UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

PEDRO HENRIQUE RABELO LEÃO DE OLIVEIRA

DETECÇÃO E RASTREAMENTO DE OBJETOS APLICADOS NA ANÁLISE DA COMUNICAÇÃO VISUAL EM LAGARTOS

Ouro Preto
2026

PEDRO HENRIQUE RABELO LEÃO DE OLIVEIRA

DETECÇÃO E RASTREAMENTO DE OBJETOS APLICADOS NA ANÁLISE DA COMUNICAÇÃO VISUAL EM LAGARTOS

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Jadson Castro Gertrudes
Coorientador: Luis Gustavo de Oliveira Miranda

Ouro Preto
2026

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E BIOLOGICAS
DEPARTAMENTO DE COMPUTACAO

FOLHA DE APROVAÇÃO

Pedro Henrique Rabelo Leão de Oliveira

Detecção e Rastreamento de Objetos Aplicados na Análise da Comunicação Visual em Largatos

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Ciência da Computação

Aprovada em 27 de Fevereiro de 2026

Membros da banca

Jadson Castro Gertrudes (Orientador) - Doutor - Universidade Federal de Ouro Preto
Luis Gustavo de Oliveira Miranda (Coorientador) - Bacharel - PPGCC/ DECOM/ UFOP
Hugo Eduardo Ziviani (Examinador) - Mestre - Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Ouro Preto
Arthur Negrão de Faria Martins da Costa (Examinador) - Bacharel - Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Ouro Preto

Jadson Castro Gertrudes, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 27/02/2026

Documento assinado eletronicamente por **Jadson Castro Gertrudes**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 02/03/2026, às 22:01, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **1062468** e o código CRC **91FB6290**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: 3135591692 - www.ufop.br

Dedico este trabalho aos meus pais, Patrícia e Renilson, aos meus irmãos, Luís e Maria, e à minha namorada, Lorraine.

Agradecimentos

Antes de tudo, agradeço à Deus, pela oportunidade de estar concluindo uma etapa tão almejada em minha vida! Agradeço também à equipe espiritual do Templo Renascer e aos meus guardiões pelo acompanhamento e proteção em cada passo da minha caminhada!

Agradeço aos meus pais, Patrícia e Renilson, por todo o amor, apoio e pelas diversas oportunidades que me proporcionaram e continuam proporcionando ao longo da vida; graças a elas, concluo hoje minha graduação, um passo fundamental para o meu futuro. Expresso minha gratidão também por toda a preocupação, pelo suporte constante e por sempre tornarem minha jornada mais leve, permitindo que eu dedicasse o máximo de energia aos meus estudos e à minha evolução profissional. Agradeço por todos os conselhos, pelo acolhimento e por se fazerem presentes, mesmo não estando mais fisicamente próximos. Aos meus irmãos, Luís e Maria, que estão sempre ao meu lado, compartilhando momentos, cuidando e protegendo uns aos outros, meu muito obrigado. Vocês me mostram diariamente o significado da verdadeira amizade, na qual a confiança e a conexão vão além do material; vocês são meus companheiros de vida! Aos quatro: vocês são a minha base e jamais me esquecerei de tudo o que já fizeram por mim.

Agradeço à minha namorada, Lorraine, por todo o amor e por me acompanhar em todos os momentos ao longo destes últimos anos, sejam eles de alegria e risadas, de reflexão com conversas sérias e conselhos genuínos, ou de aprendizado, em que crescemos juntos. Você esteve ao meu lado nas fases mais desafiadoras desse processo e foi essencial para que eu tivesse força para superá-las com um sorriso no rosto. Com você, todo o processo se tornou mais leve e aprendi a enxergar a vida de uma maneira diferente. Obrigado por despertar a minha melhor versão; você foi o maior presente que Ouro Preto poderia me dar! Agradeço também aos meus sogros, Érica e Marcos, pelo carinho com que sempre me receberam de braços abertos, pelos momentos de descontração e pelo acolhimento, que me fizeram sentir em casa e em família, mesmo morando em uma cidade diferente da qual vivi por toda a minha vida.

Aos amigos que fiz durante minha trajetória na UFOP, agradeço por todos os momentos vividos dentro e fora da faculdade, desde os intervalos das aulas até as nossas confraternizações em fins de períodos e idas à rodízios de pizza. Agradeço por todo o conhecimento trocado e por cada aprendizado! Sem dúvida, levarei essas lembranças para toda a vida.

Agradeço ao meu orientador, Jadson, e ao meu coorientador, Luis, pela oportunidade de aprendizado e pelo apoio no desenvolvimento deste trabalho, que marca a conclusão de uma etapa tão importante na minha vida. Estendo meus agradecimentos a todo o corpo docente que contribuiu para a minha formação durante o curso, pois todos foram essenciais para a minha evolução técnica e pessoal nesse período.

Por fim, agradeço à minha madrinha e a todos os meus tios, tias, primos e primas que

sempre torceram e continuam torcendo pelo meu sucesso pessoal e profissional!

Resumo

A análise da comunicação visual em lagartos do gênero *Tropidurus* na área da etologia enfrenta desafios metodológicos, uma vez que a observação manual é um processo demorado, subjetivo e sujeito a inconsistências. Embora abordagens automatizadas anteriores tenham sido propostas, elas carecem de uma validação quantitativa rigorosa de sua etapa de detecção, deixando uma lacuna sobre sua real precisão e robustez. Este trabalho propõe-se a aprimorar a detecção e o rastreamento da movimentação da cabeça desses animais em vídeos, visando possibilitar uma extração de sinais de comunicação mais precisa para estudos comportamentais. Para tal, investiga-se o potencial de arquiteturas modernas de visão computacional, com foco no modelo YOLOv12 para detecção e em sua integração com o rastreador Filtro de Kalman. A metodologia adotada comparou o desempenho da arquitetura YOLOv12 contra o modelo *baseline* YOLOv5 e avaliou o uso dos algoritmos de rastreamento *BoT-SORT* e *ByteTrack*, que usam o Filtro de Kalman, para garantir a continuidade dos dados ao longo do tempo. Os resultados mostraram a superioridade do YOLOv12, que aumentou a revocação da classe "*Head*"de 0,809 para 0,915 e melhorou a precisão do ajuste das caixas delimitadoras, obtendo um mAP50-95 de 0,471 para 0,619. Na etapa de rastreamento, o sistema detectou o animal em 100% dos quadros nos vídeos de teste, garantindo a manutenção da identidade e a estabilidade necessária para calcular a angulação entre a cabeça e o corpo. Conclui-se que o sistema desenvolvido é uma ferramenta robusta, capaz de gerar dados consistentes para estudos comportamentais, superando dificuldades como camuflagem e variações do ambiente.

Palavras-chave: Visão Computacional, Detecção de Objetos, Rastreamento, Aprendizado Profundo, YOLO, Filtro de Kalman, Análise Comportamental Animal.

Abstract

The analysis of visual communication in *Tropidurus* lizards within ethology faces methodological challenges, as manual observation is a time-consuming, subjective, and inconsistency-prone process. Although previous automated approaches have been proposed, they lack rigorous quantitative validation of their detection stage, leaving a gap regarding their actual precision and robustness. This work aims to enhance the detection and tracking of these animals' head movements in videos, aiming to enable more precise communication signal extraction for behavioral studies. To this end, the potential of modern computer vision architectures is investigated, focusing on the YOLOv12 model for detection and its integration with the Kalman Filter tracker. The adopted methodology compared the performance of the YOLOv12 architecture against the YOLOv5 baseline model and evaluated the use of the *BoT-SORT* and *ByteTrack* tracking algorithms, which utilize the Kalman Filter, to ensure data continuity over time. The results showed the superiority of YOLOv12, which increased the "Head" class recall from 0.809 to 0.915 and improved the bounding box adjustment precision, increasing mAP50-95 from 0.471 to 0.619. In the tracking stage, the system detected the animal in 100% of the frames in the test videos, ensuring identity maintenance and the necessary stability to calculate the angulation between the head and body. It is concluded that the developed system is a robust tool capable of generating consistent data for behavioral studies, overcoming difficulties such as camouflage and environmental variations.

Keywords: Computer Vision, Object Detection, Tracking, Deep Learning, YOLO, Kalman Filter, Animal Behavioral Analysis.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

Bi-RNN *Bidirectional Recurrent Neural Network*. 24

CNN *Convolutional Neural Network*. 9, 11, 12, 15, 19, 21, 23, 24

DPM *Deformable Parts Models*. 15

E-ELAN *Extended Efficient Layer Aggregation Networks*. 18

EKF *Extended Kalman Filter*. 23

FPN *Feature Pyramid Network*. 18

FPS *frames* por segundo. 16

GELAN *Generalized Efficient Layer Aggregation Networks*. 18

HOG *Histogram of Oriented Gradients*. 15

IA Inteligência Artificial. viii, 7

IoU Interseção sobre União. 33–35

LSTM *Long Short-Term Memory*. 24

mAP *Mean Average Precision*. 19

mAP50 *Mean Average Precision 50*. 33, 34

mAP50-95 *Mean Average Precision 50-95*. 30, 31, 33, 34, 38, 45

MF *Median Filter*. 24

PAN *Path Aggregation Network*. 18

PGI *Programmable Gradient Information*. 18

R-CNN *Region-based Convolutional Neural Network*. 15, 16

ReLU *Rectified Linear Unit*. 13

RNAs Redes Neurais Artificiais. 7–9, 11

SGF *Savitzky-Golay Filter*. 24

SIFT *Scale-Invariant Feature Transform*. 15

SPP *Spatial Pyramid Pooling*. 18

SPPF *Spatial Pyramid Pooling Fast*. 18

SURF *Speeded Up Robust Features*. 15

TBD *Tracking-by-Detection*. 21

UKF *Unscented Kalman Filter*. 23

YOLO *You Only Look Once*. viii, 1–5, 8, 9, 15–20, 22–24, 26–32, 35, 37–39, 41, 43, 45, 46

Sumário

1 Introdução

A comunicação animal é um comportamento determinante para a transferências de informações que garantem desde a sobrevivência até a manutenção das relações sociais no reino animal. Ela tem papel essencial na defesa de territórios, na condição reprodutiva dos indivíduos, no alerta de predadores e no estabelecimento de hierarquias sociais (KAPLAN, 2014). Essa troca de informações ocorre por meio de sinais que podem ser transmitidos por diversos canais, como o olfativo, o tátil, o sonoro e o visual. Dentre eles, a comunicação visual se destaca pela sua riqueza de possibilidades e complexidade (PASSOS, 2016).

A comunicação visual dos animais acontece tanto por meio de características físicas do indivíduo, como suas cores e formas do corpo, quanto por meio de comportamentos dinâmicos, envolvendo mudanças nas posturas e alterações corporais específicas (PALERMO-NETO; ALVES, 2010). Os lagartos do gênero *Tropidurus* são um exemplo de seres que se comunicam por meio desses comportamentos dinâmicos, realizando movimentos de cabeça e flexões corporais, conhecidos como *head-bobs* e *push-ups* (MARTINS, 1994). Esses sinais são emitidos ao levantar e abaixar a cabeça e o tronco em uma série de movimentos e carregam informações sobre a identidade da espécie, o sexo e o contexto social do indivíduo. Passos (2016) encontrou em seu trabalho padrões de comunicação em três espécies de lagartos do gênero *Tropidurus*, entretanto, componentes dessas exibições não foram explorados.

O estudo aprofundado desses sinais emitidos pelos lagartos, contudo, representa um desafio metodológico considerável. A análise tradicional, baseada na observação direta e anotação manual por especialistas, é uma tarefa que exige esforço e um grande dispêndio de tempo, sujeita à subjetividade do observador e logisticamente desafiadora (ZENÓBIO et al., 2023). Visando superar tais barreiras, a Visão Computacional é uma área que pode oferecer um conjunto de ferramentas para automatizar e escalar a análise comportamental.

Em Zenóbio et al. (2023) foi desenvolvido um sistema no qual permite a extração de padrões presentes nas comunicações dos lagartos e, na etapa inicial, é aplicado um algoritmo de aprendizado profundo, mais especificamente o modelo *You Only Look Once* (YOLO)v4, para a detecção da cabeça e do corpo do indivíduo, possibilitando a extração automatizada dessas exibições em forma de sinais. Contudo, o trabalho focou na avaliação do agrupamento dos sinais emitidos, sem apresentar uma avaliação quantitativa específica para a etapa de detecção, deixando em aberto a sua real precisão e robustez. O campo da inteligência artificial, por sua vez, avança rapidamente. Diante disso, surge a questão central que esta monografia busca responder: Qual é o desempenho da abordagem de detecção original quando avaliada e como modelos mais modernos, incluindo a integração com métodos de rastreamento, se comparam a ela em termos de precisão e robustez para os estudos de comunicação dos lagartos?

Esta monografia propõe responder a essa pergunta por meio da implementação e da avaliação rigorosa de arquiteturas computacionais modernas, comparando-as diretamente com o método de referência a fim de buscar uma precisão maior, até em casos de movimentos sutis ou ambientes ruidosos existentes na natureza, para a análise da comunicação visual nesses animais. Embora o trabalho de Zenóbio et al. (2023) descreva o uso do modelo YOLOv4 na etapa inicial de detecção, a análise do software Reptilerecon, registrado no INPI, revelou que a arquitetura encontrada na implementação efetiva é o YOLOv5. Portanto, para fins de precisão técnica e fidelidade ao software desenvolvido, este trabalho considerará a versão 5 do YOLO como o *baseline* da pesquisa.

1.1 Justificativa

Conforme a literatura aponta, a comunicação animal é um mecanismo essencial para a manutenção da vida e da organização social em inúmeras espécies, mediando comportamentos como a proteção de territórios, a reprodução e o alerta sobre ameaças (PASSOS, 2016; PALERMO-NETO; ALVES, 2010). Entretanto, a abordagem tradicional baseada na observação direta por especialistas é uma tarefa que demanda esforço e uma grande quantidade de tempo. Ademais, a análise também está sujeita à subjetividade do observador, o que pode causar inconsistências e erros, principalmente quando se trata de movimentos rápidos e complexos. Com isso, a robustez da análise comportamental do animal é comprometida, além de que padrões mais sutis emitidos pelos animais, talvez até desconhecidos, podem passar despercebidos (PASSOS, 2016; ZENÓBIO et al., 2023).

Diante desse cenário, este trabalho busca contribuir para a superação dessas barreiras através do desenvolvimento, estudo e aprimoramento de técnicas de visão computacional voltadas à análise comportamental. Embora a abordagem proposta em Zenóbio et al. (2023) represente um avanço significativo ao automatizar parte do processo, a ausência de uma validação quantitativa da sua etapa de detecção impede a avaliação de sua eficácia em cenários desafiadores e limita sua reprodutibilidade e aprimoramento. A presente monografia se justifica, portanto, na proposta e na validação de uma solução automatizada que não só otimiza o processo de coleta de dados, mas também oferece maior precisão e objetividade na detecção dos lagartos, estabelecendo uma ponte entre a Ciência da Computação e a Biologia. Ao aprimorar o método de detecção proposto em Zenóbio et al. (2023), o trabalho pretende permitir análises mais eficazes e com um nível de detalhe maior dos sinais emitidos pelos indivíduos, potencializando novas descobertas sobre a comunicação dos *Tropidurus*. A motivação para este projeto consiste na oportunidade de aplicar os conhecimentos adquiridos em inteligência artificial para contribuir diretamente para o avanço da pesquisa biológica na análise comportamental de animais, permitindo a identificação de possíveis novos padrões nas comunicações e análises mais acuradas e reforçando o papel da tecnologia como uma aliada fundamental para a ciência.

1.2 Objetivos

O objetivo principal deste trabalho é aprimorar a etapa de detecção e rastreamento da movimentação da cabeça de lagartos do gênero *Tropidurus* em vídeos realizada no trabalho de Zenóbio et al. (2023), que permite a posterior extração dessas exibições em forma de sinais para que depois sejam encontrados e estudados padrões entre esses sinais emitidos. Assim, será realizada a implementação e validação de modelos mais recentes de visão computacional, além de uma análise comparativa de desempenho com a abordagem proposta por Zenóbio et al. (2023) para essa detecção.

Para atingir isso, os seguintes objetivos específicos foram definidos:

- Realizar uma revisão bibliográfica sobre a utilização de versões recentes do modelo YOLO para detecção de animais e sobre a integração do YOLO com o filtro de Kalman;
- Construir uma base de dados robusta para realização dos experimentos a partir do conjunto de vídeos fornecidos por Passos (2016) e utilizado por Zenóbio et al. (2023);
- Implementar e treinar um modelo de detecção de cabeça e corpo de lagartos do gênero *Tropidurus* utilizando a versão 12 da arquitetura YOLO.
- Avaliar a integração do YOLOv12 a um filtro de Kalman para suavizar e rastrear a trajetória da cabeça do animal, combinando, assim, um detector e um rastreador, o que permite verificar se essa integração proporcionará melhores resultados em relação ao detector sozinho para esse tipo de tarefa;
- Avaliar o desempenho do modelo YOLOv5, identificado como a arquitetura utilizada atualmente na implementação do software Reptilerecon (ZENÓBIO et al., 2023), a fim de estabelecer uma *baseline* quantificada para a tarefa de detecção;
- Avaliar e comparar o desempenho dos dois modelos propostos nesta monografia com o modelo *baseline* para essa detecção, agora com sua performance quantificada, a partir da base de dados construída em cima dos vídeos utilizados nesse trabalho original e métricas como Precisão, Revocação, *Mean Average Precision* 50 e *Mean Average Precision* 50-95;
- Disponibilizar os modelos treinados e os algoritmos desenvolvidos para a comunidade científica, fomentando a reprodutibilidade e a colaboração em pesquisas futuras.

1.3 Organização do Trabalho

Para tanto, o presente trabalho está estruturado da seguinte forma. A introdução é realizada neste capítulo, onde se discorre sobre a contextualização do tema, sua justificativa e os objetivos principais e específicos traçados. No Capítulo 2, é realizada uma fundamentação teórica

acerca do conhecimento necessário para entendimento do restante do trabalho e uma revisão da literatura, trazendo trabalhos relacionados com uso do YOLO para detecção de animais e da integração do YOLO com o filtro de Kalman na literatura. O Capítulo 3 trata da metodologia e do desenvolvimento realizado ao longo do trabalho para a implementação e avaliação dos modelos propostos. No Capítulo 4, são abordados e discutidos os resultados obtidos com a metodologia seguida. Por fim, temos no Capítulo 5 a conclusão do trabalho, com considerações finais, e possíveis trabalhos futuros a serem realizados.

2 Revisão Bibliográfica

Este capítulo apresenta a fundamentação teórica e a revisão de literatura necessárias para o desenvolvimento desta monografia. Inicialmente, estabelece-se o contexto biológico da comunicação visual em lagartos e os conceitos de visão computacional e aprendizado profundo. Na sequência, o capítulo adota uma organização temática para abordar as técnicas centrais do trabalho: a detecção de objetos e o rastreamento. Nessas seções, são discutidos, respectivamente, a arquitetura do modelo YOLO e o algoritmo de Filtro de Kalman, integrando a explicação de seus princípios de funcionamento com a análise de trabalhos relacionados recentes. Essa abordagem visa contextualizar o estado da arte e fundamentar as escolhas metodológicas adotadas nos experimentos.

2.1 Comunicação Visual no Comportamento Animal

A comunicação é um comportamento essencial para a sobrevivência e a manutenção das relações sociais no reino animal. De forma geral, ela pode ser entendida como um processo de transferência de informação quando o comportamento de um indivíduo emite uma informação a outro, da mesma ou de outra espécie, por meio de um sinal, com a intenção de demonstrar algo e podendo acontecer através dos canais visual, olfativo, tátil ou sonoro (ENGELL, 2009; PALERMO-NETO; ALVES, 2010). Essa comunicação cumpre funções essenciais entre os animais, como a defesa de recursos ou de territórios, a demonstração relacionada a condição reprodutiva do indivíduo, o alerta sobre a presença de predadores ou de presas, a atração de parceiros e o estabelecimento de hierarquias sociais (KAPLAN, 2014).

A comunicação visual é um dos tipos mais ricos e diversificados, indo desde características físicas estáticas do animal até comportamentos dinâmicos, sendo classificadas em duas categorias, a emblemática e a ostentatória (PALERMO-NETO; ALVES, 2010). A primeira, ocorre através das cores e formas do corpo dos invíduos, como, por exemplo, a cauda do pavão que diz para a fêmea sobre a condição reprodutiva do macho de acordo com as cores e o número de "olhos"em suas penas. Já a ostentatória, ocorre através de posturas e alterações corporais específicas, como acontece, por exemplo, entre espécies de lagartos que se comunicam por meio de exibições visuais estereotipadas, fazendo movimentos de cabeça, conhecidos como *head-bobs*, e flexões corporais, os *push-ups* (PASSOS, 2016).

Aprofundando um pouco mais na comunicação dos lagartos, durante as exibições dos *head-bobs* e *push-ups*, a cabeça e o tronco do animal são levantados e abaixados em uma série de movimentos carregando informações sobre a identidade da espécie, do indivíduo e do sexo (MARTINS, 1994). Esses sinais emitidos têm funções diretamente ligadas ao contexto social, sendo classificados como exibições de afirmação, que são os que acontecem em baixa intensidade

e na ausência de outros indivíduos para transmitir identidade, e exibições de desafios, que são os de maior intensidade e direcionados a intrusos territoriais (MARTINS, 1993). Além da complexidade estrutural no sistema de comunicação dos lagartos, existe uma variação significativa no uso dos sinais entre os sexos e os indivíduos. Foram encontradas diferenças consistentes e repetíveis entre indivíduos em quase todos os aspectos da exibição, o que pode servir como base para o reconhecimento individual (MARTINS, 1991). As diferenças sexuais também são marcantes, os machos utilizam exibições mais intensas e complexas do que as fêmeas, um dimorfismo que se acentua durante a estação reprodutiva (MARTINS, 1994). Essa complexidade estrutural e funcional torna o sistema de comunicação visual dos lagartos um campo fértil para o estudo da evolução do comportamento animal.

2.1.1 Limitações na Análise Comportamental dos Animais e a Visão Computacional

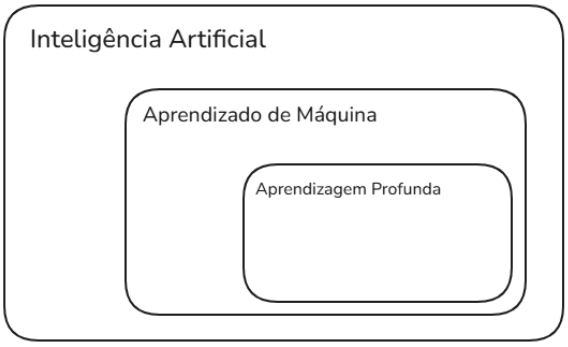
A análise do comportamento animal tradicionalmente fundamentada na observação direta por especialistas enfrenta limitações significativas que podem impactar a robustez e a escala dos estudos (ZENÓBIO et al., 2023). Uma das principais barreiras é o investimento de tempo e esforço exigido, uma vez que existe uma grande quantidade de horas de vídeos para serem analisadas para que os especialistas encontrem padrões nos comportamentos dos animais e identifiquem a intenção do indivíduo, fora o tempo gasto para gravação desses vídeos. Isso torna a compreensão dos sistemas sociais uma tarefa árdua, principalmente quando o objetivo é analisar interações comportamentais diretas, uma vez que ela é infrequente na maioria dos animais (PASSOS, 2016). Além disso, a observação humana é limitada por restrições logísticas, sendo particularmente desafiador registrar comportamentos que ocorrem em contextos de difícil acesso, como à noite ou no subsolo (FORMICA et al., 2010). Outro ponto crítico é a subjetividade inerente ao observador humano, que pode levar a inconsistências na coleta de dados, diminuindo a robustez dos estudos. Por fim, a análise humana pode não ser capaz de identificar todos os padrões de sinalização, especialmente os mais sutis, correndo o risco de não perceber novos padrões naquele domínio (ZENÓBIO et al., 2023).

Diante dessa limitações, a visão computacional é uma área que pode ser utilizada como uma ferramenta poderosa para superar esses desafios. Ela permite que sejam desenvolvidas soluções que trazem mais precisão, escala e eficiência à análise comportamental através de sistemas que podem processar, analisar e interpretar informações visuais de forma automatizada (ZENÓBIO et al., 2023). Com técnicas de aprendizado de máquina é possível desenvolver algoritmos robustos que permitem essa automatização com a realização de tarefas como a detecção de objetos, que irá localizar e identificar os animais durante um vídeo permitindo a análise de seus comportamentos por meio de uma determinada estratégia (SUN; SUN; CHEN, 2024), sendo possível realizar um agrupamento posteriormente, por exemplo, de padrões encontrados para uma análise mais eficaz dos especialistas, como feito em (ZENÓBIO et al., 2023).

2.2 Aprendizado de Máquina para Visão Computacional

Para compreender as técnicas modernas de visão computacional, é essencial entender os conceitos presentes na relação hierárquica entre os campos da Inteligência Artificial (IA), Aprendizado de Máquina e Aprendizagem profunda. Estas são áreas com escopos distintos, mas que estão organizadas como subconjunto uma da outra (NGUYEN et al., 2019). Na Figura 2.1 podemos ver essa relação.

Figura 2.1 – Relação hierárquica entre IA, Aprendizado de Máquina e Aprendizagem Profunda.



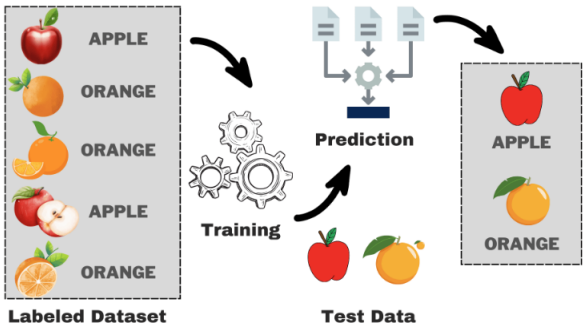
Fonte: Elaborado pelo autor

A IA é o campo mais amplo entre os três, com o objetivo geral de desenvolver técnicas que permitam aos computadores ter um tipo de "inteligência", simulando o pensamento humano (TIAN, 2020). Dentro da IA, encontra-se o Aprendizado de Máquina, conhecido por *machine learning*, que, por sua vez, é um subcampo focado em técnicas que permitem aos sistemas aprender por conta própria a partir de dados e experiências passadas para melhorar seu desempenho em uma tarefa específica (NGUYEN et al., 2019). Esse campo é considerado a principal forma de tornar os computadores inteligentes (TIAN, 2020). Por fim, a Aprendizagem Profunda, também conhecida como *deep learning*, é um subcampo do Aprendizado de Máquina que utiliza Redes Neurais Artificiais (RNAs) com múltiplas camadas, essas que simulam as redes neurais humanas. Seu objetivo é aprender representações de dados com múltiplos níveis de abstração, conseguindo uma complexidade maior nos padrões aprendidos (LECUN; BENGIO; HINTON, 2015). A capacidade da Aprendizagem Profunda de descobrir e aprender automaticamente as *features* necessárias para a tarefa em questão diretamente dos dados brutos, como os *pixels* de uma imagem, é o que destaca essa área. Com isso, por exemplo, não é mais necessário um especialista projetar manualmente um extrator de características. Essa capacidade de extração automática de representações complexas é o que permitiu os avanços mais significativos em visão computacional (LECUN; BENGIO; HINTON, 2015).

Dentro do Aprendizado de Máquina, os algoritmos normalmente são agrupados em diferentes paradigmas de aprendizado, sendo os dois principais o supervisionado e o não supervisionado (LASKOV et al., 2005). No aprendizado supervisionado, o modelo aprende a partir de um conjunto de dados previamente rotulado por humanos. Nesse caso, cada exemplo

de treinamento consiste em um par de dados: um conjunto de dados de entrada e uma saída desejada ou "rótulo verdadeiro". Com isso, o objetivo é o algoritmo ajustar seus parâmetros internos, para que tenha as saídas corretas de acordo com as entradas, minimizando o erro entre a previsão e o rótulo verdadeiro (FAGER et al., 2021). Um exemplo disso é o modelo YOLO, que utiliza imagens anotadas para aprender a prever os *bounding boxes* (REDMON et al., 2016). A Figura 2.2 traz um exemplo do fluxo de um aprendizado supervisionado. O grande benefício dessa abordagem é seu alto desempenho, já que os métodos supervisionados frequentemente superam os não supervisionados em termos de precisão, em um contexto em que os dados de teste sigam a mesma distribuição dos dados de treinamento. Entretanto, a principal desvantagem desse tipo de aprendizado é a dependência de dados rotulados, sendo que, dependendo da base de dados, o processo de anotação pode ser extremamente demorado (LASKOV et al., 2005).

Figura 2.2 – Fluxo de um aprendizado supervisionado.

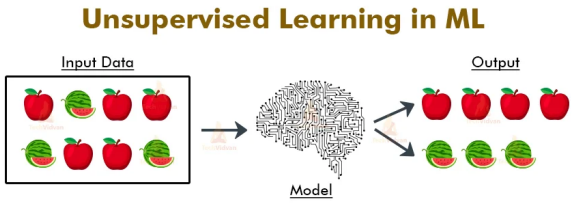


Fonte: Mehreen (2023).

Por outro lado, o aprendizado não supervisionado não necessita de dados rotulados. Ele é usado para extrair informações de um conjunto de dados, com o objetivo de descobrir, por conta própria, padrões, estruturas e relações ocultas nos dados (FAGER et al., 2021). Em vez de prever rótulos como saída, técnicas buscam agrupar dados similares, como o *clustering*, ou aprender representações compactas dos dados, como os *autoencoders*. A principal vantagem desse tipo de aprendizado é que ele não exige um processo custoso na rotulação de dados e seu desempenho não é afetado por padrões desconhecido, já que para ele todos os dados são por natureza desconhecidos. Entretanto, por ele não passar pelo processo de treinamento aprendendo com dados já anotados, ele acaba perdendo na precisão do modelo (FAGER et al., 2021). Na Figura 2.3 é possível ver um exemplo de fluxo de aprendizado não supervisionado. Portanto, a escolha entre os paradigmas de aprendizado envolve um balanço entre a precisão desejada e os recursos disponíveis, como a quantidade de dados rotulados e a capacidade computacional.

Diante da capacidade de aprendizado de representações complexas citada anteriormente, a Aprendizagem Profunda superou os métodos tradicionais em uma vasta gama de tarefas de visão computacional, como reconhecimento e detecção (LECUN; BENGIO; HINTON, 2015). Essa capacidade é visível nos modelos de RNAs e, em particular para o processamento de imagens, as

Figura 2.3 – Fluxo de um aprendizado não supervisionado.



Fonte: Team (2020).

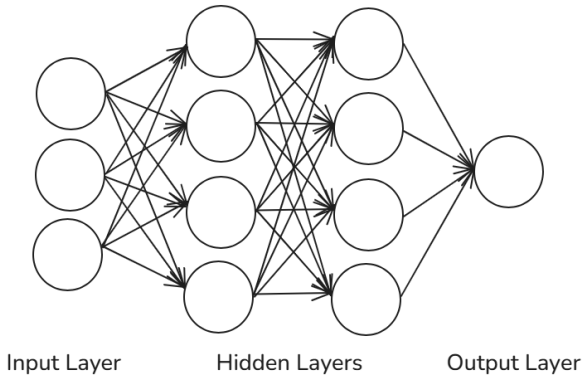
Convolutional Neural Network (CNN)s, que é uma especialização da primeira. Ambas as redes são a base para o YOLO e serão descritas em detalhes na subseções 2.2.1 e 2.2.2.

2.2.1 Redes Neurais Artificiais

As RNAs são modelos computacionais que simulam o funcionamento das redes neurais humanas. Elas são organizadas em camadas compostas por unidades de processamento simples que estão interconectadas, chamadas de neurônios artificiais (O'SHEA; NASH, 2015). Esses neurônios trabalham de forma distribuída e coletiva, buscando aprender com os dados e otimizar uma saída final. Com essa arquitetura, as RNAs possuem uma capacidade de aprender padrões complexos, não lineares e de se adaptar a novos dados, o que as torna uma das ferramentas mais poderosas no aprendizado de máquina (RANA et al., 2018).

Sua arquitetura, basicamente, é organizada em camadas, geralmente existindo uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída (DESHMUKH, 2018). Primeiro, a camada de entrada, chamada também de *input layer*, recebe os dados brutos de entrada para a tarefa. Por sua vez, as camadas ocultas, ou *hidden layers*, são responsáveis por realizar a maior parte do processamento e o aprendizado a partir dos dados. Por fim, a camada de saída, ou *output layer*, produz o resultado final do modelo. Na Figura 2.4 é mostrado uma representação visual dessa arquitetura.

Figura 2.4 – Rede Neural Artificial.



Fonte: Elaborado pelo autor

Quando uma rede tem duas ou mais camadas ocultas, ela é considerada uma rede neural profunda, caso contrário ela é uma rede rasa (SARKER, 2021). Teoricamente, quanto mais camadas em uma rede, mais complexos serão os padrões aprendidos e mais *features* não lineares ela consegue obter, o que induz a ideia de quanto maior o número de camadas melhor. Entretanto, o funcionamento geral não é este, quanto maior o número de camadas, maior o risco de *overfitting* na rede, ou seja, a rede se sobreajusta aos dados presentes no treinamento e não consegue generalizar para novos dados, e maior a demanda de poder computacional para o ajuste dos seus parâmetros (LECUN; BENGIO; HINTON, 2015; O'SHEA; NASH, 2015). Portanto, deve-se existir um equilíbrio entre a complexidade do modelo com a dificuldade da tarefa e dos dados que serão utilizados por ele (SARKER, 2021).

Cada neurônio em uma camada recebe as saídas dos neurônios da camada anterior e essas conexões possuem pesos associados a elas, que são os parâmetros ajustáveis e representam a "força" da conexão entre os dois neurônios em questão. Dentro de cada neurônio, primeiro, é calculado uma soma ponderada de todas as suas entradas e, logo em seguida, o resultado dessa soma passa por uma função de ativação. Essa função, que é não-linear, é responsável por introduzir a capacidade da rede de aprender relações complexas e não-lineares nos dados, determinando se o neurônio será "ativado"e qual será o seu sinal de saída (RANA et al., 2018). Esse processamento pode ser observado pela seguinte equação matemática:

$$y = \phi\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{2.1}$$

onde:

- x_i: entrada i do neurônio.
- w_i: peso associado à entrada x_i.
- b: viés (*bias*), termo adicional que permite deslocar a função de ativação.
- n: número de conexões de entrada do neurônio.
- ϕ: função de ativação.
- y: saída do neurônio após o processamento.

O processo pelo qual uma rede neural aprende é chamado de treinamento, e, geralmente nas redes de múltiplas camadas, isso ocorre pelo *backpropagation* (RANA et al., 2018). Esse tipo de treinamento ocorre em um ciclo de duas fases: a primeira é o *forward pass* e a segunda é o *backward pass*. No *forward pass* o modelo recebe um conjunto de dados como entrada e a informação flui através das camadas para produzir uma saída. Já no *backward pass*, a previsão feita é comparada com o resultado correto e um erro é calculado. O algoritmo de *backpropagation* utiliza, então, uma técnica de otimização baseada em descida de gradiente para propagar esse

erro de volta através da rede, ajustando os pesos de cada conexão com o objetivo de minimizar o erro total da rede ao máximo. Ao final do treinamento, espera-se que a rede consiga fazer boas predições de forma generalizada para novos conjuntos de dados.

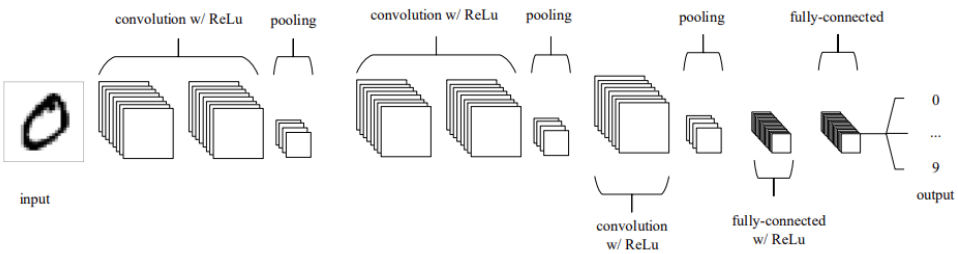
Diante disso, as RNAs formam a base para arquiteturas mais avançadas e especializadas. Uma especialização particularmente importante dessas redes, projetada especificamente para lidar com dados que possuem uma topologia de grade, como as imagens, são as CNNs (ALBAWI; MOHAMMED; AL-ZAWI, 2017), que serão detalhadas na próxima seção.

2.2.2 Redes Neurais Convolucionais

As redes neurais convolucionais, ou CNNs, são uma classe das RNAs profundas, sendo uma das arquiteturas mais populares atualmente, principalmente quando se diz respeito ao processamento de vídeos e imagens (ALBAWI; MOHAMMED; AL-ZAWI, 2017). Elas já são projetadas especificamente com esse foco de processar e extrair informações de dados que possuem uma topologia de grade, como uma imagem, aprendendo automaticamente características presentes nessa imagem, como bordas ou até objetos. Diferentemente de uma rede neural tradicional, as CNNs, ao invés de tratar a entrada como um vetor plano, exploram a forte correlação espacial entre pixels vizinhos de uma imagem, o que cria uma arquitetura mais simples e eficiente para esse tipo de tarefa, com menos parâmetros e conexões (O'SHEA; NASH, 2015). É essa característica que faz com que esse tipo de rede neural seja altamente eficaz em tarefas de reconhecimento de padrões na visão computacional, como detecção de objetos e classificação de imagens.

A arquitetura de uma CNN é composta por diversas camadas empilhadas, entre elas as principais são as camadas de convolução, as camadas de *pooling* e as camadas totalmente conectadas (O'SHEA; NASH, 2015). As duas primeiras são as principais responsáveis pela extração de características da imagem, enquanto a camada totalmente conectada mapeia essas informações extraídas para uma saída final, como uma classificação (YAMASHITA et al., 2018). Na Figura 2.5, é apresentado um exemplo de arquitetura de uma rede neural convolucional, trabalhando com a tarefa de classificação de números.

Figura 2.5 – Exemplo de Rede Neural Convolucional.

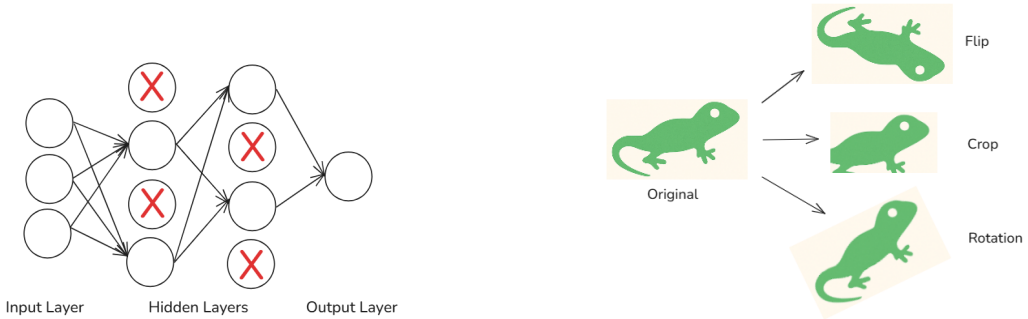


Fonte: O'Shea e Nash (2015).

O treinamento de uma CNN tem como objetivo aprender o valores ideais para os parâmetros da rede, que, nesse caso, são os pesos dos filtros convolucionais e os pesos das camadas

totalmente conectadas, visando otimizar o modelo para ter os melhores resultados possíveis e minimizar os erros obtidos. Os principais desafios enfrentados pelas CNNs são a necessidade de grandes conjuntos de dados para ter um resultado satisfatório após o treinamento, que pode ser contornado em alguns contextos com a utilização de *transfer learning*, e o *overfitting*, que pode ser mitigado a partir de técnicas como *data augmentation*, *dropout* e *weight decay* (YAMASHITA et al., 2018). Essas técnicas consistem, respectivamente, em aumentar os dados de treino aplicando transformações nas imagens originais, desativar neurônios aleatoriamente durante o treinamento e penalizar pesos de grande magnitude, sendo que as duas últimas visam prevenir a complexidade excessiva do modelo, forçando-o a aprender características mais robustas e generalizáveis. Na Figura 2.6 é possível visualizar exemplos de *data augmentation* e de *dropout* na prática.

Figura 2.6 – Exemplos de técnicas de regularização e aumento de dados em redes neurais.



(a) Exemplo de *Dropout*.

(b) Exemplo de *Data Augmentation*.

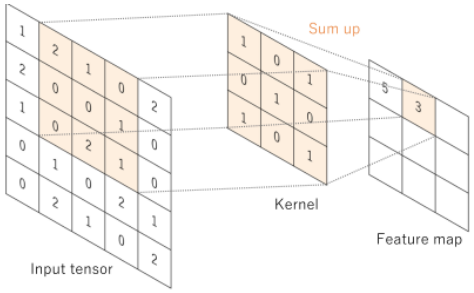
Fonte: Elaborado pelo autor.

2.2.2.1 Camada de Convolução

A camada de convolução é o coração da CNN e é onde ocorre a maior parte do processamento computacional. Esta camada realiza a operação de convolução, que consiste em aplicar um filtro, também conhecido como *kernel*, que é uma matriz de pesos aprendidos ao longo do treinamento, sobre a imagem de entrada (ALBAWI; MOHAMMED; AL-ZAWI, 2017). O filtro desliza pela imagem e calcula, em cada posição, o produto escalar entre seus pesos e a parte da imagem que ele está cobrindo de acordo com seu tamanho. O resultado desse processo gera um mapa de características que indica a presença de uma característica específica em diferentes locais da imagem, como uma borda ou uma cor (YAMASHITA et al., 2018). Na Figura 2.7, é possível ver essa operação sendo realizada.

As principais ideias dessa camada são: a conectividade local, já que os neurônios do mapa de características são conectados apenas a uma região da camada anterior, fazendo com que a rede aprenda primeiro características locais; os pesos compartilhados, já que um mesmo filtro é utilizado por toda a imagem, logo, o modelo aprende a identificar a característica em questão independente de onde apareça na imagem, e, isso garante também que, um número muito menor de parâmetros seja utilizado; e os múltiplos filtros, utilizados paralelamente pela rede para

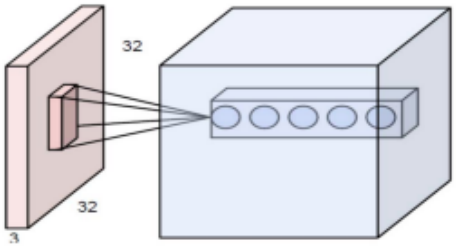
Figura 2.7 – Exemplo de operação de convolução.



Fonte: Yamashita et al. (2018).

detectar diferentes características na imagem, gerando diversos mapas de características de uma vez (O'SHEA; NASH, 2015), como pode ser visto na Figura 2.8.

Figura 2.8 – Múltiplos filtros extraíndo diferentes características de uma mesma região da imagem.



Fonte: Albawi, Mohammed e Al-Zawi (2017).

Por fim, após a convolução, é aplicada um função de ativação não linear no mapa de características, como a *Rectified Linear Unit* (ReLU). A ReLU é a mais utilizada atualmente por ser eficiente e já mitigar problemas como o de *vanishing gradient* (ALBAWI; MOHAMMED; AL-ZAWI, 2017). Ela é calculada da segunte forma:

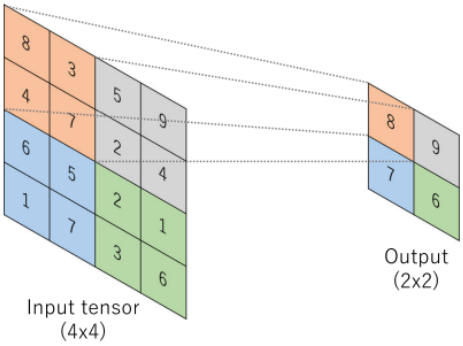
$$\phi(x) = \max(0, x) \tag{2.2}$$

2.2.2.2 Camada de *Pooling*

A camada de *pooling* normalmente segue uma camada de convolução e tem como objetivo reduzir a dimensionalidade espacial dos mapas de características gerados, visando a diminuição de complexidade da rede, controlar o *overfitting* e ser mais robustas a variações na imagem, como translações (O'SHEA; NASH, 2015; YAMASHITA et al., 2018). As operações das camadas de *pooling* são definidas por hiperparâmetros e elas não possuem parâmetros aprendidos.

A operação mais comum é o *Max Pooling*, onde, basicamente, uma janela vai deslizando sobre o mapa de características e o maior valor na região em que a janela está sobreposta é selecionado, descartando os outros valores, como pode ser visto na Figura 2.9. Com isso, a característica considerada mais forte da região é mantida (YAMASHITA et al., 2018).

Figura 2.9 – Exemplo de operação de *Max Pooling*.



Fonte: Yamashita et al. (2018).

2.2.2.3 Camada Totalmente Conectada

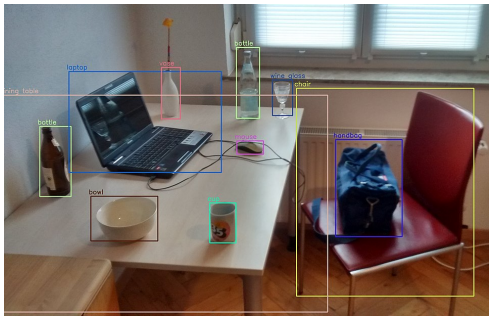
Por fim, após várias camadas de convolução e *pooling*, os mapas de características finais geralmente passam por uma camada *flatten*, onde são transformados em um vetor unidimensional, e são levadas para uma ou mais camadas totalmente conectadas. Essas camadas são responsáveis pelo mapeamento dessas características extraídas para ter uma saída final de predição para a tarefa (YAMASHITA et al., 2018).

2.3 Detecção de Objetos

A detecção de objetos, uma das tarefas mais importantes e desafiadoras em visão computacional, tem o objetivo de determinar a localização precisa de objetos presentes em uma imagem ou vídeo e de classificá-los, trazendo a informação do que eles se tratam de acordo com a tarefa (SUN; SUN; CHEN, 2024). Uma das formas de obter a localização desses objetos é através de caixas delimitadoras, mais conhecidas como *bounding boxes*, que basicamente são retângulos que contornam os objetos detectados, fornecendo informações como as coordenadas, altura e largura dos mesmos. Na Figura 2.10, é possível ver um exemplo de detecção por *bounding boxes*. Portanto, a detecção tem uma complexidade ainda maior que a classificação, uma vez que ela tem que retornar a localização dos objetos e o rótulo dos mesmos (SATHIARAJ; SATHIARAJ; BEWOOR, 2021).

Com a capacidade de localizar e identificar objetos de forma rápida em fotos e vídeos, a detecção de objetos tem uma grande variedade de possibilidades de aplicações no mundo (DIWAKAR; RAJ, 2022). Quando diz respeito a sistemas inteligentes no trânsito, podemos encontrar essa detecção sendo utilizada em carros autônomos, para a detecção de pessoas, de outros veículos, de placas e sinais de trânsito, e em sistemas de monitoramento, para cidades que desejam ter uma central inteligente para tomar ações rápidas em casos de acidentes, infrações e melhorias no trânsito. Já na área de segurança e vigilância, ela é utilizada para monitoramento realizando detecção facial e identificação de atividades suspeitas em tempo real. Na área industrial,

Figura 2.10 – Exemplo de detecção de objetos com *bounding boxes*.



Fonte: MTheiler (2019).

ela é extremamente útil para a garantia de uso de equipamentos de segurança adequados para cada cenário. Por fim, podemos citar outras aplicações como em imagens médicas de radiografia, para monitoramento ambiental, tanto em área terrestre quanto marítima, na agricultura e em robótica industrial (DIWAKAR; RAJ, 2022).

De maneira geral, podemos abordar a evolução ocorrida nos métodos de detecção de objeto dividindo-a em duas partes, os métodos anteriores ao *deep learning* e os métodos posteriores a ele, uma vez que o *deep learning* foi um ponto de virada nessa área (DIWAKAR; RAJ, 2022). Os métodos tradicionais iniciais, referentes à primeira parte, dependiam de características extraídas manualmente, como por exemplo o *Histogram of Oriented Gradients* (HOG), o *Scale-Invariant Feature Transform* (SIFT) e o *Speeded Up Robust Features* (SURF) (SUN; SUN; CHEN, 2024). Após o surgimento e uso do *deep learning*, as CNNs revolucionaram essa área e os métodos mais utilizados passaram a ser divididos em duas categorias: os detectores de dois estágios, que primeiro geram várias possíveis regiões onde pode estar um objeto e, depois, aplicam um classificador baseado em CNN em cada uma delas, e os detectores de um estágio, que tratam a detecção como um único problema de regressão, prevendo as coordenadas dos objetos e as possibilidades de classes através do processamento da imagem inteira de uma só vez (DIWAKAR; RAJ, 2022). Exemplos de detectores de dois estágios e um estágio, respectivamente, são *Fast Region-based Convolutional Neural Network* (R-CNN) e YOLO. Os detectores de um estágio muitas vezes se destacam entre as duas categorias pela sua velocidade, o que é essencial em aplicações em tempo real. O YOLO, atualmente, tem sido bastante utilizado pela sua velocidade e precisão, processando a imagem em uma única passada na rede neural (REDMON et al., 2016).

2.3.1 Modelo YOLO

Como exposto, a detecção de objetos é uma tarefa fundamental em visão computacional. Tradicionalmente, classificadores eram reaproveitados para realizar essa detecção e as técnicas não enxergavam a imagem inteira durante as fases de treinamento e teste. Alguns sistemas usavam uma abordagem de janela deslizante, aplicando o classificador em várias localizações e escalas da imagem, como *Deformable Parts Models* (DPM) (FELZENSZWALB et al., 2010), outros já

geravam *bounding boxes* potenciais e depois executavam um classificador nessa regiões, como o *Fast* R-CNN (GIRSHICK, 2015). Entretanto, o *pipeline* desses sistemas eram complexos, lentos e difícil de otimizar, já que cada componente precisava ser treinado separadamente. Superando essas limitações, o YOLO trata a tarefa de detecção de objetos como um problema de regressão, usando uma única rede neural que prevê as *bounding boxes* e as probabilidades de classes presentes em cada uma em uma única avaliação. Dessa forma, o *pipeline* é simplificado sendo uma única rede que pode ser otimizada de ponta a ponta diretamente (REDMON et al., 2016). E, baseado nessas características que o nome *"You Only Look Once"* (em português, você só vê uma vez) foi definido para o modelo.

O funcionamento do sistema se baseia em uma divisão da imagem de entrada em uma grade de S x S células. A célula responsável por detectar um objeto é aquela que tem o centro deste localizado dentro de si. Com isso, cada célula da grade prevê B *bounding boxes*, onde cada previsão contém cinco informações: as coordenadas do centro da *bounding box* em questão relativas à celula (x, y), a largura (w) e a altura (h) relativas à imagem e uma potuação de confiança, indicando a certeza do modelo que aquela *bounding box* contém um objeto e a precisão que ela foi prevista. Essa pontuação de confiança pode ser representada como:

$$Pr(Objeto) * IOU(pred, verdade) \tag{2.3}$$

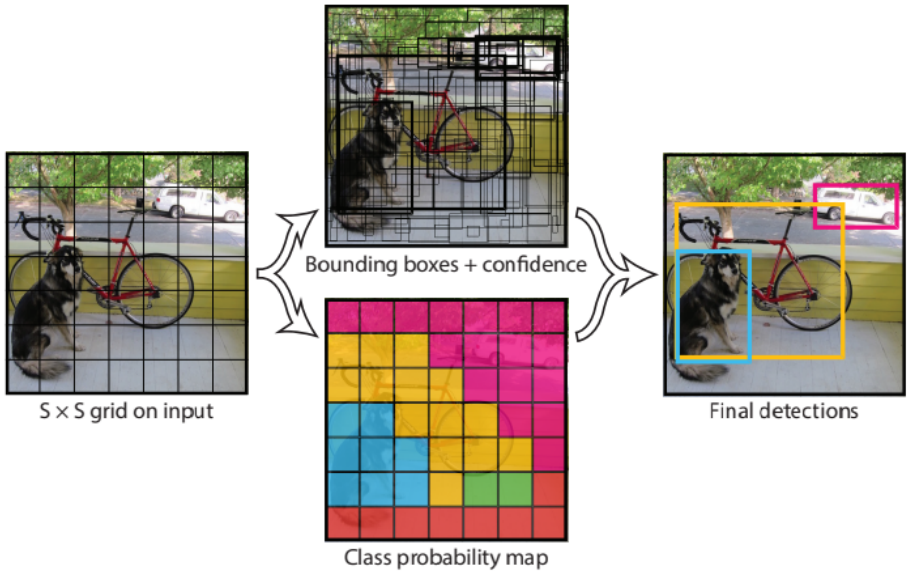
onde, IOU é a Interseção sobre União entre as *bounding boxes* prevista e real. Cada célula prevê também C probabilidades de classe condicionais, prevendo a possibilidade de um objeto pertencer a uma determinada classe, assumindo que já existe um objeto naquela célula. Apenas um conjunto de probabilidades é previsto por célula, independentemente do número de *bounding boxes*. Ao final, as predições são codificadas em um tensor de dimensão S x S x (B * 5 + C) (REDMON et al., 2016). Em um *dataset* como o PASCAL VOC, por exemplo, que possui 20 classes, utilizando uma grade 7x7 e 2 *bounding boxes* por célula, a saída é um tensor de dimensão 7x7x30. Uma representação desse funcionamento pode ser visualizada na Figura 2.11.

A arquitetura do YOLO foi inspirada no modelo GoogLeNet (SZEGEDY et al., 2014). A primeira versão padrão do modelo é constituída por 24 camadas convolucionais para extração de características e por 2 camadas totalmente conectadas responsáveis por prever as probabilidades e coordenas finais. O YOLO utiliza uma abordagem com camadas de redução de 1x1 seguidas por camadas convolucionais 3x3. A arquitetura completa pode ser vista na Figura 2.12.

Para aplicações que exigem ainda mais velocidade na predição, foi desenvolvida uma versão chamada *Fast* YOLO, que possui apenas 9 camadas convolucionais, permitindo uma performance de mais de 150 *frames* por segundo (FPS). A camada final de ambas as versões utiliza uma função de ativação linear, enquanto todas as outras camadas utilizam uma função de ativação linear retificada (*leaky rectified linear activation*), apresentada a seguir:

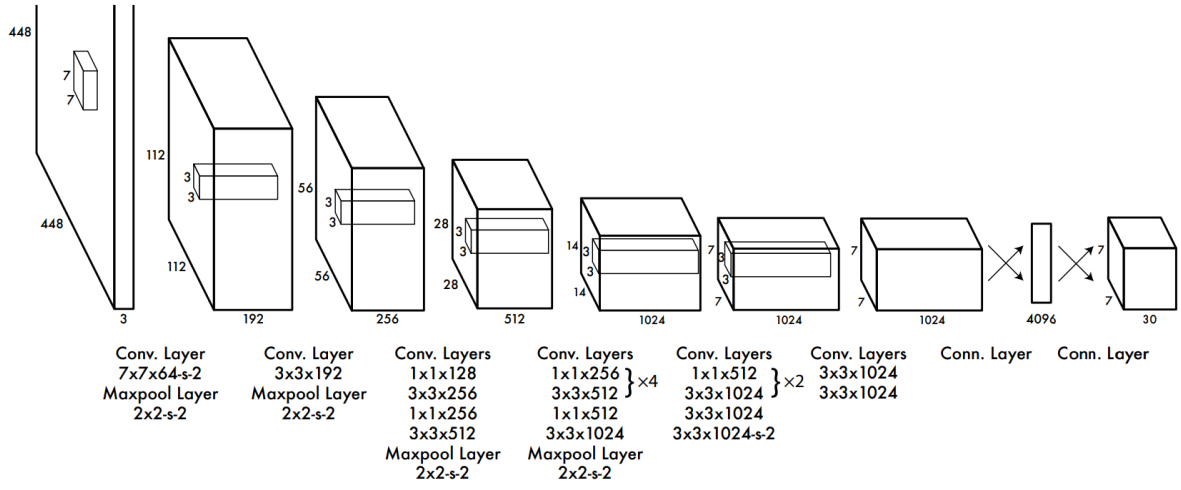
$$\phi(x) = \begin{cases} x & \text{, se } x > 0 \\ 0.1x & \text{, caso contrário} \end{cases} \tag{2.4}$$

Figura 2.11 – Modelo YOLOv1.



Fonte: Redmon et al. (2016).

Figura 2.12 – Arquitetura do modelo YOLOv1.



Fonte: (REDMON et al., 2016)

O treinamento do YOLO é dividido em duas fases principais: pré-treinamento para classificação e treinamento para detecção. Durante a primeira, as 20 primeiras camadas convolucionais do modelo são pré-treinadas no *dataset* de classificação *ImageNet*, que contém 1000 classes. Já na fase de treinamento para tarefa detecção, quatro novas camadas convolucionais e duas camadas totalmente conectadas são adicionadas para adaptar o modelo para essa tarefa. A resolução da imagem de entrada também é redimensionada para 448x448 visando uma resolução mais fina para uma melhor detecção.

A *loss function* utilizada pelo modelo é um erro quadrático somado com múltiplas partes,

que pondera os diferentes tipos de erro de forma distinta. Para estabilizar o treinamento, o erro de localização (coordenadas da detecção) recebe um peso maior, enquanto o erro de confiança para *bounding boxes* que não contêm objetos recebe um peso menor. Além disso, para tratar erros em *bounding boxes* pequenas e grandes de forma mais justa, o modelo prevê a raiz quadrada da largura e da altura. Por fim, para prevenção de *overfitting*, uma camada de *dropout* com taxa de 0,5 é utilizada após a primeira camada conectada e a técnica de *data augmentation* é aplicada, fazendo o escalonamento e translações aleatórias da imagem.

Após essa primeira versão do modelo, outras versões subsequentes foram desenvolvidas focando em refinar essa base (JEGHAM et al., 2025). O YOLOv2 incorporou o *backbone Darknet-19*, normalização em lote e técnicas de *data augmentation*. O YOLOv3 adotou o *Darknet-53* e introduziu detecção em múltiplas escalas com uma arquitetura inspirada na *Feature Pyramid Network* (FPN). O YOLOv4 ampliou essa abordagem com módulos como *Spatial Pyramid Pooling* (SPP) e *Path Aggregation Network* (PAN), enquanto o YOLOv5 marcou a transição para o *framework PyTorch*, trazendo o módulo *Spatial Pyramid Pooling Fast* (SPPF). Posteriormente, as versões continuaram a inovar com novas arquiteturas e módulos, como a *RepVGG* e *CSPStackRep* no YOLOv6, *Extended Efficient Layer Aggregation Networks* (E-ELAN) no YOLOv7, e a expansão funcional do YOLOv8, que passou a contemplar tarefas além da detecção, incluindo segmentação semântica, estimação de pose e detecção de *bounding boxes* delimitadoras orientadas. O YOLOv9, por sua vez, introduziu melhorias no treinamento com o *Programmable Gradient Information* (PGI) e a *Generalized Efficient Layer Aggregation Networks* (GELAN). As versões mais recentes focaram em otimizações de ponta, como o YOLOv10 que eliminou a necessidade de *Non-Maximum Suppression* com uma abordagem de detecção *One-to-One*. Já os YOLOv11 e YOLOv12 incorporaram mecanismos de atenção e aprimoramentos estruturais para melhorar a detecção em cenários complexos. Essa trajetória demonstra a busca pelo aprimoramento da precisão, velocidade e versatilidade do YOLO, consolidando-se como um dos principais frameworks para aplicações em visão computacional. Todo esse acumulo de melhorias ao longo das gerações do modelo, com destaque para a maior capacidade de generalização e precisão na versão 12, mesmo em ambientes com ruído visual que é essencial para o contexto do trabalho, fundamenta a hipótese de que os avanços recentes do YOLO podem contribuir significativamente para a análise automatizada da comunicação visual em lagartos.

2.3.2 YOLO na Literatura

Nesta subseção, serão analisados trabalhos que aplicam o algoritmo YOLO para a detecção de animais, estabelecendo sua eficácia e robustez para este tipo de tarefa.

Em Zenóbio et al. (2023) foi proposto um modelo com o objetivo de extrair, analisar e identificar padrões nos sinais de comunicação de lagartos do gênero Tropidurus, visando superar as limitações de abordagens anteriores como a análise manual por parte dos biólogos e técnicas de *Template Matching*. Para isso, inicialmente o modelo utilizou o YOLOv4 para detecção da

posição da cabeça e do corpo do lagarto e, a partir das *bounding boxes*, era traçado uma reta entre o centro delas, permitindo o cálculo da angulação entre as duas partes do animal em cada quadro. Esse cálculo geométrico, fundamentado na modelagem proposta por (SPONG; HUTCHINSON; VIDYASAGAR, 2005), garante que a métrica extraída seja invariante às possíveis movimentações da câmera, focando exclusivamente na postura do animal. Com essa angulação, uma série temporal que representa o movimento de cabeceio foi extraída, depois ocorreu a segmentação dos sinais através da técnica baseada em uma janela deslizante, a compressão dos sinais segmentados por uma rede *autoencoder* e, por fim, algoritmos de aprendizado não supervisionado foram aplicados para encontrar padrões presentes nesses sinais. Para treinamento do YOLO, foram utilizados 600 quadros rotulados manualmente. Com os experimentos, realizados com 44 vídeos, cada um com um único lagarto, foram identificados 10 grupos de padrões de sinais distintos. Este trabalho é relevante por aplicar com sucesso o YOLO no mesmo contexto de análise de comportamento animal que o presente trabalho. Contudo, é importante ressaltar que o código-fonte do software Reptilerecon, produto desse estudo registrado no INPI, utiliza a versão 5 do YOLO. Dessa forma, será ela a adotada aqui para fins de comparação.

No contexto de animais selvagens, visando uma solução de baixo custo e baixo consumo de energia para o monitoramento contínuo da vida selvagem, em (NISHANTH; RAKESHKUMAR; THENMOZHI, 2025) foi proposto um sistema de detecção de animais selvagens em tempo real. O trabalho teve como objetivo superar as limitações de métodos tradicionais e na ineficiência de modelos de detecção mais antigos para aplicações em tempo real presentes nos dispositivos de borda. Justamente, por isso, foi escolhido o YOLOv10 para ser implementado em uma placa *NVIDIA Jetson Nano*, tendo em vista sua maior acurácia, velocidade de inferência e melhor adaptação em condições adversas. Nos experimentos, realizados com o *dataset Wild Animall*, os autores compararam diretamente o desempenho do YOLOv10 com o do YOLOv5. Os resultados demonstraram que o YOLOv10 alcançou um *Mean Average Precision* (mAP)@0.5 de 0,934, enquanto o YOLOv5 alcançou 0,89, além de obter valores superiores de precisão e de *recall*. É importante destacar que este trabalho valida o YOLOv10 como um detector de ponta para aplicações de monitoramento de animais.

Ainda nesse contexto, mas agora refente a cenários nas áreas agrícolas, em (PREETHI et al., 2025) foi proposto um sistema para detecção e desestimulação da vida selvagens nessas áreas, visando evitar perdas financeiras, danos ambientais e riscos à segurança causados pelo conflito entre humanos e animais selvagens. Para isso, foi utilizado o modelo YOLOv11 para detecção de animais em tempo real a partir de vídeos ao vivo e uma CNN para fazer a classificação da espécie do animal detectado. Quando uma espécie de animal considerada uma ameaça é detectada, um dissuasor ultrassônico é acionado repelindo o animal através de ondas sonoras de alta frequência, sem causar danos a ele. Nos experimentos realizados com o YOLO, foram utilizadas mais de 10000 imagens e o modelo alcançou uma acurácia de detecção de 95%, uma precisão de 93% e um recall de 91%. Já o sistema de dissuasão atingiu uma taxa de sucesso de 87%. É interessante notar que, este trabalho apresenta uma tecnologia de detecção, classificação e uma rápida ação

de resposta automática em relação aos animais, validando o uso de modelos como YOLO em sistema de gestão ambiental.

Já em (RADHAKRISHNAN et al., 2024), foi proposto um sistema automatizado que detecta a presença de animais, como lagarto, rato ou barata, no contexto de restaurantes, visando garantir a higiene das cozinhas em tempo real. O objetivo desse trabalho é mitigar riscos à saúde pública através desse sistema automatizado, superando as limitações dos métodos de inspeção manual. Para a identificação das violações a partir de câmeras, foi utilizado o detector YOLOv8 integrado com a plataforma *Firebase* para armazenamento em nuvem e notificação de alertas para um aplicativo *Android*. Para os experimentos, foi utilizado um *dataset* com 9400 imagens para treinar o modelo e foi alcançada uma acurácia geral de 89%. Vale destacar que, o modelo foi treinado e avaliado para identificar, entre outras violações, a presença de lagartos, o que valida o uso do YOLOv8 para esse contexto e justifica a utilização de versões recentes desse modelo para o presente trabalho.

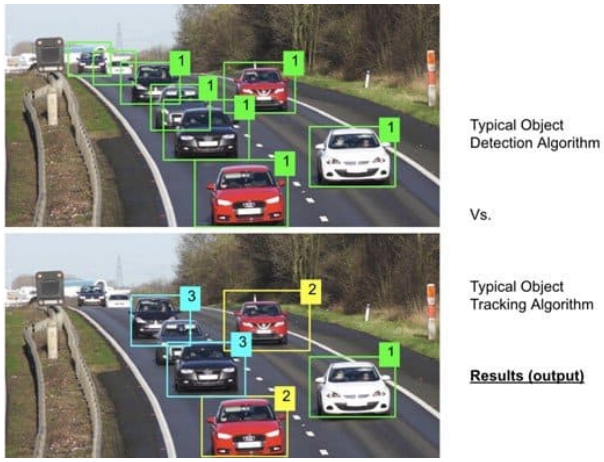
Em suma, a análise desses trabalhos corrobora a utilização de versões mais recentes do detector YOLO, uma vez que estas demonstram um desempenho superior a versões mais antigas, conforme mostrado pelos experimentos realizados em Nishanth, Rakeshkumar e Thenmozhi (2025), Jegham et al. (2025). Ademais, os estudos Zenóbio et al. (2023), Radhakrishnan et al. (2024) apresentam resultados satisfatórios com a aplicação do YOLO em cenários que envolvem a detecção de lagartos, além de todos os outros estarem aplicados para a tarefa de detecção de animais, o que confirma o potencial desta técnica para tal tarefa perante os resultados obtidos. Portanto, diante dessas evidências, o presente trabalho empregará versões mais recentes desse modelo em seus experimentos.

2.4 Rastreamento de Objetos

O rastreamento de objetos, ou *tracking*, em visão computacional é o processo de localizar um ou mais objetos em movimento ao longo de uma sequência de vídeo. Diferente da detecção de objetos, que opera em quadros individuais de forma independente, o rastreamento adiciona a dimensão temporal, conectando as detecções de um mesmo objeto através dos quadros para entender sua dinâmica (JOSHI et al., 2018). O resultado desse processo é a geração de uma trajetória, que descreve o caminho percorrido pelo objeto, permitindo uma análise muito mais rica de seu comportamento. Na Figura 2.13, é realizada uma comparação visual entre a detecção e o rastreamento de objetos.

O rastreamento de objetos tem diversas aplicações e é essencial em múltiplos domínios (JOSHI et al., 2018). Um desses campos é o de vigilância por vídeo, onde o rastreamento é usado para monitorar atividades, detectar comportamentos anormais e proteger locais sensíveis como bancos ou instalações de defesa. Outra área em que se destaca o uso dessa técnica é a análise de humanos, que vai desde o rastreamento de pessoas até o reconhecimento de ações e atividades.

Figura 2.13 – Comparação entre detecção e rastreamento de objetos.



Fonte: Klingler (2023).

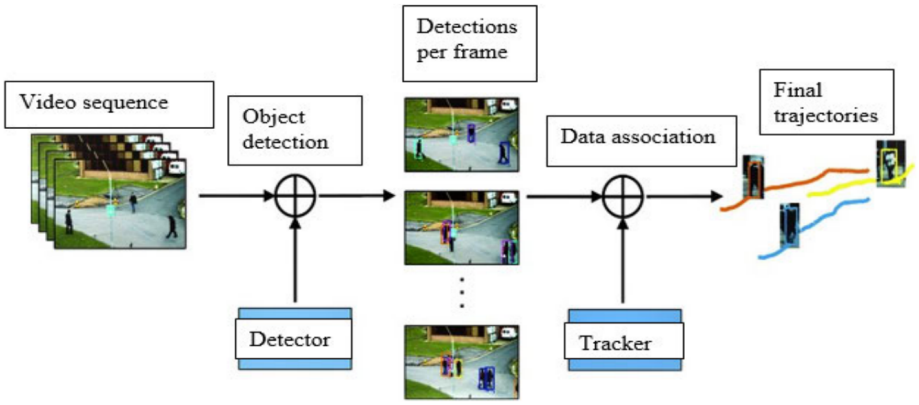
Além disso, o rastreamento de veículos também é essencial no que diz respeito ao monitoramento de trânsito em rodovias.

A importância do rastreamento está na sua capacidade de manter a identidade de um objeto ao longo do tempo. Manter a identidade é crucial para extrair informações valiosas como velocidade, aceleração, direção do movimento e padrões de comportamento complexos. O principal desafio para isso, especialmente em sistemas de múltiplos objetos, é o problema da associação de dados, que consiste em associar corretamente as detecções do quadro atual às trajetórias existentes (KALAKE; WAN; HOU, 2021).

Dentre as diversas abordagens para o rastreamento, a evolução das CNNs contribuiu para o avanço do paradigma de rastreamento por detecção (*Tracking-by-Detection* (TBD)) (KALAKE; WAN; HOU, 2021). Nessa estratégia, o processo é dividido em duas etapas: primeiro, um detector de objetos é aplicado para encontrar todos os alvos em um quadro, e, logo em seguida, um algoritmo de rastreamento utiliza essas detecções para gerar e conectar os segmentos de trajetória, associando as detecções atuais com as trajetórias passadas. A qualidade e a robustez do detector inicial são, portanto, fundamentais para o sucesso de todo o sistema de rastreamento. Na Figura 2.14, é possível ver o funcionamento de um rastreamento por detecção.

No que concerne o rastreamento por detecção, a etapa de associação frequentemente depende da capacidade de predizer a próxima posição de um objeto, especialmente em casos de oclusão. Para essa tarefa de estimação de estado a partir de medições que contêm ruído, um dos algoritmos mais clássicos na literatura é o Filtro de Kalman (JOSHI et al., 2018). A próxima seção detalhará o funcionamento e os princípios deste método.

Figura 2.14 – Rastreamento por detecção.



Fonte: Kalake, Wan e Hou (2021).

2.4.1 Filtro de Kalman

O Filtro de Kalman, desenvolvido por Rudolf Kálmán em 1960, é um algoritmo de estimação de estado que se tornou uma ferramenta presente em diversas áreas, desde o controle de espaçonaves até a robótica e o processamento de sinais (JWO; BISWAL, 2023). Originalmente, ele é um algoritmo recursivo de processamento de dados que combina as medições atuais disponíveis, mesmo que ruidosas e imprecisas, com o conhecimento prévio sobre o sistema para produzir uma estimativa otimizada das variáveis de interesse, minimizando o erro quadrático médio. Uma de suas vantagens é a eficiência computacional e os baixos custos de memória, pois ele não precisa armazenar todo o histórico de medições, dependendo apenas do estado anterior para calcular o estado atual (PEI et al., 2019).

O propósito fundamental dessa técnica é estimar o estado de um sistema que não é estático. No contexto da visão computacional e do rastreamento, o estado de um objeto pode ser um vetor contendo sua posição, velocidade e aceleração (JOSHI et al., 2018). O Filtro de Kalman tem a capacidade de lidar com a incerteza, que está presente tanto no modelo do sistema, pequenas variações no movimento que não podem ser previstas, quanto nas medições, ruído do sensor ou imprecisões do detector (CHEN, 2012). Por exemplo, ao rastrear um lagarto, a posição da sua cabeça detectada pelo YOLO em cada quadro pode variar ligeiramente, mesmo que o animal esteja parado. O Filtro de Kalman utiliza essas medições ruidosas para produzir uma estimativa da posição que tende a ser mais próxima da real.

O funcionamento do Filtro de Kalman é um ciclo recursivo dividido em duas fases: a predição e a atualização (RAMASUBRAMANIAN; RANGASWAMY; REDDY, 2014). A fase de predição consiste na utilização do estado estimado do passo anterior e de um modelo de movimento, por exemplo, assumindo uma velocidade constante, para prever qual será o estado no momento atual. Junto com essa previsão, o filtro também propaga a incerteza associada a essa estimativa, representada por uma matriz de covariância (MONTELLA, 2011). Já na fase de

atualização, uma nova medição do sistema é obtida, por exemplo, no contexto desta monografia, uma nova posição da cabeça do lagarto é detectada. Esta medição, que também possui sua própria incerteza, é usada para corrigir a estimativa feita na fase de predição. O filtro, basicamente, dá mais peso ao que for mais confiável entre a medição e a previsão, calculando um valor que pondera a incerteza de ambas. O resultado é uma nova estimativa, que é mais precisa do que a medição ou a previsão sozinhas, e será utilizada como ponto de partida para a fase de predição do próximo ciclo.

O filtro de Kalman, a *priori*, é um estimador linear que assume que o ruído do sistema e da medição seguem uma distribuição Gaussiana (MONTELLA, 2011). Porém, para sistemas com movimentos não lineares, comuns em robótica e análise de comportamento, são utilizadas extensões como o *Extended Kalman Filter* (EKF) e o *Unscented Kalman Filter* (UKF), que linearizam ou aproximam as funções não lineares para aplicar os mesmos princípios de predição e correção (CHEN, 2012).

2.4.2 Aplicações de Rastreamento Híbrido (YOLO + Filtro de Kalman)

Para uma detecção mais precisa, aprimorando o rastreamento e a estabilidade da análise de movimento, o detector YOLO é integrado com o algoritmo de rastreamento de Filtro de Kalman. Nesta subseção, serão apresentados estudos que realizaram essa combinação.

No âmbito de trânsito, os rastreadores vêm cada vez mais sendo utilizados, em específico o Filtro de Kalman. No que diz respeito a isso, em Azimjonov e Özmen (2021) foi proposta uma abordagem utilizando um detector de objeto e um rastreador trabalhando em conjunto com o objetivo de monitorar e estimar o fluxo de tráfego em rodovias. Para o detector, a combinação do modelo YOLO com um classificador CNN secundário obteve os melhores resultados, atingindo uma acurácia de 95.45% e superando, assim, as limitações na tarefa de classificação de veículos enfrentadas pelo YOLO. Já para o rastreador, foram implementadas e comparadas duas abordagens, uma utilizando o Filtro de Kalman e a outra utilizando um rastreador baseado em *Bounding Box*. Apesar de resultados muito similares, ambos obtendo acurácias médias acima de 90% em todos os testes utilizando o YOLO combinado com a CNN, o rastreador baseado em *bounding box* apresentou um resultado melhor, considerando a queda de desempenho do Filtro de Kalman no momento em que o número de objetos vai aumentando na imagem. Entretanto, no contexto de análise de movimento de lagartos, com um número muito menor de objetos na imagem quando comparado com um cenário de trânsito, seu desempenho satisfatório tende a se manter.

Visando abordar o desafio de localização de alvos para robôs móveis, que sofrem com a perda de informação de profundidade ao usar visão monocular, Wei (2024) propôs um sistema que integrava o modelo YOLOv5s com um Filtro de Kalman aprimorado, com o objetivo de aumentar a precisão da localização ao fundir dados de um câmera com medição de um lidar, um radar a laser. Ao invés de tratar todas as medições de forma igual para o Filtro de Kalman, o algoritmo atribui pesos de confiança diferentes a cada observação com base em sua distância

e direção, usando esses pesos para ajustar dinamicamente os parâmetros de ruído do filtro e otimizar a estimativa da posição. Para validação, foram realizados experimentos em um robô real que compararam o método proposto com o Filtro de Kalman tradicional e a filtragem por média simples. Os resultados demonstraram que o algoritmo aprimorado obteve uma vantagem na precisão da localização, mantendo a razão entre o erro e a distância abaixo de 1% em testes de até 7 metros, o que evidencia sua estabilidade e superioridade. É importante pontuar que este estudo reforça a eficácia da combinação YOLO com Filtro de Kalman, mas destaca que a performance do sistema é maximizada quando o filtro é adaptado.

No contexto de esporte, visando aprimorar o desempenho de jogadores de basquete através da análise do movimento desses atletas durante o treinamento para obter parâmetros como aceleração, velocidade e tempo de reação dos jogadores. Para isso, Chen (2024) propôs um modelo dividido em múltiplos estágios, iniciando com um pré-processamento que utiliza um *Median Filter* (MF) e um *Savitzky-Golay Filter* (SGF) para redução de ruído. Em seguida, o sistema emprega o detector YOLOv8 para identificar os jogadores , o modelo *DeepLabV3+* para realizar a estimação de pose e, por fim, utiliza o Filtro de Kalman para o rastreamento da pose ao longo do tempo. O YOLO foi escolhido pelo autor como um modelo robusto para a detecção dos jogadores e o Filtro de Kalman foi justificado como o modelo mais adequado para estimar o movimento dinâmico desses jogadores e prever estados futuros com base em estimativas passadas. Os experimentos, realizados sobre o *dataset Basketball Action Dataset*, compararam o modelo proposto com outras abordagens como CNN, *Bidirectional Recurrent Neural Network* (Bi-RNN) e *fuzzy Long Short-Term Memory* (LSTM). Os resultados indicam que o sistema proposto alcançou uma acurácia superior de 98,71%, superando todos os outros métodos testados.

Ainda nos cenários esportivos, focando no reconhecimento de gestos, em (WU; FAN, 2024) foi proposto um modelo que combinou o detector YOLOv5 e o Filtro de Kalman para aumentar a acurácia, estabilidade e o desempenho em tempo real na análise de posturas. A abordagem utilizou o YOLOv5, com uma estrutura *backbone* mais leve para uma maior eficiência, para detectar os gestos humanos, enquanto o Filtro de Kalman foi empregado para pré-processar e otimizar os dados de pose em uma série temporal, rastreando as trajetórias para corrigir ruídos e desfoques de movimento. Para os experimentos, foi utilizado um *dataset* de gestos de tênis e o modelo proposto foi comparado com as abordagens de Filtro de Kalman e YOLO isoladas. O modelo combinado obteve os melhores resultados, alcançando uma acurácia de aproximadamente 93%.

Diante desses trabalhos, é visível a importância da combinação do Filtro de Kalman no uso do modelo de detecção YOLO no que diz respeito a estabilidade, precisão e robustez no rastreamento de alvos em sequências de vídeo. O filtro atua como um otimizador da trajetória temporal, suavizando as instabilidades e corrigindo ruídos inerentes às detecções quadro a quadro do YOLO. Essa capacidade de prever e corrigir a posição de um objeto torna a extração de

dados de movimento mais confiável. Entretanto, como visto em Azimjonov e Özmen (2021), essa combinação depende do contexto e das características do problema para dar certo, o que justifica a avaliação desta abordagem como um aprimoramento em relação ao uso do detector de forma isolada no âmbito da análise de comportamento animal, como proposto neste trabalho.

3 Desenvolvimento

Este capítulo apresenta a metodologia empregada para atingir os objetivos propostos nesta monografia, detalhando as etapas do processo de desenvolvimento e avaliação experimental. Inicialmente, a seção 3.1 descreve a origem e a composição do conjunto de vídeos de lagartos do gênero *Tropidurus*, bem como o pré-processamento e a anotação das imagens para a construção da base de dados. Em seguida, a seção 3.2 detalha as arquiteturas utilizadas, com foco no detector de objetos YOLOv12 e na sua integração com algoritmos de rastreamento baseados no Filtro de Kalman. Nesta etapa, também são apresentados o processo de otimização de hiperparâmetros e a configuração final dos modelos. A seção 3.3 estabelece as métricas de desempenho empregadas para quantificar a eficácia das soluções propostas. Por fim, a seção 3.4 descreve o ambiente computacional, incluindo hardware e software, utilizado para a execução de todos os experimentos.

3.1 Base de Dados

Similar ao que foi realizado em Zenóbio et al. (2023), este trabalho utiliza o conjunto de vídeos construído e fornecido por Passos (2016), em sua tese de doutorado. O acervo é composto por 46 vídeos, sendo 44 utilizados neste trabalho, que registram lagartos do gênero *Tropidurus* em seu *habitat* natural. O objetivo original da coleta foi investigar a organização social e a comunicação visual dessas espécies, com ênfase nos sinais emitidos através de movimentos de cabeça (*headbobs*) e flexões corporais (*push-ups*). A coleta de dados concentrou-se em três espécies do grupo *Tropidurus semitaeniatus*: os *Tropidurus helenae*, filmados na Serra da Capivara, no município de Coronel José Dias, no Piauí; os *Tropidurus jaguaribanus*, filmados no vale do rio Jaguaribe, no município de São João do Jaguaribe, Ceará; e os *Tropidurus pinima*, filmados no Morro de Santo Inácio, no município de Gentio do Ouro, na Bahia.

A captura dos vídeos foi executada a partir de câmeras de alta definição com zoom óptico de no mínimo 30x, montadas sobre tripés para garantir a estabilidade. Todos os vídeos foram gravados a uma taxa de 30 quadros por segundo (fps). Além disso, as gravações foram realizadas a uma distância controlada de 5 a 10 metros dos animais, um procedimento adotado para minimizar a interferência do observador nos comportamentos naturais dos lagartos. Na Figura 3.1 é apresentado um quadro retirado de um dos vídeos gravados.

Para a construção da base de dados utilizada nos experimentos desta monografia, foi realizado um pré-processamento rigoroso na base original de vídeos, seguindo as seguintes etapas:

1. **Análise Temporal:** Foi conduzido um estudo do tempo de duração dos vídeos originais,

Figura 3.1 – Quadro extraído de um dos vídeos da base de dados.

Fonte: Retirada da base de dados de Passos (2016)

gerando dados estatísticos para auxiliar na seleção equilibrada das amostras.

2. **Filtragem:** Três vídeos foram descartados por apresentarem conteúdo praticamente idêntico a outros arquivos da base, restando 41 vídeos.

3. **Seleção para Rastreamento:** Dentre os vídeos restantes, três foram selecionados exclusivamente para testar a etapa de rastreamento. Para estes, todos os quadros foram rotulados (75, 81 e 84 quadros, respectivamente), totalizando 240 quadros para a validação do *tracking*.

4. **Divisão para Detecção:** Os 38 vídeos restantes foram destinados ao treinamento e avaliação do detector YOLO. Estes foram divididos em subconjuntos de treinamento (75% - 26 vídeos), validação (15% - 6 vídeos) e teste (15% - 6 vídeos). A distribuição dos vídeos entre os subconjuntos buscou equilibrar o tempo de duração deles, a variação de poses dos animais e as características visuais dos quadros.

5. **Extração de Quadros:** Por fim, desses 38 vídeos, foram extraídos aproximadamente 13 quadros por vídeo, selecionados de forma a maximizar a variância visual, pegando quadros espaçados por um intervalo de tempo.

Uma vez que as abordagens utilizando o YOLO e a combinação do YOLO com Filtro de Kalman aprendem a partir de um treinamento supervisionado, após ter sido realizada a extração dos quadros como descrito anteriormente, ocorreu uma rotulação dos mesmos. Essa rotulação será descrita na subseção a seguir.

3.1.1 Anotação das imagens

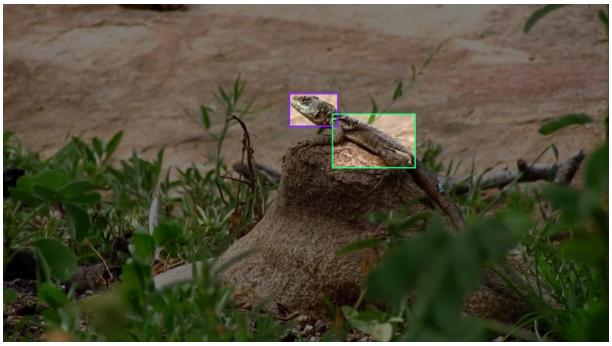
Para viabilizar o aprendizado supervisionado e o *fine-tuning* do modelo YOLO, a rotulação das imagens foi conduzida manualmente por meio da plataforma *Roboflow*[1].

Nesta etapa, definiu-se um critério específico para a demarcação das regiões de interesse: além da cabeça do animal, diferente de Zenóbio et al. (2023), optou-se por rotular o corpo do

[1] Disponível em: <https://roboflow.com/>. Acesso em: 14 jan. 2026.

lagarto excluindo a cauda. Essa decisão estratégica visa aumentar a precisão no estudo da variação angular entre a cabeça e o tronco do animal, dado que a posição da cauda pode introduzir ruído na identificação dos movimentos de *head-bobs* e *push-ups* se movimentando sozinha, o que consequentemente alteraria a posição da *bounding box* do corpo. A Figura 3.2 ilustra um exemplo dessa anotação.

Figura 3.2 – Imagem rotulada manualmente.



Elaborado pelo autor.

O formato de anotação segue o padrão YOLO, exigindo um arquivo de texto (.txt) para cada imagem, onde cada linha contém:

```
class_id   x_center   y_center   width   height
```

onde as coordenadas e dimensões da *bounding box* são normalizadas em relação à imagem e:

- **class_id**: É a classe do objeto identificado, representado por um número inteiro (por exemplo, 0 para "cabeça"e 1 para "corpo").
- **x_center**: É a coordenada do centro da *bounding box* no eixo horizontal normalizada em relação à imagem.
- **y_center**: É a coordenada do centro da *bounding box* no eixo vertical normalizada em relação à imagem.
- **width**: É a dimensão de largura da *bounding box* normalizada em relação à imagem.
- **height**: É a dimensão de altura da *bounding box* normalizada em relação à imagem.

3.1.2 Configuração dos Conjuntos de Dados Experimentais

Posterior à extração de quadros dos vídeos selecionados para cada subconjunto, treino, validação e teste, e dos vídeos selecionados para a avaliação do *tracking*, com a anotação manual tendo sido realizada para todos eles, foi gerado um *dataset* final utilizado nos experimentos, com o objetivo de avaliar a robustez e a qualidade dos modelos, tornando possível compará-los.

O *dataset* utilizado para experimentação e comparação dos modelos de detecção YOLOv5 e YOLOv12 seguiu a proporção de 75%, 15% e 15% respectivamente para treino, validação e teste. Além disso, visando aumentar a variabilidade e a capacidade de generalização dos modelos, foram aplicadas as seguintes técnicas de *data augmentation*, totalizando ao final em 987 imagens para o treino, 76 imagens para a validação e 74 imagens para o teste:

- ***Flip*:** Horizontal.
- **Rotação:** Variação aleatória entre -15° e +15°.
- **Brilho:** Variação aleatória entre -25% e +25%.

Finalmente, para a etapa de experimentação envolvendo modelos de *tracking*, foi criado um subconjunto para cada um dos três vídeos separados para essa fase, cada um contendo todos os quadros rotulados do vídeo em questão. Esses conjuntos foram utilizados para a realização dos testes e comparação dos modelos YOLO com e sem o Filtro de Kalman.

3.2 Modelos de *Deep Learning* e *Tracking*

Para os experimentos, responsável pela detecção da cabeça e do corpo dos lagartos nos quadros de vídeo, o modelo de *deep learning* empregado foi a versão 12 do YOLO. A escolha deste modelo se deve ao seu desempenho de ponta na detecção de objetos em tempo real, que introduz avanços significativos em precisão e eficiência computacional em relação às suas versões anteriores. Uma vantagem crucial do YOLOv12 para este trabalho está na incorporação do mecanismo de atenção denominado "*Area Attention*", que permite ao modelo focar em regiões de interesse com maior eficácia, uma característica particularmente útil para a detecção de alvos pequenos, parcialmente ocluídos ou que se camuflam com o ambiente, desafios esses que são comuns na identificação de lagartos em seus habitats naturais. Além disso, a arquitetura do modelo se beneficia das redes R-ELAN (*Residual Efficient Layer Aggregation Networks*), que aprimoram a capacidade de aprender características mais robustas e detalhadas.

A fase de treinamento do modelo foi realizada por meio de transferência de aprendizado. O YOLOv12, pré-treinado em um grande conjunto de dados, passou por um processo de *fine-tuning* para especializá-lo no contexto específico da detecção de lagartos. Optou-se por não treinar o modelo do zero (*from scratch*), já que essa abordagem seria inviável, uma vez que demandaria recursos computacionais massivos e descartaria o conhecimento prévio que o modelo já possui para identificar características fundamentais como bordas, texturas e formas.

Para a quantificação do desempenho do modelo de detecção utilizado por Zenóbio et al. (2023), o modelo YOLOv5 também foi avaliado sob as mesmas condições experimentais.

Já o segundo modelo deste trabalho que integra detecção de objetos baseada em *deep learning* com um algoritmo de *tracking*, foi composto pelo YOLOv12 integrado ao Filtro de

Kalman. Para a finalidade de extração da trajetória de um objeto ao longo do vídeo, a função desse filtro é suavizar as medições ruidosas provenientes do detector e manter uma estimativa de posição mesmo em quadros onde a detecção falha. O Filtro de Kalman é um estimador clássico para essa tarefa, operando em um ciclo de predição e correção em que, a cada passo de tempo, ele prevê o próximo estado do objeto com base em seu estado anterior e, em seguida, atualiza essa previsão ponderando-a com a nova medição fornecida pelo YOLOv12, com base na confiança relativa de cada um.

Para essa etapa de rastreamento, optou-se pela utilização de algoritmos de estado da arte já integrados ao ecossistema do *Ultralytics*: o *BoT-SORT* e o *ByteTrack*. Ambos os algoritmos fundamentam-se no Filtro de Kalman para a estimativa de estado e predição de movimento, mas incorporam estratégias robustas de associação de dados para lidar com desafios comuns em vídeos de animais, como oclusões momentâneas e movimentos não-lineares abruptos da cabeça dos lagartos. O *ByteTrack* destaca-se por aproveitar detecções de baixa confiança, recuperando objetos que seriam descartados por limiares rígidos, enquanto o *BoT-SORT* combina informações de movimento da câmera e descritores de aparência para aprimorar a consistência da identidade do alvo ao longo do tempo. A adoção dessas ferramentas permite unir a precisão de detecção do YOLOv12 com mecanismos de rastreamento altamente otimizados e validados na literatura.

3.2.1 Otimização de Hiperparâmetros

Visando extrair o máximo desempenho dos modelos e garantir uma comparação justa entre os cenários, foi conduzida uma etapa de otimização de hiperparâmetros (*hyperparameter tuning*) utilizando o *framework* Optuna. O espaço de busca configurado abrangeu parâmetros críticos como a taxa de aprendizado inicial (*lr0*), *momentum*, decaimento de peso (*weight_decay*), ganhos das funções de perda de caixa (*box*) e de classificação (*cls*), a escolha do otimizador (entre SGD e AdamW) e a taxa de *dropout*. Para a busca, foram executadas 40 tentativas para cada conjunto de dados, onde cada tentativa consistiu em um treinamento parcial de 25 épocas. A função objetivo estabelecida para o algoritmo de otimização foi a maximização da métrica *Mean Average Precision 50-95* (mAP50-95) no conjunto de validação, selecionando-se, ao final, a combinação de hiperparâmetros que resultou na maior precisão de localização e classificação.

A definição dos tipos de busca para cada hiperparâmetro se baseia na natureza de cada variável. Para a taxa de aprendizado inicial e o decaimento de peso, adotou-se a distribuição log-uniforme, abordagem essencial quando o intervalo de busca abrange múltiplas ordens de magnitude, permitindo que o algoritmo explore com a mesma densidade tanto valores muito pequenos (por exemplo, 10^{-5}) quanto valores maiores (por exemplo, 10^{-2}). Para parâmetros que possuem intervalos limitados e lineares, como o *momentum*, a taxa de *dropout* e os ganhos das funções de perda, optou-se pela distribuição uniforme, garantindo uma amostragem homogênea dentro dos limites estabelecidos. Por fim, o otimizador foi tratado como uma variável categórica, permitindo a seleção direta entre algoritmos distintos (SGD e AdamW) para verificar qual

Tabela 3.1 – Espaço de busca de hiperparâmetros definido para a otimização com Optuna.

Hiperparâmetro	**Tipo de Busca**	**Intervalo / Valores**
Taxa de aprendizado inicial (*lr0*)	Log-uniforme	$[1 \times 10^{-5}, 1 \times 10^{-1}]$
Momentum	Uniforme	$[0,6, 0,99]$
Decaimento de peso (*weight_decay*)	Log-uniforme	$[1 \times 10^{-5}, 1 \times 10^{-3}]$
Ganho da perda de caixa (*box*)	Uniforme	$[0,02, 0,2]$
Ganho da perda de classificação (*cls*)	Uniforme	$[0,2, 4,0]$
Otimizador	Categórico	{SGD, AdamW}
Dropout	Uniforme	$[0,0, 0,25]$

Fonte: elaborado pelo autor.

dinâmica de atualização de pesos melhor se adapta ao problema.

3.2.2 Hiperparâmetros Finais

Após a execução do processo de otimização descrito na subseção anterior, obteve-se um conjunto de hiperparâmetros que maximizou a métrica mAP50-95 no conjunto de validação. A Tabela 3.2 apresenta os valores finais selecionados para a taxa de aprendizado, *momentum*, decaimento de peso, ganhos das funções de perda, otimizador e taxa de *dropout*.

É importante ressaltar que, visando garantir uma comparação justa e isolar a arquitetura da rede como a variável principal de análise, estes mesmos hiperparâmetros otimizados foram aplicados tanto no treinamento do modelo proposto (YOLOv12) quanto no treinamento do modelo *baseline* (YOLOv5). Dessa forma, é garantido que eventuais diferenças de desempenho sejam atribuídas às capacidades intrínsecas de cada modelo e não a uma configuração de treinamento desigual.

Tabela 3.2 – Hiperparâmetros finais obtidos via otimização com Optuna e utilizados nos experimentos.

Hiperparâmetro	**Valor Selecionado**
Taxa de aprendizado inicial (lr0)	$6,62 \times 10^{-5}$
Momentum	0,901
Decaimento de peso (weight_decay)	$6,37 \times 10^{-4}$
Ganho da perda de caixa (box)	0,119
Ganho da perda de classificação (cls)	0,872
Taxa de Dropout	0,216
Otimizador	AdamW

Fonte: elaborado pelo autor.

3.2.3 Extração de Sinais e Cálculo da Angulação na Etapa de Rastreamento

Após a etapa de rastreamento, que assegura a identidade e a consistência temporal das *bounding boxes*, as coordenadas espaciais obtidas são processadas para gerar os sinais comportamentais de interesse. Para traduzir a movimentação do animal em uma métrica relevante

e invariante à rotação da câmera, adotou-se a abordagem geométrica utilizada por Zenóbio et al. (2023), fundamentada em (SPONG; HUTCHINSON; VIDYASAGAR, 2005).

A métrica principal extraída é a angulação relativa (θ) entre a cabeça e o corpo do lagarto. O cálculo utiliza três pontos de referência no plano da imagem a cada quadro:

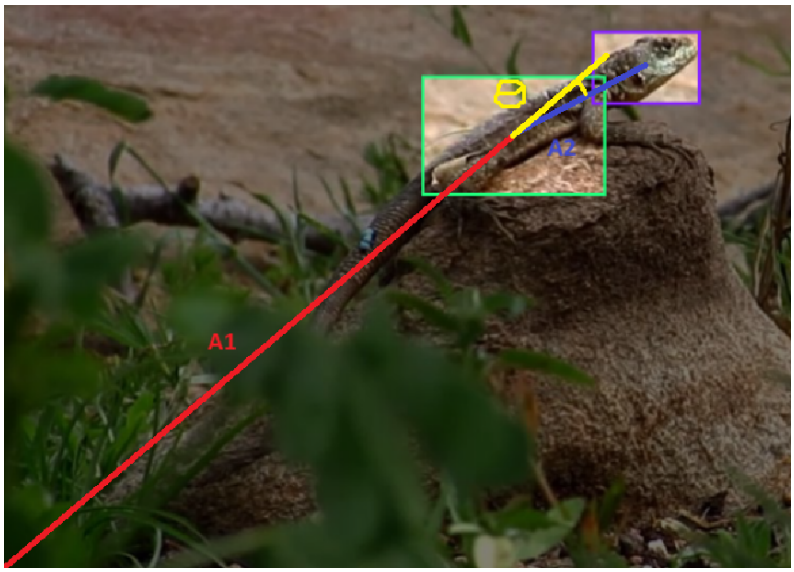
- A origem do sistema de coordenadas (canto inferior esquerdo do vídeo);
- O centro da *bounding box* do corpo;
- O centro da *bounding box* da cabeça.

Sejam a_1 a distância euclidiana entre a origem e o centro do corpo, a_2 a distância entre o centro do corpo e o centro da cabeça e x e y as coordenadas do centro da cabeça. A variável auxiliar D e o ângulo θ são calculados conforme as Equações 3.1 e 3.2:

$$D = \frac{x^2 + y^2 - a_1^2 - a_2^2}{2 \cdot a_1 \cdot a_2} \tag{3.1}$$

$$\theta = \arctan\left(\frac{\sqrt{1 - D^2}}{D}\right) \tag{3.2}$$

Figura 3.3 – Cálculo da angulação relativa (θ) entre a cabeça e o corpo do lagarto.



Fonte: Elaborado pelo autor.

É importante ressaltar que, diferentemente do trabalho original de Zenóbio et al. (2023), que aplicava um limiar de movimentação mínima na cabeça do lagarto (1 pixel) para filtrar ruídos estáticos, neste trabalho optou-se por não utilizar filtros de pós-processamento. Essa decisão tem como objetivo permitir a avaliação da estabilidade bruta fornecida pelos algoritmos de rastreamento e pelo detector YOLOv12, expondo qualquer instabilidade presente no sinal gerado.

Dessa forma, a série temporal resultante representa a variação angular direta quadro a quadro, servindo como base para a análise comparativa da suavidade das trajetórias apresentada no Capítulo 4.

3.3 Avaliação dos Modelos

Para quantificar o desempenho dos modelos ao longo dos experimentos, foram adotadas métricas específicas, descritas na subseção a seguir, para as duas etapas centrais deste trabalho: a detecção e o rastreamento. A escolha deste conjunto de métricas visa fornecer uma análise completa e rigorosa da performance em cada fase das soluções propostas.

3.3.1 Métricas de Avaliação

As métricas permitem que os modelos desenvolvidos sejam avaliados quantitativamente de acordo com seus desempenhos. Neste trabalho, que se concentra na detecção e posterior rastreamento da cabeça de um lagarto em vídeo, foram selecionadas métricas específicas para cada uma dessas etapas. Para a fase de detecção, as métricas utilizadas são a Precisão, Revocação, *Mean Average Precision 50* (mAP50) e o mAP50-95. Já para a fase de rastreamento, as métricas são a Interseção sobre União (IoU) Média ao longo dos quadros e a Distância Euclidiana Média entre Centros.

3.3.1.1 Precisão e Revocação

As métricas de Precisão e Revocação (*Recall*) são utilizadas para avaliar a qualidade das detecções. A classificação de uma predição como correta ou incorreta depende fundamentalmente do critério de IoU, que mede a sobreposição entre a *bounding box* predita e a real. Uma detecção é considerada um Verdadeiro Positivo (VP) se sua IoU com a anotação real correspondente for superior a um limiar pré-definido, caso contrário, é classificada como um Falso Positivo (FP). A ausência de uma detecção para um objeto real existente é contabilizada como um Falso Negativo (FN).

- **Precisão:** Mede a proporção de detecções classificadas como positivas que são de fato corretas. Representa a acurácia das predições positivas do modelo e é definida pela equação 3.3. Seus valores variam no intervalo de $[0, 1]$, onde 1 representa a precisão perfeita (nenhum falso positivo) e 0 o pior caso.

$$\text{Precisão} = \frac{VP}{VP + FP} \tag{3.3}$$

- **Revocação:** Mede a proporção de objetos reais que foram corretamente identificados pelo modelo. Indica a capacidade do modelo de encontrar todas as instâncias relevantes e é

calculada pela equação 3.4. Assim como a Precisão, seus valores variam em $[0, 1]$, onde 1 indica a revocação perfeita (nenhum falso negativo) e 0 o pior caso.

$$\text{Revocação} = \frac{VP}{VP + FN} \tag{3.4}$$

3.3.1.2 Interseção sobre União (IoU)

A Interseção sobre União é uma métrica que avalia a sobreposição entre as *bounding boxes* predita pelo modelo (B_{pred}) e a real anotada (B_{real}). O valor da IoU é calculado como a razão entre a área da interseção e a área da união das duas caixas. Um valor de IoU igual a 1 indica uma sobreposição perfeita, enquanto um valor igual a 0 indica ausência de sobreposição. A métrica é definida pela equação 3.5.

$$\text{IoU} = \frac{\text{Área}(B_{pred} \cap B_{real})}{\text{Área}(B_{pred} \cup B_{real})} \tag{3.5}$$

3.3.1.3 *Mean Average Precision* 50 (mAP50)

O *Mean Average Precision* (mAP) é a principal métrica para avaliar o desempenho geral de modelos de detecção de objetos, sendo calculado a partir da curva de Precisão-Revocação. A variante mAP50 calcula a Precisão Média (AP) para cada classe utilizando um limiar fixo para a classificação de uma detecção como correta. O critério para esta classificação é baseado na IoU, exigindo uma sobreposição mínima de 50% (IoU > 0,5) entre a caixa predita e a real para que a detecção seja considerada um Verdadeiro Positivo. Como uma média de valores de precisão, seus resultados também variam no intervalo de $[0, 1]$, onde 1 representa o desempenho perfeito e 0 a pior performance possível.

3.3.1.4 *Mean Average Precision* 50-95 (mAP50-95)

Visando uma avaliação mais rigorosa da precisão na localização das *bounding boxes*, também foi utilizada a métrica mAP50-95. Diferentemente do mAP50, que utiliza um limiar de IoU único e mais permissivo, esta versão calcula a média dos valores de mAP em múltiplos limiares de IoU, variando de 0,5 a 0,95, com um incremento de 0,05. Dessa forma, a IoU é utilizada como base em dez diferentes níveis de exigência. Esta métrica penaliza fortemente modelos que, apesar de detectarem o objeto, não conseguem delimitar sua localização com alta precisão, fornecendo uma visão mais completa do desempenho do detector. Assim como as demais métricas baseadas em precisão, seus valores estão contidos no intervalo de $[0, 1]$, com 1 sendo o melhor resultado alcançável.

3.3.1.5 Interseção sobre União Média

Para avaliar a precisão do rastreamento ao longo dos quadros de um vídeo, a métrica de IoU é estendida. A IoU Média é calculada como a média aritmética dos valores de IoU entre a *bounding box* rastreada e a anotação real para cada quadro do vídeo. Sendo N o número total de quadros, B_{pred_t} a *bounding box* predita no quadro t e B_{real_t} a *bounding box* real no quadro t, a métrica é definida pela equação 3.6. O resultado desta métrica também pertence ao intervalo $[0, 1]$, onde 1 significa um rastreamento perfeito ao longo de todos os quadros e 0 indica uma falha total de sobreposição.

$$\text{IoU}_{\text{média}} = \frac{1}{N} \sum_{t=1}^{N} \frac{\text{Área}(B_{pred_t} \cap B_{real_t})}{\text{Área}(B_{pred_t} \cup B_{real_t})} \tag{3.6}$$

3.3.1.6 Distância Euclidiana Média entre Centros

A Distância Euclidiana Média entre Centros irá avaliar a precisão da localização do objeto rastreado. Ela identifica o erro de posicionamento medindo a distância espacial entre o ponto central da *bounding box* predita e o ponto central da *bounding box* real, calculando a média desses valores ao longo de todos os quadros do vídeo. Dessa forma, quanto maior a distância pior é o rastreamento.

Sejam (x_{pred_t}, y_{pred_t}) e (x_{real_t}, y_{real_t}) as coordenadas dos centros das *bounding boxes* predita e real, respectivamente, no quadro t, e N o número total de quadros. A métrica é definida pela equação 3.7.

$$\text{Distância Média} = \frac{1}{N} \sum_{t=1}^{N} \sqrt{(x_{pred_t} - x_{real_t})^2 + (y_{pred_t} - y_{real_t})^2} \tag{3.7}$$

3.4 Ambiente Experimental e Ferramentas

O experimento foi implementado em ambiente Python, com o auxílio das seguintes bibliotecas e frameworks:

- **Ultralytics 8.4.10:** Biblioteca principal que fornece a implementação da arquitetura YOLOv12 e dos algoritmos de rastreamento (*ByteTrack* e *BoT-SORT*), além das ferramentas para treinamento e inferência.
- **Python 3.11.11:** Linguagem de programação utilizada para a execução de todos os scripts.
- **Roboflow 1.2.13:** Ferramenta utilizada para a exportação do conjunto de dados para o formato compatível com o YOLO.

- **Optuna 4.7.0:** Framework utilizado para a automação e gerenciamento do processo de otimização de hiperparâmetros.
- **OpenCV 4.11.0.86:** Biblioteca de visão computacional utilizada para a leitura e processamento dos arquivos de vídeo e imagens.
- **NumPy 2.1.3:** Biblioteca fundamental para a computação científica, utilizada para manipulação de arrays e operações matriciais.
- **Matplotlib 3.10.3:** Biblioteca empregada para a geração de gráficos e visualização dos resultados, como as curvas de aprendizado e métricas de avaliação.

Todos os treinamentos, otimizações e testes foram realizados na infraestrutura do Laboratório CSI Lab da Universidade Federal de Ouro Preto. Foram utilizadas três máquinas de trabalho de alto desempenho, denominadas *Cisco*, *Nevasca* e *Maeve*, equipadas com a GPU NVIDIA RTX 3090 de 24 GB de VRAM, 128 GB de memória RAM e variando entre os processadores Intel Core i9-10900 e AMD Ryzen 3960X, garantindo a capacidade computacional necessária para o processamento massivo de dados e treinamento eficiente dos modelos.

4 Resultados

Este capítulo apresenta e discute os resultados experimentais obtidos, visando validar a eficácia da metodologia proposta neste trabalho para a análise automatizada do comportamento de lagartos. A apresentação dos dados está estruturada em duas etapas principais: avaliação da detecção, comparando-se o desempenho do modelo proposto, YOLOv12, com o *baseline*, YOLOv5; e avaliação do rastreamento, analisando a consistência temporal das trajetórias geradas e verificando o impacto da integração do Filtro de Kalman na suavização do movimento e manutenção da identidade do alvo.

4.1 Etapa de Detecção

Nesta seção, são apresentados os parâmetros de treinamento e a análise comparativa quantitativa e qualitativa entre as arquiteturas de detecção avaliadas.

4.1.1 Configuração Experimental

Para garantir a equidade na comparação entre o modelo *baseline* e a arquitetura proposta, ambos foram submetidos ao mesmo protocolo de treinamento e validação. Utilizou-se o conjunto de hiperparâmetros otimizados apresentados anteriormente na Tabela 3.2 (Capítulo 3).

Além dos hiperparâmetros de otimização, foram definidas configurações de execução padronizadas:

- **Resolução de Entrada:** As imagens foram mantidas na resolução original de entrada da rede de 640×640 pixels.
- ***Epochs*:** Definiu-se um total de 100 épocas para o treinamento.
- ***Early Stopping*:** Configurou-se um mecanismo de *patience* de 20 épocas, interrompendo o treinamento caso não houvesse melhoria na métrica de validação por 20 ciclos consecutivos, prevenindo o *overfitting* e otimizando o uso de recursos computacionais.

4.1.2 Análise dos resultados

Após a execução dos treinamentos e a avaliação no conjunto de teste, os resultados demonstraram uma superioridade consistente do modelo YOLOv12 em relação ao *baseline* em todas as métricas de precisão e recuperação avaliadas. A Tabela 4.1 sumariza estes dados comparativos.

Tabela 4.1 – Comparativo detalhado de desempenho e ganho percentual absoluto: YOLOv5 vs YOLOv12.

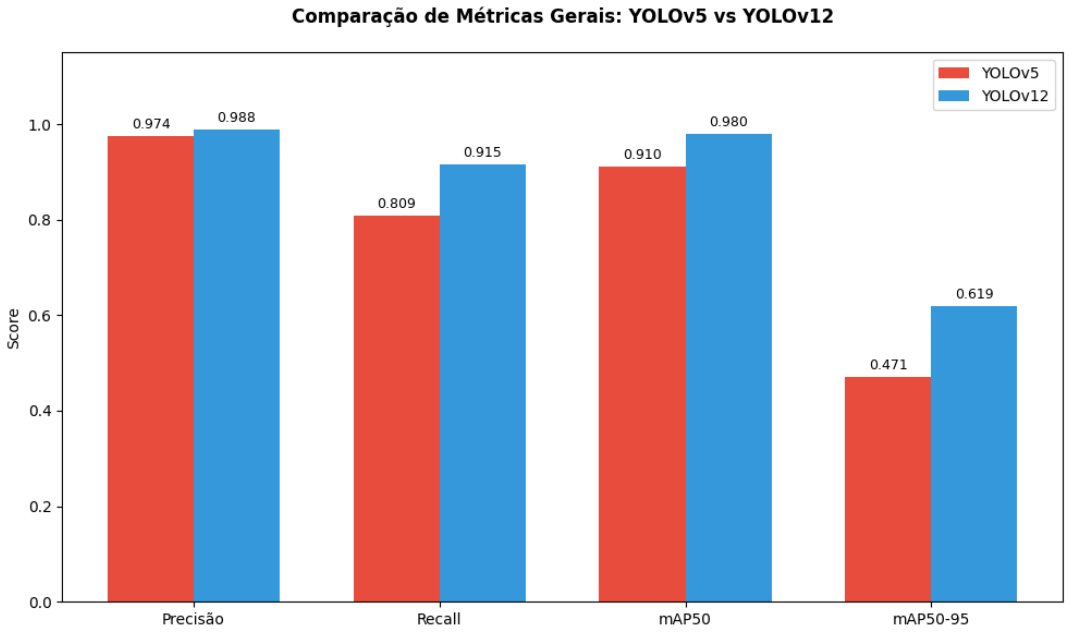
Classe	Modelo	Precisão	Revocação	mAP@50	mAP@50-95
Geral	YOLOv5	0,974	0,809	0,910	0,471
	YOLOv12	0,988	0,915	0,980	0,619
	Diferença	**+0,014**	**+0,106**	**+0,070**	**+0,148**
Head	YOLOv5	1,000	0,712	0,910	0,504
	YOLOv12	1,000	0,858	0,971	0,624
	Diferença	0,000	**+0,146**	**+0,061**	**+0,120**
Body	YOLOv5	0,949	0,905	0,910	0,437
	YOLOv12	0,977	0,973	0,990	0,615
	Diferença	**+0,028**	**+0,068**	**+0,080**	**+0,178**

Fonte: Elaborado pelo autor.

A análise da Tabela 4.1, complementada pela visualização gráfica na Figura 4.1, evidencia que o maior ganho proporcionado pelo YOLOv12 reside na métrica de Revocação, que saltou de 0,809 para 0,915 no cenário geral. Este aumento é crucial para a aplicação proposta, pois indica que o novo modelo falha significativamente menos em detectar o animal presente na cena, ou seja, tem uma redução de falsos negativos, garantindo uma coleta de dados comportamentais mais contínua.

Outro avanço notável observa-se na métrica mAP50-95, que avalia a qualidade do ajuste da *bounding box*. O YOLOv12 obteve 0,619 contra 0,471 do *baseline*, representando um ganho de aproximadamente 31%. Isso indica que as detecções do modelo proposto estão muito mais ajustadas ao contorno real do animal, o que é determinante para a precisão do centróide e, consequentemente, para reduzir ruídos no cálculo da angulação entre cabeça e corpo.

Figura 4.1 – Comparação de Métricas Gerais: YOLOv5 vs YOLOv12.



Fonte: Elaborado pelo autor.

Ao segmentar a análise por classes, observa-se que o ganho de desempenho não foi uniforme, sendo mais acentuado na classe "*Head*"(cabeça do lagarto), historicamente a mais desafiadora devido às suas dimensões reduzidas e maior mobilidade. Conforme ilustrado nas Figura 4.2 e Figura 4.3, enquanto a classe "*Body*" já apresentava bons índices no *baseline*, a classe "*Head*" teve um salto de revocação de 0,712 para 0,858.

Figura 4.2 – Métricas para a classe "Head".

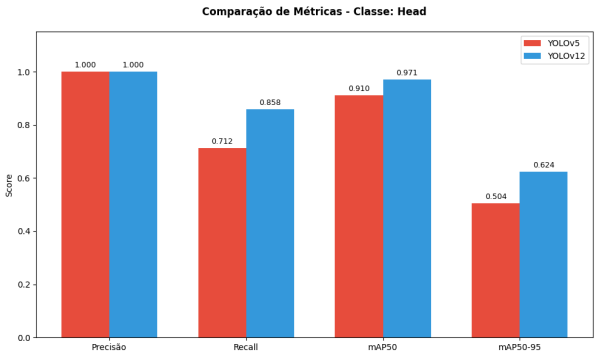
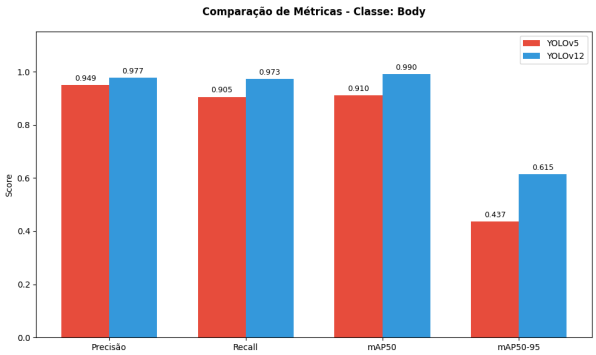


Figura 4.3 – Métricas para a classe "Body".



Fonte: Elaborado pelo autor.

Para compreender a natureza dos erros corrigidos pela nova arquitetura, foram analisadas as matrizes de confusão normalizadas apresentadas na Figura 4.4 e na Figura 4.5. No modelo YOLOv5 (Figura 4.4), nota-se que a classe "*Head*"apresenta uma taxa de acerto de 0,76, mas com 24% das instâncias sendo classificadas incorretamente como *background* (falsos negativos). Além disso, a linha referente ao *background* mostra que o modelo confundia o fundo com a cabeça em 24% dos casos (falsos positivos).

Em contrapartida, o YOLOv12 (Figura 4.5) reduziu a taxa de cabeças não detectadas para 16% e a confusão do fundo com a cabeça também para 16%. Esses dados corroboram a hipótese de que os mecanismos de atenção da arquitetura v12 são mais eficazes em discriminar pequenos objetos de interesse em meio à complexidade visual do habitat natural, reduzindo tanto a perda de informação quanto a geração de ruído ou alucinações.

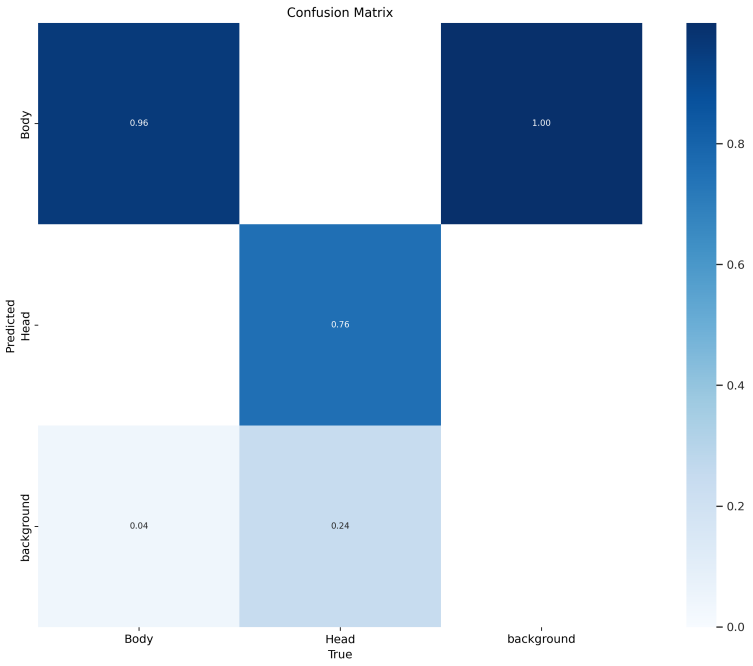
4.2 Etapa de Rastreamento

Após validar a eficácia do modelo YOLOv12 na detecção quadro a quadro, a segunda etapa dos experimentos consistiu na avaliação do desempenho do rastreamento temporal. O objetivo desta fase é garantir a consistência da identidade do animal ao longo do vídeo e a estabilidade das trajetórias geradas, requisitos fundamentais para a extração limpa dos sinais de comunicação (*head-bobs*).

Para esta análise, foram comparadas três abordagens:

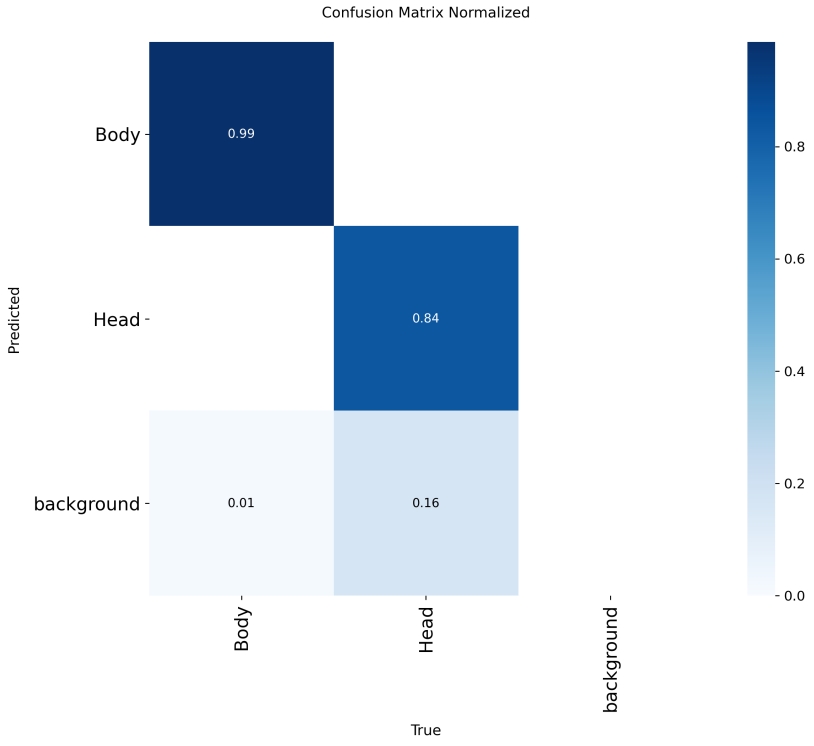
1. **YOLOv12 Puro:** Detecção quadro a quadro sem associação temporal.

Figura 4.4 – Matriz de Confusão: YOLOv5.



Fonte: Elaborado pelo autor.

Figura 4.5 – Matriz de Confusão: YOLOv12.



Fonte: Elaborado pelo autor.

2. **YOLOv12 + BoT-SORT:** Integração com rastreador que usa o Filtro de Kalman utilizando características de aparência e movimento.

3. **YOLOv12 + ByteTrack:** Integração com rastreador que usa o Filtro de Kalman baseado em associação de baixo limiar de confiança.

4.2.1 Análise Quantitativa dos Resultados

Os testes foram conduzidos em três vídeos selecionados especificamente para esta validação, totalizando 240 quadros rotulados manualmente. É importante destacar um resultado preliminar determinante: durante a execução desses vídeos de teste, o modelo YOLOv12 demonstrou uma boa robustez, atingindo uma taxa de detecção de 100%, não havendo ocorrência de quadros onde o animal (cabeça ou corpo) deixasse de ser detectado pelo modelo.

Devido a essa alta performance do detector, o papel dos algoritmos de rastreamento (*BoT-SORT* e *ByteTrack*) nestes experimentos concentrou-se não na recuperação de falhas de detecção, o que seria necessário se o YOLO falhasse, mas sim na estabilização das *bounding boxes* e na manutenção da identidade.

A Tabela 4.2 apresenta o resumo geral das métricas obtidas, calculadas como a média ponderada entre os três vídeos analisados.

Tabela 4.2 – Resultados comparativos das métricas de rastreamento (Média dos 3 vídeos).

Classe	Método	IoU Médio	Desvio Padrão IoU	Distância Média (px)
Body	YOLOv12 (Puro)	0,8390	0,0409	10,03
	+ BoT-SORT	0,8397	**0,0406**	**9,80**
	+ ByteTrack	**0,8409**	0,0421	9,83
Head	YOLOv12 (Puro)	**0,8976**	**0,0323**	**2,86**
	+ BoT-SORT	0,8939	0,0342	3,18
	+ ByteTrack	0,8906	0,0363	3,21

Fonte: Elaborada pelo autor.

A análise da Tabela 4.2 revela comportamentos distintos decorrentes da natureza dos movimentos de cada parte do animal. Para a classe "*Body*", observa-se que a aplicação dos rastreadores trouxe um pequeno ganho, porém positivo. O *ByteTrack* obteve o melhor IoU Médio (0,8409) e o *BoT-SORT* a menor Distância Euclidiana Média (9,80 px). Em termos de estabilidade, o *BoT-SORT* destacou-se com o menor desvio padrão de IoU (0,0406), indicando que ele foi capaz de manter a consistência das *bounding boxes* com menos flutuações quadro a quadro. Como o corpo do lagarto possui um movimento mais lento e linear, o modelo de predição do Filtro de Kalman consegue ajustar as caixas com eficiência, reduzindo tanto o erro de posicionamento quanto a variabilidade em relação à detecção pura.

Por outro lado, para a classe "*Head*", nota-se um fenômeno interessante onde o YOLOv12 puro apresentou métricas superiores aos métodos com rastreamento (IoU de 0,8976 contra 0,8939

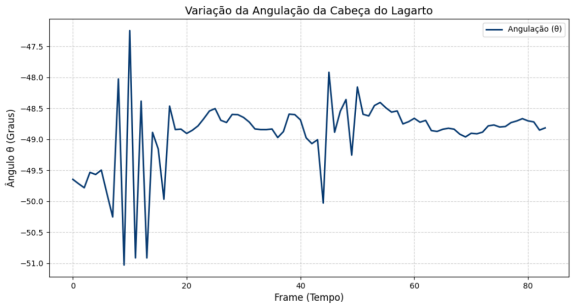
do *BoT-SORT*). Essa superioridade reflete-se também na estabilidade, visto que o detector puro obteve o menor desvio padrão (0,0323), enquanto a introdução dos rastreadores aumentou a dispersão dos valores de IoU. Isso ocorre devido à natureza do movimento de *head-bob*, que trata-se de um movimento rápido, de alta frequência e com mudanças bruscas de direção. O Filtro de Kalman, por padrão, assume um modelo de velocidade constante, tendendo a suavizar essas mudanças bruscas. Essa tentativa de suavização resulta em um leve "atraso" da caixa rastreada em relação ao movimento real da cabeça, o que acaba penalizando a métrica de IoU e introduzindo uma maior variância nos resultados em comparação à detecção direta.

4.2.2 Análise Qualitativa dos Resultados: Suavização do Sinal de Angulação

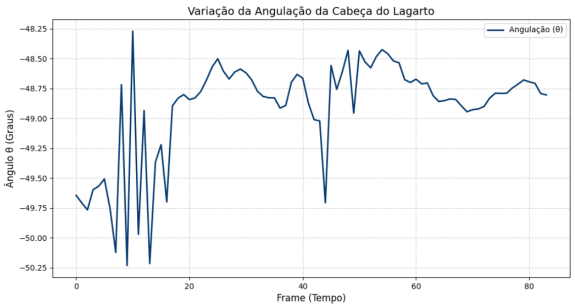
Embora as métricas de sobreposição (IoU) da cabeça tenham sido ligeiramente inferiores com o uso de rastreadores, a análise qualitativa dos sinais gerados justifica sua aplicação. O objetivo final do sistema não é apenas a detecção *pixel-a-pixel*, mas a geração de uma série temporal da angulação (θ) consistente para a análise comportamental final.

As Figuras 4.6, 4.7 e 4.8 apresentam as curvas de variação angular relativa extraídas dos três vídeos de teste.

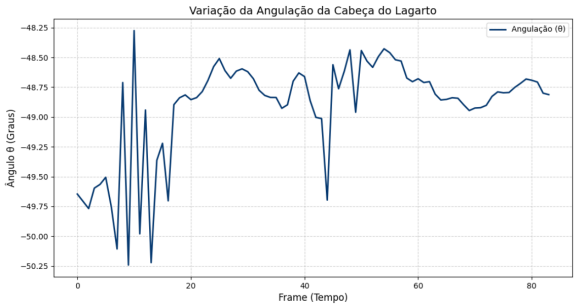
Figura 4.6 – Comparação da variação angular relativa no vídeo 1.



YOLOv12 Puro



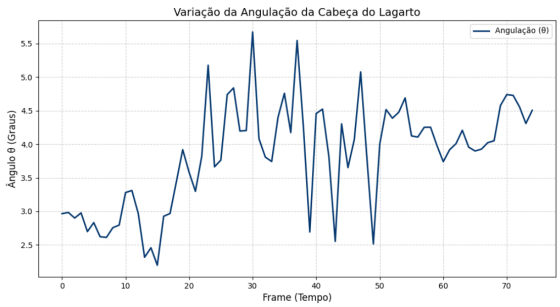
+ BoT-SORT



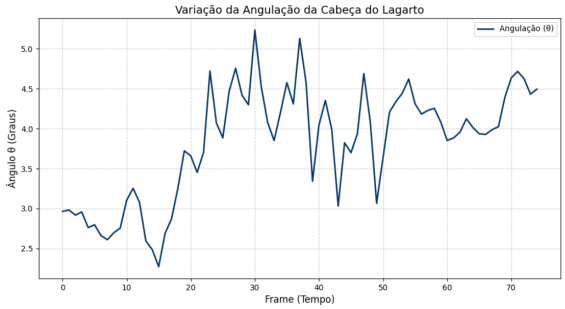
+ ByteTrack

Fonte: Elaborado pelo autor.

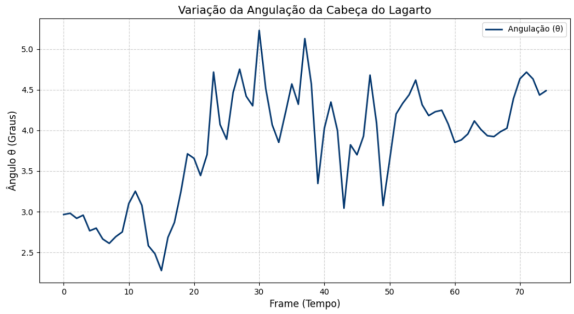
Figura 4.7 – Comparação da variação angular relativa no vídeo 2.



YOLOv12 Puro



+ BoT-SORT



+ ByteTrack

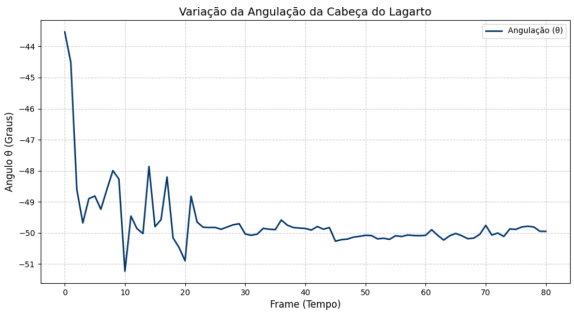
Fonte: Elaborado pelo autor.

Observando os gráficos, percebe-se que as curvas geradas pelos rastreadores, *BoT-SORT* e *ByteTrack*, tendem a apresentar um comportamento similar ao do detector puro, preservando a morfologia dos sinais de comunicação. Portanto, a leve redução na precisão espacial da cabeça, discutida na subseção anterior, não resultou em distorção significativa do sinal biológico.

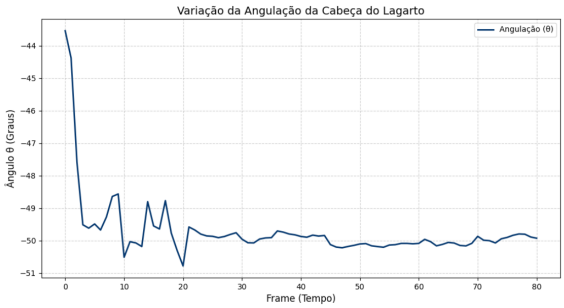
A principal contribuição da etapa de rastreamento, neste cenário de alta precisão do detector, reside na garantia da identidade, interessante principalmente em contextos em que mais de um objeto da mesma classe aparecem no mesmo momento. Enquanto o YOLOv12 puro trata cada quadro como um evento independente, o que poderia levar a trocas de identidade caso houvesse mais de um lagarto ou objetos similares na cena, os algoritmos *BoT-SORT* e *ByteTrack* mantêm um vínculo temporal. Isso assegura que o sinal extraído pertença ao mesmo indivíduo do início ao fim do vídeo, o que é indispensável para a automação da análise comportamental em larga escala.

Por fim, dentre os rastreadores avaliados, o *BoT-SORT* apresentou o menor erro de distância para a classe corpo (9,80 px) e métricas competitivas para a cabeça, demonstrando ser uma solução equilibrada para compor a arquitetura final do sistema.

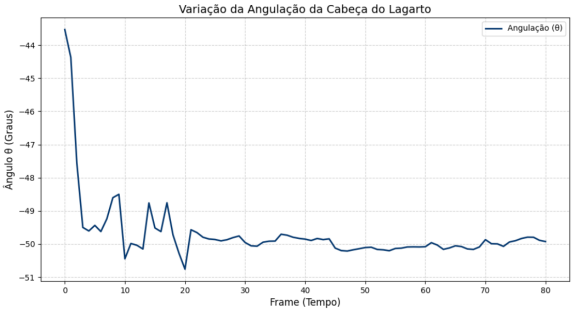
Figura 4.8 – Comparação da variação angular relativa no vídeo 3.



YOLOv12 Puro



+ BoT-SORT



+ ByteTrack

Fonte: Elaborado pelo autor.

5 Considerações Finais

O presente trabalho dedicou-se ao desenvolvimento e validação de uma abordagem computacional automatizada para a análise do comportamento de lagartos do gênero *Tropidurus*. A motivação central residiu na necessidade de superar as limitações dos métodos manuais de etologia e melhorar os resultados do método automatizado utilizado pelo software Reptilerecon (ZENÓBIO et al., 2023), fornecendo uma ferramenta capaz de operar em cenários naturais e não controlados, sendo possível a extração de sinais de comunicação visual com precisão e consistência.

5.1 Conclusões

Os experimentos realizados permitiram avaliar a evolução das arquiteturas de detecção de objetos e seu impacto direto na qualidade dos dados extraídos. A análise comparativa entre o modelo *baseline* (YOLOv5) e a arquitetura proposta (YOLOv12) evidenciou que a modernização do detector foi determinante para a robustez do sistema. O YOLOv12 demonstrou uma superioridade significativa, especialmente na métrica de revocação para a classe "*Head*"(cabeça do lagarto), que saltou de 0,809 para 0,915 no conjunto de teste. Esse ganho mitiga um dos maiores desafios da área, a perda de rastreamento de pequenos objetos em movimento rápido. Além disso, a melhoria na métrica mAP50-95, de 0,471 para 0,619, confirmou que as *bounding boxes* geradas pelo novo modelo estão mais ajustadas à anatomia do animal, resultando em centróides mais precisos para o cálculo da angulação.

Outro ponto de destaque foi a validação em ambiente real, uma vez que os vídeos utilizados nos experimentos foram capturados na natureza, apresentando desafios típicos como variação de iluminação, fundos complexos e camuflagem natural dos répteis. Mesmo diante dessas adversidades, o modelo YOLOv12 proposto atingiu uma taxa de detecção de 100% nos vídeos selecionados para a etapa de rastreamento, demonstrando que a solução é viável para aplicações de campo, fora de ambientes de laboratório controlados.

Na etapa de rastreamento, a integração dos algoritmos *BoT-SORT* e *ByteTrack* comprovou-se eficaz para a manutenção da identidade do animal, eliminando possíveis ambiguidades temporais. Embora tenha sido observado que o Filtro de Kalman tende a suavizar levemente os movimentos de alta frequência dos *head-bobs*, a consistência espacial garantida pelos rastreadores oferece um sinal limpo e contínuo. A estratégia adotada de rotular a classe "*Body*"excluindo a cauda mostrou-se, empiricamente, uma decisão acertada para evitar ruídos decorrentes de movimentos independentes do tronco, gerando séries temporais de angulação estáveis e aptas para análises futuras.

Conclui-se, portanto, que a arquitetura proposta atinge os objetivos do trabalho, entregando um *pipeline* de visão computacional robusto e validado quantitativamente, que serve como base sólida para a automação de estudos etológicos em larga escala.

5.2 Trabalhos Futuros

A complexidade do comportamento animal e os avanços contínuos na visão computacional abrem diversas frentes para aprimoramento e expansão deste trabalho. Sugere-se, para pesquisas futuras:

- **Análise de Interações Sociais:** Extensão do sistema para cenários com múltiplos lagartos no mesmo vídeo. Isso envolve o desafio de manter a identidade correta de cada indivíduo durante interações complexas e a análise de oclusões mútuas.
- **Estudo Comparativo de Morfologia (Com Cauda vs. Sem Cauda):** Neste trabalho, adotou-se a premissa de que excluir a cauda da *bounding box* do corpo reduz o ruído do sinal, visto que a cauda possui alta flexibilidade. Um trabalho futuro relevante seria treinar um modelo alternativo incluindo a cauda e comparar estatisticamente os sinais extraídos de ambos os modelos contra um *ground truth* manual, quantificando o impacto real dessa escolha metodológica na precisão da angulação.
- **Detecção por Pontos-Chave (*Pose Estimation*):** Substituição das *bounding boxes* por técnicas de estimativa de pose (como YOLO-Pose). A detecção de *keypoints* anatômicos específicos eliminaria o ruído causado pelo fundo dentro da caixa delimitadora e permitiria um cálculo geométrico da angulação muito mais preciso.
- **Exploração de Arquiteturas Baseadas em Attention:** Avaliação do uso de *Visual Transformers*. A capacidade desses modelos de capturar dependências globais na imagem pode oferecer maior robustez em cenários de camuflagem intensa, onde as redes convolucionais tradicionais podem ter dificuldade em distinguir a textura do animal da textura do ambiente.
- **Fusão de Métodos (*Ensemble*):** Implementação de uma abordagem de combinação de modelos, utilizando a média dos sinais extraídos por diferentes técnicas ou detectores para tornar o sistema tolerante a falhas pontuais de um algoritmo específico.

Referências

ALBAWI, S.; MOHAMMED, T. A.; AL-ZAWI, S. Understanding of a convolutional neural network. In: *2017 International Conference on Engineering and Technology (ICET)*. [S.l.: s.n.], 2017. p. 1–6.

AZIMJONOV, J.; ÖZMEN, A. A real-time vehicle detection and a novel vehicle tracking systems for estimating and monitoring traffic flow on highways. *Adv. Eng. Informatics*, v. 50, p. 101393, 2021. Disponível em: <https://doi.org/10.1016/j.aei.2021.101393>.

CHEN, C. Kinetic motion tracking of basketball training motion data capture and analysis using kalman filter and deeplabv3+. In: IEEE. *2024 International Conference on Distributed Systems, Computer Networks and Cybersecurity (ICDSCNC)*. [S.l.], 2024. p. 1–5.

CHEN, S. Y. Kalman filter for robot vision: A survey. *IEEE Transactions on Industrial Electronics*, v. 59, n. 11, p. 4409–4420, 2012.

DESHMUKH, P. S. Travel time prediction using neural networks: A literature review. In: *2018 International Conference on Information , Communication, Engineering and Technology (ICICET)*. [S.l.: s.n.], 2018. p. 1–5.

DIWAKAR, D.; RAJ, D. Recent object detection techniques: A survey. *International Journal of Image, Graphics and Signal Processing*, v. 14, p. 47–60, 04 2022.

ENGELL, M. Animal behavior, ninth edition. john alcock. *Integrative and Comparative Biology - INTEGR COMP BIOL*, v. 49, p. 608–609, 10 2009.

FAGER, P.; HANSON, R.; FASTH-BERGLUND Åsa; EKERED, S. Supervised and unsupervised learning in vision-guided robotic bin picking applications for mixed-model assembly. *Procedia CIRP*, v. 104, p. 1304–1309, 2021. ISSN 2212-8271. 54th CIRP CMS 2021 - Towards Digitalized Manufacturing 4.0. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2212827121011173>.

FELZENSZWALB, P. F.; GIRSHICK, R. B.; MCALLESTER, D.; RAMANAN, D. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 32, n. 9, p. 1627–1645, 2010.

FORMICA, V. A.; AUGAT, M. E.; BARNARD, M. E.; BUTTERFIELD, R. E.; WOOD, C. W.; BRODIE, E. D. Using home range estimates to construct social networks for species with indirect behavioral interactions. *Behavioral Ecology and Sociobiology*, v. 64, n. 7, p. 1199–1208, jul. 2010. ISSN 1432-0762. Disponível em: <https://doi.org/10.1007/s00265-010-0957-5>.

GIRSHICK, R. *Fast R-CNN*. 2015. Disponível em: <https://arxiv.org/abs/1504.08083>.

JEGHAM, N.; KOH, C. Y.; ABDELATTI, M.; HENDAWI, A. *YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions*. 2025. Disponível em: <https://arxiv.org/abs/2411.00201>.

JOSHI, R. C.; JOSHI, M.; SINGH, A. G.; MATHUR, S. Object detection, classification and tracking methods for video surveillance: A review. In: *2018 4th International Conference on Computing Communication and Automation (ICCCA)*. [S.l.: s.n.], 2018. p. 1–7.

JWO, D.-J.; BISWAL, A. Implementation and performance analysis of kalman filters with consistency validation. *Mathematics*, v. 11, n. 3, 2023. ISSN 2227-7390. Disponível em: <https://www.mdpi.com/2227-7390/11/3/521>.

KALAKE, L.; WAN, W.; HOU, L. Analysis based on recent deep learning approaches applied in real-time multi-object tracking: A review. *IEEE Access*, v. 9, p. 32650–32671, 2021.

KAPLAN, G. Animal communication. *WIREs Cognitive Science*, v. 5, n. 6, p. 661–677, 2014. Disponível em: <https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wcs.1321>.

KLINGLER, N. *Object tracking in computer vision*. 2023. <https://viso.ai/deep-learning/object-tracking/>. Imagem. Acesso em: 29 jun. 2025.

LASKOV, P.; DÜSSEL, P.; SCHÄFER, C.; RIECK, K. Learning intrusion detection: Supervised or unsupervised? In: ROLI, F.; VITULANO, S. (Ed.). *Image Analysis and Processing – ICIAP 2005*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 50–57. ISBN 978-3-540-31866-8.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, v. 521, n. 7553, p. 436–444, maio 2015. ISSN 1476-4687. Disponível em: <https://doi.org/10.1038/nature14539>.

MARTINS, E. P. Individual and sex differences in the use of the push-up display by the sagebrush lizard, sceloporus graciosus. *Animal Behaviour*, v. 41, n. 3, p. 403–416, 1991. ISSN 0003-3472. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0003347205808413>.

MARTINS, E. P. Contextual use of the push-up display by the sagebrush lizard, sceloporus graciosus. *Animal Behaviour*, v. 45, n. 1, p. 25–36, 1993. ISSN 0003-3472. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0003347283710031>.

MARTINS, E. P. Structural complexity in a lizard communication system: The sceloporus graciosus “push-up” display. *Copeia*, [American Society of Ichthyologists and Herpetologists (ASIH), Allen Press], v. 1994, n. 4, p. 944–955, 1994. ISSN 00458511, 19385110. Disponível em: <http://www.jstor.org/stable/1446717>.

MEHREEN, K. *Understanding Supervised Learning: Theory and Overview*. 2023. <https://www.kdnuggets.com/understanding-supervised-learning-theory-and-overview>. Imagem. Acesso em: 30 jun. 2025.

MONTELLA, C. *The Kalman Filter and Related Algorithms: A Literature Review*. 2011.

MTHEILER. *Detected-with-YOLO–Schreibtisch-mit-Objekten.jpg*. 2019. <https://commons.wikimedia.org/wiki/File:Detected-with-YOLO--Schreibtisch-mit-Objekten.jpg>. Imagem. Acesso em: 21 jul. 2025.

NGUYEN, G.; DLUGOLINSKY, S.; BOBÁK, M.; TRAN, V.; GARCÍA, A. L.; HEREDIA, I.; MALÍK, P.; HLUCHÝ, L. Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey. *Artificial Intelligence Review*, v. 52, n. 1, p. 77–124, jun. 2019. ISSN 1573-7462. Disponível em: <https://doi.org/10.1007/s10462-018-09679-z>.

NISHANTH, I.; RAKESHKUMAR, R.; THENMOZHI, V. Real-time wild animal detection using yolov10 algorithm. In: IEEE. *2025 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*. [S.l.], 2025. p. 1–6.

O’SHEA, K.; NASH, R. *An Introduction to Convolutional Neural Networks*. 2015. Disponível em: <https://arxiv.org/abs/1511.08458>.

PALERMO-NETO, J.; ALVES, G. J. A comunicação dos animais. *Revista CFMV*, v. 16, n. 49, p. 24–34, 2010. Disponível em: <https://repositorio.usp.br/bitstream/handle/BDPI/2309/art.PALERMO_a_comunicacao_dos_animais.pdf?sequence=1>.

PASSOS, D. C. *Área de vida, organização social e comunicação visual de Tropidurus do grupo semitaeniatus (Squamata: Tropiduridae)*. Tese (Tese de Doutorado) — Universidade do Estado do Rio de Janeiro, Instituto de Biologia Roberto Alcântara Gomes, Rio de Janeiro, 2016.

PEI, Y.; BISWAS, S.; FUSSELL, D. S.; PINGALI, K. *An Elementary Introduction to Kalman Filtering*. 2019. Disponível em: <https://arxiv.org/abs/1710.04055>.

PREETHI, K.; VINITHA, A.; VINOTHIGA, V.; MAHALAKSHMI, V.; KUMARARAJA, V. Ai-driven wildlife detection and management system for agricultural protection. In: IEEE. *2025 3rd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA)*. [S.l.], 2025. p. 1–4.

RADHAKRISHNAN, A.; SAMUEL, S.; JOHN, S. S.; REJI, R. A.; JOHN, S.; JACOB, L. E.; VINOD, D. An automated alert system for monitoring the hygiene in restaurants using machine vision. In: IEEE. *2024 1st International Conference on Trends in Engineering Systems and Technologies (ICTEST)*. [S.l.], 2024. p. 1–6.

RAMASUBRAMANIAN, M.; RANGASWAMY, M. A. D.; REDDY, G. N. V. R. A survey study on detecting and tracking objective methods. In: *2014 IEEE National Conference on Emerging Trends In New & Renewable Energy Sources And Energy Management (NCET NRES EM)*. [S.l.: s.n.], 2014. p. 159–164.

RANA, A.; RAWAT, A. S.; BIJALWAN, A.; BAHUGUNA, H. Application of multi layer (perceptron) artificial neural network in the diagnosis system: A systematic review. In: *2018 International Conference on Research in Intelligent and Computing in Engineering (RICE)*. [S.l.: s.n.], 2018. p. 1–6.

REDMON, J.; DIVVALA, S.; GIRSHICK, R.; FARHADI, A. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. Disponível em: <https://arxiv.org/abs/1506.02640>.

SARKER, I. H. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Computer Science*, v. 2, n. 6, p. 420, ago. 2021. ISSN 2661-8907. Disponível em: <https://doi.org/10.1007/s42979-021-00815-1>.

SATHIARAJ, S.; SATHIARAJ, S.; BEWOOR, L. Object detection techniques: A state-of-the-art survey and challenges. *SSRN Electronic Journal*, 01 2021.

SPONG, M.; HUTCHINSON, S.; VIDYASAGAR, M. *Robot Modeling and Control*. Wiley, 2005. ISBN 9780471649908. Disponível em: <https://books.google.com.br/books?id=A0OXDwAAQBAJ>.

SUN, Y.; SUN, Z.; CHEN, W. The evolution of object detection methods. *Engineering Applications of Artificial Intelligence*, v. 133, p. 108458, 2024. ISSN 0952-1976. Disponível em: <https://www.sciencedirect.com/science/article/pii/S095219762400616X>.

SZEGEDY, C.; LIU, W.; JIA, Y.; SERMANET, P.; REED, S.; ANGUELOV, D.; ERHAN, D.; VANHOUCKE, V.; RABINOVICH, A. *Going Deeper with Convolutions*. 2014. Disponível em: <https://arxiv.org/abs/1409.4842>.

TEAM, T. *Unsupervised Learning – Machine Learning Algorithms*. 2020. <https://techvidvan.com/tutorials/unsupervised-learning/>. Imagem. Acesso em: 30 jun. 2025.

TIAN, J. The human resources development applications of machine learning in the view of artificial intelligence. In: *2020 IEEE 3rd International Conference on Computer and Communication Engineering Technology (CCET)*. [S.l.: s.n.], 2020. p. 39–43.

WEI, C. Improvement of kalman filtering in mobile robot target recognition and intelligent localization research. In: IEEE. *2024 International Conference on Information Technology, Comunication Ecosystem and Management (ITCEM)*. [S.l.], 2024. p. 278–283.

WU, T.; FAN, X. A novel gesture recognition model under sports scenarios based on kalman filtering and yolov5 algorithm. *IEEE Access*, v. 12, p. 64886–64896, 2024. Disponível em: <https://doi.org/10.1109/ACCESS.2024.3394705>.

YAMASHITA, R.; NISHIO, M.; DO, R. K. G.; TOGASHI, K. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, v. 9, n. 4, p. 611–629, ago. 2018. ISSN 1869-4101. Disponível em: <https://doi.org/10.1007/s13244-018-0639-9>.

ZENÓBIO, J. G. F.; SILVA, P. H. L.; LUZ, E. J. da S.; MOREIRA, G. J. P.; GALDINO, C. A. B.; GERTRUDES, J. C. Reptilerecon: Um arcabouço para extração e análise de sinais de lagartos. In: *Proceedings of the 38th Brazilian Symposium on Databases, SBBD 2023, Belo Horizonte, MG, Brazil, September 25-29, 2023*. SBC, 2023. p. 154–166. Disponível em: <https://doi.org/10.5753/sbbd.2023.231703>.