustração do framework.



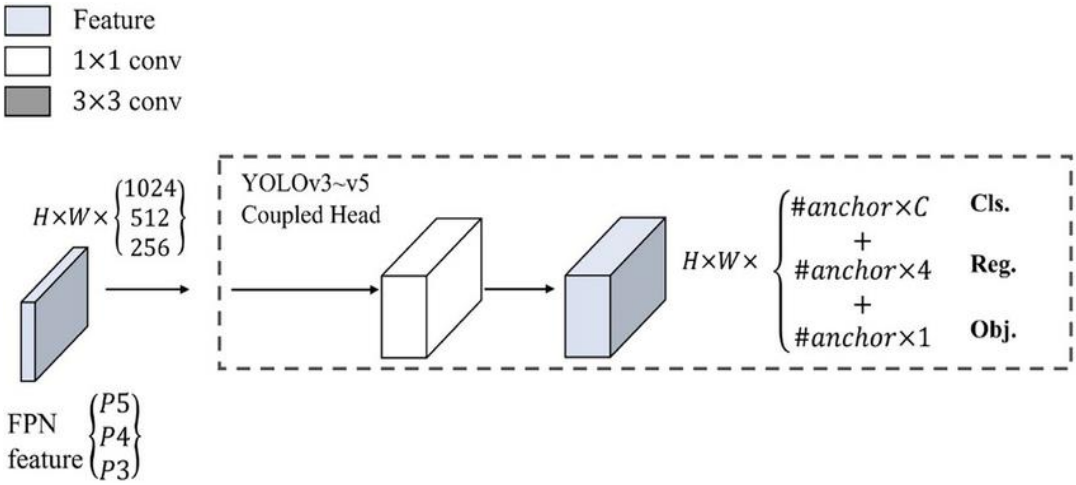
Fonte: Bochkovskiy et al. (2020).

2.4.5.2.4 Head

Após o PAN, a cabeça (*head*) do YOLOv3 é utilizada, Figura 18. Ela consiste em três sub-redes de detecção operando em mapas de recursos de diferentes escalas. A head inclui camadas convolucionais para processar recursos, prever coordenadas da caixa delimitadora (x, y, largura, altura e confiança) e determinar as probabilidades de classe para objetos detectados. O uso de *anchor boxes* auxilia na previsão das caixas delimitadoras, e a supressão não-máxima

é aplicada para refinar o conjunto final de previsões, garantindo resultados precisos e sem redundância. A *head* transforma os recursos extraídos pelo *backbone* e pelo *neck* em previsões acionáveis de detecção de objetos.

Figura 18 - YOLOv3 Head.



Fonte: Bochkovskiy et al. (2020).

2.4.5.3 Visão Geral do YOLOv5

YOLOv5 é um modelo de detecção de objetos introduzido em 2020 pela Ultralytics LLC (2020), criadora dos modelos YOLOv1 e YOLOv3. O YOLOv5 alcança performance de ponta no conjunto de dados de referência COCO Aydin (2020), sendo também rápido e eficiente para treinar e implementar. YOLOv5 introduziu diversas mudanças na arquitetura, sendo a mais notável a prática padronizada de estruturar o modelo em três componentes: *backbone* (cadeia principal), *neck* (pescoço) e *head* (cabeça).

- **Backbone:** A cadeia principal do YOLOv5 é o Darknet53, uma nova arquitetura de rede que foca na extração de características, caracterizada por pequenas janelas de filtro e conexões residuais. As conexões parciais entre estágios (CSP) permitem que a arquitetura alcance um fluxo de gradiente mais rico enquanto reduz a computação, conforme proposto por Wang et al. (2019).
- **Neck:** O *neck*, conforme descrito por Teven et al. (2023), conecta a cadeia principal à cabeça, cujo objetivo é agregar e refinar as características extraídas pela cadeia principal, com foco em melhorar a informação espacial e semântica em diferentes escalas. Um módulo de *Pooling* Piramidal Espacial (SPP) remove a restrição de

tamanho fixo da rede, o que elimina a necessidade de distorcer, aumentar ou recortar imagens. Isso é seguido por um módulo de Rede de Agregação de Caminhos CSP (WANG et al., 2019), que incorpora os recursos aprendidos na cadeia principal e encurta o caminho da informação entre as camadas inferiores e superiores.

- **Head:** A cabeça do YOLOv5 consiste em três ramos, cada um prevendo uma escala de características diferentes. Na publicação original do modelo (AYDIN, 2020), os criadores usaram três tamanhos de célula de grade de 13 x 13, 26 x 26 e 52 x 52, com cada célula de grade prevendo B = 3 caixas delimitadoras. Cada cabeça produz caixas delimitadoras, probabilidades de classe e scores de confiança. Finalmente, a rede usa Supressão Não-Máxima (NMS) para filtrar caixas delimitadoras sobrepostas (HOSANG, 2017).

O YOLOv5 incorpora caixas âncoras, que são caixas delimitadoras de tamanho fixo usadas para prever a localização e o tamanho dos objetos dentro de uma imagem. Em vez de prever caixas delimitadoras arbitrárias para cada instância do objeto, o modelo prevê as coordenadas das caixas âncoras com relações de aspecto e escalas predefinidas e as ajusta para se adequarem à instância do objeto.

2.4.5.4 YOLOv8

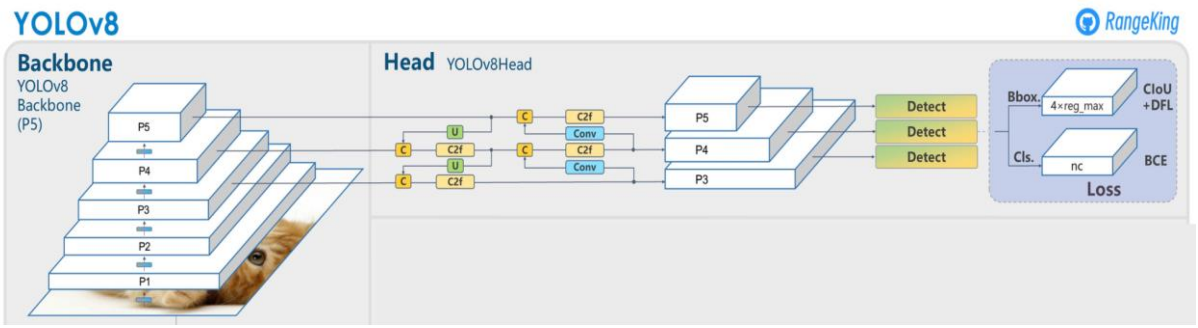
O YOLOv8 introduz diversas melhorias em comparação com as versões anteriores do YOLO, como uma nova arquitetura Figura 19, de rede neural que utiliza a Rede Piramidal de Características (FPN) e a Rede de Agregação de Caminhos (PAN), além de uma nova ferramenta de rotulagem que simplifica o processo de anotação. Essa ferramenta de rotulagem contém vários recursos úteis, como rotulagem automática, atalhos de rotulagem e teclas de atalho personalizáveis. A combinação desses recursos torna mais fácil anotar imagens para treinar o modelo (Solawetz, 2023).

A FPN funciona reduzindo gradualmente a resolução espacial da imagem de entrada, ao mesmo tempo que aumenta o número de canais de características. Isso resulta na criação de mapas de características capazes de detectar objetos em diferentes escalas e resoluções. A arquitetura PAN, por outro lado, agrega características de diferentes níveis da rede por meio de conexões de salto (*skip connections*). Ao fazer isso, a rede pode capturar melhor características

em várias escalas e resoluções, o que é crucial para detectar com precisão objetos de diferentes tamanhos e formas (TREVEN et al., 2023).

Solawetz (2023) traz uma breve análise arquitetural do YOLOv8, onde foi destacado algumas mudanças importantes para o ganho de performance em comparação com versões anteriores.

Figura 19 - Arquitetura YOLOv8.



Fonte: Solawetz (2023).

- **Detecção sem âncoras (*Anchor Free Detection*):** o YOLOv8 é um modelo livre de âncoras. Isso significa que ele prediz diretamente o centro de um objeto, em vez do deslocamento de uma caixa âncora conhecida. As caixas âncoras eram uma parte notoriamente complicada dos modelos YOLO anteriores, pois elas podiam representar a distribuição das caixas do benchmark alvo, mas não a distribuição do conjunto de dados personalizado. A detecção sem âncoras reduz o número de previsões de caixas, o que acelera a Supressão Não Máxima (NMS), uma etapa complicada de pós-processamento que filtra as detecções candidatas após a inferência (SOLAWETZ, 2023).
- **Novas Convoluções:** a primeira convolução 6x6 do tronco (stem) foi substituída por uma 3x3, o bloco de construção principal foi alterado e o C2f substituiu o C3. O módulo é resumido na imagem abaixo, onde "f" é o número de características, "e" é a taxa de expansão e CBS é um bloco composto por uma Convolução, uma BatchNorm e uma SiLU. No C2f, todas as saídas do *Bottleneck* (apelido para duas convoluções 3x3 com conexões residuais) são concatenadas. Enquanto no C3, apenas a saída do último *Bottleneck* era usada. O *Bottleneck* é o mesmo do YOLOv5, mas o tamanho do kernel da primeira convolução foi alterado de 1x1 para 3x3. A partir dessas informações,

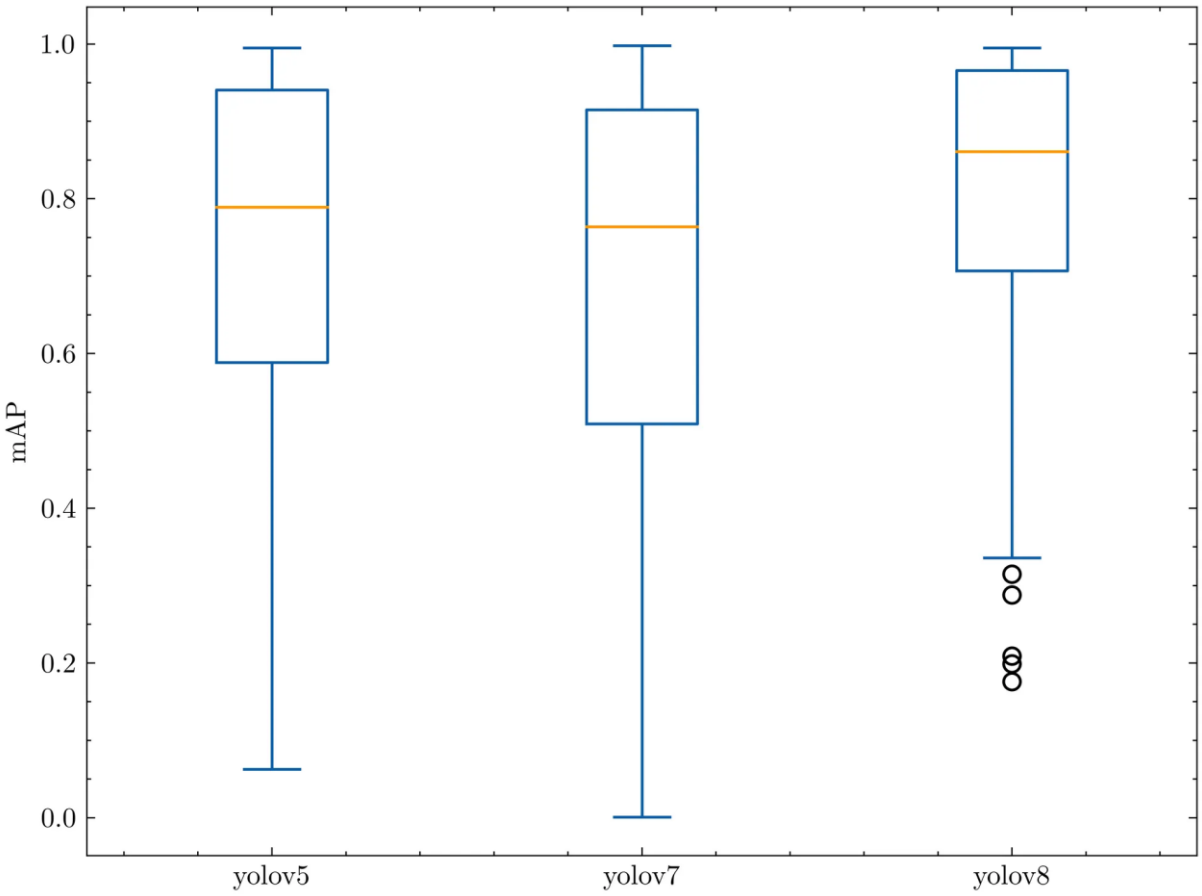
podemos ver que o YOLOv8 está começando a voltar para o bloco ResNet definido em 2015. No pescoço da rede (neck), os recursos são concatenados diretamente sem forçar as mesmas dimensões de canais. Isso reduz a contagem de parâmetros e o tamanho geral dos tensores (SOLAWETZ, 2023).

- **Finalizando o Aumento Mosaico:** a pesquisa em aprendizado profundo tende a se concentrar na arquitetura do modelo, mas a rotina de treinamento no YOLOv5 e YOLOv8 é uma parte essencial do seu sucesso. O YOLOv8 aumenta as imagens durante o treinamento online. Em cada época, o modelo vê uma variação ligeiramente diferente das imagens que foram fornecidas. Um desses aumentos é chamado de aumento mosaico. Isso envolve unir quatro imagens, forçando o modelo a aprender objetos em novos locais, em oclusão parcial e contra diferentes pixels do entorno. No entanto, esse aumento demonstrou empiricamente degradar o desempenho se realizado durante toda a rotina de treinamento. É vantajoso desligá-lo nas últimas dez épocas de treinamento. Esse tipo de mudança é um exemplo da atenção cuidadosa que a modelagem YOLO recebeu ao longo do tempo no repositório YOLOv5 e na pesquisa do YOLOv8 (SOLAWETZ, 2023).

2.4.5.5 Comparação entre YOLOv5 e YOLOv8

A razão pela qual o YOLOv8 é constantemente comparado ao YOLOv5 se dá por causa que o desempenho e as métricas do YOLOv5 estão próximos do YOLOv8. No entanto, o YOLOv8 supera o YOLOv5 em aspectos como um melhor mAP, como visto na Figura 20. Junto com um melhor mAP, isso mostra que o YOLOv8 tem menos *outliers* quando comparado ao RF100. RF100 é um conjunto de dados de 100 amostras do universo Roboflow, que é um repositório de 100.000 conjuntos de dados. Também é possível ver o YOLOv8 superando o YOLOv5 para cada categoria do RF100. Na Figura 21, é possivel ver que o YOLOv8 produz resultados semelhantes ou até melhores em comparação com o YOLOv5 (SOLAWETZ, 2023).

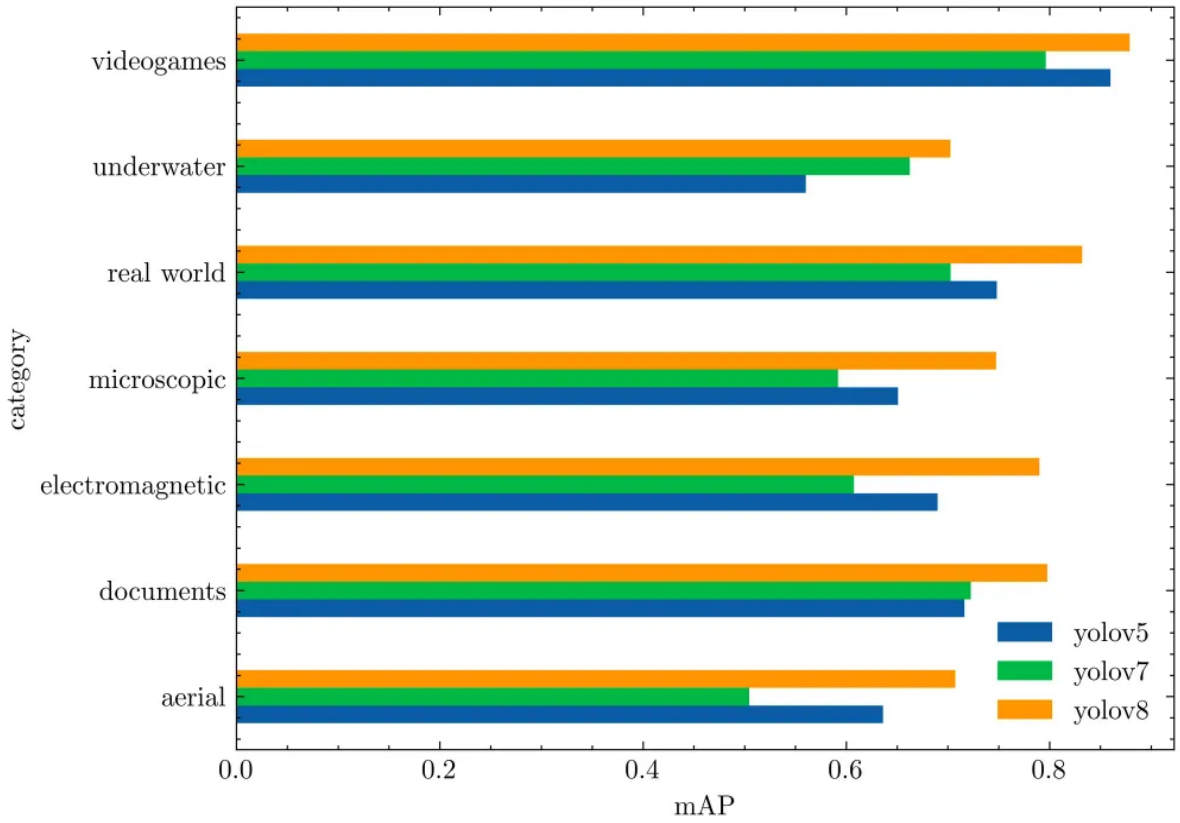
Figura 20 - Desempenho do mAP50 para o dataset RF100.



Fonte: Solawetz (2023).

Como mencionado anteriormente, o YOLOv8 usa uma nova arquitetura que combina módulos FAN e PAN. O FPN é usado para gerar mapas de recursos em várias escalas e resoluções, enquanto o PAN é usado para agregar recursos de diferentes níveis da rede para melhorar a precisão. Os resultados dos módulos FAN e PAN combinados são melhores do que o YOLOv5, que usa uma versão modificada da arquitetura CSPDarknet. Esta versão modificada do CSPDarknet é baseada em conexões parciais entre estágios, o que melhora o fluxo de informações entre diferentes partes da rede.

Figura 21 - Desempenho do mAP50 para as categorias do dataset RF100.



Fonte: Solawetz (2023).

Outra diferença entre os dois modelos é baseada em seus dados de treinamento. O YOLOv8 foi treinado em um conjunto de dados maior e mais diversos em comparação com o YOLOv5. O YOLOv8 foi treinado em uma mistura do conjunto de dados COCO e vários outros conjuntos de dados, enquanto o YOLOv5 foi treinado principalmente no conjunto de dados COCO. Por causa disso, o YOLOv8 tem um melhor desempenho em uma ampla variedade de imagens.

2.4.6 Inteligência Artificial Embarcada

Ao longo dos anos, o desenvolvimento da inteligência artificial e suas aplicações reduziu bastante a complexidade de muitos modelos de aprendizado de máquina, tornando mais fácil implantá-los em dispositivos com restrição de recursos. Além disso, surgiu o suporte correspondente para modelos e algoritmos nesses dispositivos (DICK et al., 2019).

O conceito de IA embarcada foi introduzido pela primeira vez no trabalho de Guo (2012), que propôs que a IoT poderia evoluir para a Web Semântica das Coisas (W2T) e enfatizou que a inteligência embarcada sobre indivíduos, o meio ambiente e a sociedade poderia aumentar o número de usuários de sistemas IoT existentes, promover a sustentabilidade ambiental e aumentar a consciência social. Desenvolvimentos recentes em IA embarcada são descritos como a combinação de IA embarcada com tecnologia IoT (GUO, 2012).

O desenvolvimento atual da IA embarcada é bidirecional: a otimização de modelos e algoritmos de IA reduz a dificuldade de implantá-los em dispositivos embarcados, enquanto os aceleradores de hardware em dispositivos embarcados aumentam o suporte para modelos e algoritmos de IA. Além disso, os recursos de hardware estão sendo desenvolvidos, e a IA está avançando rapidamente em dispositivos móveis. Por exemplo, Poniszewska-Maranda (2018) descreve a implantação de redes neurais em telefones celulares, e também existem redes neurais projetadas especificamente para dispositivos móveis, como MobileNet de Howard et al. (2017).

Para implantar inteligência artificial em dispositivos embarcados, vários aspectos precisam ser considerados: (1) escolha de uma plataforma de implantação adequada, (2) uso de aceleradores de hardware, (3) aplicação de técnicas de compressão de modelo e (4) aprimoramento do suporte de hardware para algoritmos e modelos de IA (DENG et al. 2020).

No contexto deste trabalho, a inteligência artificial embarcada seria o objetivo maior da pesquisa. Uma vez estabelecido o melhor modelo, este poderá ser embarcado em um drone, dentro do projeto maior no qual este trabalho está inserido. Sendo assim, a seleção dos algoritmos também considerou o que a literatura apresenta sobre IA Embarcada.

3 TRABALHOS RELACIONADOS

3.1 DEFINIÇÕES DE BUSCA

Para realizar as pesquisas, foi utilizado dois motores de pesquisas acadêmicas chamados de Google Acadêmico e Scopus, escolhidos por sua capacidade de pesquisar estudos publicados em livros periódicos, anais de eventos e bases de instituições de ensino. As buscas utilizadas foram:

Quadro 1 – Strings de busca.

Identificação	***String* de busca**
B01	Exposed soil artificial intelligence
B02	Pasture degradation machine learning
B03	Exposed soil artificial intelligence yolo
B04	Pasture degradation machine learning yolo
B05	Degradação de pastagens e inteligência artificial

Fonte: Autoria Própria.

Critérios de Seleção

- O trabalho deve estar disponível gratuitamente na internet para a leitura;
- O trabalho deve ter sido publicado entre 2019 e 2023;
- O trabalho pode ter sido escrito nos idiomas inglês ou português;
- O trabalho deve ter relação com a aplicação de aprendizado de máquina no mundo do agronegócio.

Critérios de Exclusão

- O trabalho não estar disponível gratuitamente na internet para a leitura;
- O trabalho não foi publicado entre 2019 e 2023;
- O trabalho não foi escrito nos idiomas inglês ou português;
- O trabalho não tem relação com a aplicação de aprendizado de máquina no mundo do agronegócio;
- O trabalho não é um resumo ou uma postagem em blog.

3.2 HYBRID MACHINE LEARNING METHODS COMBINED WITH COMPUTER VISION APPROACHES TO ESTIMATE BIOPHYSICAL PARAMETERS OF PASTURES

Franco et al. (2023) aborda a importância da gestão agrícola na segurança alimentar global devido ao crescimento populacional. O uso de tecnologias no campo é destacado como uma solução para atender às demandas futuras, especialmente a recuperação de pastagens degradadas, que afetam a produção animal. O sensoriamento remoto é essencial para obter informações fenotípicas das plantas para a extração de atributos relevantes. Para resolver esse desafio, o trabalho explora o uso de recursos computacionais, como visão computacional e algoritmos de aprendizado de máquina, para extrair informações de imagens das pastagens e otimizar o processo de seleção de atributos e ajuste de hiperparâmetros dos modelos. Foram utilizados três modelos de aprendizado de máquina, SVR (*Support vector regression*), LASSO (*Least absolute shrinkage and selection operator*) e em especial a MLP (*Multi-Layer Perceptron*) que teve o modelo com maior desempenho na estimativa da altura e biomassa das pastagens de *Panicum maximum* observadas. Para os dois problemas estudados nesse trabalho, foi determinado um coeficiente médio de 0.495 para estimar biomassa (R^2) e 0.656 para estimar altura. Além disso, foi provado que o índice TGI (índice sensível a clorofila nas plantas) é relevante para as predições do MLP e LASSO.

3.3 DIAGNOSIS OF DEGRADED PASTURES USING AN IMPROVED NDVI-BASED REMOTE SENSING APPROACH: AN APPLICATION TO THE ENVIRONMENTAL PROTECTION AREA OF UBERABA RIVER BASIN (MINAS GERAIS, BRAZIL)

A pesquisa de Valle Júnior et al. (2019) se concentra na área da Bacia do Rio Uberaba, em Minas Gerais, onde mais da metade das pastagens é considerada degradada. A metodologia proposta envolveu a coleta de dados de solo e a análise de imagens de satélite, com o objetivo de mapear áreas de pastagens degradadas. O uso do Índice de Vegetação por Diferença Normalizada (NDVI) é proposto como uma ferramenta para identificar pastagens degradadas. O NDVI é uma métrica que se baseia em imagens de satélite e tem sido usado em pesquisas ambientais para avaliar a saúde das plantas.

Como parâmetros foi utilizado dados de solo, incluindo nutrientes, matéria orgânica, textura do solo e resistência à penetração, para diferenciar entre pastagens saudáveis e degradadas. A metodologia envolveu a análise estatística dos parâmetros de solo e a criação de modelos de regressão não linear para estabelecer "impressões digitais" NDVI sazonais para diferentes tipos de pastagens. Em suma, a metodologia proposta, que combina dados de solo e imagens de satélite NDVI, é eficaz na identificação e mapeamento de pastagens degradadas, conseguindo prever com acurácia 84,1% da área testada.

3.4 DRIVERS OF DEGRADATION OF PASTURES IN THE CERRADO NORTH OF MINAS GERAIS – BR

Silva et al. (2023) foca nas pastagens da região norte de Minas Gerais. Esta região possui 89 municípios, sendo 84 deles localizados na borda do Cerrado, o bioma predominante na área. Os pesquisadores utilizaram o Índice de Vegetação por Diferença Normalizada (NDVI) para mapear a degradação das pastagens. Além disso, variáveis relacionadas ao relevo e fatores climáticos também foram selecionadas. O trabalho define quatro classes de degradação das pastagens: Ausência de Degradação, Degradação Leve, Degradação Moderada e Degradação Severa, com base no NDVI. Para isso, foi calculado o Índice de Degradação das Pastagens (PDI), considerando a área de pastagens nas diferentes classes de degradação.

A análise estatística foi realizada com um modelo de regressão linear múltipla, incluindo variáveis climáticas, tais como temperatura do ar, precipitação anual e sazonalidade da precipitação. Por fim, o estudo revelou que toda a área analisada apresenta algum grau de degradação das pastagens. A maioria (79%) possui pastagens levemente degradadas, enquanto 21% exibem níveis moderados de degradação. As variáveis climáticas, como temperatura do ar, sazonalidade da precipitação e precipitação anual, explicaram significativamente (50%) a degradação das pastagens na região. Foi possível concluir também que as temperaturas elevadas e sazonalidade na distribuição de chuvas estão associadas a maiores níveis de degradação, enquanto áreas com maior precipitação mantêm pastagens de melhor qualidade.

3.5 ESTIMATING VEGETATION INDEX FOR OUTDOOR FREE-RANGE PIG PRODUCTION USING YOLO

Oh et al. (2023) tinha como objetivo estimar quantitativamente o nível de dano na área de pastagem na produção de suínos ao ar livre em regime extensivo, utilizando um Veículo

Aéreo Não Tripulado (VANT) com sensor de imagem RGB. Dez imagens de campos de milho foram capturadas por um VANT ao longo de aproximadamente duas semanas, período no qual os porcos poderiam pastar livremente em um campo de milho medindo 100 × 50 m². O VANT utilizado neste trabalho foi o Phantom 2 Vision fabricado pela DJI e as imagens foram capturadas aproximadamente no mesmo horário e posição durante as duas semanas analisadas.

As imagens capturadas foram redimensionadas em forma de grande, em imagens menores de 448x448, que é o tamanho de entrada para treinamento no modelo YOLOv4, resultando em um conjunto de dados de 320 imagens, que, passaram por um processo que aumento de dados de tal maneira que a base final continha 24.768 imagens. As imagens poderiam possuir 3 rótulos: (1) Milho intacto, (2) Milho danificado, (3) Milho com resíduos. A rede foi treinada por um máximo de 50 épocas e com tamanho de *mini-batch* em 32. Em relação a parâmetros de treino, foi utilizado a taxa de treinamento inicial em 0,001, além do uso do otimizador Adam.

Na parte de hardware, o autor tinha disponível um processador Intel i9-12900 e um acelerador gráfico NVIDIA RTX-A6000. A respeito de resultados, o autor não traz métricas sobre o treinamento do modelo, mas sim, resultados sobre a contagem de milho nas imagens com o passar dos dias. Essa análise mostra que no primeiro dia, quase que todas as ocorrências do objeto milho eram da classe Milho intacto e já no nono dia, quase já não existiam mais exemplares dele. Portanto o autor afirma que o modelo se mostrou eficiente para estimar a taxa de ocupação da pastagem com milho e que poderia ser adaptado e utilizado para qualquer outro tipo de cultura.

3.6 CONSIDERAÇÕES SOBRE OS TRABALHOS CORRELATOS

No Quadro 2 tem-se um comparativo entre o trabalho proposto e os trabalhos correlatos. As variáveis consideradas no comparativo foram a fonte de dados utilizada, o modelo utilizado e o índice.

Quadro 2 - Quadro comparativo dos trabalhos relacionados.

Trabalho Correlato	Base de Dados	Modelo Utilizado	Índice
Hybrid machine learning methods combined with computer vision approaches to estimate biophysical parameters of pastures	Criada a partir de fotos comuns (câmera ou celular)	SVR, LASSO e MLP	TGI
Diagnosis of degraded pastures using an improved NDVI-based remote sensing approach: an application to the environmental protection area of Uberaba river basin (Minas Gerais, Brazil)	Imagens de Satélite	Regressão não linear	NDVI
Drivers of degradation of pastures in the cerrado north of Minas Gerais – BR	Imagens de Satélite	Regressão linear múltipla	NDVI
Estimating vegetation index for outdoor free-range pig production using yolo	Imagens de Drone	YOLOv4	Nenhum
Trabalho Proposto	Imagens de Drone	YOLOv4 e YOLOv8	Nenhum

Fonte: Autoria Própria.

Verifica-se que a fonte de dados preponderante se refere a imagens de drones. Já os modelos utilizados têm-se a regressão e os modelos Yolo. E considerando o modelo Yolo, o mesmo não utilizou índices.

4 DESENVOLVIMENTO

Neste capítulo será apresentado o desenvolvimento do projeto. Para isto será apresentado como foi feita a coleta das imagens, o processo de classificação, processo de pré-processamento.

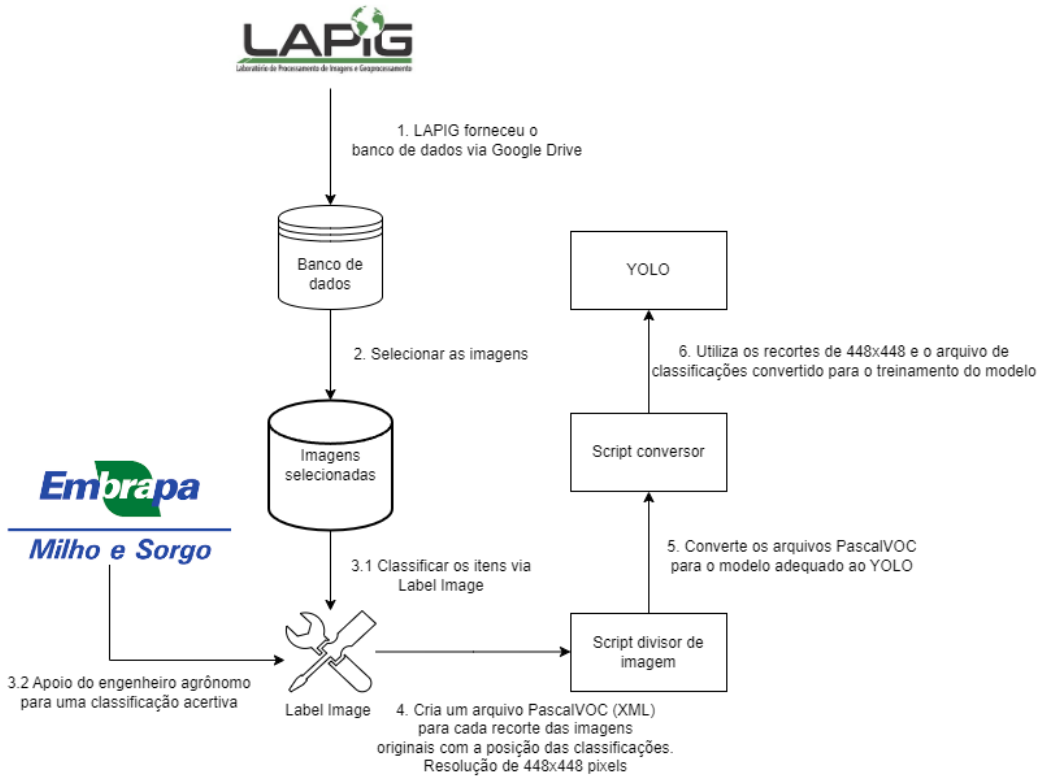
4.1 VISÃO GERAL DO PROJETO

Os algoritmos YOLOv4 e YOLOv8 foram selecionados para o desenvolvimento deste trabalho. O YOLOv4 foi escolhido dentre as demais versões disponíveis do YOLO, pois, durante as pesquisas na literatura, esta versão estava presente na grande maioria dos trabalhos relacionados à visão computacional na agricultura. Já o YOLOv8 foi escolhido por ser a versão mais recente, prometendo um aumento de desempenho e precisão em comparação ao YOLOv4.

O projeto proposto visa utilizar tecnologias avançadas, como o YOLOv4 e YOLOv8, aliadas aos princípios da Agricultura de Precisão e à integração de TIC, para identificar áreas de solo exposto e cupinzeiros em pastagens brasileiras. Isso proporcionará uma abordagem inovadora e eficaz para analisar os níveis de degradação das pastagens, contribuindo para a redução da degradação e promovendo práticas mais sustentáveis na agropecuária brasileira.

Além disso, esse projeto é viabilizado pela EMBRAPA Milho e Sorgo de Minas Gerais e pelo Laboratório de Processamento de Imagens e Geoprocessamento (LAPIG) da UFG. A EMBRAPA tem como papel fornecer apoio especialista para a classificação das imagens e o LAPIG fornece a base de dados.

Figura 22 - Visão geral do projeto.



Fonte: Autoria Própria.

4.2 CARACTERÍSTICAS DA BASE DE DADOS

A coletânea de imagens e vídeos fornecido pelo LAPIG contém imagens de pastagens dos biomas Amazônia e Cerrado localizadas em Goiás e Mato Grosso que foram retiradas no período de março de 2022 até dezembro de 2022. Segundo o LAPIG, essas imagens foram feitas com o objetivo de ajudar na caracterização das pastagens em nível de degradação e avaliar a cobertura de solo (em % de solo exposto).

Nesses campos foram utilizados 3 modelos de drones, conforme apresenta o Quadro 3.

Quadro 3 – Informações técnicas dos drones utilizados.

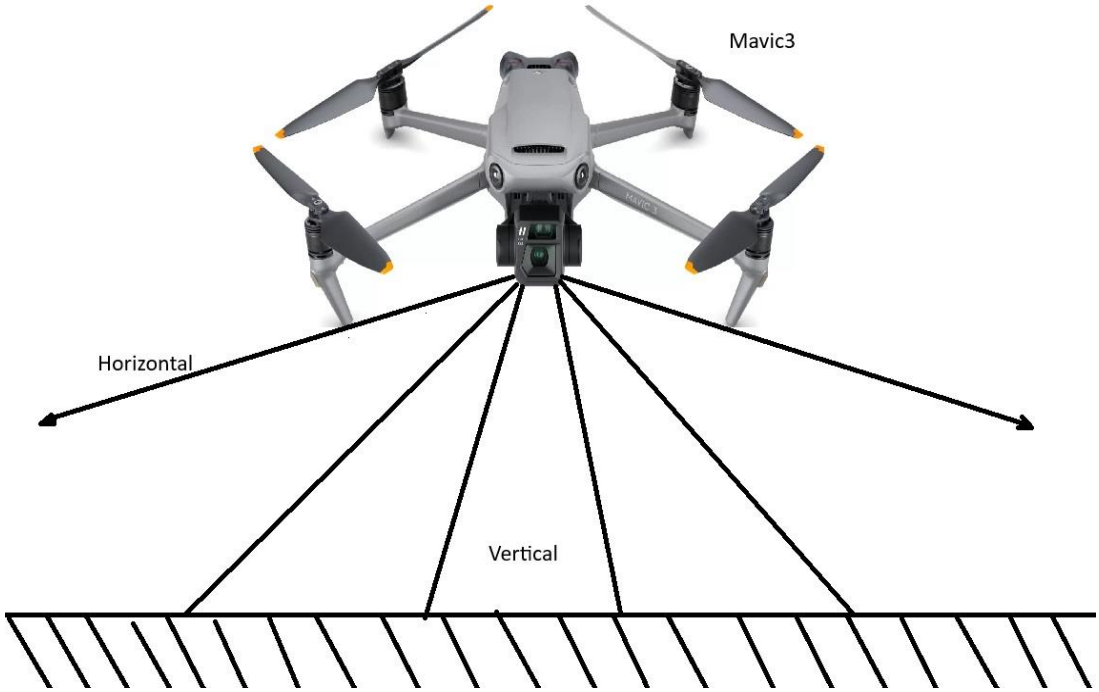
Modelo	Vídeo	Foto
(Parrot) Anafi	4K Cinema: 4096x2160 24fps	Wide: 21MP (5344x4016) / 4:3

	4K UHD: 3840x2160 24/25/30fps FHD: 1920x1080 24/25/30/48/50/60fps	Objetiva retilínea: 16MP (4608x3456) / 4:3
(DJI) Mavic 3	5.1K: 5120×2700 24/25/30/48/50fps 4K Cinema: 4.096×2.160 24/25/30/48/50/60fps 4K: 3.840×2.160 24/25/30/48/50/60fps FHD: 1920×1080 24/25/30/48/50/60fps	20MP (5280×3956) / 4:3
(DJI) Phantom 4	4K UHD: 4096×2160 24/25fps 4K: 3840×2160 24/25/30fps 2.7K: 2704×1520 24/25/30fps FHD: 1920×1080 24/25/30/48/50/60/120fps HD: 1280×720 24/25/30/48/50/60fps	12MP (4000×3000) / 4:3

Fonte: Autoria própria.

Por padrão, a maioria das imagens foram retiradas com resolução 4K e proporção 4:3 porém sem um padrão de captura, logo, as imagens se enquadram em diferentes ângulos de visão (Figura 23), altura do voo entre outras características, que impactam no resultado do treinamento.

Figura 23 - Ângulos de visão.



Fonte: Autoria Própria.

Essas imagens foram disponibilizadas via Google Drive, contamos com uma base de dados privada de 2042 imagens e 228 vídeos de 4 campos diferentes. Por ser uma base grande de imagens cruas, ou seja, sem rotulações, apenas 100 imagens do campo 4 foram utilizadas nesse trabalho. Pela falta de padrão na captura das imagens, dada as perspectivas do ângulo de visão, foi feita a separação das imagens em dois tipos:

1. Imagens com ângulo de visão de perspectiva horizontal, contabilizando 50 imagens;
2. Imagens com ângulo de visão de perspectiva vertical, contabilizando 50 imagens.

Todas as imagens selecionadas possuem pelo menos 1 item classificado dos três rótulos estipulados: Pasto; Solo exposto; Cupim.

Quadro 4 – Levantamento do tamanho da base de dados fornecida.

Campo	Imagens	Vídeos
Campo 1	718	110

Campo 2	327	53
Campo 3	194	56
Campo 4	803	9
Total	2042	228

Fonte: Autoria própria.

4.3 CLASSIFICAÇÃO

Para facilitar o processo de classificação foi utilizado uma plataforma open source chamada Label Studio. Nele é possível importar as imagens e criar grupos de classificação facilitando a organização da base de dados. Para exportar as imagens classificadas é possível exportar diretamente os valores das *bounding boxes* para o formato do YOLO em TXT. Os objetos que estão sendo rotulados para a proposta deste trabalho são: Pasto; Solo exposto; Cupim.

O processo de classificação foi um trabalho que demandou muito tempo entre 30 minutos a mais de 4 horas por imagem, pois, como a ideia é fazer uma análise de área na imagem providenciada, é necessário rotular as imagens por completo. Portanto, dependendo da complexidade das imagens, ou seja, imagens que não são tão homogêneas e demandam uma quantidade muito grande de rótulos para alcançar a marcação total da imagem, foram responsáveis por uma grande parcela do esforço desse trabalho. Esse tipo de imagem pode acabar tendo entre 1000 a mais de 3000+ rótulos (Figuras 26 e 27) e imagens mais homogêneas possuem entre 50 a 999 rótulos (Figuras 24 e 25).

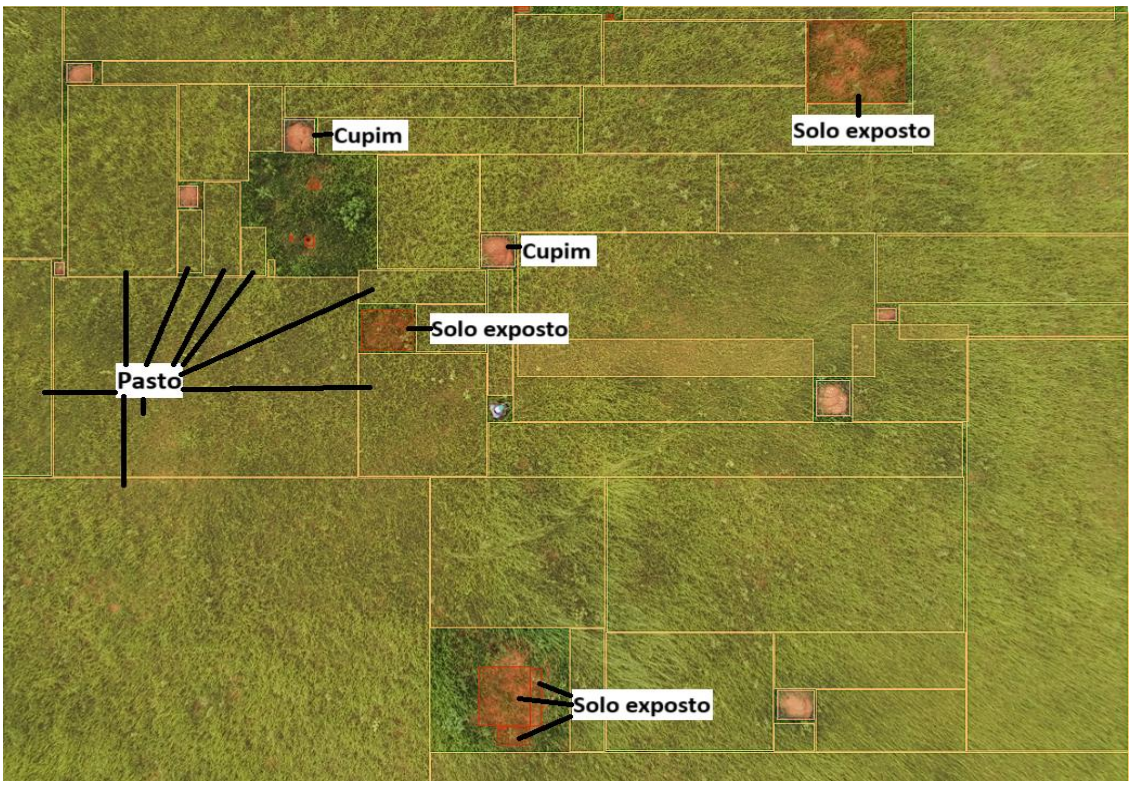
Dado o grande esforço necessário para a classificação das imagens foram utilizadas apenas 50 imagens das 100 selecionadas. Também foi mantido a proporção de 50% para cada ângulo de visão selecionado.

Figura 24 - Exemplo de imagem sem rotulações do Campo 4.

Fonte: Autoria Própria.

Figura 25 - Exemplo de imagem com rotulações do Campo 4.

Amarelo - Pasto; Vermelho - Solo Exposto; Bege - Cupim;



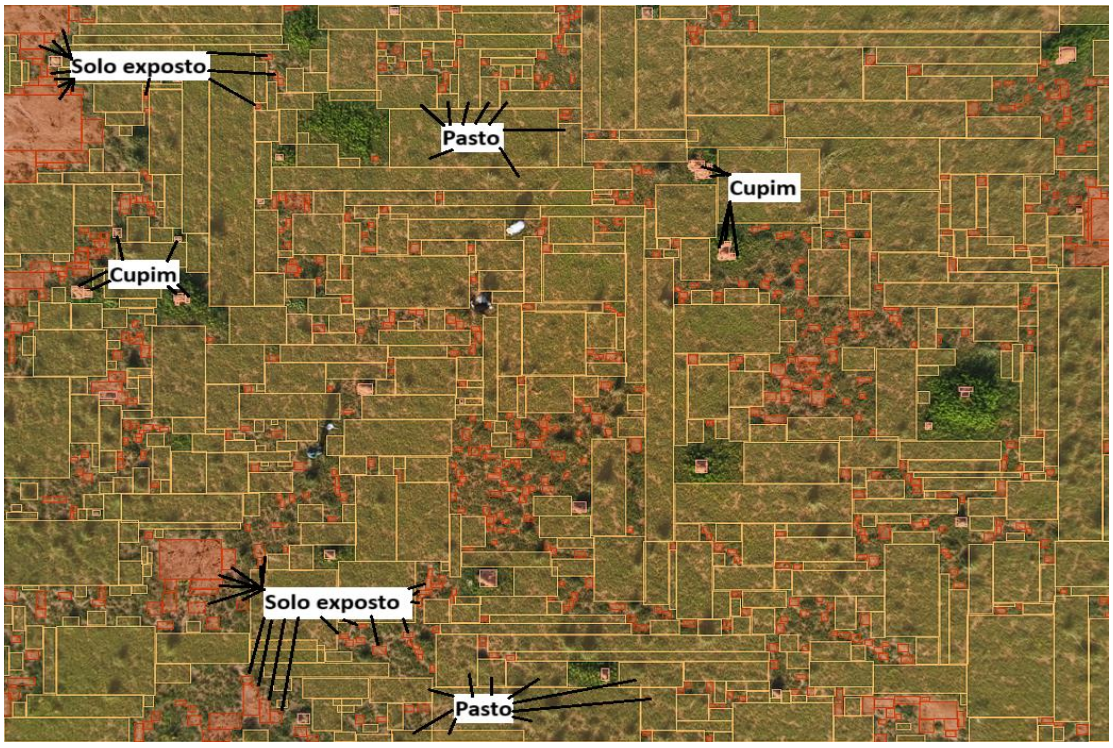
Fonte: Autoria Própria.

Figura 26 - Exemplo de imagem complexa sem rotulações do Campo 4.

Fonte: Autoria Própria.

Figura 27 - Exemplo de imagem complexa com rotulações do Campo 4.

Amarelo - Pasto; Vermelho - Solo Exposto; Bege - Cupim;



Fonte: Autoria Própria.

4.4 PRÉ-PROCESSAMENTO

Após a classificação das imagens na plataforma Label Studio e a exportação das imagens e seus respectivos conjuntos de *bounding boxes* no formato YOLO em TXT, foi aplicado uma série de transformações nas imagens, fazendo o que chamamos de Aumento de dados, neste caso, para ter um aumento de 100% no tamanho da base de dados.

A aplicação das transformações foi feita em cima das imagens já rotuladas, visando poupar tempo e não ter que classificar mais uma quantidade significativa de imagens. Por esse motivo, não foram utilizadas transformações que alteram de alguma forma as coordenadas das *bounding boxes* já marcadas, já que, manter as rotulações em suas devidas posições seria um complicador para o projeto, sendo assim, transformações como rotação e zoom foram descartadas.

Foi desenvolvido um script Python utilizando a biblioteca *imgaug*, que serve para aumento de dados em experimentos de aprendizado de máquina. Essa biblioteca suporta uma grande variedade de técnicas de aumento, permitindo combina-los e executa-los de forma randômica ou em múltiplos núcleos de processador. As transformações foram utilizadas de forma sequencial em cima de cada imagem para torná-la o mais diferente possível da imagem original. Sendo assim, as transformações utilizadas no pré-processamento para aumento de dados foram:

- Alteração de brilho pixel a pixel (80% - 120%);
- Alteração de contraste (60% - 140%);
- Espelhamento da imagem na vertical (50% de chance);
- Espelhamento da imagem na horizontal (50% de chance);
- Desfoque aleatório de 0 (sem desfoque) a 3 (desfoque significativo) com 50% de chance de ser aplicado.

Código Fonte – Script para aplicar as transformações de aumento de dados.

```
import os
import cv2
import imgaug.augmenters as iaa
import imgaug as ia


def adjust_bbox(bbox, mask):
```

```
    # Encontrar interseção da bounding box com a máscara preta
    intersection = bbox.intersection(mask)
    # Calcular área da interseção
    intersection_area = intersection.area
    # Se a interseção for maior que um limite, redimensionar a bounding box
    if intersection_area > 0.5 * bbox.area:
        # Encontrar a maior área retangular dentro da imagem não preta
        max_bbox = mask.find_biggest_bounding_box().bounding_box
        # Redimensionar a bounding box para caber dentro da área máxima
        bbox = bbox.clip(max_bbox)
    return bbox

# Diretório onde estão suas imagens originais
input_dir = "data/images"
# Diretório onde você deseja salvar as imagens aumentadas
output_dir = "output"

# Carregue as imagens originais e suas anotações (bounding boxes)
images = []
annotations = []  # Lista de listas contendo as anotações para cada imagem

for filename in os.listdir(input_dir):
    if filename.endswith(".JPG"):
        img_path = os.path.join(input_dir, filename)
        img = cv2.imread(img_path)
        images.append(img)

        boxes: list[ia.BoundingBox] = []
        with open(f'data/labels/{filename.split('.')[0]}.txt', 'r') as f:
            lines = f.readlines()
            for line in lines:
                [ID, X, Y, W, H] = map(float, line.split(' '))
                x_min = (X - W / 2) * img.shape[1]
                y_min = (Y - H / 2) * img.shape[0]
                x_max = (X + W / 2) * img.shape[1]
                y_max = (Y + H / 2) * img.shape[0]
                box = ia.BoundingBox(x1=x_min, y1=y_min, x2=x_max, y2=y_max, label=int(ID))
                boxes.append(box)
        annotations.append(boxes)
```

```
# Defina as transformações de aumento de dados
augmentation = iaa.Sequential([
    iaa.Fliplr(0.5),  # Espelhar horizontalmente
    iaa.Flipud(0.5),  # Espelhar verticalmente
    iaa.Multiply((0.8, 1.2)),  # Ajuste de brilho
    iaa.LinearContrast((0.6, 1.4)),  # Ajuste de contraste
    iaa.Sometimes(0.5, iaa.GaussianBlur((0.0, 3.0)))  # Desfoque gaussiano
])

# Aplique as transformações às imagens e ajuste as bounding boxes
augmented_images = []
augmented_annotations: list[list[ia.BoundingBox]] = []

for img, ann in zip(images, annotations):
    transformed = augmentation(image=img, bounding_boxes=ann)
    augmented_images.append(transformed[0])
    augmented_annotations.append(transformed[1])

# Salve as imagens aumentadas e suas anotações
for i, (aug_img, aug_ann) in enumerate(zip(augmented_images, augmented_annotations)):
    output_path = os.path.join(output_dir, f"images/imagem_aumentada_{i}.JPG")
    cv2.imwrite(output_path, aug_img)
    img = cv2.imread(output_path)

    # Salva as anotações para a imagem aumentada (no formato YOLO)
    with open(f"{output_dir}/labels/imagem_aumentada_{i}.txt", "w") as f:
        for ann in aug_ann:
            x_center = (ann.x1 + ann.x2) / (2 * img.shape[1])
            y_center = (ann.y1 + ann.y2) / (2 * img.shape[0])
            width = (ann.x2 - ann.x1) / img.shape[1]
            height = (ann.y2 - ann.y1) / img.shape[0]

            f.write(f"{ann.label}      {x_center:.16f}      {y_center:.16f}      {width:.16f}
{height:.16f}\n")

print(f"{len(augmented_images)} imagens aumentadas foram salvas em {output_dir}")
```

4.5 ESPECIFICAÇÕES E IMPLEMENTAÇÃO

A implementação e treinamento dos modelos YOLOv4 e YOLOv8 foram feitos de maneiras diferentes e em máquinas diferentes. Para o YOLOv4 foi utilizado a implementação do framework *darknet* disponibilizada por BOCHKOVSKIY (2020). Para o treinamento desse modelo foi utilizada um notebook Acer Aspire Nitro 5 AN515-44-R8HN com as seguintes configurações:

- Processador AMD Ryzen 7 4800H 8 núcleos / 16 threads de 2.90 GHz até 4.20 GHz;
- 16 GB de memória RAM DDR4 3200 MHz;
- Placa de vídeo dedicada GTX 1650ti com 4GB VRAM GDDR6; e
- Sistema operacional Windows 11, versão 23H2.

Para o YOLOv8 foi implementado um script Python utilizando as ferramentas da biblioteca da Ultralytics LLC (2023), instituição responsável por sua concepção. Como pode ser observado no Quadro 5 o YOLOv8 possui diversas implementações disponíveis para utilizar, porém, foi selecionado apenas duas, a YOLOv8n e a YOLOv8x que foram analisadas neste trabalho. Para o treinamento foi utilizado uma máquina com as seguintes configurações:

- Processador AMD Ryzen 7 5700X 8 núcleos / 16 threads de 3.40 GHz até 4.60 GHz;
- 32 GB de memória RAM DDR4 2666MHz;
- Placa de vídeo dedicada AMD Radeon RX 7900XTX com 24GB VRAM GDDR6; e
- Sistema operacional Linux Zorin OS Core 17.1.

Dada a utilização de uma placa de vídeo AMD nesta máquina para o treinamento dos modelos YOLOv8, foi preciso instalar e configurar um software específico da AMD chamado ROCm, que atualmente situa-se como uma alternativa ao CUDA, software proprietário da NVIDIA. O AMD ROCm é um conjunto de software de código aberto projetado para programar unidades de processamento gráfico (GPUs) da AMD para computação de alto desempenho, inteligência artificial e aprendizado de máquina. Atualmente o ROCm tem um suporte melhor a plataformas Linux do que Windows, por isso da necessidade de utilizar um sistema operacional Linux para o treinamento dos modelos YOLOv8.

Código Fonte – Script para o treinamento do YOLOv8.

```
import ultralytics
from ultralytics import YOLO
import time
import datetime
import os

ultralytics.checks()
# carrega o modelo disponibilizado pela Utralytics
model = YOLO("yolov8n.yaml") # build new model from scratch
# realiza o treinamento
model.train(data="config.yaml", epochs=15000, batch=8, patience=0) # train model
```

Quadro 5 - Comparativo dos modelos disponíveis para o YOLOv8.

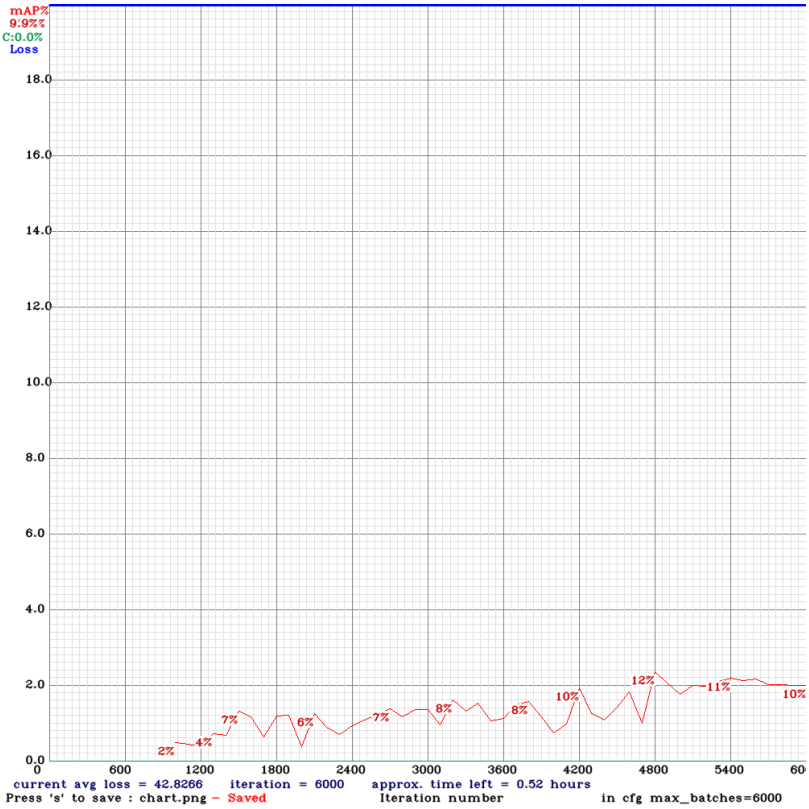
Modelo	**Tamanho (pixels)**	**mAP 50-95**	**Velocidade CPU ONNX (ms)**	**Velocidade A100 TensorRT (ms)**	**Parâmetros (milhões)**	**FLOPs (bilhões)**
YOLOv8n	640	37.3	80.4 / 12,43 fps	0.99 / 1010 fps	3.2	8.7
YOLOv8s	640	44.9	128.4 / 7,78 fps	1.20 / 833 fps	11.2	28.6
YOLOv8m	640	50.2	234.7 / 4,26 fps	1.83 / 546 fps	25.9	78.9
YOLOv8l	640	52.9	375.2 / 2,66 fps	2.39 / 418 fps	43.7	165.2
YOLOv8x	640	53.9	479.1 / 2 fps	3.53 / 283 fps	68.2	257.8

Fonte: Ultralytics LLC (2023).

5 RESULTADOS

Todos os treinamentos realizados foram feitos utilizando os hiperparâmetros padrões e sem nenhum modelo pré treinado a partir de transfer learning, para manter a comparação o mais justa possível entre os modelos. Apenas no processo de fine-tuning houve alteração nos hiperparâmetros, visando melhorar o desempenho do modelo YOLOv8 selecionado, que será discutido em seguida. A rede do modelo YOLOv4 foi treinada por um total de 6000 épocas utilizando o *dataset* completo pré-processado de 100 imagens. O desempenho durante o treinamento pode ser observado na Figura 28 e teve um mAP50 de 9,9%, sendo considerado insatisfatório. Foi feito também um treinamento posterior com 12000 épocas, porém, não gerou um resultado melhor do que o primeiro treinamento.

Figura 28 – Desempenho durante o treinamento do YOLOv4 com 100 imagens no *dataset*.

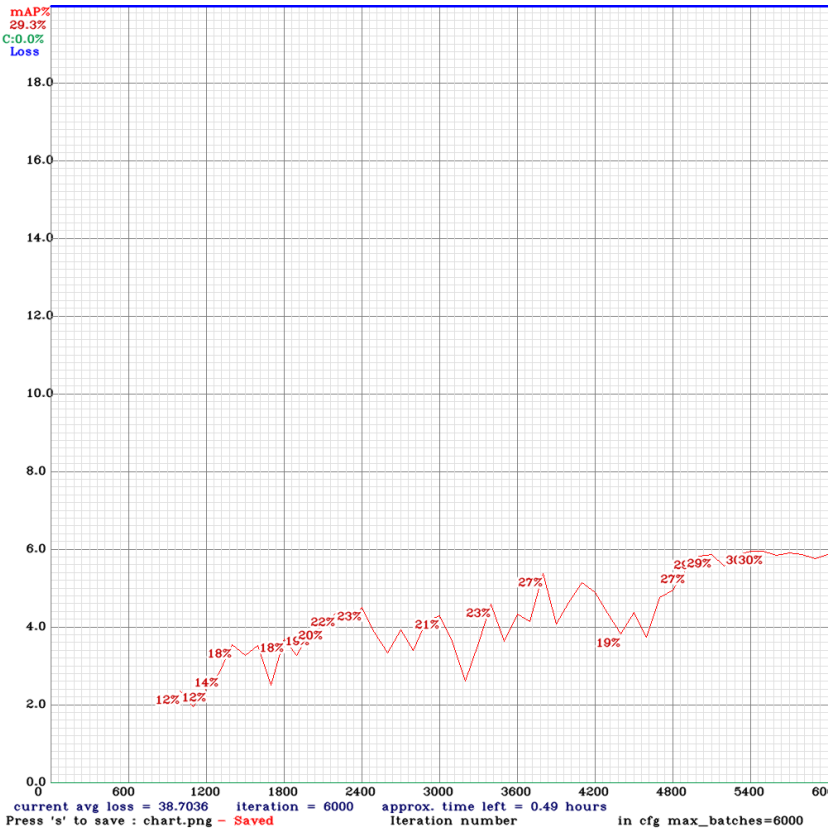


Fonte: Autoria própria.

Dado o resultado insatisfatório do treinamento anterior, foi alterado o dataset para treinamento. Visando o objetivo de analisar a área da imagem, foi utilizado apenas as imagens de ângulo de visão de perspectiva vertical. Portanto, o desempenho durante o treinamento da

rede o modelo YOLOv4 com esse *dataset* pode ser observado na Figura 29, tendo um mAP50 de 29,3%, mesmo assim o resultado final desse treinamento continua insatisfatório apesar da melhora na precisão.

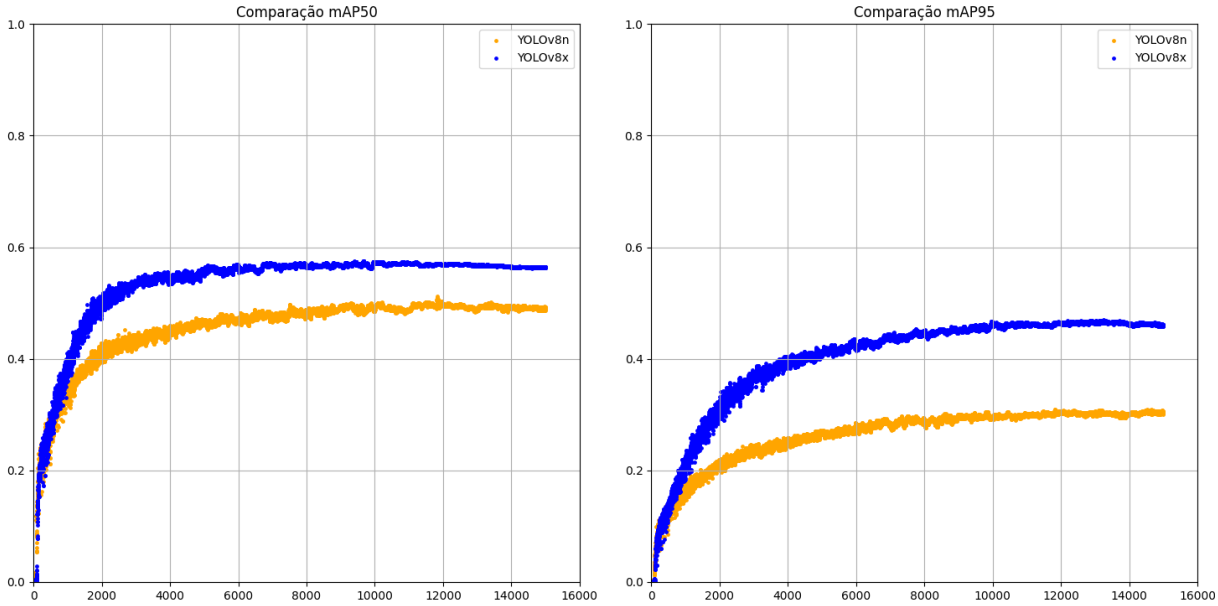
Figura 29 – Desempenho durante o treinamento do YOLOv4 com 50 imagens no dataset.



Fonte: Autoria própria.

Com uma melhora de aproximadamente 300% em comparação com o treinamento feito com o *dataset* com 100 imagens, esse *dataset* com 50 imagens de ângulo de visão de perspectiva vertical seguiu adiante para o treinamento dos modelos YOLOv8n e YOLOv8x já apresentados anteriormente. Os modelos YOLOv8 também foram treinados com os hiperparâmetros padrões por 15000 épocas. O desempenho durante o treinamento desses dois modelos pode ser observado na Figura 30, onde o modelo YOLOv8n em sua melhor época teve um mAP50 de 49,4%, já o modelo YOLOv8x em sua melhor época teve um mAP50 de 56,7%.

Figura 30 – Desempenho durante o treinamento dos modelos YOLOv8 com 50 imagens no *dataset*.



Fonte: Autoria própria.

Seguindo o propósito de criar um modelo embarcável e eficiente, o modelo YOLOv8n foi selecionado para passar por um processo de *fine-tuning* através dos otimizadores disponíveis no modo de *tuning* de um modelo YOLOv8 pelas ferramentas da Ultralytics. No Quadro 6, pode-se observar os otimizadores que foram selecionados e executados por 1500 épocas por 15 iterações cada um.

Quadro 6 – Comparativo de desempenho entre os otimizadores.

Otimizador	**Precisão**	**Recall**	**mAP50**	**mAP50-95**
YOLOv8n Original	0,759	0,321	0,494	0,306
Adam	0,784	0,369	0,539	0,348
AdamW	0,737	0,328	0,495	0,297
NAdam	-	-	-	-
RAdam	0,744	0,354	0,514	0,314
RMSProp	0,291	0,214	0,257	0,114
SGD	0,784	0,326	0,504	0,311

Fonte: Autoria própria.

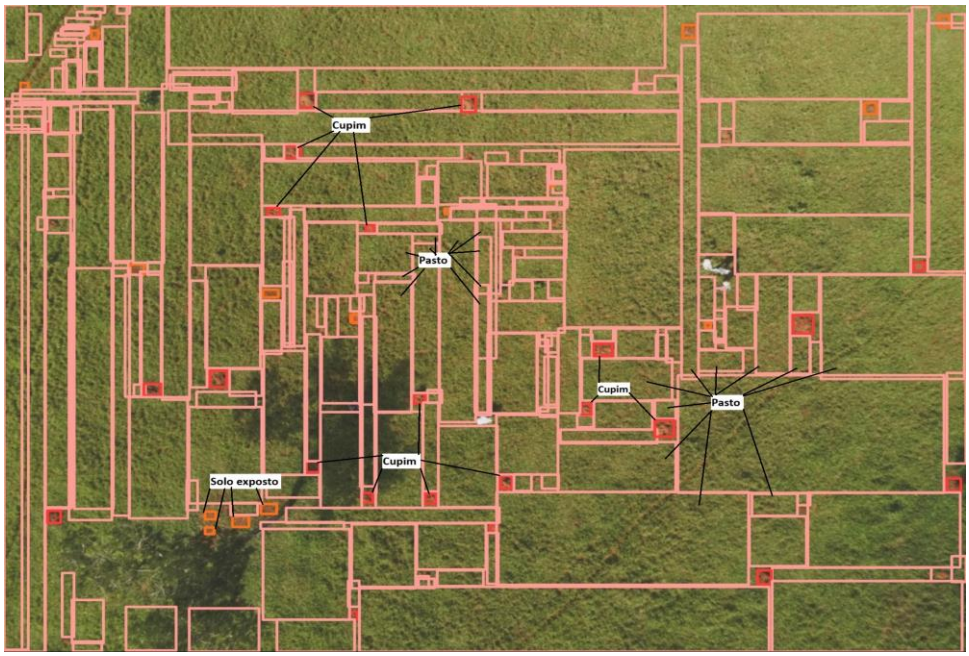
Código Fonte – Script para fine-tuning com os algoritmos disponíveis para o YOLOv8.

```
from ultralytics import YOLO


optimizers = ["SGD", "Adam", "AdamW", "NAdam", "RAdam", "RMSProp", "auto"]


for optimizer in optimizers:
    model = YOLO("/home/ian/yolov8/train-default-v8n/weights/best.pt")


    model.tune(epochs=1000, iterations=150, optimizer=optimizer, plots=False, save=False,
val=False)
```

Para a análise de área de cada classe foi implementado um script em Python que a partir das predições feitas pelo modelo YOLO, fosse calculado a área que cada classe ocupa na imagem, consequentemente, qual a porcentagem de participação da classe na imagem analisada. Algumas questões precisaram ser levadas em conta para essa análise, como a sobreposição de predições que pode ser observado na Figura 31. O método utilizado foi a Intersecção sobre União (IOU), que serve para descrever a extensão de sobreposição entre duas marcações. Além disso, foi seguido as seguintes regras sobre os casos de sobreposição entre duas marcações:

1. Se uma marcação está sobrepondo outra da mesma classe, será descontado da maior área, o valor respectivo a área de interseção entre as duas marcações; e
2. Se uma marcação está sobrepondo outra de classes diferentes, será descontado da maior área o valor respectivo a área de interseção entre as duas marcações. Já que em muitos casos nessa aplicação, elementos menores acabam sendo marcados dentro de elementos maiores.

Figura 31 - Imagem avaliada pelo modelo YOLOv8n Otimizado (Adam).



Fonte: Autoria própria.

Dessa forma, tendo o valor de área em pixels da imagem original e os valores de área em pixels respectivos de cada classe detectada, é possível calcular a porcentagem de ocupação de cada classe na imagem. Os resultados obtidos para a Figura 31, podem ser observados no Quadro 7. Essa análise permite que junto a um especialista seja definido níveis de degradação, que podem ser mais adequadas e adaptadas à realidade de cada situação.

Quadro 7– Ocupação de cada classe na predição da Figura 31.

Otimizador	**Ocupação**
Pasto	90,66%
Solo exposto	0,59%
Cupim	0,24%
Ignorado	8,51%

Fonte: Autoria própria.

Código Fonte – Script para calcular a IOU e porcentagem de ocupação de cada classe na imagem.

```
from ultralytics import YOLO
import os


def intersection_over_union(bbox1, bbox2):
  # Calcula as coordenadas da intersecção do retângulo
  xmin_intersection = max(bbox1[0], bbox2[0])
  ymin_intersection = max(bbox1[1], bbox2[1])
  xmax_intersection = min(bbox1[2], bbox2[2])
  ymax_intersection = min(bbox1[3], bbox2[3])


  # Calcula a área da intersecção
  area_intersection = max(0, xmax_intersection - xmin_intersection) * max(0,
ymax_intersection - ymin_intersection)


  # Calcula a área de união
  area_bbox1 = (bbox1[2] - bbox1[0]) * (bbox1[3] - bbox1[1])
  area_bbox2 = (bbox2[2] - bbox2[0]) * (bbox2[3] - bbox2[1])
  area_union = area_bbox1 + area_bbox2 - area_intersection


  # Calcula a IOU
  return area_intersection / area_union if area_union > 0 else 0


base_path = "/home/ian/yolov8/data/images/val"
# base_path = "/home/ian/yolov8/Todas as Seleções"
validation_images = os.listdir(base_path)
detection_folder = "/home/ian/yolov8/detection"
classes = [
    "Cupim",
    "Pasto",
    "Solo exposto",
]


# Carrega o modelo YOLOv8
model = YOLO('/home/ian/yolov8/runs/detect/tune2/weights/best.pt')
```

```
for img in validation_images:
    [results] = model.predict(task="detect", source=base_path+"/"+img, show_labels=False, show_conf=False, save=True, device="0", augment=False)

    # Dictionary to store class area sums
    class_areas = {}

    # Total image area
    image_height,image_width  = results.orig_shape
    print(image_width, image_height)
    total_image_area = image_width * image_height

    # Process detections
    for i, detection_i in enumerate(results.boxes):
        class_id_i = int(detection_i.cls)
        xmin_i, ymin_i, xmax_i, ymax_i = detection_i.xyxy[0]
        area_i = (xmax_i - xmin_i) * (ymax_i - ymin_i)

        for j in range(i + 1, len(results.boxes)):
            detection_j = results.boxes[j]
            class_id_j = int(detection_i.cls)
            xmin_j, ymin_j, xmax_j, ymax_j = detection_j.xyxy[0]

            # Calculate IOU
            iou = intersection_over_union([xmin_i, ymin_i, xmax_i, ymax_i], [xmin_j, ymin_j, xmax_j, ymax_j])

            if class_id_i == class_id_j:
                area_i -= iou * ((xmax_i - xmin_i) * (ymax_i - ymin_i))
            else:
                # Subtract overlap from larger box
                if (xmax_i - xmin_i) * (ymax_i - ymin_i) > (xmax_j - xmin_j) * (ymax_j - ymin_j):
                    area_i -= iou * ((xmax_i - xmin_i) * (ymax_i - ymin_i))
                else:
                    area_j = (xmax_j - xmin_j) * (ymax_j - ymin_j)  # Define area_j within the loop for class_j
                    area_j -= iou * area_j
```

```
        # Update class area sum
        class_areas[class_id_i] = class_areas.get(class_id_i, 0) + area_i

    # Calculate class representation percentages
    class_percentages = {class_id: (area / total_image_area) * 100 for class_id,
area in class_areas.items()}

    # Print class representation percentages
    for class_id, percentage in class_percentages.items():
        print(f"Class {classes[class_id]} representation: {percentage:.2f}%")
```

5.1 DISCUSSÃO

Este estudo demonstra as dificuldades na detecção precisa e confiável de áreas de pasto, áreas de solo exposto e cupinzeiros em pastagens brasileiras utilizando modelos YOLO de inteligência artificial para detecção de objetos. Os resultados obtidos com YOLOv4 e os modelos YOLOv8, podem ser observados no Quadro 8, no qual apresentaram um desempenho insatisfatório, já que um mAP50 de aproximadamente 50% é como se fosse um chute de certo ou errado. Se for considerado as classificações mais difíceis, mensuradas pelo mAP50-95, nenhum dos modelos alcança nem 50% de precisão.

Quadro 8 – Comparativo entre os modelos analisados no conjunto de validação.

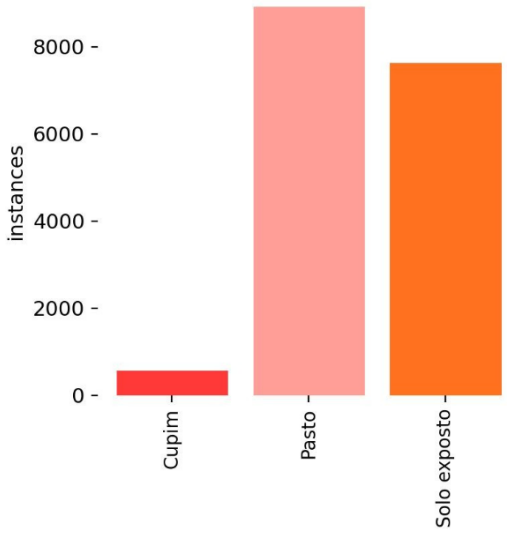
Otimizador	**mAP50**	**mAP50-95**
YOLOv4 Darknet	29,3%	-
YOLOv8n	49,4%	30,6%
YOLOv8n Adam	53,9%	34,8%
YOLOv8x	56,7%	47,2%

Fonte: Autoria própria.

Ao analisar a situação do trabalho, as métricas de precisão ruins podem ser justificadas por alguns fatores limitantes:

- **Inconsistência na qualidade do dataset:** as imagens que constituem a base de dados fornecida pelo LAPIG não seguiram nenhum padrão na captura, como, altura do drone, ângulo da câmera, posicionamento do drone, horário do dia, entre outros fatores que podem ser padronizados. Por mais que para o treinamento foi selecionado imagens com ângulo de visão parecidas, ainda assim, não seguiam um padrão.
- **Base de dados limitada:** a base de dados rotulada é pequena, apesar de aplicado o processo de aumento de dados, seria ideal que fosse utilizado uma base rotulada com um número muito maior de exemplares. Essa limitação se deu principalmente pelo consumo de tempo envolvida na rotulação das imagens, que demandam de 1h a 4h cada.
- **Desbalanceamento de classes:** visando fazer a identificação de todos os elementos desejados na imagem, um dos produtos dessa necessidade pode ser observado na Figura 32, que foi o desbalanceamento de exemplares para cada classe. Essa situação pode acarretar em uma dificuldade do modelo em identificar classes menos representadas.

Figura 32 - Imagem avaliada pelo modelo YOLOv8n Otimizado (Adam).

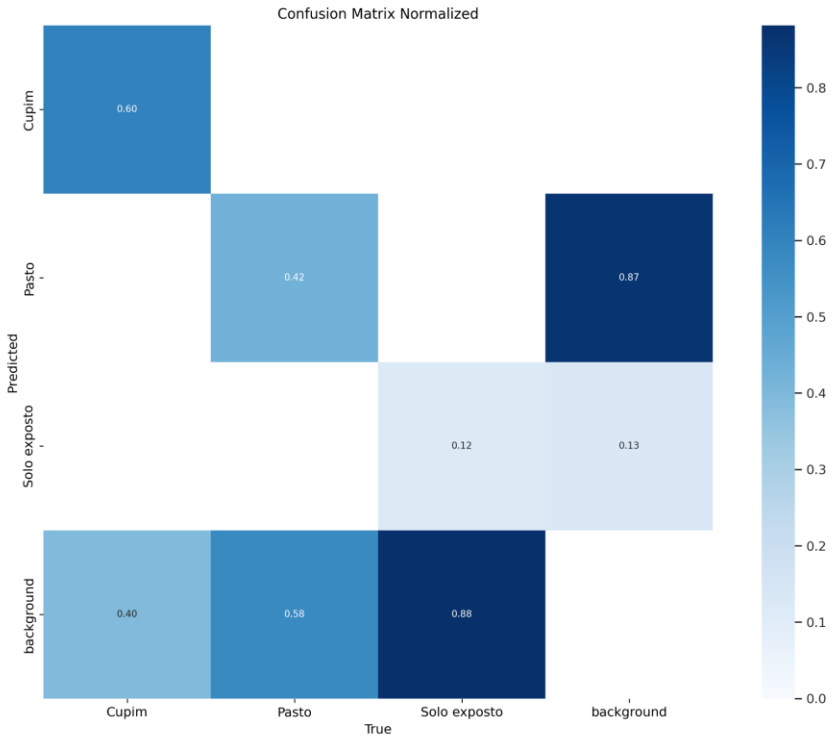


Fonte: Autoria própria.

- **Marcação imperfeita dos rótulos:** apesar do acompanhamento e disponibilidade do engenheiro agrônomo durante o processo de rotulação das imagens da base de dados, muito trabalho precisou ser feito em sua ausência, por causa da grande demanda de tempo para a rotulação. Possivelmente marcações mais precisas e adequadas devem resultar em predições mais assertivas. Como pode ser observado na Figura 33, a matriz de confusão indica que a classe "Pasto" é a classe mais bem detectada pelo modelo, com

precisão de 87% e recall de 40%. Isso significa que o modelo identifica corretamente a maioria dos exemplos de pasto, mas também comete alguns erros, classificando incorretamente alguns exemplos de solo exposto ou cupim como pasto. A classe "Solo Exposto" tem um desempenho menos satisfatório que o "Pasto", com precisão de 42% e recall de 58%. Isso significa que o modelo comete mais erros ao detectar solo exposto. A classe "Cupim" é a classe com o pior desempenho, com precisão de 58% e recall de 12%. Isso significa que o modelo comete muitos erros ao detectar cupim, classificando incorretamente a maioria dos exemplos como pasto ou solo exposto.

Figura 33 – Matriz de confusão normalizada do modelo YOLOv8n Otimizado (Adam).



Fonte: Autoria própria

6 CONCLUSÕES

A integração de soluções baseadas em TIC na agricultura possibilita o alcance da segurança de produção, redução dos impactos ambientais e a sustentabilidade agrícola. Neste sentido, este trabalho busca colaborar com os Objetivos de Desenvolvimento Sustentáveis 2 e 11 das Nações Unidas[1]. Para esta colaboração considera a identificação de áreas de pasto, solo exposto e cupinzeiros em pastagens a partir de aprendizado de máquina e visão computacional. Para isto, estabeleceu-se alguns objetivos específicos, (OE1) criar uma base de dados com imagens rotuladas de áreas de pasto, solo exposto e cupinzeiros, (OE2) Selecionar os algoritmos de aprendizado de máquina, (OE3) analisar o desempenho dos algoritmos, (OE4) criar o modelo de classificação, (OE5) propor uma análise de área degradada X área produtiva.

Considerando a execução do OE1, para a criação da base de dados, foi utilizado a base de dados fornecida pelo LAPIG, que continha aproximadamente 2 mil imagens retiradas por drones durante visitas de campo. A base adequada para treinamento foi criada com a utilização da ferramenta Label Studio, onde apenas uma porção das imagens fornecidas pelo LAPIG foi selecionado e rotulado na plataforma. Porém, as imagens não possuem nenhum padrão de captura, seja em termos de posição do drone, altura do drone, ângulo da câmera, entre outros parâmetros que são essenciais para garantir a qualidade da base de dados e bons resultados. Além disso, não foi possível utilizar toda a base de dados fornecida para o treinamento devido à grande demanda de tempo para a rotulação das imagens, foi, outro ponto que comprometeu o resultado do treinamento do modelo, já que uma quantidade reduzida de exemplares compromete a generalização de características.

A seleção dos algoritmos de aprendizado de máquina foi feita após uma revisão da literatura, onde foi constatado que a maioria das aplicações de algoritmos de aprendizado de máquina relacionado a análise de degradação em pastagens, é feita através da análise de imagens de satélites utilizando índices como o NDVI e o TGI. Portanto, a seleção dos algoritmos YOLOv4 e YOLOv8 foi feita pelo amplo uso desse modelo na área de visão computacional, atendendo assim o OE2.

A criação do modelo de classificação foi feita através do treinamento de um modelo base YOLO e generalizando-o para o propósito desse trabalho. Dada a análise do desempenho,

[1] https://brasil.un.org/pt-br/sdgs

observa-se que os resultados foram insatisfatórios em termos de precisão e confiabilidade da predição, e, a principal causa foi a baixa qualidade da base de dados e quantidade reduzida de exemplares para o treinamento. Porém, apesar de resultados pouco satisfatórios, a abordagem é promissora para ser feita a detecção dos elementos selecionados e para a análise de degradação através de área produtiva X área degradada. Possivelmente, caso ajustes sejam feitos em questão de qualidade da captura das imagens, qualidade das rotulações e quantidade de exemplares, resultados melhores devem ser obtidos.

Diante da análise dos objetivos específicos e do objetivo geral deste trabalho, é possível concluir que houve avanços significativos na proposta de identificar áreas de solo exposto e cupinzeiros em pastagens por meio do aprendizado de máquina. A criação da base de dados e a seleção dos algoritmos demonstram um esforço em direção à modernização do manejo das pastagens no Brasil. No entanto, os resultados obtidos, embora pouco satisfatórios em termos de precisão e recall, apontam para a necessidade de aprimoramentos futuros.

Para melhorar o desempenho do trabalho, recomenda-se a padronização na captura das imagens, a ampliação da base de dados rotulada e a otimização dos processos de treinamento dos modelos de aprendizado de máquina. Essas medidas podem contribuir significativamente para a melhoria da precisão e eficácia do modelo de classificação proposto, possibilitando uma identificação mais precisa de áreas de pasto, solo exposto e cupinzeiros em pastagens.

Por fim, conclui-se que os objetivos propostos para esse trabalho foram atendidos, já que uma base de dados foi criada e adequada para treinamento, os modelos de aprendizado de máquina YOLOv4 e YOLOv8 foram selecionados e avaliados diante do propósito do trabalho, de tal forma que como produto foi criado 4 modelos de classificação (YOLOv4, YOLOv8n, YOLOv8 Adam e YOLOv8x), a análise de detecções também foi feita para relacionar área de pasto (produtiva) com área degradada.

6.1 TRABALHOS FUTUROS

Visando melhorar o desempenho, um dos problemas que deve ser resolvido, é a quantidade limitada da base de dados. Para isso, recortar as imagens em tamanhos menores pode ser uma alternativa complementar ao processo de aumento de dados. Além disso, para

evitar o desbalanceamento de classes, treinar um modelo para cada classe pode ser uma opção.

Como apresentado anteriormente sobre inteligência artificial embarcada, seria interessante integrar o modelo de detecção (YOLOv4, YOLOv8 ou outros) em um drone equipado com uma câmera, como os utilizados para a captura das imagens da base de dados. Isso permitiria a coleta de imagens e análise em tempo real durante sobrevoos das áreas de pastagem. A validação do modelo em campo, comparando as detecções feitas pelo drone com as áreas reais de pasto, solo exposto e cupinzeiros, seria fundamental para avaliar sua eficácia.

Além da metodologia de detecção de objetos, pode ser considerada a exploração de métodos de segmentação de imagens. A segmentação permite identificar regiões específicas dentro de uma imagem, como áreas de pasto, áreas de solo exposto ou cupinzeiros. Algoritmos como U-Net, Mask R-CNN, DeepLab, ou até mesmo o próprio YOLOv8 que disponibiliza modelos específicos para tarefas de segmentação, podem ser úteis para essa tarefa.

Como dito anteriormente, possuir um padrão de captura de imagens é um fator que contribui para melhorar o a qualidade dos dados. Isso inclui especificações técnicas para câmeras (resolução, ângulo de visão, iluminação) e procedimentos padronizados para coleta (altura do voo, velocidade do drone, sobreposição de imagens). Portanto, propor um protocolo bem definido ajudará a criar uma base de dados mais consistente e confiável. Em conjunto com a proposta de criar um protocolo para a captura das imagens, é ideal que seja proposto também diretrizes claras contando com o apoio de especialistas para a classificação e rotulação das imagens. Isso envolve definir critérios para distinguir diferentes classes (solo exposto, cupinzeiros, pasto) e garantir que os rótulos sejam mais consistentes.

REFERÊNCIAS BIBLIOGRÁFICAS

AHMAD, L.; NABI, F. **Agriculture 5.0: Artificial Intelligence, IoT and Machine Learning**. [S.l.]: CRC Press, 2021.

AYDIN, Burchan; SINGHA, Subroto. YOLOv5. 2020.

BERNARDI, A. C. de C.; NAIME, J. de M.; RESENDE, A. V.; BASSOI, L. H.; INAMASU, R. Y. Agricultura de precisão: resultados de um novo olhar. EMBRAPA, 2014. Disponível em: ⟨ https://www.embrapa.br/busca-de-publicacoes/-/publicacao/1002959/agricultura-de-precisao-resultados-de-um-novo-olhar ⟩.

BISHOP, C. M. **Pattern Recognition and Machine Learning (Information Science and Statistics)**. [S.l.]: Springer, 2006.

BOCHKOVSKIY, A.; WANG, C-Y.; LIAO, H-Y. M. **YOLOv4: Optimal Speed and Accuracy of Object Detection**. arXiv, 2020. DOI: /10.48550/arXiv.2004.10934

BOW, S.-T. **Pattern Recognition and Image Preprocessing (Signal Processing and Communications)**. CRC Press, 2002. Disponível em: ⟨https://books.google.com.br/books?id=5gEqB7C4yAMC&lpg=PP11&ots=q2pAeTUAJ-&dq=pattern%20recognition%20in%20images&lr&hl=pt-BR&pg=PR9#v=onepage&q=pattern%20recognition%20in%20images&f=false⟩.

Cai, Z. & Peng, C. (2021). A study on training fine-tuning of convolutional neural networks [Proceedings of the 2021 13th International Conference on Knowledge and Smart Technology (KST), pp. 84-89]. DOI: /10.1109/KST51265.2021.9415793

CARVALHO, Wellyngton Tadeu Vilela; MINIGHIN, Duarte Carvalho; GONÇALVES, Lúcio Carlos; VILLANOVA, Daiana Francisca Quirino; MAURICIO, Rogério Martins; PEREIRA, Renata Vitarele Gimenes. Pastagens degradadas e técnicas de recuperação: revisão. **Pubvet**, [S.L.], v. 11, n. 10, p. 1036-1045, out. 2017. Editora MV Valero. DOI: /10.22256/pubvet.v11n10.1036-1045.

DENG, Bin Liang; LI, Guoqiang; HAN, Song; SHI, Lei; XIE, Yanzhi. Model Compression and Hardware Acceleration for Neural Networks: A Comprehensive Survey. Proceedings of the IEEE. v. 108, p. 485-532, 2020.

DIAS-FILHO, M. B. **Degradação de pastagens, o que é e como evitar**. EMBRAPA, 2017. Acessado em 7 de outubro de 2023. Disponível em: ⟨https://www.infoteca.cnptia.embrapa.br/Infoteca/bitstream/doc/1070416/1/TC1117CartilhaPastagemV04.pdf⟩.

DIAS-FILHO, M. **Vamos falar sobre pastagens: fatos, dicas e recomendações**. 2022. Acessado em 04 de agosto de 2023. Disponível em: https://www.researchgate.net/publication/358189917_Vamos_falar_sobre_pastagens_fatos_dicas_e_recomendacoes

DIAS-FILHO, M. **Degradação de pastagens: conceitos, processos e estratégias de recuperação e de prevenção**. 2023. Acessado em 04 agosto de 2023. Disponível em: https://www.researchgate.net/publication/371958772_Degradacao_de_pastagens_conceitos_processos_e_estrategias_de_recuperacao_e_de_prevencao.

DICK, Richard P.; SHANG, Li; WOLF, Michael; YANG, Sheng-Wen. Embedded Intelligence in the Internet-of-Things. IEEE Design & Test. v. 37, p. 7-27, 2019.

FACELI, K.; LORENA, A. C.; GAMA, J.; ALMEIDA, T. A.; CARVALHO, A. C. P. **Inteligência Artificial** - Uma Abordagem de Aprendizado de Máquina. [S.l.]: LTC, 2021.

FEIGL, B. J., MELILLO, J.; CERRI, C.C. Changes in the origin and quality of soil organic matter after pasture introduction in Rondônia (Brazil). **Plant Soil** 175, 21–29 (1995). https://doi.org/10.1007/BF02413007

FLORES, R. S.; EUCLIDES, V. P. B.; ABRÃO, M. P. C.; GALBEIRO, S.; DIFANTE, G. dos S.; BARBOSA, R. A. Desempenho animal, produção de forragem e características estruturais dos capins marandu e xaraés submetidos a intensidades de pastejo. **Forragicultura R. Bras. Zootec**. 37 (8) Ago 2008. https://doi.org/10.1590/S1516-35982008000800004

FRANCO, V.R., HOTT, M.C., ANDRADE, R.G. et al. Hybrid machine learning methods combined with computer vision approaches to estimate biophysical parameters of pastures. **Evol. Intel**. 16, 1271–1284 (2023). https://doi.org/10.1007/s12065-022-00736-9

GÉRON, Aurélien. **Mãos à Obra: Aprendizado de máquina com scikit-learn, keras & tensorflow**. 2. ed. Rio de Janeiro: Alta Books Editora, 2021. 640 p.

GOLUB, T. R., SLONIM, D. K., TAMAYO, P., HUARD, C., GAASENBEEK, M., MESIROV, J. P., COLLER, H., LOH, M. L., DOWNING, J. R., CALIGIURI, M. A., BLOOMFIELD, C. D., & LANDER, E. S. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. **Science** (New York, N.Y.), 286(5439), 531–537. https://doi.org/10.1126/science.286.5439.531

GUIMARAES, A. C. D. O.; INOUE, M. H. O.; IKEDA, F. S. O. **Estratégias de manejo de plantas daninhas para novas fronteiras agrícolas**. Curitiba: SBCPD, 2018.

GUO, Bo; ZHANG, Dong; YU, Zhiping; LIANG, Yuxing; WANG, Zhiping; ZHOU, Xiaopeng. From the internet of things to embedded intelligence. World Wide Web. v. 16, p. 399-420, 2012.

HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence. v. 37, p. 1-14, jun. 2014.

HOSANG, Jan Hendrik; BENENSON, Rodrigo; SCHIELE, Bernt. Learning non-maximum suppression. CoRR, 2017. DOI: abs/1705.02950.

HOWARD, Andrew G.; ZHU, Menglong; CHEN, Bo; KALENICHENKO, Dmitry; WANG, Weijia; WEYAND, Tobias; ANDREETTO, Marco; ADAM, Hartmut. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017. DOI: 10.48550/arXiv.1704.04861

HUANG, S.; CAI, N.; PACHECO, P. P.; NARRANDES, S.; WANG, Y.; XU, W. Applications of support vector machine (svm) learning in cancer genomics. **Cancer Genomics & Proteomics**, Inter-national Institute of Anticancer Research, v. 15, n. 1, p. 41–51, 2018. ISSN 1109-6535. Disponível em: <https://cgp.iiarjournals.org/content/15/1/41>.

JAPKOWICZ, Nathalie; SHAH, Mohak. **Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation**. Springer, 2011

VALLE JÚNIOR, R. F.; SIQUEIRA, H. E.; VALERA, C. A.; OLIVEIRA, C. F.; FERNANDES, L. F. S.; MOURA, J. P.; PACHECO, F. A. L. Diagnosis of degraded pastures using na improved ndvi-based remote sensing approach: An application to the environmental protection area of uberaba river basin (Minas Gerais, Brazil). 2019. **Remote Sensing Applications: Society and Environment** Volume 14, April 2019, Pages 20-33 https://doi.org/10.1016/j.rsase.2019.02.001.

KLUTHCOUSKI, J.; CORDEIRO, L. A. M.; OLIVEIRA, G. C. P. de. **Braquiária na agropecuária brasileira: uma história de sucesso.** EMBRAPA, 2013. Acessado em 10 de setembro de 2023. Disponível em: https://www.embrapa.br/busca-de-publicacoes/-/publicacao/982611/braquiaria-na-agropecuaria-brasileira-uma-historia-de-sucesso⟩.

LECUN, Y.; BOSER, B.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W.; JACKEL, L. D. Backpropagation applied to handwritten zip code recognition. Backpropagation Applied to Handwritten Zip Code Recognition. **Neural Comput** 1989; 1 (4): 541–551. doi: https://doi.org/10.1162/neco.1989.1.4.541

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278–2324, 1998. DOI: 1109/5.726791.

LEVINE, S., FINN, J., PENG, X., ABBEEL, P., & LEVINE, Y. Learning to walk with deep reinforcement learning. arXiv preprint arXiv:1412.6980. 2016. DOI: https://arxiv.org/abs/1412.6980

LOSHCHILOV I., HUTTER F. DECOUPLED WEIGHT DECAY REGULARIZATION. arXiv preprint arXiv:1711.05101v3 2019. DOI: https://arxiv.org/abs/1711.05101v3

MAPBIOMAS. Pastagens brasileiras ocupam Área equivalente a todo o estado do amazonas. 2021. Acessado em 04 de agosto de 2023. Disponível em: https://brasil.mapbiomas.org/2021/10/13/ pastagens-brasileiras-ocupam-area-equivalente-a-todo-o-estado-do-amazonas/.

MASSRUHÁ, S. M. F. S.; LEITE, M. A. d. A.; OLIVEIRA, S. R. d. M.; MEIRA,

C. A. A.; JUNIOR, A. L.; BOLFE, E. L. **Agricultura digital: pesquisa, desenvolvimento e inovação nas cadeias produtivas**. EMBRAPA, 2020. Acessado em 08 de agosto de 2023. Disponível em: https://www.embrapa.br/busca-de-publicacoes/-/publicacao/1126213/ agricultura-digital-pesquisa-desenvolvimento-e-inovacao-nas-cadeias-produtivas.

MITCHELL, T. M. Machine Learning. [S.l.]: McGraw-Hill Education, 1997.
MYFARM. **Agricultura 5.0**. 2021. Acessado em 10de setembro de 2023. Disponível em: https://www.myfarm.com.br/
agricultura-5-0/.

MÜLLER, Andreas C.; GUIDO, Sarah. **Introduction to Machine Learning with Python**. O'Reilly Media, 2016.

NORVIG, P.; RUSSELL, S. **Artificial Intelligence**: A Modern Approach, Global Edition. [S.l.]: Pearson Education Limited, 2021.

OH, Sang-Hyon; PARK, Hee-Mun; PARK, Jin-Hyun. Estimating vegetation index for outdoor free-range pig production using YOLO. Journal of Animal Science and Technology. v. 65, n. 3, p. 638-651, maio 2023. DOI: 10.5187/jast.2023.e41.

OLSEN, A., KONOVALOV, D.A., PHILIPPA, B. et al. **DeepWeeds**: A Multiclass Weed Species Image Dataset for Deep Learning. Sci Rep 9, 2058 (2019). https://doi.org/10.1038/s41598-018-38343-3

PONISZEWSKA-MARANDA, Agnieszka; KACZMAREK, Dariusz; KRYVINSKA, Nataliia; XHAFA, Fatmir. Studying usability of AI in the IoT systems/paradigm through embedding NN techniques into mobile smart service system. Computing. v. 101, p. 1661-1685, 2018.

REDMON, J.; DIVVALA, S.; GIRSHICK, R.; FARHADI, A. **You only look once: Unified, real-time object detection**. 2015. Acessado em 20 de agosto de 2023. Disponível em: https://www.cv-foundation.org/openaccess/content cvpr 2016/papers/Redmon You Only Look CVPR 2016 paper.pdf⟩.

RUDER S. An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747v2 2017. DOI: https://arxiv.org/abs/1609.04747v2

SAIZ-RUBIO, V.; ROVIRA-MÁS, F. From smart farming towards agriculture 5.0: A review on crop data management. 2020. **Agronomy** 2020, 10(2), 207; https://doi.org/10.3390/agronomy10020207

SEMENTES, J. **Agricultura digital: Revolução no agro!** 2022. Acessado em 03 de agosto de 2023. Disponível em: https: //jhsementes.com/blog/agricultura-digital-revolucao-digital-no-agro.

SERRÃO, E. A. S.; FALESI, I. C.; VEIGA, J. B. da. **Produtividade de pastagens cultivadas em solos de baixa fertilidade das Áreas de floresta da amazônia brasileira**.

1982. Acessado em 08 de agosto de 2023.Disponível em: https://ainfo.cnptia.embrapa.br/digital/bitstream/doc/980424/1/ Produtividade-de-pastagens-cultivadas-em-solos.pdf.

SILVA, L. A. P. da; SENA-SOUZA, J. P.; SOUZA, C. P. de; SILVA, C. R.; BOLFE, E. L.; CHAGAS-REIS, C. C.; LEITE, M. E. Drivers of degradation of pastures in the cerrado north of minas gerais - br. 2023. **RA'EGA**, Curitiba, PR, V.57, p. 66 – 80, 08/2023 https://revistas.ufpr.br/raega http://dx.doi.org/10.5380/raega.v57i0.92339

SOLAWETZ, Jacob; . What is YOLOv8? The Ultimate Guide. [2024]. Blog Roboflow. 11 jan. 2023. Disponível em: < https://blog.roboflow.com/whats-new-in-yolov8/> Acesso em: 03 mai. 2024.

SUTSKEVER, I., MARTENS, J., DAHL, G., & HINTON, G. On the importance of initialization and momentum in deep learning. In *ICML*. 2013.

TOMBE, R. Computer vision for smart farming and sustainable agriculture. In: **IEEE. 2020 IST-Africa Conference** (IST-Africa). [S.l.], 2020. p. 1–8.

TREVEN, Juan R.; CORDOVA-ESPARAZA, Diana M. A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond. 2023. Disponível em: https://arxiv.org/pdf/2304.00501.pdf.

ULTRALYTICS LLC. Ultralytics YOLOv8. Versão 8.0.0. 2023. Disponível em: https://github.com/ultralytics/ultralytics.

ULTRALYTICS LLC. Ultralytics YOLOv5. 2020. Disponível em: https://github.com/ultralytics/ultralytics.

VALÉRIO, J. R. **Cupins-de-montículo em pastagens**. 2006. Acessado em 07 de setembro de 2023. Disponível em: https: //ainfo.cnptia.embrapa.br/digital/bitstream/CNPGC-2009-09/12409/1/DOC160.pdf.
VILLAFUERTE, A.; VALADARES, F. G.; CAMPOLINA, G. F.; SILVA, M. G. P. da. Agricultura 4.0 - estudo de inovação disruptiva no agronegócio brasileiro. **ISTI**, v. 7, n. 1, 2018. Acessado em 12 de agosto de 2023. Disponível em: https://www.api.org.br/conferences/index.php/ISTI2018/ISTI2018/ paper/viewFile/567/276.

WADT, P. G. S.; PEREIRA, J. E. S.; GONC ̧ALVES, R. C.; SOUZA, C. B. da Costa de; ALVES, L. da S. **Práticas de conservação do Solo e Recuperação de Áreas Degradadas. Rio Branco**: EMBRAPA, 2003. Acessado em 23 de agosto de 2023. Disponível em: https://www.embrapa.br/busca-de-publicacoes/-/publicacao/1126213/agricultura-digital-pesquisa-desenvolvimento-e-inovacao-nas-cadeias-produtivas.

WANG, Chien-Yao; LIAO, Hong-Yuan Mark; YEH, I-Hau; WU, Yueh-Hua; CHEN, Ping-Yang; HSIEH, Jun-Wei. CSPNet: A new backbone that can enhance learning capability of CNN. CoRR, 2019. DOI: abs/1911.11929.