



Trabalho de Conclusão de Curso

**Uma Metodologia para Detecção e Rastreamento
de Múltiplos Cães em Baías utilizando YOLOv11
com Avaliação de Algoritmos de Rastreamento**

Cauê Bittencourt Ângelo dos Santos
cbas@ic.ufal.br

Orientadores:

Thales Miranda de Almeida Vieira
Danielly Spósito Pessoa de Melo

Maceió, Maio de 2025

Cauê Bittencourt Ângelo dos Santos

Uma Metodologia para Detecção e Rastreamento de Múltiplos Cães em Baías utilizando YOLOv11 com Avaliação de Algoritmos de Rastreamento

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação do Instituto de Computação da Universidade Federal de Alagoas.

Orientadores:

Thales Miranda de Almeida Vieira

Danielly Spósito Pessoa de Melo

Catlogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico
Bibliotecária: Taciana Sousa dos Santos – CRB-4 – 2062

S237m Santos, Cauê Bittencourt Ângelo dos.

Uma metodologia para detecção e rastreamento de múltiplos cães em
baías utilizando YOLOv11 com avaliação de algoritmos de rastreamento /
Cauê Bittencourt Ângelo dos Santos. – 2025.
61 f. : il. color.

Orientador: Thales Miranda de Almeida Vieira.

Coorientador: Danielly Spósito Pessoa de Melo.

Monografia (Trabalho de Conclusão de Curso em Ciência da
Computação) – Universidade Federal de Alagoas. Instituto de Computação.
Maceió, 2025.

Bibliografia: f. 58-61.

1. Rastreamento de múltiplos objetos. 2. Visão computacional. 3. Cães –
Monitoramento. I. Título.

CDU: 004

Agradecimentos

Este Trabalho de Conclusão de Curso marca o encerramento de um ciclo importante, iniciado em 2014, e que só foi possível graças ao apoio e incentivo de muitas pessoas, às quais expresso aqui minha sincera gratidão.

Em primeiro lugar, agradeço imensamente à minha família, pelo apoio incondicional e pela paciência ao longo dessa jornada. Aos meus queridos amigos Gustavo e Bruna, deixo meu agradecimento pela amizade, pelos momentos de descontração e, sobretudo, por acompanharem e apoiarem a construção deste TCC.

Manifesto minha profunda gratidão ao meu orientador, Prof. Dr. Thales Miranda de Almeida Vieira, pela visão crítica, por me apresentar ao tema e por compartilhar generosamente seus conhecimentos. À minha coorientadora, Dra. Danielly Spósito Pessoa de Melo, agradeço pela leitura atenta, pelo incentivo e pelo apoio contínuo no desenvolvimento desta pesquisa e também em outros tantos momentos. A orientação, a disponibilidade e a paciência de ambos foram essenciais para a realização deste trabalho.

Agradeço também aos professores Fábio Cunha de Albuquerque e Roberta Vilhena Vieira Lopes, por demonstrarem compreensão e humanidade no trabalho da coordenação do curso e pelo apoio e acolhida diante das dificuldades enfrentadas ao longo desses quase onze anos de trajetória.

Registro ainda meu sincero agradecimento ao Coletivo Autista da UFAL, pelo apoio, pela amizade e pela construção conjunta de um ambiente universitário mais justo e acolhedor. Vocês tornaram minha trajetória na UFAL mais suportável e significativa.

Agradeço, igualmente, aos professores Adriano Oliveira Barbosa e Marcelo Costa Oliveira por aceitarem compor a banca avaliadora do trabalho e pela generosidade em compartilhar seus conhecimentos para o aprimoramento desta pesquisa.

Por fim, agradeço a toda a equipe da Lacom e do Instituto de Computação da UFAL. Em especial, minha gratidão à Giuliana Arecippo, à Sarah Lima, à Ana Ferreira, à Lúcia Maria e ao Marcelo Gusmão, pela gentileza e disponibilidade de sempre.

Cauê Santos

Estivera a olhar fixamente para o alto daquele barril, durante alguns minutos, e o que agora me causava surpresa era o fato de que não houvesse percebido mais cedo a tal coisa ali situada. Aproximei-me e toquei-a com a mão. Era um gato preto, um gato bem grande, tão grande como Plutão, e totalmente semelhante a ele, exceto em um ponto. Plutão não tinha pelos brancos em parte alguma do corpo, mas este gato tinha uma larga, embora imprecisa mancha branca, cobrindo quase toda a região do peito. Logo que o toquei, ele imediatamente se levantou, ronronou alto, esfregou-se contra minha mão e pareceu satisfeito com o meu carinho. Era, pois, aquela a criatura mesma que eu procurava.

– Edgar Allan Poe, *O Gato Preto*

Resumo

O Rastreamento de Múltiplos Animais (MAT), um subcampo do Rastreamento de Múltiplos Objetos (MOT), é uma ferramenta essencial em visão computacional para aplicações como monitoramento comportamental e bem-estar animal. No entanto, rastrear múltiplos cães em baias utilizando uma visão superior (*top-down*) apresenta desafios significativos devido a oclusões frequentes e alta similaridade visual entre animais. Este trabalho teve como objetivo desenvolver e avaliar uma metodologia completa de detecção e rastreamento para este cenário específico. A metodologia envolveu a criação de *datasets* anotados, o ajuste fino (*fine-tuning*) de um detector de objetos YOLO11x sobre dados próprios, e uma avaliação comparativa de quatro algoritmos de rastreamento estado da arte (ByteTrack, BoT-SORT, HybridSORT, Deep OC-SORT). O detector YOLO11x ajustado alcançou um desempenho robusto no conjunto de teste ($mAP@50-95 = 86.9\%$). Na avaliação do rastreamento, BoT-SORT apresentou a maior acurácia geral ($MOTA = 87.4\%$), enquanto HybridSORT demonstrou a melhor consistência na manutenção da identidade ($IDF1 = 67.3\%$, menor IDSW), revelando um *trade-off* entre essas métricas. Conclui-se que a combinação do detector YOLO11x com BoT-SORT ou HybridSORT oferece soluções eficazes para o problema, sendo a escolha dependente da prioridade da aplicação final (*Recall* vs. fidelidade de ID). Este trabalho contribui com uma análise comparativa detalhada e uma metodologia validada para o rastreamento MOT de cães em visão superior.

Palavras-chave: Rastreamento de Múltiplos Animais; MAT; Rastreamento de Múltiplos Objetos; MOT; BoT-SORT; HybridSORT; Detecção de Objetos; YOLOv11; YOLO11x; *Deep Learning*; Visão Computacional; Cães; Visão Superior;

Abstract

Multiple Animal Tracking (MAT), a subfield of Multiple Object Tracking (MOT), is an essential computer vision tool for applications such as behavioral monitoring and animal welfare. However, tracking multiple dogs in kennels using a top-down view presents significant challenges due to frequent occlusions and high visual similarity between individuals. This work aimed to develop and evaluate a complete detection and tracking methodology for this specific scenario. The methodology involved creating annotated datasets, fine-tuning a YOLO11x object detector on custom data, and a comparative evaluation of four state-of-the-art tracking algorithms (ByteTrack, BoT-SORT, HybridSORT, Deep OC-SORT). The fine-tuned YOLO11x detector achieved robust performance on the test set ($mAP@50-95 = 86.9\%$). In the tracking evaluation, BoT-SORT yielded the highest overall accuracy ($MOTA = 87.4\%$), while HybridSORT demonstrated the best identity consistency ($IDF1 = 67.3\%$, lowest IDSW), revealing a trade-off between these metrics. It is concluded that combining the YOLO11x detector with either BoT-SORT or HybridSORT offers effective solutions for the problem, with the choice depending on the final application's priority (overall tracking coverage vs. identity fidelity). This work contributes a detailed comparative analysis and a validated methodology for top-down MOT of dogs.

Keywords: Multiple Animal Tracking, MAT, Multiple Object Tracking, MOT, BoT-SORT, HybridSORT, Object Detection, YOLOv11, YOLO11x, Deep Learning, Computer Vision, Dogs, Top-Down View.

Lista de Figuras

1.1	Exemplo de MOT. Observa-se que, visualmente, os animais (c) são mais difíceis de distinguir do que os carros (a) e os pedestres (b).	10
1.2	Exemplo de imagem captada com visão superior.	12
2.1	Exemplo de uma rede neural profunda (arquitetura VGG16), pode-se notar a estrutura de empilhamento de camadas.	19
2.2	Ilustração esquemática de uma operação de convolução 2D válida, mostrando um <i>kernel</i> deslizando sobre a imagem de entrada para gerar um mapa de características.	20
2.3	Ilustração de <i>bounding boxes</i> delimitando os objetos de interesse (cães) em uma imagem.	24
2.4	Modelagem da YOLO original.	25
2.5	Arquitetura da YOLO original.	25
2.6	Benchmark comparando a YOLOv11 com os modelos antigos.	26
2.7	Ilustração de trajetórias estimadas para múltiplos objetos (cães) ao longo de um vídeo.	27
3.1	Visão geral da nossa abordagem.	37
3.2	Exemplo de utilização da ferramenta CVAT para a tarefa de anotação.	39
4.1	Gráficos das métricas de treinamento e validação ao longo das épocas para um dos treinamentos realizados.	44
4.2	Exemplo de detecção bem-sucedida (pose, borramento).	46
4.3	Exemplo de detecção bem-sucedida (proximidade).	47
4.4	Exemplo de detecção bem-sucedida (nitidez, iluminação).	47
4.5	Exemplo de detecção bem-sucedida (contraste, iluminação).	48
4.6	Exemplo de dificuldade na detecção (baixo contraste com estrutura preta).	48
4.7	Exemplo de dificuldade na detecção (cães muito próximos, baixo contraste).	49
4.8	Comparativo visual do rastreamento de objetos pelos algoritmos BoT-SORT, ByteTrack, HybridSORT e Deep OC-SORT. Cada linha apresenta quadros extraídos nos mesmos instantes de tempo (sincronizados) de uma sequência de vídeo (Câmera 9).	52
4.9	Comparativo visual do rastreamento de objetos pelos algoritmos BoT-SORT, ByteTrack, HybridSORT e Deep OC-SORT. Cada linha apresenta quadros extraídos nos mesmos instantes de tempo (sincronizados) de uma sequência de vídeo (Câmera 7).	53

Conteúdo

1	Introdução	10
1.1	Motivação	10
1.2	Objetivo	13
1.3	Solução Proposta	13
1.4	Contribuições	14
1.5	Estrutura do texto	15
2	Fundamentação Teórica	16
2.1	Aprendizado de Máquina (<i>Machine Learning</i>)	16
2.2	Aprendizado Profundo (<i>Deep Learning</i>)	17
2.2.1	Redes Neurais (<i>Neural Networks</i>)	17
2.2.2	Redes Neurais Convolucionais (CNNs)	18
2.2.3	Treinamento de Redes Profundas	21
2.3	Detecção de Objetos via <i>Deep Learning</i>	23
2.3.1	YOLO (You Only Look Once)	23
2.4	Rastreamento de Múltiplos Objetos (MOT) via <i>Deep Learning</i>	26
2.4.1	Paradigmas Principais (Foco em <i>Tracking-by-Detection</i>)	28
2.4.2	Técnicas Comuns de Associação de Dados em TBD	28
2.4.3	Algoritmos de Rastreamento Relevantes	29
2.5	Métricas de Avaliação	32
2.5.1	Métricas para Detecção de Objetos	32
2.5.2	Métricas para Rastreamento de Múltiplos Objetos (MOT)	34
3	Metodologia	37
3.1	Aquisição e características das imagens	38
3.2	Compilação dos dados supervisionados	38
3.3	Detecção de Objetos (Cães)	38
3.4	Rastreamento de Múltiplos Objetos (MOT)	39
4	Experimentos e Resultados	41
4.1	Compilação e Caracterização dos <i>Datasets</i>	41
4.1.1	<i>Dataset</i> para Detecção de Objetos (Imagens)	41
4.1.2	<i>Dataset</i> para Avaliação de Rastreamento (Vídeos)	42
4.2	Treinamento, Avaliação e Seleção do Detector	43
4.2.1	Configuração do Treinamento	43
4.2.2	Avaliação de Desempenho e Seleção do <i>Checkpoint</i> Final	44
4.3	Avaliação Comparativa dos Algoritmos de Rastreamento	49
4.3.1	Configuração da Avaliação	49
4.3.2	Resultados Quantitativos	50

CONTEÚDO

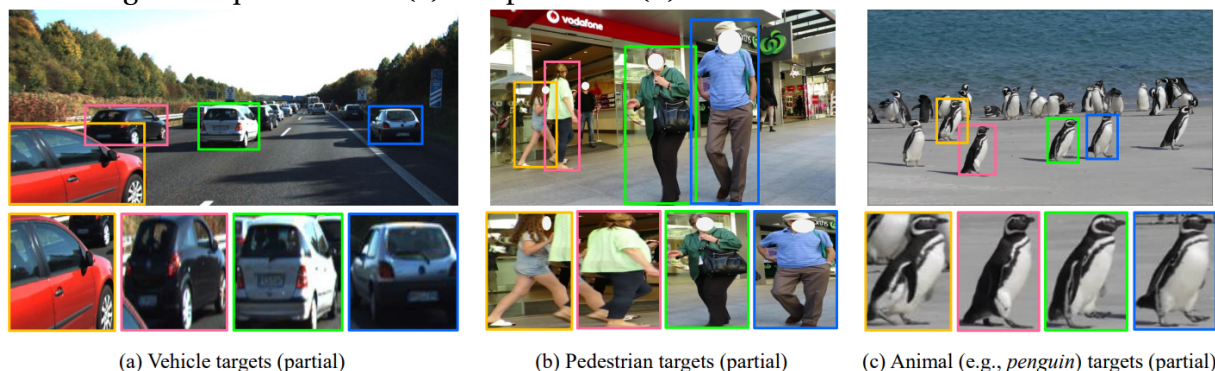
4.3.3	Análise Quantitativa Detalhada	50
4.3.4	Análise Qualitativa	51
4.4	Discussão Geral	54
4.4.1	Síntese do Desempenho e Validação da Abordagem	54
4.4.2	Análise Comparativa e Escolha do Melhor Rastreador	54
5	Conclusões e Trabalhos Futuros	56
5.1	Síntese dos Resultados e Conclusões Principais	56
5.2	Contribuições Principais	57
5.3	Limitações e Trabalhos Futuros	57
5.4	Considerações Finais	58
	Referências	60

Introdução

1.1 Motivação

Nas últimas décadas, os algoritmos de Aprendizado Profundo (*Deep Learning*) tornaram-se ferramentas essenciais para resolver problemas complexos, sendo amplamente aplicados na área de Visão Computacional. Dentro desse campo, o rastreamento de objetos é uma tarefa fundamental, que ocorre após a detecção inicial, atribuindo um identificador único a cada objeto para acompanhar suas trajetórias ao longo dos quadros de um vídeo. Quando múltiplos objetos precisam ser rastreados simultaneamente, a tarefa é denominada Rastreamento de Múltiplos Objetos (*MOT - Multiple Object Tracking*), um problema com crescente aplicabilidade hoje em dia, abrangendo desde o monitoramento de pedestres e veículos até o complexo rastreamento de animais (BASHAR et al., 2022) (Figura 1.1).

Figura 1.1: Exemplo de MOT. Observa-se que, visualmente, os animais (c) são mais difíceis de distinguir do que os carros (a) e os pedestres (b).



Fonte: Libo Zhang et al. (2022).

O Rastreamento de Múltiplos Animais (*MAT - Multiple Animal Tracking*), um subcampo do MOT, foca especificamente em atribuir identificadores únicos a cada animal ao longo

dos quadros de um vídeo, fornecendo *insights* valiosos sobre o comportamento individual e do grupo. Suas aplicações estendem-se por áreas como agricultura, pecuária e pesquisa ecológica (LIU et al., 2024). Essa tecnologia possibilita o monitoramento da saúde e bem-estar animal (detectando reações anormais para prevenir doenças), a preservação de espécies ameaçadas (acompanhando rotas de migração e fatores que as influenciam) e a identificação automática de padrões comportamentais (como interações sociais, competição, cooperação e hábitos alimentares). Um benefício transversal importante é a redução significativa da necessidade de trabalho manual e de recursos em tarefas como: monitoramento contínuo, busca por animais ou medições (como tamanho e massa em aquicultura), substituindo métodos tradicionais, que são frequentemente laboriosos, demorados e limitados em escala (LIU et al., 2024).

O rastreamento de múltiplos cães em baias tem uma importância prática direta com grande potencial de mercado, especialmente para a saúde e bem-estar animal. Um sistema automatizado pode, por exemplo, auxiliar na detecção precoce de doenças em canis ou clínicas, ao identificar mudanças sutis em padrões comportamentais que poderiam passar despercebidas pela observação humana. Ele também pode ajudar na pesquisa farmacêutica, realizando testes de novos medicamentos ou tratamentos, permitindo quantificar objetivamente os efeitos de novos medicamentos ou tratamentos – como um aumento na frequência de coceira decorrente de alguma medicação, por exemplo – fornecendo dados que são difíceis de obter de maneira consistente através da observação manual.

Segundo Liu et al. (2024), o rastreamento de múltiplos animais apresenta desafios únicos em comparação ao rastreamento de humanos, principalmente por causa das variações na aparência e nos padrões de movimento dos animais, conforme ilustrado na Figura 1.1. Dentre os principais desafios, destacam-se: (1) a necessidade de algoritmos robustos que consigam distinguir animais de fundos complexos e diferenciar indivíduos visualmente muito semelhantes; (2) lidar com movimentos frequentemente erráticos e rápidos, que intensificam problemas de oclusão (parcial ou total), deformações e aumentam o risco de trocas de identidade (*ID switches*), e (3) a diferença visual mínima entre indivíduos da mesma espécie, que frequentemente confunde os algoritmos e leva à perda do alvo ou trocas de ID. Adicionalmente, cenários específicos como a visão superior (*top-down*), comum em monitoramentos controlados como o deste trabalho (Figura 1.2), podem anular características distintivas visíveis em outras perspectivas, tornando modelos pré-treinados em conjuntos de dados (*datasets*) genéricos, como o COCO (LIN et al., 2014), frequentemente inadequados e exigindo abordagens adaptadas ou com ajuste fino.

Felizmente, com o avanço contínuo das tecnologias de *hardware* e de algoritmos, a pesquisa em MAT tem se expandido rapidamente. O uso de métodos baseados em imagem, sem marcadores, permite o acompanhamento de um maior número de indivíduos com maior resolução espaço-temporal, em comparação com métodos baseados em sensores, além de ser menos invasivo. Isso abre portas para aplicações que podem reduzir custos operacionais

Figura 1.2: Exemplo de imagem captada com visão superior.



Fonte: dados da pesquisa.

na criação de animais, auxiliar na proteção de espécies ameaçadas e promover o bem-estar animal através de monitoramento não invasivo e automatizado, substituindo tarefas manuais extenuantes (LIU et al., 2024). Diante desse cenário promissor e da rápida evolução dos algoritmos, torna-se crucial a avaliação comparativa das metodologias de rastreamento mais recentes em contextos específicos e desafiadores, como o monitoramento de cães em baias com visão superior, o que justifica a investigação realizada no presente trabalho. Com este propósito, será investigado também se a YOLOv11 (KHANAM; HUSSAIN, 2024), uma arquitetura de detecção de objetos no nível do estado da arte, é capaz de ser treinada para detectar cães em baias com visão superior, com alta acurácia.

1.2 Objetivo

Este trabalho tem como objetivo principal desenvolver e avaliar uma metodologia para detecção e rastreamento de cães, contendo uma análise comparativa de diferentes técnicas de rastreamento, em vídeos gravados em ambiente controlado com câmeras em visão superior. Inicialmente, treina-se um modelo YOLO11x da Ultralytics (JOCHER; QIU; CHAURASIA, 2023) com ajuste fino (*fine-tuning*) para realizar a detecção de cães em ambientes com visão superior. A partir dessas detecções iniciais, compara-se o desempenho de múltiplos algoritmos de MOT para o rastreamento dos animais. Especificamente, esta pesquisa busca:

- Construir e anotar bases de dados de imagens e vídeos de cães em baias, utilizando a ferramenta *Computer Vision Annotation Tool* (CVAT) ¹ e empregando auxílio semi-automático do modelo SAM2 (RAVI et al., 2025) para a anotação dos vídeos;
- Realizar o *fine-tuning* de um modelo detector de objetos YOLO11x para a tarefa específica de detecção de cães nas condições da base de vídeos;
- Avaliar o desempenho do detector de objetos ajustado (*fine-tuned*) de forma isolada (*standalone*), utilizando métricas padrão como *mean Average Precision* (mAP), Precisão e *Recall*;
- Avaliar e comparar o desempenho de múltiplos algoritmos de rastreamento estado da arte, incluindo ByteTrack (ZHANG, Y. et al., 2022), BoT-SORT (AHARON; ORFAIG; BOBROVSKY, 2022), HybridSort (YANG et al., 2024) e Deep OC-SORT (MAGGIOLINO et al., 2023), quando alimentados pelas detecções do modelo YOLO11x ajustado, utilizando métricas padrão de MOT (MOTA, IDF1, IDSW, Frag, Precisão, *Recall*);
- Identificar as soluções de rastreamento mais adequadas para o contexto específico do trabalho, analisando os *trade-offs* entre diferentes métricas de desempenho;
- Analisar criticamente as dificuldades e limitações encontradas durante o desenvolvimento e avaliação da solução.

1.3 Solução Proposta

A solução proposta neste trabalho segue um *pipeline* bem definido para abordar o desafio do rastreamento de cães em visão superior:

1. **Aquisição e Preparação de Dados:** Utilização de vídeos gravados por câmeras fixas no teto das baias. Criação de dois *datasets* distintos: um de imagens para detecção (com

¹<https://docs.cvat.ai/docs/manual/advanced/formats/format-mot/>

divisão Treino/Validação/Teste baseada em câmeras para garantir generalização) e um de vídeos para avaliação do rastreamento. A anotação foi realizada via CVAT, com auxílio do SAM2 para pré-anotação dos vídeos;

2. **Detecção de Objetos:** Emprego de um detector baseado na arquitetura YOLOv11, com ajuste fino sobre o *dataset* de treinamento específico para cães neste contexto;
3. **Rastreamento Múltiplo de Objetos:** Adoção do paradigma *tracking-by-detection*, onde as detecções do YOLO11x alimentam diferentes algoritmos de rastreamento modernos para comparação, incluindo ByteTrack, BoT-SORT, HybridSort e Deep OC-SORT;
4. **Avaliação:** Avaliação quantitativa tanto do detector (usando métricas como mAP, Precisão, *Recall*) quanto dos algoritmos de rastreamento (usando métricas MOTA, IDF1, IDSW, Frag, Precision, Recall), complementada por análise qualitativa.

Ao integrar essas etapas, busca-se uma solução robusta, cuja eficácia é validada por meio de uma análise comparativa detalhada, para o monitoramento de cães no ambiente proposto.

1.4 Contribuições

Este trabalho apresenta as seguintes contribuições para a área de Visão Computacional aplicada ao monitoramento animal:

1. **Identificação e Análise Comparativa de Rastreadores:** A principal contribuição reside na identificação e análise comparativa do desempenho dos algoritmos de rastreamento de múltiplos objetos (MOT) mais adequados para o cenário específico de monitoramento de cães em baías com visão superior, destacando os *trade-offs* entre diferentes métricas. Destaca-se que, para tal, foi necessário realizar o treinamento, via *fine-tuning*, e avaliação de um detector YOLO11x adequado para o cenário proposto;
2. **Metodologia de Avaliação:** Contribui-se também com a descrição detalhada de uma metodologia de avaliação robusta, incluindo uma estratégia de divisão de dados baseada em câmeras para garantir a generalização e um processo de seleção de detector orientado ao desempenho da tarefa final de rastreamento;
3. **Aplicação de Anotação Semi-Automática:** Embora o *dataset* em si não seja público devido à origem dos dados, demonstra-se o ganho de eficiência no processo de anotação de vídeos para rastreamento ao empregar uma ferramenta estado da arte como o SAM2. O relato prático desta abordagem serve como um indicativo útil para trabalhos futuros que enfrentem desafios de anotação em larga escala.

1.5 Estrutura do texto

Esta pesquisa está organizada em 5 capítulos:

- O **Capítulo 1 (Introdução)**, este capítulo, apresenta a motivação, os objetivos, a solução proposta, as contribuições e a estrutura geral do texto;
- O **Capítulo 2 (Fundamentação Teórica)** introduz os conceitos essenciais de Aprendizado de Máquina e Aprendizado Profundo, detalha as técnicas de Detecção de Objetos com foco na arquitetura YOLO e aborda os princípios do Rastreamento de Múltiplos Objetos (MOT), incluindo paradigmas, algoritmos relevantes e métricas de avaliação;
- O **Capítulo 3 (Metodologia)** descreve o *pipeline* metodológico adotado, abrangendo a aquisição de imagens, a compilação e anotação dos dados, a abordagem para detecção de objetos e a estratégia para rastreamento, bem como as métricas selecionadas para avaliação;
- O **Capítulo 4 (Experimentos e Resultados)** detalha a configuração experimental completa, desde a caracterização dos *datasets* e o processo de treinamento, avaliação e seleção do detector YOLO11x, até a configuração da avaliação comparativa dos algoritmos de rastreamento. Em seguida, este mesmo capítulo apresenta, analisa e discute os resultados quantitativos e qualitativos obtidos, abordando os principais achados da avaliação do detector e da comparação entre os rastreadores, bem como as limitações identificadas no estudo;
- O **Capítulo 5 (Conclusões e Trabalhos Futuros)** sintetiza as principais conclusões do trabalho, recapitulando as contribuições e apontando direções para trabalhos futuros.

2

Fundamentação Teórica

Neste capítulo, apresentamos a fundamentação teórica necessária para a compreensão do trabalho proposto. Através do método da revisão bibliográfica, exploramos conceitos-chave, teorias e metodologias relevantes nos campos de estudo de Visão Computacional e *Deep Learning*.

2.1 Aprendizado de Máquina (*Machine Learning*)

[Mitchell \(1997\)](#) apresenta uma definição ampla de Aprendizado de Máquina em seu trabalho. Segundo o autor, trata-se de uma subárea da Inteligência Artificial que abrange o desenvolvimento de programas de computador projetados para melhorar seu desempenho em tarefas específicas por meio de aprendizagem baseada em dados, fornecidos pela experiência adquirida. Diversos métodos e técnicas compõem a subárea do Aprendizado de Máquina, dentre os quais destacam-se os tipos de aprendizados, são eles: aprendizado não supervisionado, aprendizado por reforço e aprendizado supervisionado.

O aprendizado não supervisionado consiste na aprendizagem de padrões ou estruturas ocultas em dados não rotulados, a tarefa usual desse tipo de aprendizado é a clusterização, ou seja: "detectar agrupamentos potencialmente úteis de exemplos de entrada" ([RUSSELL; NORVIG, 2009](#)). Já o aprendizado por reforço implica em um algoritmo baseado em uma série de reforços, recompensas ou penalizações, sendo assim um processo de aprendizagem fundamentado em tentativa e erro ([RUSSELL; NORVIG, 2009](#)).

Por último, tem-se o tipo de aprendizado prevalente neste trabalho, o supervisionado. Nele, treina-se um agente a partir de um conjunto de dados rotulados, compostos de pares de entrada e seu rótulo (indicando a saída esperada). A principal tarefa atribuída ao sistema no aprendizado supervisionado é a previsão ou classificação, baseada nesse conjunto de dados, através do aprendizado de uma função que mapeia uma entrada à saída esperada.

Neste contexto, o conjunto de rótulos, geralmente obtido por anotação manual por especialistas, é denominado *ground truth*, servindo como referência absoluta para treinar o modelo e avaliar seu desempenho.

A área de aplicação do aprendizado de máquina supervisionado é vasta e diversificada, abrangendo desde a detecção de fraudes em transações financeiras, onde são analisados padrões de gastos para sinalizar atividades suspeitas, até o campo de Processamento de Linguagem Natural (PLN). Na área de Visão Computacional, o aprendizado de máquina supervisionado está presente em sistemas que identificam e classificam objetos em imagens e vídeos. Isso possibilita uma variedade de aplicações, incluindo o reconhecimento facial em *smartphones*, sistemas avançados de vigilância, diagnósticos médicos por imagem e reidentificação de animais para estudos de dinâmicas populacionais, ecologia comportamental e comunitária.

2.2 Aprendizado Profundo (*Deep Learning*)

Abordagens tradicionais de aprendizado de máquina, as chamadas *Shallow Learning* ou Aprendizado Raso, geralmente se concentram na aprendizagem de uma ou duas camadas de representações dos dados (CHOLLET, 2021). Em contraste, Chollet (2021) aponta que o *Deep Learning*, ou Aprendizado Profundo, constitui uma subárea do Aprendizado de Máquina que se concentra no estudo e implementação de algoritmos capazes de aprender representações complexas e significativas dos dados, por meio de camadas sucessivas de transformações. A quantidade de camadas que um modelo possui é chamada de profundidade de um modelo. Assim, o termo "profundo" no conceito Aprendizado Profundo se refere ao empilhamento dessas camadas, portanto, ao poder de representação do mesmo.

No contexto do *Deep Learning* moderno, é comum o uso de dezenas ou até mesmo centenas de camadas sucessivas, tais representações são aprendidas automaticamente a partir dos dados de treinamento. Chollet (2021) apresenta ainda uma metáfora que auxilia na compreensão do conceito, ao afirmar que se pode pensar em uma rede profunda como um processo de destilação da informação em múltiplas etapas. Nessa metáfora, a informação passa por filtros sucessivos, saindo cada vez mais "purificada", ou seja, mais útil para realizar determinada tarefa.

2.2.1 Redes Neurais (*Neural Networks*)

Em *Deep Learning*, as representações em camadas são aprendidas por meio de modelos denominados Redes Neurais (*Neural Networks*), as quais são estruturadas em camadas empilhadas umas sobre as outras. Essas camadas formam modelos complexos capazes de aprender representações dos dados, progressivamente mais abstratas.

O termo Rede Neural tem origem na neurobiologia e pode ser definido como um conjunto interconectado de unidades, ou nós, cuja funcionalidade é vagamente baseada no neurônio animal, essas unidades são chamadas de neurônios (GURNEY, 1997). Cada neurônio processa sinais de entrada por meio de conexões com pesos associados e aplica uma função de ativação não-linear ao resultado para gerar sua saída. Os pesos representam a força de conexão do neurônio com outros neurônios na camada anterior da rede (GURNEY, 1997), e é justamente neles que o aprendizado da rede fica armazenado.

O empilhamento dessas camadas permite aprender relações não-lineares complexas presentes nos dados (LECUN; BENGIO; HINTON, 2015). É por meio da exposição da rede a múltiplos exemplos desses dados rotulados, no processo chamado de treinamento da rede, que ela modifica iterativamente seus pesos, aprendendo assim as características discriminantes necessárias para realizar a tarefa proposta (LECUN; BENGIO; HINTON, 2015).

2.2.2 Redes Neurais Convolucionais (CNNs)

As Redes Neurais Convolucionais (CNNs) representam uma arquitetura de rede neural projetada especialmente para o processamento de dados que seguem uma topologia de grade, como é o caso de imagens bidimensionais (GOODFELLOW; BENGIO; COURVILLE, 2016). A característica distintiva das CNNs é a aplicação da operação matemática de convolução - um tipo especializado de operação linear - em pelo menos uma de suas camadas, ao invés da multiplicação de matrizes genérica utilizada em redes totalmente conectadas (GOODFELLOW; BENGIO; COURVILLE, 2016).

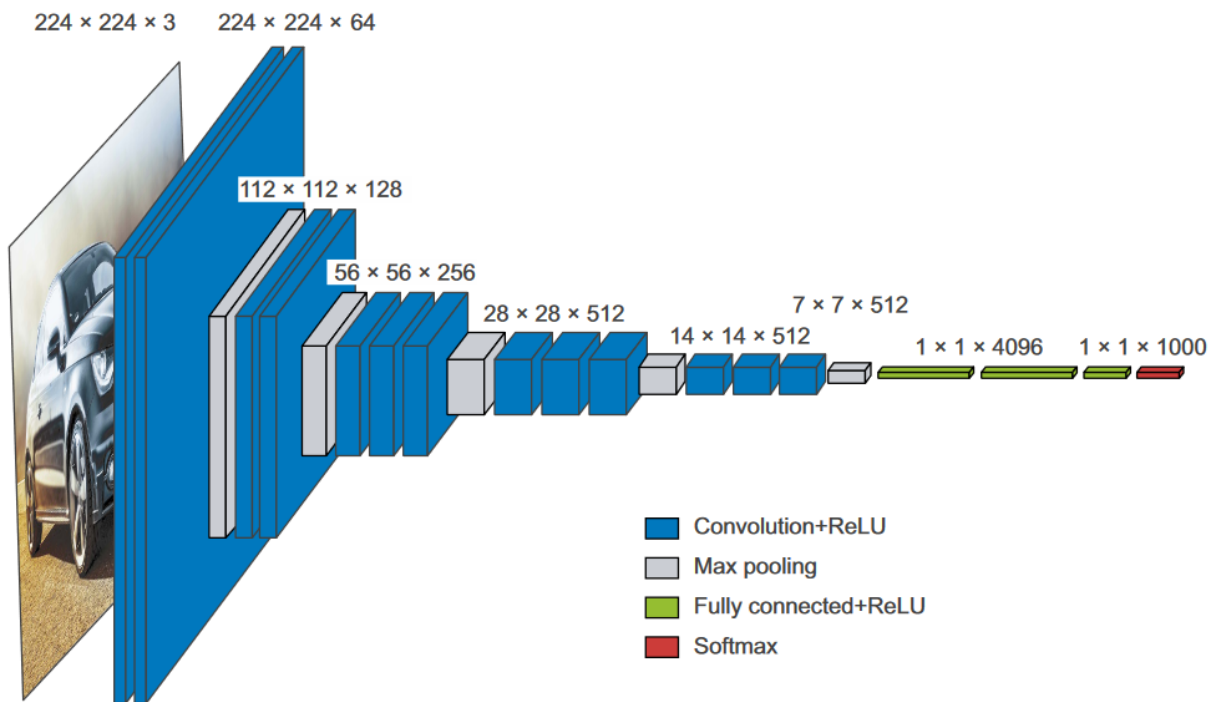
Essas redes utilizam camadas de convolução para extrair características cruciais das imagens, tais como bordas, texturas e formas, tornando-as particularmente eficazes para a extração de representações significativas de uma imagem. Uma representação visual das camadas pode ser observada na Figura 2.1.

Operação de Convolução 2D

A operação central da CNN é a convolução discreta bidimensional. Conceitualmente, um filtro (*kernel*), que é uma pequena matriz de pesos aprendidos, desliza sobre a imagem de entrada. Em cada posição, calcula-se a soma ponderada dos pixels da imagem cobertos pelo filtro, gerando um valor de saída que compõe o mapa de características (*feature map*). Esse mapa representa a ativação de um determinado padrão (definido pelos pesos do filtro) naquela região da imagem. A Figura 2.2 ilustra este processo. Matematicamente, para uma imagem de entrada I e um *kernel* K , a convolução discreta 2D (*cross-correlation* na prática de *Deep Learning*) que gera o mapa de saída S é frequentemente definida como:

$$S(i, j) = (K * I)(i, j) = \sum_m \sum_n I(i + m, j + n) K(m, n). \quad (2.1)$$

Figura 2.1: Exemplo de uma rede neural profunda (arquitetura VGG16), pode-se notar a estrutura de empilhamento de camadas.



Fonte: Chollet (2021).

onde (i, j) são as coordenadas no mapa de características de saída e (m, n) iteram sobre os elementos do *kernel* K . Parâmetros importantes incluem o tamanho do *kernel*, o *stride* (passo do deslizamento) e o *padding* (adição de bordas à imagem de entrada) (CHOLLET, 2021).

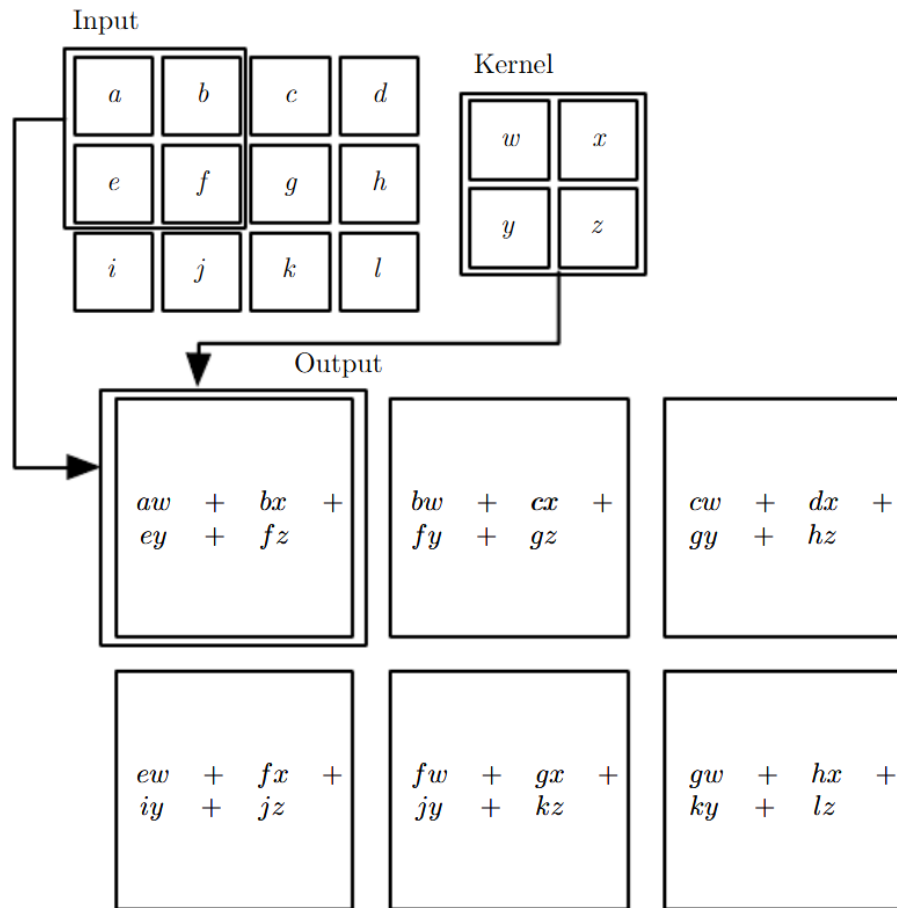
Função de Ativação Não-Linear

Após a operação de convolução, que é uma transformação linear, aplica-se tipicamente uma função de ativação não-linear a cada elemento do mapa de características resultante. Esta etapa de não-linearidade é essencial, pois permite que a rede neural aprenda padrões e relações complexas nos dados. Embora existam diversas funções de ativação (como sigmoide, softmax ou tangente hiperbólica), a Unidade Linear Retificada (ReLU - *Rectified Linear Unit*) é uma escolha comum e excelente para camadas intermediárias da rede (GOOD-FELLOW; BENGIO; COURVILLE, 2016). Sua definição matemática é simples:

$$g(z) = \max(0, z). \quad (2.2)$$

É importante notar que existem diversas generalizações e variantes da ReLU (como Leaky ReLU, PReLU). A maioria dessas alternativas apresenta desempenho comparável ao da ReLU padrão em muitas tarefas e, ocasionalmente, pode oferecer vantagens ou um desem-

Figura 2.2: Ilustração esquemática de uma operação de convolução 2D válida, mostrando um *kernel* deslizando sobre a imagem de entrada para gerar um mapa de características.



Fonte: Goodfellow, Bengio e Courville (2016).

penho ligeiramente superior em contextos específicos (GOODFELLOW; BENGIO; COURVILLE, 2016). No entanto, a ReLU básica (Equação 2.2) continua sendo um ponto de partida robusto e amplamente utilizado.

Camadas de *Pooling*

Outro componente fundamental das CNNs são as camadas de *pooling* (agrupamento). Sua função principal é reduzir progressivamente a resolução espacial dos mapas de características, tornando a representação mais robusta a pequenas translações e distorções na entrada, além de diminuir o número de parâmetros e a carga computacional. A operação de *pooling* mais utilizada é o *Max Pooling*, que extrai o valor máximo de uma pequena janela (2x2, por exemplo) que desliza sobre o mapa de características, com um certo passo (*stride*) (GOODFELLOW; BENGIO; COURVILLE, 2016).

Estrutura Típica e Camadas Totalmente Conectadas

Uma arquitetura CNN típica consiste no empilhamento de múltiplos blocos, onde cada bloco geralmente contém uma ou mais camadas convolucionais, seguidas por uma função de ativação (como a ReLU) e, opcionalmente, uma camada de *pooling*. As camadas iniciais tendem a aprender características de baixo nível (bordas e texturas, por exemplo), enquanto camadas mais profundas aprendem padrões mais complexos e abstratos (CHOLLET, 2021). Após as camadas convolucionais e de *pooling*, que atuam como extratores de características, é comum adicionar uma ou mais camadas totalmente conectadas (FC - *Fully Connected*) na parte final da rede. Essas camadas recebem o vetor de características achatado (*flattened*) e realizam a tarefa final, como classificação de objetos ou regressão de coordenadas de *bounding boxes* (no caso de alguns detectores).

Arquiteturas Notáveis

O sucesso das CNNs levou ao desenvolvimento de diversas arquiteturas que se tornaram marcos na área, cada uma com inovações específicas em termos de profundidade, blocos de construção ou eficiência. Exemplos notáveis incluem a AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2017), VGG (SIMONYAN; ZISSERMAN, 2015), GoogLeNet/Inception (SZEGEDY et al., 2015) e ResNet (*Residual Networks*) (HE et al., 2016).

2.2.3 Treinamento de Redes Profundas

O processo pelo qual uma rede neural "aprende" é conhecido como treinamento. O objetivo central é ajustar iterativamente os pesos da rede para que ela consiga mapear corretamente as entradas (por exemplo, imagens) para as saídas desejadas (localização e classe de objetos, por exemplo), minimizando uma função de perda (*loss function*) que quantifica a discrepância entre as previsões do modelo e os rótulos verdadeiros. O algoritmo fundamental para calcular como os pesos devem ser ajustados é o *backpropagation* (RUMELHART; HINTON; WILLIAMS, 1986), que utiliza o gradiente da função de perda em relação a cada peso para propagar o sinal de erro de volta através das camadas, guiando assim a atualização (GOODFELLOW; BENGIO; COURVILLE, 2016).

Os gradientes calculados são então utilizados por um algoritmo de otimização para realizar a atualização efetiva dos pesos. Embora diversas opções existam, otimizadores com taxa de aprendizado adaptativa, como o AdamW (LOSHCHILOV; HUTTER, 2019) (empregado neste trabalho), são frequentemente escolhidos devido à sua eficiência. Eles adaptam a taxa de aprendizado para diferentes parâmetros e incorporam o conceito de *Momentum* (POLYAK, 1964) para ajudar a acelerar e estabilizar a convergência do treinamento (GOODFELLOW; BENGIO; COURVILLE, 2016). O treinamento tipicamente ocorre ao longo de múltiplas épocas, onde cada época representa uma passagem completa por todo o conjunto de

dados de treinamento.

Dado seu grande número de parâmetros, as redes neurais profundas são particularmente suscetíveis ao *overfitting* (superajuste). Este fenômeno descreve a situação em que o modelo se ajusta excessivamente aos dados de treinamento, incluindo ruídos e particularidades, perdendo assim sua capacidade de generalizar para dados novos e não vistos. Para combater o *overfitting*, técnicas de regularização são cruciais. O *Dropout* (SRIVASTAVA et al., 2014) é uma abordagem eficaz e amplamente utilizada nesse sentido.

O mecanismo do *Dropout*, aplicado durante a fase de treinamento, consiste em desativar (zerar), aleatoriamente a cada iteração, a saída de uma fração pré-definida das unidades (neurônios) de uma camada. A lógica por trás dessa abordagem, como explica Chollet (2021), é que introduzir esse ruído força a rede a aprender representações mais distribuídas e robustas, evitando a dependência excessiva de neurônios específicos e a memorização de padrões não significativos dos dados de treinamento.

Como o objetivo é desenvolver modelos que possam generalizar com sucesso para novos dados, é essencial medir confiavelmente essa capacidade. A prática padrão envolve dividir os dados em conjuntos de treinamento, validação e teste. Treina-se o modelo nos dados de treino e avalia-se seu desempenho no conjunto de validação para guiar ajustes de hiperparâmetros ou outras decisões de modelagem. No entanto, o ajuste contínuo baseado no desempenho de validação pode levar ao *overfitting* nesse próprio conjunto de validação, mesmo que ele não seja usado diretamente para o cálculo dos gradientes (CHOLLET, 2021). Por isso, uma vez que o modelo é considerado pronto (após ajuste e seleção), ele é avaliado uma única vez no conjunto de teste, que deve ser mantido intocado até então e ser o mais representativo possível dos dados reais da aplicação, para obter uma estimativa imparcial da sua capacidade de generalização.

Para determinar o ponto ótimo de interrupção do treinamento, evitando que o modelo comece a superajustar ao conjunto de treino ao mesmo tempo que se busca o melhor desempenho no conjunto de validação, utiliza-se comumente a técnica de *early stopping* (parada antecipada). Nela, monitora-se uma métrica de interesse (perda ou acurácia, por exemplo) e o treinamento é interrompido se essa métrica não melhorar por um número definido de épocas consecutivas, parâmetro conhecido por *patience* (paciência) (CHOLLET, 2021).

Durante o treinamento, salva-se periodicamente o estado da rede (pesos e, às vezes, estado do otimizador) em arquivos chamados *checkpoints*. Tipicamente, o *checkpoint* correspondente ao melhor desempenho alcançado no conjunto de validação (antes da parada antecipada, se utilizada) é guardado e selecionado como o "melhor *checkpoint*" para a avaliação final no conjunto de teste e para uso posterior (CHOLLET, 2021).

Por fim, Chollet (2021) aponta que uma abordagem comum e muito eficaz para aprendizado profundo em conjuntos de dados de imagens relativamente pequenos é usar um modelo pré-treinado, também conhecida por Transferência de Aprendizado (*Transfer Learning*). Um modelo pré-treinado é um modelo que já foi treinado em um *dataset* mas-

sivo, como COCO (LIN et al., 2014). A ideia é aproveitar o conhecimento sobre características visuais genéricas (como bordas, texturas e formas) aprendido por este modelo e continuar o treinamento no *dataset* alvo com uma taxa de aprendizado geralmente menor. Isso permite transferir o conhecimento de características visuais genéricas aprendidas no pré-treinamento, acelerando a convergência e frequentemente resultando em maior desempenho final.

Uma das maneiras mais comuns de realizar o *Transfer Learning* é através do *fine-tuning* (ajuste fino). Segundo Chollet (2021), o *fine-tuning* consiste em "descongelar" (permitir a atualização dos pesos), durante o treinamento, algumas das camadas superiores da base do modelo pré-treinado. Isso é chamado de ajuste fino porque modifica ligeiramente as representações mais abstratas do modelo reutilizado para torná-las mais relevantes para o problema específico em mãos.

2.3 Detecção de Objetos via *Deep Learning*

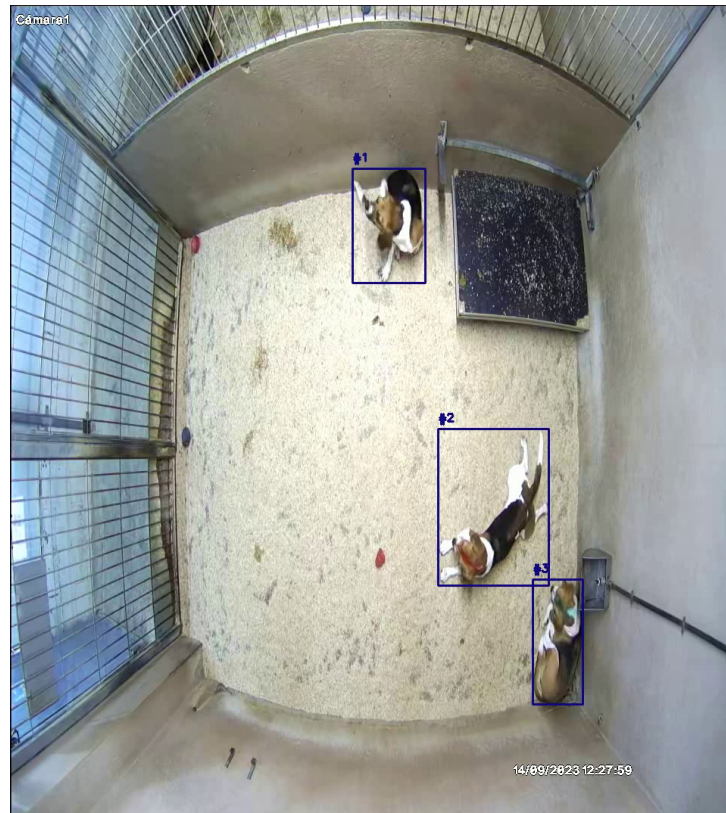
A detecção de objetos é uma tarefa fundamental em visão computacional que visa não apenas classificar os objetos presentes em uma imagem, mas também localizar sua posição espacial exata. Isso é tipicamente realizado através da predição de uma *bounding box* (caixa delimitadora) – geralmente um retângulo – que envolve o objeto de interesse, associada a um rótulo de classe ("cão", "pessoa", "carro", por exemplo) e um *score* de confiança (CHOLLET, 2021). A Figura 2.3 ilustra o conceito. Abordagens modernas utilizam predominantemente redes neurais profundas para realizar esta tarefa.

2.3.1 YOLO (You Only Look Once)

A família de detectores YOLO (You Only Look Once) representa uma abordagem influente e popular para a detecção de objetos em tempo real. Proposta originalmente por Redmon et al. (2016), sua filosofia central é tratar a detecção como um problema de regressão, utilizando uma única rede neural convolucional (CNN) para prever simultaneamente as localizações das *bounding boxes* e as probabilidades de classe diretamente a partir da imagem inteira em uma única passagem (*single pass*). Essa abordagem unificada permite que a YOLO seja extremamente rápida comparada a abordagens anteriores (como as *two-stage*) que dependem de múltiplos estágios (proposta de região, classificação, refinamento) (REDMON et al., 2016).

No *framework* original, a imagem de entrada é dividida em uma grade (*grid*) $S \times S$, conforme ilustrado na Figura 2.4. Se o centro de um objeto recai sobre uma célula da grade, essa célula torna-se responsável por detectar aquele objeto. Cada célula da grade prediz *B bounding boxes* e um *score* de confiança para cada uma delas. A confiança reflete a probabilidade de que a caixa contenha um objeto $\Pr(\text{Object})$ e quão precisa é a predição da caixa

Figura 2.3: Ilustração de *bounding boxes* delimitando os objetos de interesse (cães) em uma imagem.



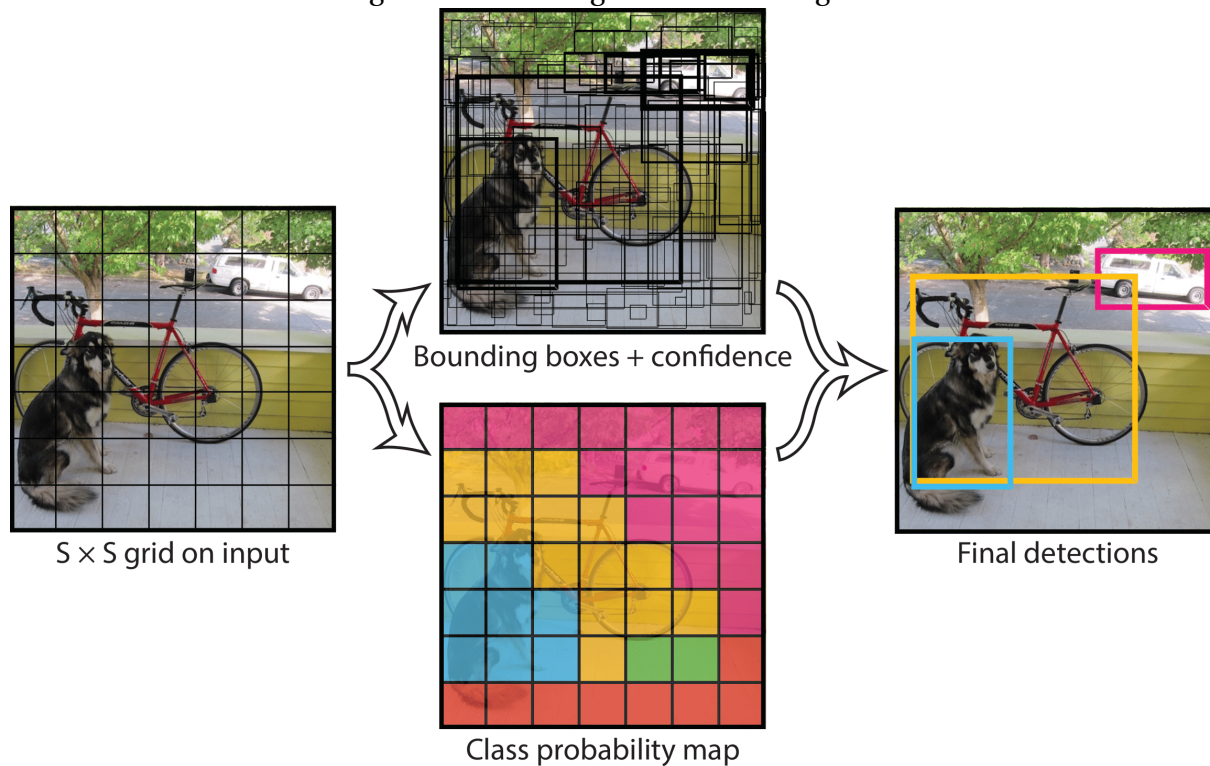
Fonte: elaborado pelo autor.

IOU_{pred}^{truth} , definida formalmente como $Pr(Object) \times IOU_{pred}^{truth}$ (REDMON et al., 2016). IOU será detalhado na Seção 2.5.1. Cada predição de *bounding box* consiste em 5 valores: as coordenadas do centro (x, y) relativas aos limites da célula da grade, a largura w e altura h relativas à imagem inteira e o *score* de confiança. Adicionalmente, cada célula da grade prediz C probabilidades condicionais de classe, $Pr(Class_i|Object)$, que indicam a probabilidade de um objeto pertencer a uma classe específica, dado que um objeto está presente naquela célula (REDMON et al., 2016).

A arquitetura de rede YOLO original (Figura 2.5) foi inspirada no GoogLeNet (SZEGEDY et al., 2015), consistindo em 24 camadas convolucionais seguidas por 2 camadas totalmente conectadas. As camadas convolucionais iniciais extraem características da imagem, enquanto as camadas totalmente conectadas preveem as probabilidades de classe e as coordenadas das caixas (REDMON et al., 2016). A rede utiliza camadas de redução 1×1 seguidas por convoluções 3×3 , além de uma função de ativação Leaky ReLU nas camadas ocultas e uma ativação linear na camada final.

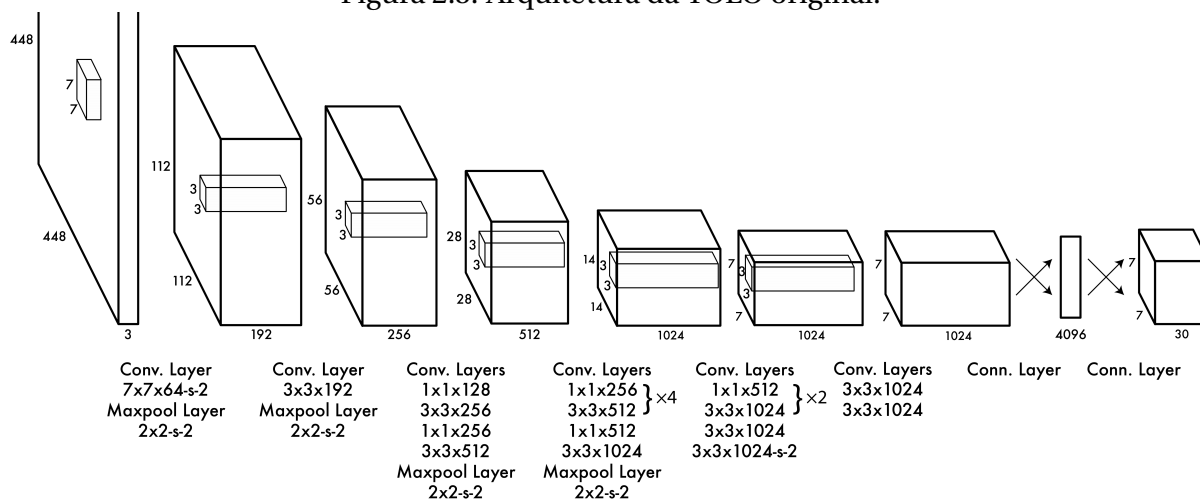
É crucial notar, entretanto, que a família YOLO evoluiu significativamente desde sua concepção original. A YOLOv11, versão utilizada neste trabalho, representa um avanço considerável sobre seus predecessores. Conforme descrito por Khanam e Hussain (2024), a YOLOv11 introduz aprimoramentos arquiteturais chave, que visam otimizar o balanço entre acurácia

Figura 2.4: Modelagem da YOLO original.



Fonte: Redmon et al. (2016).

Figura 2.5: Arquitetura da YOLO original.

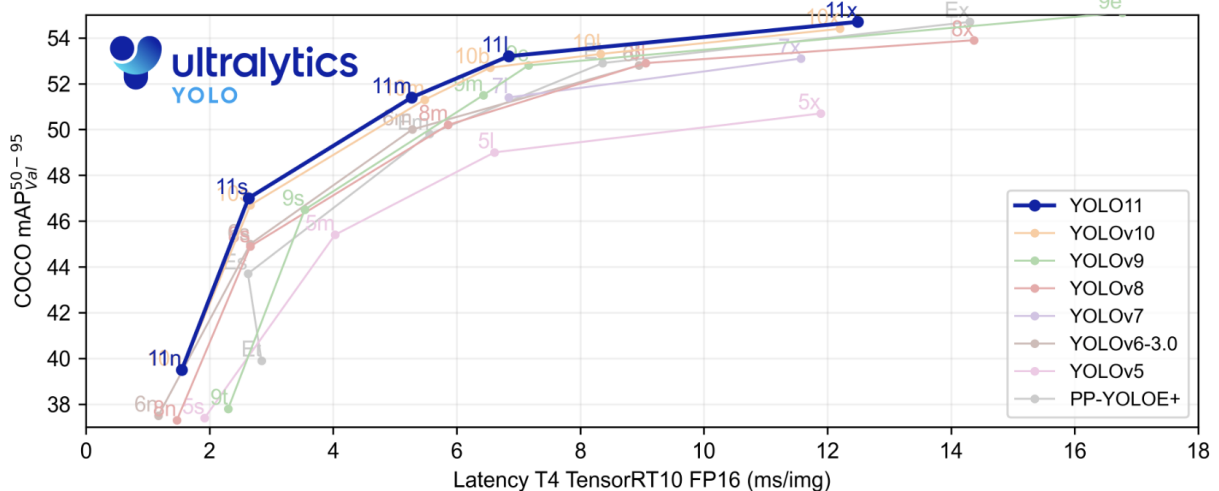


Fonte: Redmon et al. (2016).

(mAP) e velocidade, tornando a YOLOv11 uma ferramenta poderosa e versátil para diversas tarefas de visão computacional em tempo real, incluindo não apenas detecção, mas também segmentação de instância e estimativa de pose, como ilustra o *benchmark* na Figura 2.6, onde podemos ver o melhor desempenho da YOLOv11 em comparação aos modelos anteriores.

É importante distinguir a nomenclatura utilizada para referenciar a arquitetura conceitual, YOLOv11, cujos avanços foram descritos por [Khanam e Hussain \(2024\)](#), da implementação específica utilizada neste trabalho, disponibilizada pela biblioteca Ultralytics ([JOCHER; QIU; CHAURASIA, 2023](#)) e referenciada por essa fonte como YOLO11x. Para os experimentos, empregou-se a variante *extra-large* ("x") desta arquitetura, buscando maior acurácia geral. Esta nomenclatura (YOLO11x) será utilizada doravante ao se referir ao modelo detector específico empregado nas etapas experimentais deste TCC.

Figura 2.6: Benchmark comparando a YOLOv11 com os modelos antigos.



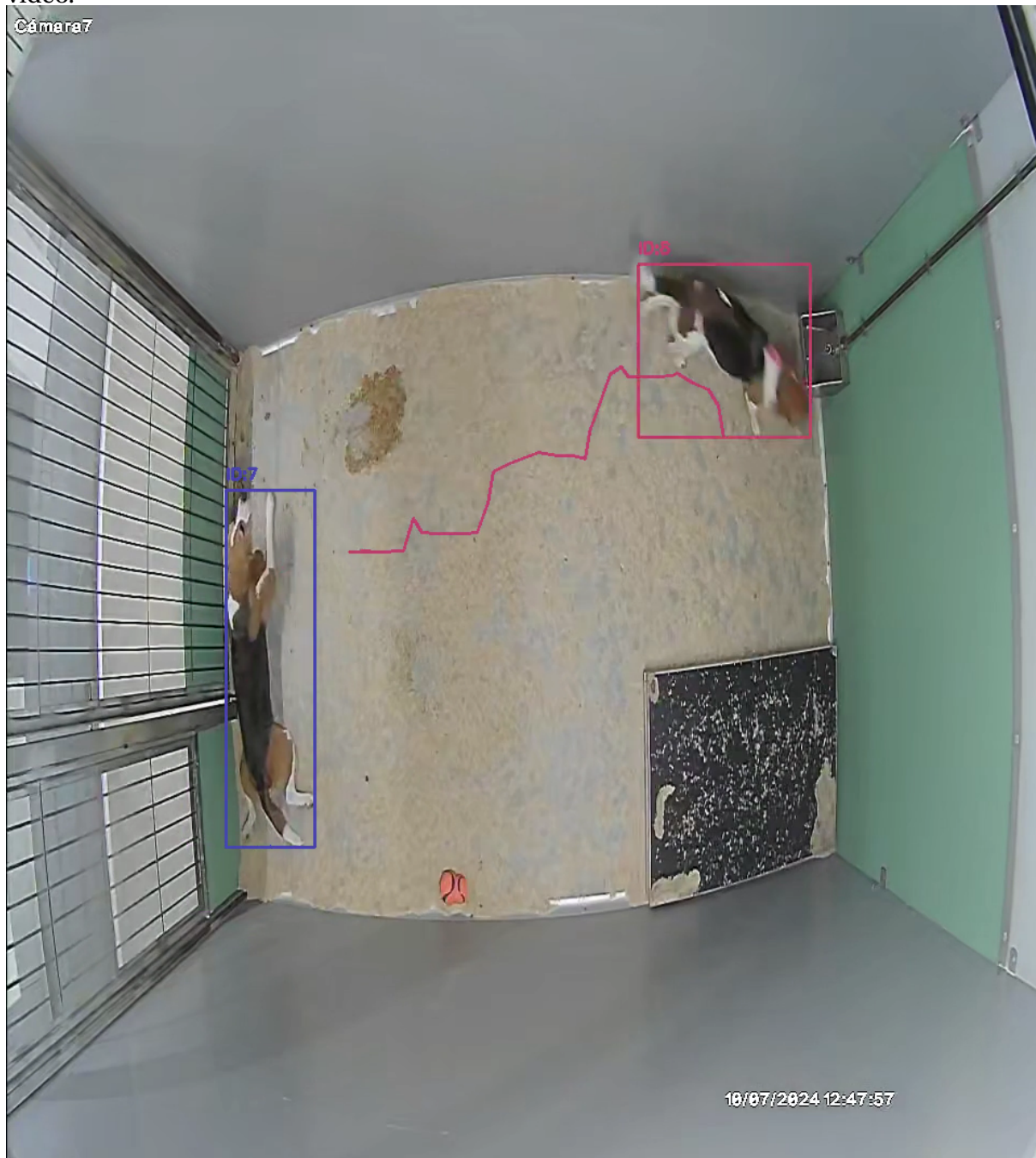
Fonte: [Khanam e Hussain \(2024\)](#).

2.4 Rastreamento de Múltiplos Objetos (MOT) via *Deep Learning*

O Rastreamento de Objetos (*Tracking*) é a tarefa de localizar e seguir um ou mais objetos de interesse ao longo dos quadros (*frames*) de uma sequência de vídeo. Quando o objetivo é rastrear múltiplos objetos simultaneamente, mantendo uma identidade única para cada um, a tarefa é denominada Rastreamento de Múltiplos Objetos (MOT). Formalmente, o MOT visa estimar as trajetórias dos objetos ao longo do tempo, geralmente representadas por uma sequência de *bounding boxes* associadas a um identificador (ID) único para cada objeto ([ZHANG, Y. et al., 2022](#)). Esta é uma tecnologia essencial para diversas aplicações, desde vigilância e carros autônomos até a análise comportamental de animais. A Figura 2.7

ilustra trajetórias estimadas, a partir da qual se pode inferir o deslocamento dos indivíduos; por exemplo, que o cão à esquerda permaneceu parado enquanto o da direita se moveu vindo da esquerda, perto do outro cão, para perto do comedouro à direita.

Figura 2.7: Ilustração de trajetórias estimadas para múltiplos objetos (cães) ao longo de um vídeo.



Fonte: elaborado pelo autor.

2.4.1 Paradigmas Principais (Foco em *Tracking-by-Detection*)

A abordagem predominante, e utilizada neste trabalho, para MOT é conhecida como *Tracking-by-Detection* (TBD). Nela, segundo [Bochinski, Eiselein e Sikora \(2017\)](#), um detector de objetos - como o YOLO11x, detalhado na Seção 2.3.1 - é aplicado a cada *frame* do vídeo de forma independente para gerar um conjunto de detecções (geralmente *bounding boxes*) para os objetos de interesse. Em um segundo passo, um algoritmo de associação de dados é então executado para conectar essas detecções através dos quadros, formando e estendendo as trajetórias individuais e atribuindo/mantendo as identidades dos objetos. A qualidade da performance do detector de objetos é, portanto, crucial para o sucesso da abordagem TBD ([BEWLEY et al., 2016](#)).

2.4.2 Técnicas Comuns de Associação de Dados em TBD

A etapa de associação de dados é o componente central dos algoritmos de rastreamento TBD, responsável por decidir quais detecções no quadro atual correspondem a quais trajetórias já existentes. Para tomar essa decisão, diversas fontes de informação e técnicas são comumente combinadas:

Predição de Movimento

Modelos de movimento são usados para estimar a posição provável de um objeto no quadro atual com base em seu histórico. O Filtro de Kalman (KF) ([KALMAN, 1960](#)) é amplamente adotado para essa finalidade, especialmente para modelar movimentos relativamente lineares e prever o estado futuro (posição, velocidade) de uma trajetória e a incerteza associada, permitindo comparar detecções com as posições preditas.

O Filtro de Kalman é um algoritmo recursivo computacionalmente eficiente para estimar o estado de um sistema dinâmico a partir de uma série de medições incompletas ou ruidosas. No contexto do rastreamento de objetos, seu objetivo é prever a localização de um objeto no quadro atual com base em seu histórico de movimento. O funcionamento do filtro é baseado em um ciclo de duas etapas: predição e atualização.

1. **Etapa de Predição:** Nesta fase, o filtro utiliza o estado estimado do objeto no instante anterior (por exemplo, sua posição x, y e velocidade v_x, v_y) para prever seu estado no instante atual. Essa predição é feita com base em um modelo de movimento, que geralmente assume uma dinâmica linear (como movimento com velocidade constante). O resultado não é uma única posição, mas sim uma estimativa acompanhada de uma matriz de covariância, que quantifica a incerteza associada a essa predição. Em outras palavras, o filtro prevê onde o objeto provavelmente estará e o quão confiante ele está sobre essa previsão.

2. **Etapa de Atualização (ou Correção):** Após a predição, uma nova medição se torna disponível — no caso do rastreamento, a detecção do objeto (a *bounding box* fornecida pelo YOLO) no quadro atual. O filtro então compara sua predição com a medição real. A etapa de atualização combina a predição e a medição, ponderando-as de acordo com suas respectivas incertezas. Se a incerteza da predição for alta e a da medição for baixa, o filtro confiará mais na nova medição, e vice-versa. O resultado é um novo estado estimado, mais preciso e com uma incerteza reduzida, que servirá como base para o próximo ciclo de predição.

A grande vantagem do Filtro de Kalman no rastreamento é sua capacidade de lidar com oclusões curtas e imprecisões na detecção. Ao prever a localização esperada de um objeto, ele define uma região de busca, facilitando a associação correta da identidade do objeto entre quadros consecutivos, mesmo que o detector falhe momentaneamente ou a aparência do objeto mude. No entanto, é importante notar que sua suposição de linearidade pode ser uma limitação em cenários com movimentos altamente erráticos e não lineares.

Proximidade Espacial (Sobreposição de *Bounding Boxes*)

A proximidade espacial entre uma detecção atual e a posição predita de uma trajetória é um forte indicativo de correspondência, especialmente em curtos intervalos de tempo. A métrica de Interseção sobre União (IoU), que será detalhada posteriormente na Seção 2.5.1, é a medida mais comum para quantificar essa similaridade espacial na etapa de associação.

Similaridade de Aparência

Quando o movimento ou a sobreposição não são suficientes para desambiguar associações (por exemplo, durante oclusões), a aparência visual do objeto se torna crucial. Técnicas de Re-Identificação (Re-ID), frequentemente baseadas em *deep learning*, são usadas para extrair um vetor de características (*embedding*) que descreve a aparência única de cada detecção. A similaridade entre os *embeddings* de uma detecção e de uma trajetória existente pode então ser usada para reforçar a associação correta e manter a identidade, mesmo após longas oclusões (WOJKE; BEWLEY; PAULUS, 2017).

Os algoritmos de rastreamento modernos tipicamente calculam um custo ou uma afinidade combinando essas diferentes pistas (movimento, sobreposição, aparência) e utilizam algoritmos de otimização, como o Algoritmo Húngaro (KUHN, 1955), para encontrar a melhor correspondência global entre detecções e trajetórias a cada quadro.

2.4.3 Algoritmos de Rastreamento Relevantes

Nesta seção, descreve-se brevemente os princípios dos algoritmos de rastreamento selecionados para comparação experimental neste trabalho, todos operando sob o paradigma

Tracking-by-Detection e utilizando combinações das técnicas de associação discutidas na Seção 2.4.2.

ByteTrack

O ByteTrack introduz uma abordagem inovadora para o MOT, diferenciando-se pela sua estratégia de integrar detecções de baixa confiança ao processo de associação de trajetórias. Enquanto métodos tradicionais tendem a ignorar essas detecções de menor confiança, o ByteTrack as utiliza, partindo da premissa de que elas frequentemente correspondem a objetos que estão temporariamente oclusos ou em condições difíceis de detecção. (ZHANG, Y. et al., 2022)

O princípio de funcionamento do ByteTrack baseia-se em uma associação realizada em duas fases consecutivas. Na primeira fase, apenas as *bounding boxes* que possuem alta pontuação (*score*) de detecção são consideradas para associação com as trajetórias ativas (*tracklets*). Para realizar essa correspondência inicial, podem ser aplicadas técnicas padrão, como a previsão da localização futura da trajetória, utilizando um Filtro de Kalman, e a posterior associação baseada na métrica de Interseção sobre União (IoU) ou na similaridade de aparência entre a detecção e a trajetória prevista (ZHANG, Y. et al., 2022).

O passo distintivo do ByteTrack acontece na segunda fase. As trajetórias que permaneceram sem correspondência após a primeira etapa (frequentemente representando objetos que sofreram oclusão) são, então, comparadas com o conjunto de *bounding boxes* de baixa confiança, que haviam sido descartadas anteriormente. Essa segunda tentativa de associação busca conectar essas trajetórias "perdidas" às detecções de baixa confiança que apresentem maior sobreposição espacial, geralmente utilizando a métrica IoU como critério de correspondência (ZHANG, Y. et al., 2022).

BoT-SORT

O algoritmo BoT-SORT foi proposto visando aprimorar a robustez e a precisão do rastreamento MOT, especialmente em cenários desafiadores que incluem movimento da câmera (AHARON; ORFAIG; BOBROVSKY, 2022). Ele parte dos fundamentos de rastreadores anteriores como SORT (BEWLEY et al., 2016) e DeepSORT (WOJKE; BEWLEY; PAULUS, 2017), combinando modelagem de movimento e características de aparência, mas introduzindo refinamentos importantes.

Assim como seus predecessores, o BoT-SORT emprega um Filtro de Kalman para estimar o estado (posição e velocidade) de cada objeto e prever sua localização no quadro seguinte. Adicionalmente, ele integra o uso de características de aparência (Re-ID), extraídas por uma rede neural profunda, para calcular a similaridade visual entre detecções e trajetórias, auxiliando na re-identificação após oclusões ou interações (AHARON; ORFAIG; BOBROVSKY, 2022).

A etapa de associação de dados no BoT-SORT utiliza, portanto, uma combinação ponderada dessas informações de movimento (predição do KF) e aparência (similaridade Re-ID), frequentemente utilizando a métrica IoU para avaliar a proximidade espacial. O algoritmo também incorpora explicitamente mecanismos para Compensação de Movimento da Câmera (CMC) (AHARON; ORFAIG; BOBROVSKY, 2022), um recurso importante para vídeos com câmera não estática, embora menos impactante em cenários de câmeras fixas, como o deste trabalho. Essa fusão de múltiplas pistas visa alcançar um rastreamento mais estável e confiável na manutenção das identidades.

HybridSORT

O HybridSORT introduz uma estratégia distinta para o MOT, caracterizada por um *framework* que propõe o rastreamento sem uma fase de associação de dados explícita entre detecções e trajetórias. Diferentemente da maioria dos métodos de rastreamento por detecção (como SORT, DeepSORT e ByteTrack), que utilizam módulos específicos para calcular similaridades e executar a correspondência, o HybridSORT visa a simplificação do processo. A ideia é delegar parte da tarefa de manutenção da identidade ao próprio modelo detector de objetos (YANG et al., 2024)

Para isso, a metodologia do HybridSORT aproveita funcionalidades intrínsecas de detectores de objetos recentes, especialmente arquiteturas que utilizam *queries* ou que já incorporam capacidades de segmentação de instância ou rastreamento *end-to-end*. A premissa central é que a identificação de um objeto detectado pode ser mantida ou transmitida entre *frames* adjacentes através das representações internas do detector (como a estabilidade das *queries* ou *embeddings* de instância associados a um objeto), evitando assim a necessidade de um complexo componente externo de associação de dados (YANG et al., 2024)

Embora não realize associação explícita quadro a quadro, o HybridSORT ainda precisa enfrentar desafios como oclusões e falhas de detecção. Nestas situações, a estratégia predominante baseia-se na capacidade do detector de identificar novamente o objeto quando ele se torna visível. Características de aparência (Re-ID) podem ter um papel neste contexto, mas sua função é geralmente auxiliar: ajudar a reconectar trajetórias que foram interrompidas, servindo para confirmar se uma nova detecção pertence a uma identidade perdida, em vez de atuar como mecanismo primário de associação (YANG et al., 2024).

A vantagem principal do HybridSORT reside na sua potencial simplificação e no ganho de eficiência para o processo de rastreamento. Ao diminuir a dependência de heurísticas e algoritmos complexos de associação, ele busca tirar proveito direto dos avanços contínuos nas capacidades dos próprios detectores de objetos modernos (YANG et al., 2024)

Deep OC-SORT

O algoritmo Deep OC-SORT representa uma evolução do rastreador OC-SORT (CAO et al., 2023), introduzindo o uso de características de aparência profundas para aumentar a robustez do rastreamento de múltiplos objetos (MAGGIOLINO et al., 2023). Este método adota o princípio *Observation-Centric* (centrado na observação), uma estratégia que busca mitigar a deriva do Filtro de Kalman e corrigir inconsistências, especialmente em situações de oclusão ou movimentos não lineares (CAO et al., 2023). Ao priorizar as detecções (observações) mais recentes para ajustar o estado das trajetórias, reduz-se a dependência exclusiva das previsões do KF, que podem acumular erros e se tornar imprecisas ao longo do tempo (MAGGIOLINO et al., 2023).

A principal inovação, refletida no termo "Deep", é a incorporação de descritores visuais gerados por redes de Re-Identificação (Re-ID). Estas características de aparência adicionam uma fonte complementar de informação ao processo de associação de dados. Elas são particularmente úteis para restabelecer corretamente a identidade de objetos após longos períodos de oclusão, momentos em que a comparação visual pode ser mais eficaz para distinguir indivíduos do que apenas a proximidade espacial prevista (MAGGIOLINO et al., 2023).

Assim, o Deep OC-SORT integra duas abordagens de associação: a lógica de correspondência centrada na observação, herdada e aprimorada do OC-SORT (incluindo mecanismos para lidar com incertezas de movimento), e a correspondência baseada na similaridade das características Re-ID. Espera-se que esta combinação se beneficie da robustez do OC-SORT contra erros na estimação de movimento e, ao mesmo tempo, utilize as características profundas (*deep features*) para resolver ambiguidades visuais, tornando o rastreador mais efetivo em cenários complexos (MAGGIOLINO et al., 2023).

2.5 Métricas de Avaliação

A avaliação quantitativa do desempenho é crucial para comparar objetivamente algoritmos de detecção e rastreamento de objetos e para compreender suas respectivas forças e limitações. Nesta seção, são apresentadas as métricas padrão consolidadas na literatura e utilizadas neste trabalho para avaliar separadamente a etapa de detecção e a etapa de rastreamento MOT.

2.5.1 Métricas para Detecção de Objetos

As métricas de detecção quantificam a capacidade de um modelo em localizar e classificar corretamente os objetos de interesse em imagens ou *frames* de vídeo.

Interseção sobre União (IoU)

Antes de definir as métricas principais, é essencial introduzir a Interseção sobre União (IoU - *Intersection over Union*). A IoU mede o grau de sobreposição entre duas caixas delimitadoras (*bounding boxes*): a caixa predita pelo detector (B_p) e a caixa verdadeira anotada no *ground truth* (B_{gt}). É calculada como a razão entre a área da interseção e a área da união das duas caixas (EVERINGHAM et al., 2010):

$$\text{IoU}(B_p, B_{gt}) = \frac{\text{área}(B_p \cap B_{gt})}{\text{área}(B_p \cup B_{gt})}. \quad (2.3)$$

O valor da IoU varia entre 0 (sem sobreposição) e 1 (sobreposição total). Um limiar (*threshold*) de IoU é usado para determinar se uma detecção é considerada correta espacialmente (Everingham et al. (2010) utiliza um *threshold* de 0.5), classificando-a como Verdadeiro Positivo ou Falso Positivo.

Precisão e Recall

Para calcular a Precisão e o *Recall* no contexto da detecção de objetos, as predições do modelo são primeiro comparadas com as anotações verdadeiras (*ground truth*). Utilizando um limiar de Interseção sobre União (IoU) para avaliar a qualidade da localização, cada detecção é classificada como:

- **Verdadeiro Positivo (TP - *True Positive*):** Uma detecção que corresponde corretamente a um objeto real presente no *ground truth*. Isso ocorre quando a *bounding box* predita tem IoU com a caixa verdadeira acima do limiar estabelecido e a classe predita coincide com a classe verdadeira;
- **Falso Positivo (FP - *False Positive*):** Uma detecção incorreta. Isso pode ser uma detecção onde não existe objeto (*background*), uma detecção de um objeto real mas com IoU abaixo do limiar ou uma detecção com a classe incorreta;
- **Falso Negativo (FN - *False Negative*):** Refere-se a um objeto real presente no *ground truth* que o detector falhou em detectar.

A avaliação do detector frequentemente se baseia na contagem de Verdadeiros Positivos (TP), Falsos Positivos (FP) e Falsos Negativos (FN) para cada classe de objeto. Com base nessas contagens, calculam-se a Precisão e o *Recall* (EVERINGHAM et al., 2010):

- **Precisão:** Mede a fração de detecções feitas pelo modelo que são verdadeiramente corretas (TP) em relação ao total de detecções realizadas (TP + FP). Indica a capacidade do modelo de evitar detecções erradas (falsos positivos)

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \quad (2.4)$$

- **Recall:** Mede a fração de objetos reais presentes no *ground truth* que foram corretamente detectados pelo modelo (TP) em relação ao total de objetos reais (TP + FN). Indica a capacidade do modelo de encontrar todos os objetos relevantes, evitando falsos negativos

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (2.5)$$

Tipicamente, existe um *trade-off* entre Precisão e *Recall*: ajustar o limiar de confiança do detector para aumentar uma geralmente resulta na diminuição da outra. Para visualizar essa relação para diferentes limiares de confiança, tipicamente utiliza-se a Curva *Precision-Recall* (Curva PR).

Mean Average Precision (mAP)

O *Mean Average Precision* (mAP) é a métrica padrão consolidada para sumarizar o desempenho global de um detector de objetos em múltiplas classes, sendo amplamente utilizada em desafios como PASCAL VOC (EVERINGHAM et al., 2010) e COCO (LIN et al., 2014). O cálculo do mAP baseia-se na curva *Precision-Recall* (PR). Para cada classe, calcula-se a *Average Precision* (AP), que representa a área sob a curva PR (ou uma interpolação dela), fornecendo uma medida única que considera ambos *Precision* e *Recall* em diferentes níveis de confiança. O mAP é, então, a média dos valores de AP calculados para todas as classes de interesse. Em geral, um mAP mais elevado indica um detector de objetos com melhor desempenho geral.

O cálculo exato do AP está atrelado aos critérios de classificação de TP/FP (notavelmente o limiar de IoU), os quais podem diferir conforme o protocolo de avaliação adotado. Os mais utilizados são:

- **mAP@0.5 (ou mAP₅₀):** Utiliza um limiar de IoU fixo de 0.5 para considerar uma detecção como Verdadeiro Positivo (TP). É a métrica principal do desafio PASCAL VOC. Um valor alto indica que o detector é bom em encontrar os objetos, mesmo que a localização da caixa não seja perfeitamente precisa;
- **mAP@0.5:0.95 (ou mAP_{COCO}):** É a métrica principal do desafio COCO e considerada mais rigorosa. Ela calcula o AP para dez limiares de IoU diferentes (de 0.5 a 0.95, com passo de 0.05) e faz a média desses valores. Um valor alto nesta métrica indica que o detector é bom não apenas em encontrar os objetos, mas também em localizar suas *bounding boxes* com alta precisão espacial.

2.5.2 Métricas para Rastreamento de Múltiplos Objetos (MOT)

As métricas de rastreamento MOT avaliam a capacidade de um algoritmo em manter corretamente as identidades e trajetórias dos objetos ao longo de uma sequência de vídeo.

Essas métricas vão além da simples detecção por quadro, focando na consistência temporal. Métricas padronizadas são essenciais para a comparação justa entre diferentes algoritmos, sendo o MOTChallenge (MILAN et al., 2016; DENDORFER et al., 2021) um dos principais *benchmarks*, que estabelece e utiliza muitas das métricas aqui descritas.

Multiple Object Tracking Accuracy (MOTA)

A métrica MOTA (*Multiple Object Tracking Accuracy*) é uma das mais utilizadas e consolidadas para avaliação de MOT. Proposta originalmente no contexto das métricas CLEAR (STIEFELHAGEN et al., 2006), ela agrega três fontes principais de erro em um único *score*, calculado sobre todos os quadros de um vídeo, são elas: Falsos Positivos, Falsos Negativos e Trocas de Identidade. A fórmula é dada por:

$$\text{MOTA} = 1 - \frac{\sum_t (\text{FN}_t + \text{FP}_t + \text{IDSW}_t)}{\sum_t \text{GT}_t}, \quad (2.6)$$

onde t é o índice do quadro, FN_t são os falsos negativos no quadro t , FP_t são os falsos positivos no quadro t , IDSW_t são as trocas de identidade no quadro t , e GT_t é o número de objetos verdadeiros (*ground truth*) no quadro t .

O MOTA varia de $-\infty$ a 1 (ou 100%), com valores mais altos indicando melhor desempenho. Ele penaliza tanto erros de detecção (FP, FN) quanto erros de manutenção de identidade (IDSW).

Identity F1 Score (IDF1)

Enquanto o MOTA oferece uma visão geral da precisão do rastreamento quadro a quadro, o IDF1 (*Identity F1 Score*) (RISTANI et al., 2016) concentra-se especificamente na avaliação da consistência da identidade ao longo das trajetórias completas. Ele avalia quão bem as trajetórias preditas (*hypotheses*) correspondem aos *ground truth* em termos de manutenção correta dos IDs. O IDF1 é a média harmônica da Precisão de Identidade (IDP) e do Recall de Identidade (IDR):

$$\text{IDP} = \frac{\text{IDTP}}{\text{IDTP} + \text{IDFP}}, \quad (2.7)$$

$$\text{IDR} = \frac{\text{IDTP}}{\text{IDTP} + \text{IDFN}}, \quad (2.8)$$

$$\text{IDF1} = \frac{2 \times \text{IDTP}}{2 \times \text{IDTP} + \text{IDFP} + \text{IDFN}} = \frac{2 \times \text{IDP} \times \text{IDR}}{\text{IDP} + \text{IDR}}, \quad (2.9)$$

onde IDTP, IDFP e IDFN representam respectivamente os Verdadeiros Positivos, Falsos Positivos e Falsos Negativos, calculados ao nível das associações de identidade corretas ao longo de toda a sequência. O IDF1 varia de 0 a 1 (ou 100%) e é particularmente relevante

para aplicações onde é crucial manter o ID correto de cada indivíduo.

Identity Switches (IDSW)

A métrica IDSW (*Identity Switches*), que também parte das métricas CLEAR MOT (BERNARDIN; STIEFELHAGEN, 2008), quantifica o número total de vezes, ao longo do vídeo, que a identidade associada a uma trajetória rastreada é incorretamente trocada para a identidade de outra trajetória existente ou para uma nova identidade, que não é legítima. Essa métrica penaliza diretamente falhas na manutenção da consistência da identidade. Um número menor de IDSW indica um rastreador mais confiável na identificação individual, sendo este um dos componentes que impactam negativamente o cálculo do MOTA.

Fragmentations (Frag)

Similarmente ao IDSW, a métrica de Fragmentações (Frag) (BERNARDIN; STIEFELHAGEN, 2008) contabiliza o número total de vezes que uma trajetória do *ground truth* é interrompida pelo rastreador (i.e., o objeto é perdido temporariamente) e depois retomada com a mesma identidade correta. Cada interrupção conta como uma fragmentação. Embora a identidade possa ser recuperada, um número elevado de fragmentações indica um rastreamento descontínuo ou instável. Portanto, um menor número de fragmentações é ideal.

Precisão e Recall em Rastreamento

Os conceitos de Precisão e *Recall*, fundamentais na avaliação de detecção, também são aplicados no contexto de rastreamento, embora suas definições exatas possam variar ou serem parcialmente capturadas por outras métricas como MOTA e IDF1. Geralmente, no contexto de MOT:

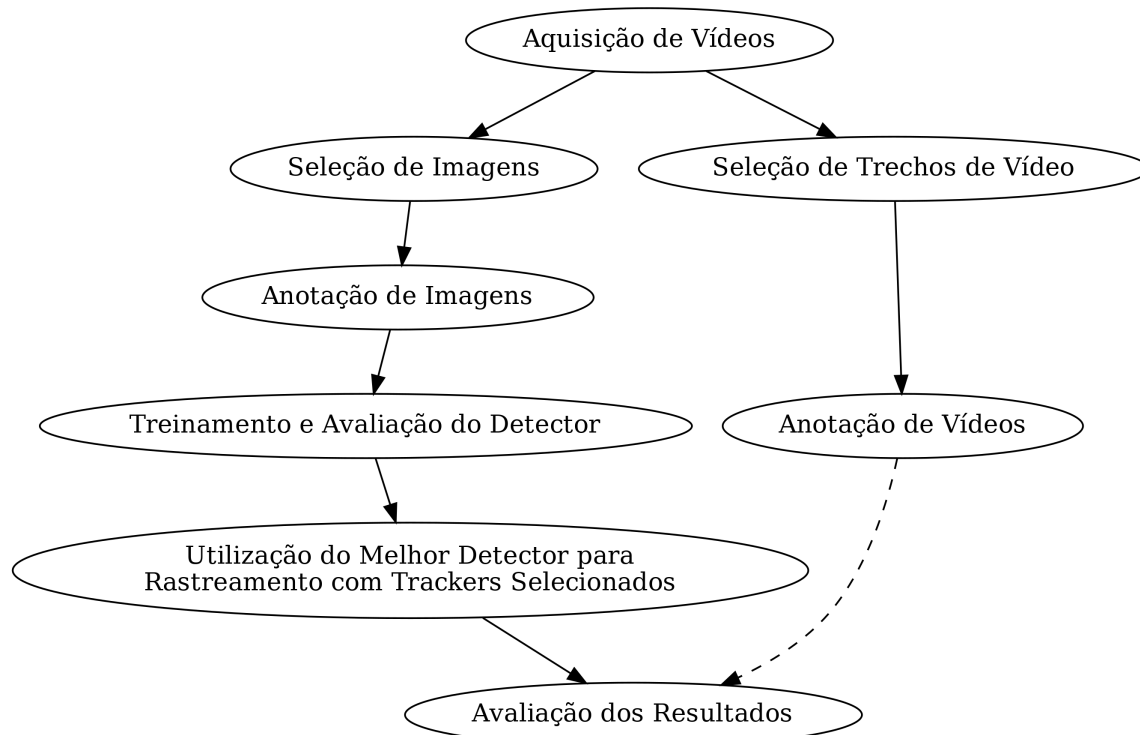
- ***Tracking Precision*** relaciona-se à fração de trajetórias (ou segmentos) preditas que são corretas, minimizando Falsos Positivos.
- ***Tracking Recall*** relaciona-se à fração de trajetórias verdadeiras que foram efetivamente rastreadas (pelo menos em parte), minimizando Falsos Negativos.

3

Metodologia

A metodologia adotada neste trabalho utiliza um *pipeline* lógico e estruturado, iniciando com a aquisição de dados, passando pela seleção e anotação de imagens e vídeos, seguindo para o treinamento e validação do modelo de detecção de objetos, sua aplicação, o subsequente rastreamento dos objetos detectados e, finalmente, a avaliação, tanto quantitativa quanto qualitativa, dos resultados obtidos. A Figura 3.1 apresenta esquematicamente este fluxo metodológico.

Figura 3.1: Visão geral da nossa abordagem.



Fonte: elaborado pelo autor.

3.1 Aquisição e características das imagens

O ponto de partida deste estudo considera imagens geradas por câmeras fixas, estrategicamente posicionadas no centro do teto de cada baia de cães monitorada, conforme Figura 1.2. Este posicionamento é considerado vantajoso por oferecer uma perspectiva constante (vista superior ou *top-view*), maximizar a área de cobertura visual do ambiente e contribuir para a redução significativa de oclusões entre os animais. As câmeras utilizadas produzem imagens coloridas no padrão RGB. Pressupõe-se que a resolução das imagens seja suficiente para a detecção confiável dos objetos, influenciando a capacidade de discernir detalhes importantes. Adicionalmente, como requisito para a etapa de rastreamento, assume-se que a taxa de quadros (*frame rate*) das gravações seja suficientemente alta para capturar a dinâmica dos animais com continuidade e precisão entre quadros consecutivos.

3.2 Compilação dos dados supervisionados

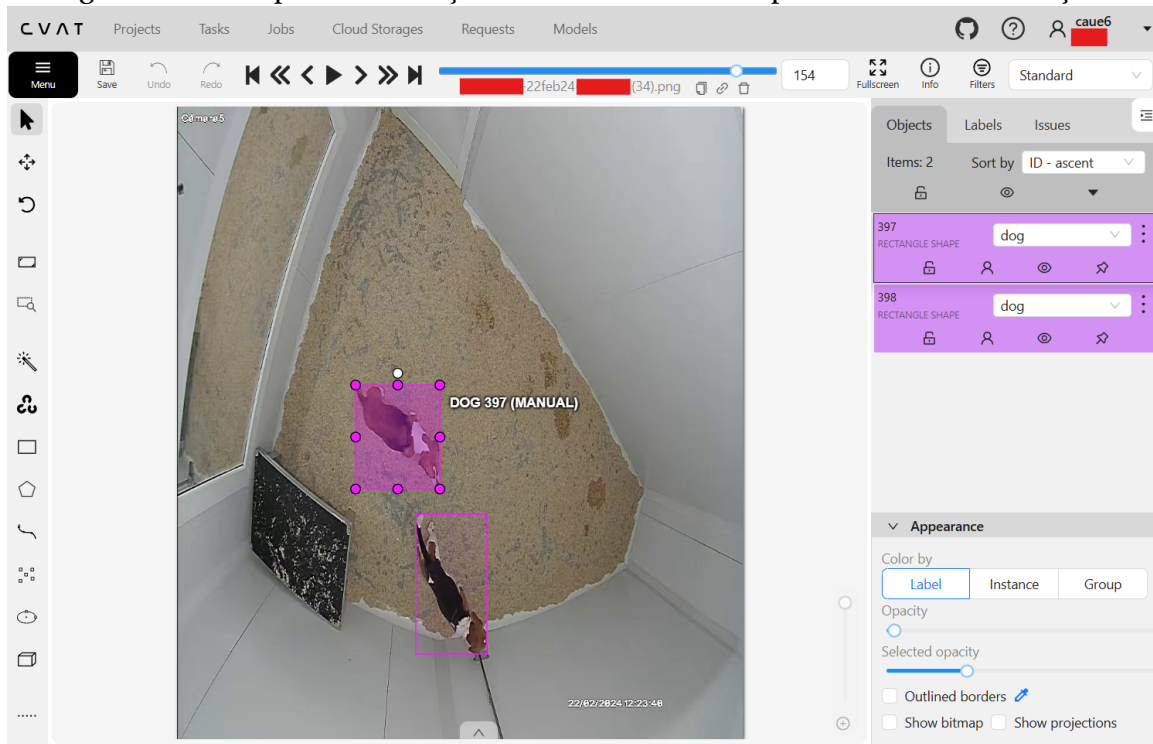
Para viabilizar o treinamento e a avaliação dos modelos de aprendizado de máquina propostos, a metodologia requer a construção de duas bases de dados distintas, ambas supervisionadas. A primeira base de dados necessária é uma composta por imagens (*frames* individuais de vídeos), que servirá para o treinamento, validação e teste do detector de objetos baseado na YOLOv11. Já a segunda base de dados é composta por vídeos, também anotados, destinada especificamente à avaliação do desempenho dos algoritmos de rastreamento em cenários dinâmicos, representativos das condições reais de uso.

A criação das anotações *ground truth* para ambas as bases de dados envolve um processo de anotação manual. Este processo consiste na identificação visual dos objetos de interesse (cães) em cada imagem ou *frame* de vídeo e na marcação precisa de suas localizações através de caixas delimitadoras (*bounding boxes*). Utilizaram-se ferramentas especializadas em anotação visual, como o *Computer Vision Annotation Tool* (CVAT), ilustrado pela Figura 3.2, para facilitar esta tarefa. Cada *bounding box* é definida como a menor caixa retangular que engloba toda a porção visível do cão. Adotou-se como regra para casos de oclusão parcial a anotação somente da parte visível do animal. O mesmo procedimento de anotação é aplicado aos vídeos destinados à avaliação do rastreamento, garantindo a consistência entre as bases.

3.3 Detecção de Objetos (Cães)

A abordagem de detecção de objetos empregada neste trabalho baseia-se em modelos de Redes Neurais Convolucionais (CNNs) pertencentes à família YOLO (*You Only Look Once*), cuja fundamentação teórica é detalhada na seção 2.3.1. A escolha por esta família de modelos justifica-se pela sua reconhecida capacidade de combinar alta precisão na localização es-

Figura 3.2: Exemplo de utilização da ferramenta CVAT para a tarefa de anotação.



Fonte: elaborado pelo autor.

pacial dos objetos com notável eficiência computacional, fatores que viabilizam potenciais aplicações em tempo real. A versão específica do modelo YOLO e a estratégia de treinamento são definidas e detalhadas na Seção 4.2.

Para avaliar o desempenho do detector de objetos de forma quantitativa, serão utilizadas métricas padrão consolidadas na literatura de Visão Computacional. As principais métricas incluem a *mean Average Precision* (mAP), calculada em diferentes limiares de Interseção sobre União (IoU) como mAP@0.5 e mAP@0.5:0.95, além da Precisão e *Recall*. Essas métricas permitem uma avaliação objetiva da capacidade do modelo em detectar e localizar corretamente os objetos de interesse (cães).

3.4 Rastreamento de Múltiplos Objetos (MOT)

Para o rastreamento dos animais ao longo do tempo nos vídeos, adota-se o paradigma de *tracking-by-detection*. Nesta abordagem, as trajetórias individuais dos objetos são construídas sequencialmente, quadro a quadro, a partir das detecções fornecidas por um modelo de detecção previamente treinado (conforme Seção 3.3). O desafio central desta técnica é a implementação de algoritmos capazes de associar corretamente as detecções de um mesmo objeto ao longo dos quadros, mantendo a consistência de sua identidade (ID) ao longo do tempo, mesmo em cenários complexos, com múltiplos indivíduos e interações.

A avaliação objetiva do desempenho dos algoritmos de rastreamento é realizada através

de um conjunto de métricas padrão da área de Rastreamento de Múltiplos Objetos (MOT), conforme proposto em desafios como o MOTChallenge. As métricas consideradas neste trabalho, e que foram melhor detalhadas na Seção 2.5.2 são: MOTA, IDF1, IDSW, Frag, Precisão e *Recall*.

A análise conjunta destas métricas fornece uma visão detalhada e multifacetada do desempenho dos algoritmos, permitindo avaliar não só a precisão geral (MOTA), mas também a qualidade da associação de identidades (IDF1, IDSW) e a continuidade das trajetórias (Frag), além da capacidade de detecção ao nível do rastreador (Precision, Recall). Essas métricas são cruciais para assegurar que a avaliação reflita o desempenho real dos algoritmos em condições práticas de monitoramento.



Experimentos e Resultados

Neste capítulo, são detalhados os procedimentos experimentais executados para implementar a metodologia descrita no Capítulo 3, incluindo a compilação e caracterização das bases de dados, o treinamento e avaliação do detector, e a avaliação comparativa dos algoritmos de rastreamento. Na sequência, são apresentados, analisados e discutidos os resultados quantitativos e qualitativos obtidos nestes experimentos.

4.1 Compilação e Caracterização dos *Datasets*

Para a execução dos experimentos, foram necessários conjuntos de dados específicos para as tarefas de detecção e rastreamento. Nesta seção, detalha-se a construção e as características desses dois conjuntos de dados, cuja criação seguiu os procedimentos definidos na Metodologia (Seção 3.2).

Os dados brutos (vídeos) que serviram de base para a construção dos *datasets* nesta pesquisa foram obtidos através de uma parceria entre o laboratório e uma empresa do setor privado. Devido a cláusulas contratuais de confidencialidade e direitos de propriedade sobre o material original, o conjunto de dados gerado e anotado não pode ser disponibilizado publicamente. No entanto, toda a metodologia empregada para a compilação, anotação e divisão dos dados é detalhadamente descrita neste capítulo, visando a máxima transparência e a replicabilidade do processo.

4.1.1 *Dataset* para Detecção de Objetos (Imagens)

A criação dos conjuntos de dados para treinamento, validação e teste do detector de objetos partiu de um acervo de gravações originárias de 13 câmeras fixas distintas, localizadas em 11 baías diferentes, coletadas em 4 períodos entre julho de 2023 e julho de 2024. As capturas ocorreram sob iluminação artificial no intervalo entre 11h45 e 17h30. É importante

notar que, embora a iluminação dentro de uma mesma baia fosse constante, a utilização de baias distintas introduziu variações de luminosidade (intensidade, tonalidade) no conjunto de dados, enriquecendo-o e contribuindo para a robustez do modelo treinado.

Seguindo a metodologia, cada baia foi equipada com uma câmera fixa centralizada no teto (conforme exemplo na Figura 1.2), capturando imagens RGB com resolução de 960x1072 pixels e a uma taxa fixa de 25 quadros por segundo (FPS). Foi constatada uma leve distorção esférica (tipo "olho de peixe") nas imagens, inerente à lente utilizada. Optou-se por não realizar uma etapa de pré-processamento para correção dessa distorção, confiando-se na capacidade de arquiteturas como YOLO para aprender a lidar com tais variações geométricas diretamente a partir dos dados.

Para avaliar a generalização do modelo para câmeras não vistas durante o treino, aplicou-se uma estratégia de divisão baseada na câmera de origem: 8 câmeras foram exclusivamente alocadas para Treinamento, 3 para Validação e 2 para Teste. A partir desta divisão, realizou-se um processo de seleção manual dos quadros para compor os conjuntos, buscando assegurar diversidade de cenários (contemplando diferentes cães, condições de iluminação e níveis de ocupação das baias). Os conjuntos resultantes foram:

- **Conjunto de Treinamento:** 1133 imagens (aprox. 72%);
- **Conjunto de Validação:** 184 imagens (aprox. 12%);
- **Conjunto de Teste:** 265 imagens (aprox. 17%).

Todas as imagens selecionadas foram então anotadas manualmente com *bounding boxes* via CVAT, seguindo as diretrizes da Seção 3.2, e as anotações exportadas no formato YOLO 1.1 ¹.

4.1.2 Dataset para Avaliação de Rastreamento (Vídeos)

Para a avaliação específica dos algoritmos de rastreamento, foi construído um *dataset* distinto, utilizando gravações contínuas realizadas em julho de 2023 e setembro de 2024. Estas gravações são provenientes de 6 registros específicos de câmeras diferentes das utilizadas no *dataset* de imagens, originários de 5 baias distintas (identificadas como Baias 5, 6, 7, 8 no período de 2023; e Baias 8, 9 no período de 2024 - Baia 8 contribuiu em ambos os períodos). As especificações das câmeras (posição, RGB, resolução e FPS) foram mantidas consistentes com as da base de detecção. As gravações também foram realizadas sob iluminação artificial, refletindo as mesmas características do dataset de detecção: constância de luz dentro de cada baia, porém com variações entre os diferentes ambientes de coleta.

Desse conjunto de gravações, foram selecionados criteriosamente 11 trechos de vídeo, somando uma duração total de 300 segundos (aproximadamente 7.500 *frames*). A seleção

¹<https://docs.cvat.ai/docs/manual/advanced/formats/format-yolo/>

buscou intencionalmente incluir segmentos que representassem desafios comuns e relevantes para o rastreamento múltiplo de animais, tais como oclusões parciais e totais entre cães, movimentos rápidos e erráticos e interações sociais complexas. A duração total foi dividida equitativamente entre os dois períodos de coleta (150 segundos de julho de 2023 e 150 segundos de setembro de 2024).

Dado o volume considerável de *frames*, o processo de anotação para este *dataset* de rastreamento foi otimizado utilizando uma abordagem semi-automática. Empregou-se o modelo *Segment Anything Model 2* (SAM2) Ravi et al. (2025) para gerar propostas iniciais de *bounding boxes* para os cães em cada frame. Estas pré-anotações foram subsequentemente importadas para a ferramenta CVAT, onde cada *bounding box* passou por um processo de verificação e ajuste fino manual, garantindo também a consistência dos IDs ao longo do tempo. As anotações finais foram exportadas no formato MOT 1.1 ².

4.2 Treinamento, Avaliação e Seleção do Detector

Nesta seção, descreve-se o processo de treinamento e avaliação do detector de objetos YOLO11x, utilizando o *dataset* de imagens detalhado na Seção 4.1.1. São apresentadas a configuração do treinamento, a avaliação quantitativa e qualitativa do desempenho e a metodologia de seleção do modelo final utilizado na etapa de rastreamento.

4.2.1 Configuração do Treinamento

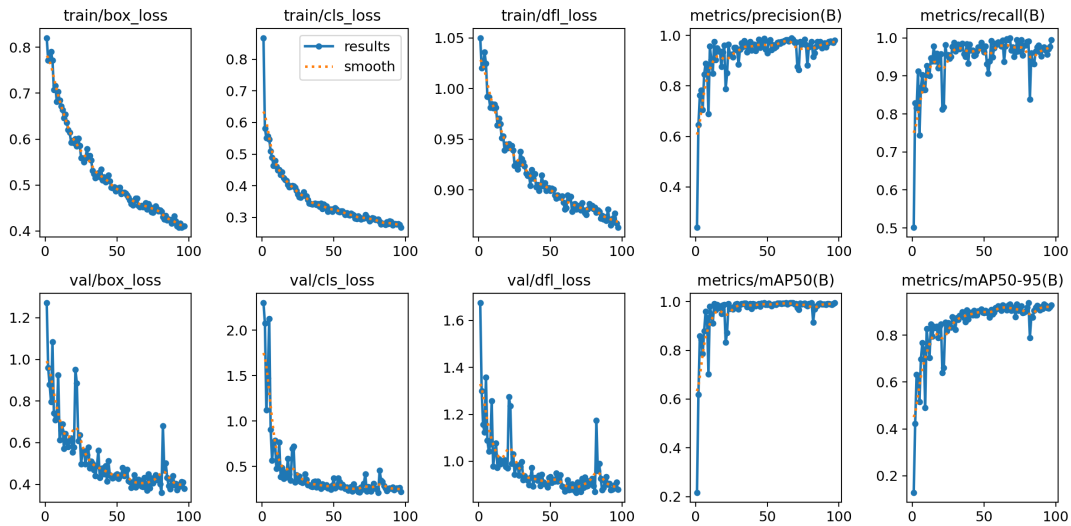
O modelo de detecção selecionado para este trabalho foi o YOLO11x, disponibilizado pela biblioteca Ultralytics (JOCHER; QIU; CHAURASIA, 2023). Como passo preliminar, avaliou-se a performance do modelo YOLO11x pré-treinado no dataset COCO ao aplicá-lo diretamente na base de imagens, sem qualquer treinamento adicional. Empiricamente, verificou-se que o desempenho foi inadequado para os objetivos da pesquisa. Embora o modelo identificasse a classe "cão", a precisão das *bounding boxes* era baixa e o número de detecções perdidas (Falsos Negativos) era alto, especialmente devido à perspectiva superior (*top-down*), que difere substancialmente das imagens do *dataset* COCO (LIN et al., 2014).

Dado o tamanho do conjunto de treinamento (1133 imagens), considerado relativamente limitado para treinar um modelo desta complexidade a partir do zero, empregou-se a estratégia de *fine-tuning*. Esta técnica consistiu em inicializar o modelo com pesos pré-treinados em um *dataset* de larga escala, COCO, e prosseguir o treinamento utilizando especificamente o nosso conjunto de dados. Tal abordagem permite que o modelo aproveite o conhecimento prévio sobre extração de características visuais e se adapte mais eficientemente à tarefa de detectar cães nas condições particulares das baías monitoradas.

²<https://docs.cvat.ai/docs/manual/advanced/formats/format-mot/>

Para otimizar o treinamento, utilizou-se o otimizador AdamW (LOSHCHILOV; HUTTER, 2019), realizando ajustes em hiperparâmetros chave como a taxa de aprendizado inicial ($lr0$) e o *momentum*. Foi empregada também a técnica de *early stopping* baseada em *patience*, que monitora o desempenho no conjunto de validação (184 imagens) para interromper o treinamento precocemente se não houver melhoria, evitando *overfitting*. Isso resultou na conclusão dos treinamentos em menos de 100 épocas, conforme ilustrado pelos gráficos de acompanhamento na Figura 4.1, embora o máximo configurado fosse 200. Como parte da busca pelos melhores hiperparâmetros, foram treinadas 12 variantes do modelo, ajustando-se parâmetros chave como a taxa de aprendizado inicial ($lr0$), *momentum* e a *patience* para o mecanismo de *early stopping*. Cada treinamento gerou no fim um melhor *checkpoint* para a configuração utilizada.

Figura 4.1: Gráficos das métricas de treinamento e validação ao longo das épocas para um dos treinamentos realizados.



Fonte: Saída do treinamento do modelo YOLO11x (Ultralytics).

4.2.2 Avaliação de Desempenho e Seleção do *Checkpoint* Final

Após o treinamento das variantes do modelo, procedeu-se à avaliação de desempenho para selecionar o melhor modelo a ser utilizado na etapa de rastreamento. Esta avaliação considerou tanto o desempenho isolado do detector quanto seu impacto na tarefa final.

Primeiramente, avaliou-se a performance *standalone* dos *checkpoints* no conjunto de validação (184 imagens) e no conjunto de teste (265 imagens). A Tabela 4.1 detalha os hiperparâmetros e o desempenho de detecção resultante para todas as 12 variantes do modelo.

A análise da Tabela 4.1 evidencia que múltiplos *checkpoints*, como o train11 e train16, também alcançaram alta performance de detecção. Contudo, como o objetivo final é o rastreamento eficaz, e métricas como mAP nem sempre se correlacionam perfeitamente com o desempenho de rastreamento, realizou-se uma avaliação adicional focada no impacto no

Tabela 4.1: Desempenho *Standalone* dos Detectores no Conjunto de Teste

<i>Checkpoint</i>	Precisão (P)	Recall (R)	mAP@50	mAP@50-95
train2	0.980	0.975	0.989	0.891
train3	0.823	0.785	0.839	0.599
train4	0.823	0.785	0.839	0.599
train8	0.762	0.935	0.922	0.743
train9	0.925	0.911	0.956	0.772
train10	0.936	0.934	0.955	0.795
train11	0.984	0.977	0.991	0.898
train12	0.953	0.955	0.978	0.807
train13	0.966	0.966	0.985	0.869
train14	0.886	0.925	0.955	0.817
train15	0.950	0.964	0.980	0.850
train16	0.973	0.955	0.982	0.911

* Valores em **negrito** na linha "train13" indicam o *checkpoint* selecionado como modelo final. Os valores em **negrito** nas demais linhas indicam, para cada métrica, os maiores valores globais observado entre as variantes.

rastreamento. Os *checkpoints* candidatos foram combinados com os quatro algoritmos de rastreamento considerados (ByteTrack, BoT-SORT, HybridSORT, Deep OC-SORT) e avaliados no dataset de vídeos. A Tabela 4.2 apresenta os resultados de MOTA e IDF1.

A análise da Tabela 4.2 demonstra que, embora o *checkpoint* train11 tenha alcançado o maior valor global de IDF1 (74.4% com HybridSORT), o *checkpoint* train13 apresentou um desempenho geral mais robusto e equilibrado. O train13 não só atingiu o maior MOTA global da avaliação (87.4% com BoT-SORT), como também manteve um desempenho altamente competitivo em IDF1 (67.3% com HybridSORT). Essa combinação de alta acurácia geral (MOTA) com uma forte, ainda que não máxima, consistência de identidade (IDF1) o tornou o candidato mais adequado como detector final, garantindo uma base sólida e versátil para a avaliação comparativa dos algoritmos de rastreamento.

Complementarmente, realizou-se uma análise qualitativa das detecções no conjunto de teste. As Figuras 4.2 a 4.5 ilustram exemplos de detecções bem-sucedidas em diferentes cenários (indicados entre parênteses no título da figura em questão), mostrando a robustez do detector a variações de pose, proximidade, contraste e iluminação. Por outro lado, as Figuras 4.6 e 4.7 exemplificam algumas dificuldades recorrentes. Notou-se maior dificuldade em localizar precisamente cães sobre uma estrutura de suporte preta de baixo contraste (Figura 4.6) e na distinção/separação de cães em contato físico ou muito próximos (Figura 4.7), o que pode levar a *bounding boxes* imprecisas ou sobrepostas.

Portanto, com base na combinação de um desempenho *standalone* satisfatório (Tabela 4.1) e, principalmente, na performance superior na tarefa final de rastreamento (Tabela 4.2), o *checkpoint* train13 do modelo YOLO11x foi selecionado como o detector final

Tabela 4.2: Comparação do Desempenho de Rastreamento (MOTA / IDF1 em %) utilizando os *Checkpoints* Candidatos do Detector YOLO11x.

<i>Checkpoint</i> Detector	ByteTrack (MOTA / IDF1)	BoT-SORT (MOTA / IDF1)	HybridSORT (MOTA / IDF1)	Deep OC-SORT (MOTA / IDF1)
train2	72.7 / 49.6	80.7 / 64.2	78.1 / 60.5	76.0 / 51.4
train3	32.9 / 37.4	53.6 / 50.3	48.5 / 43.5	38.0 / 39.9
train4	32.9 / 37.4	53.6 / 50.3	48.5 / 43.5	38.0 / 39.9
train8	33.4 / 39.9	63.1 / 53.8	55.6 / 54.5	31.5 / 44.9
train9	49.4 / 42.8	60.1 / 53.9	58.8 / 55.3	40.6 / 42.2
train10	74.1 / 58.0	82.5 / 59.7	79.1 / 68.5	77.3 / 60.3
train11	67.5 / 54.8	77.5 / 61.0	76.8 / 74.4	72.7 / 56.6
train12	76.5 / 50.2	84.2 / 60.2	83.2 / 67.8	81.7 / 55.7
train13	78.8 / 53.7	87.4 / 63.7	85.0 / 67.3	83.5 / 53.2
train14	50.5 / 45.0	76.1 / 62.1	62.5 / 57.3	50.3 / 49.3
train15	51.9 / 48.1	59.1 / 53.8	57.5 / 58.3	55.5 / 51.9
train16	65.8 / 54.8	73.7 / 59.9	70.5 / 71.1	69.0 / 59.5

* Valores em **negrito** na linha "train13" indicam o *checkpoint* selecionado como modelo final. O valor em **negrito** na coluna HybridSORT da linha "train11" indica o maior IDF1 global observado entre as variantes.

Figura 4.2: Exemplo de detecção bem-sucedida (pose, borramento).



Fonte: elaborado pelo autor.

Figura 4.3: Exemplo de detecção bem-sucedida (proximidade).



Fonte: elaborado pelo autor.

Figura 4.4: Exemplo de detecção bem-sucedida (nitidez, iluminação).



Fonte: elaborado pelo autor.

Figura 4.5: Exemplo de detecção bem-sucedida (contraste, iluminação).



Fonte: elaborado pelo autor.

Figura 4.6: Exemplo de dificuldade na detecção (baixo contraste com estrutura preta).



Fonte: elaborado pelo autor.

Figura 4.7: Exemplo de dificuldade na detecção (cães muito próximos, baixo contraste).



Fonte: elaborado pelo autor.

para a avaliação comparativa detalhada dos algoritmos de rastreamento discutida a seguir.

4.3 Avaliação Comparativa dos Algoritmos de Rastreamento

Tendo selecionado o detector final com base na análise da seção anterior, esta seção foca na avaliação comparativa dos algoritmos de rastreamento. Detalha-se a configuração utilizada para esta avaliação e, em seguida, apresentam-se e analisam-se os resultados quantitativos e qualitativos obtidos para cada rastreador considerado.

4.3.1 Configuração da Avaliação

Na fase experimental de rastreamento, utilizando as detecções fornecidas pelo modelo YOLO11x selecionado, o foco foi avaliar e comparar o desempenho de quatro algoritmos de *tracking-by-detection* proeminentes e recentes: ByteTrack (ZHANG, Y. et al., 2022), BoT-SORT (AHARON; ORFAIG; BOBROVSKY, 2022), HybridSort (YANG et al., 2024) e Deep-OC-SORT (MAGGIOLINO et al., 2023). Para assegurar uma comparação justa e reprodutível, todos esses rastreadores foram implementados e executados utilizando estritamente suas configurações padrão. Nenhuma otimização de parâmetros específica para o *dataset* deste trabalho foi realizada. Cada combinação detector+rastreador foi aplicada sobre os 11 vídeos (aproximadamente 7.500 *frames*) do *dataset* de rastreamento.

4.3.2 Resultados Quantitativos

O desempenho de cada combinação detector+rastreador foi quantificado utilizando as métricas de MOT definidas na Fundamentação Teórica (Seção 2.5.2), incluindo MOTA, IDF1, número de Trocas de Identidade (IDSW), número de Fragmentações (Frag), Precisão e *Recall*. Os resultados numéricos consolidados para os quatro algoritmos avaliados são apresentados na Tabela 4.3.

Tabela 4.3: Resultados Quantitativos Finais dos Algoritmos de Rastreamento com Detector YOLO11x (train13). Melhor desempenho por métrica em **negrito**.

Rastreador	MOTA (%)	IDF1 (%)	IDSW	Frag	Precisão (%)	Recall (%)
ByteTrack	78.8	53.7	137	234	93.2	87.1
BoT-SORT	87.4	63.7	107	151	97.7	91.0
HybridSORT	85.0	67.3	59	152	97.5	88.1
Deep OC-SORT	83.5	53.2	114	165	97.2	87.6

* Nota: Valores mais altos são melhores para MOTA, IDF1, Precisão e Recall; valores mais baixos são melhores para IDSW e Frag.

4.3.3 Análise Quantitativa Detalhada

A Tabela 4.3 permite uma análise comparativa detalhada do desempenho dos rastreadores neste cenário.

Acurácia Geral (MOTA)

O BoT-SORT alcançou a maior acurácia geral (MOTA = 87.4%), seguido de perto pelo HybridSORT (85.0%) e Deep OC-SORT (83.5%), indicando que estes três lidam bem com o balanço de erros. ByteTrack ficou abaixo (78.8%).

Consistência de Identidade (IDF1, IDSW)

O HybridSORT demonstrou clara superioridade na manutenção da identidade, com o maior IDF1 (67.3%) e o menor número de trocas de ID (IDSW = 59). BoT-SORT teve um bom IDF1 (63.7%) mas com consideravelmente mais trocas (107). Deep OC-SORT e ByteTrack tiveram os menores *scores* de IDF1 (53.2% e 53.7%, respectivamente) e um número elevado de trocas de ID (114 e 137).

Continuidade das Trajetórias (Frag)

BoT-SORT e HybridSORT foram os melhores em manter trajetórias contínuas, com números muito próximos de fragmentações (151 e 152). Deep OC-SORT foi ligeiramente pior (165) e ByteTrack apresentou a maior fragmentação (234).

Precisão/*Recall* do Rastreador

BoT-SORT liderou tanto em Precisão (97.7%) quanto em *Recall* (91.0%), indicando alta confiança nas trajetórias geradas e poucas trajetórias do *ground truth* perdidas. HybridSORT e Deep OC-SORT também tiveram Precisão muito alta (97.5% e 97.2%, respectivamente), mas *Recall* um pouco menor (88.1% e 87.6%). ByteTrack teve a menor Precisão (93.2%) e menor *Recall* (87.1%).

4.3.4 Análise Qualitativa

Complementando as métricas, a análise visual dos vídeos processados oferece alguns *insights* sobre o comportamento dos algoritmos. As Figuras 4.8 e 4.9 ilustram algumas situações encontradas durante a realização desta análise qualitativa.

Na Figura 4.8 é possível visualizar com clareza o comportamento distinto dos rastreadores. O BoT-SORT foi o único rastreador que apresentou as duas *tracks* ativas nos quatro instantes de tempo analisados, um reflexo da sua maior acurácia geral (MOTA de 87.4%) e maior *Recall* (91.0%), contudo não manteve a identidade dos objetos, começando com os IDs #1 e #4 e terminando a sequência com #6 e #7. Já o HybridSort, embora não tenha mantido as *tracks* ativas em todos os instantes, como o BoT-SORT, manteve os IDs corretos (#1 e #2), refletindo sua superioridade na manutenção da identidade, com o maior IDF1 (67.3%) e o menor número de trocas de ID (IDSW = 59). Também é possível observar a notória inferioridade do ByteTrack em relação aos demais. Sua MOTA inferior (78.8%) reflete-se em falhas visíveis, como no segundo e quarto *frames* da sequência, onde gera *bounding boxes* com tamanhos inadequados; além disso, nota-se a maior troca de IDs (IDSW = 137) e o baixo IDF1 (53.7%), começando a sequência com os IDs #5 e #6, e terminando com #11 e #14. O Deep OC-SORT passa um pouco despercebido nesta análise qualitativa, por não alcançar um desempenho tão bom quanto o BoT-SORT ou o HybridSORT, nem ter um desempenho tão baixo quanto o ByteTrack.

Já a Figura 4.9 apresenta um exemplo onde falhas na detecção (com sobreposição de cães, neste caso) impactam o desempenho do rastreamento de maneira significativa. Nota-se que nenhum rastreador conseguiu manter a identidade correta nesse exemplo.

De maneira geral, observou-se que tanto o BoT-SORT quanto o HybridSORT produzem trajetórias visualmente suaves na maior parte do tempo. O HybridSORT mostrou-se robusto em manter os IDs, inclusive durante cruzamentos, enquanto BoT-SORT destacou-se pela cobertura do rastreamento (alto *Recall*). ByteTrack frequentemente apresentou trajetórias mais fragmentadas ou *bounding boxes* menos estáveis. O Deep OC-SORT, apesar de não ter um desempenho tão bom quanto o BoT-SORT e o HybridSORT, apresentou trajetórias mais estáveis que o ByteTrack. Falhas comuns a todos os rastreadores ocorreram em sobreposições complexas e próximo à estrutura preta no canto da baía, indicando áreas onde as detecções de entrada podem falhar, evidenciando que ainda há muito espaço para aprimorar

Figura 4.8: Comparativo visual do rastreamento de objetos pelos algoritmos BoT-SORT, ByteTrack, HybridSORT e Deep OC-SORT. Cada linha apresenta quadros extraídos nos mesmos instantes de tempo (sincronizados) de uma sequência de vídeo (Câmera 9).

BoTSORT



BYTETracker



HybridSORT



DeepOCSORT



Fonte: elaborado pelo autor.

Figura 4.9: Comparativo visual do rastreamento de objetos pelos algoritmos BoT-SORT, ByteTrack, HybridSORT e Deep OC-SORT. Cada linha apresenta quadros extraídos nos mesmos instantes de tempo (sincronizados) de uma sequência de vídeo (Câmera 7).

BoTSORT



BYTETracker



HybridSORT



DeepOCSORT



Fonte: elaborado pelo autor.

tanto os métodos de rastreamento analisados quanto o modelo de detecção selecionado.

4.4 Discussão Geral

Os resultados experimentais apresentados permitem uma discussão consolidada sobre a metodologia, os desafios e as implicações deste trabalho.

4.4.1 Síntese do Desempenho e Validação da Abordagem

Os experimentos confirmaram que o detector YOLO11x, após *fine tuning*, alcançou um nível de desempenho (mAP@50-95 de 0.869) adequado para servir como base para algoritmos de rastreamento de múltiplos objetos no cenário de visão superior de cães em baías. Os altos valores de MOTA (>85%) obtidos com os melhores rastreadores (BoT-SORT e HybridSORT) validam não apenas a qualidade das detecções, mas também a estratégia de selecionar o *checkpoint* final do detector (train13) com base em sua performance na tarefa de rastreamento, otimizando o *pipeline* completo.

4.4.2 Análise Comparativa e Escolha do Melhor Rastreador

A comparação quantitativa e qualitativa entre os rastreadores selecionados revelou performances distintas, com BoT-SORT e HybridSORT destacando-se significativamente dos demais (ByteTrack e Deep OC-SORT) neste cenário. A escolha entre os dois melhores depende crucialmente das prioridades da aplicação final.

- Se o objetivo principal é maximizar a cobertura e a detecção geral das trajetórias, ou seja, garantir que a maioria dos animais seja rastreada na maior parte do tempo, mesmo que ocorram algumas trocas de identidade ou fragmentações, BoT-SORT é a escolha recomendada. Seu desempenho superior em MOTA (87.4%) e *Recall* (91.0%) o torna ideal para tarefas como monitoramento de níveis de atividade geral, contagem de animais em zonas específicas ou aplicações onde a perda completa de um animal (Falso Negativo) é mais crítica que uma troca de ID momentânea;
- Se a aplicação exige alta fidelidade na identificação individual e trajetórias contínuas e consistentes, minimizando erros de identidade e interrupções, HybridSORT é a opção ideal. Seu IDF1 mais alto (67.3%), número significativamente menor de trocas de identidade (IDSW = 59) e baixa fragmentação (Frag = 152) são cruciais para análises comportamentais detalhadas (e.g. interações sociais específicas, padrões de movimento individuais, tempo dedicado a atividades por animal) ou qualquer estudo onde a manutenção precisa da identidade ao longo do tempo é fundamental.

Portanto, a decisão deve ser guiada pela pergunta: "É mais importante saber *que* um animal está ali e segui-lo (BoT-SORT), ou saber *qual* animal específico está ali e seguir sua trajetória individual sem erros de identidade (HybridSORT)?". Os rastreadores ByteTrack e Deep OC-SORT mostraram-se menos competitivos neste cenário específico com os parâmetros padrão e não seriam as escolhas primárias com base nos resultados obtidos.

Conclusões e Trabalhos Futuros

Este trabalho abordou o desafio do monitoramento automático de múltiplos cães em baías por meio de visão computacional, utilizando uma perspectiva de câmera superior (*top-down*). O objetivo principal foi desenvolver e avaliar uma metodologia completa, englobando a detecção de objetos com um modelo YOLOv11 *fine-tuned* (ajustado) e o rastreamento de múltiplos objeto (MOT) através da comparação de algoritmos estado da arte. A metodologia empregada fundamentou-se em uma estratégia de divisão de dados, baseada em câmeras e em um processo de seleção de detector orientado à tarefa final de rastreamento.

5.1 Síntese dos Resultados e Conclusões Principais

Os experimentos demonstraram, primeiramente, a eficácia do detector de objetos YOLO11x após o *fine-tuning*. O modelo alcançou um desempenho robusto no conjunto de teste dedicado ($mAP@50-95 = 0.869$), mostrando boa capacidade de generalização para câmeras não vistas durante o treinamento e fornecendo detecções de qualidade para a etapa subsequente.

A avaliação comparativa dos algoritmos de rastreamento, utilizando as detecções da YOLO11x, revelou diferenças significativas de desempenho. Os algoritmos BoT-SORT e HybridSORT emergiram como as soluções mais promissoras no cenário específico do recorte da pesquisa. O BoT-SORT se destacou pela maior acurácia geral ($MOTA = 87.4\%$) e pelo maior *Recall*, indicando maior capacidade de seguir corretamente a maioria das trajetórias. Por outro lado, o HybridSORT apresentou a melhor consistência na manutenção da identidade dos indivíduos, alcançando o maior IDF1 (67.3%) e o menor número de trocas de identidade ($IDSW = 59$), métricas cruciais para análises comportamentais individuais. Os algoritmos ByteTrack e Deep OC-SORT apresentaram desempenho inferior nessas métricas principais.

Conclui-se, portanto, que existe um *trade-off* entre maximizar a cobertura do rastreamento (BoT-SORT) e garantir a máxima fidelidade na identidade individual (HybridSORT).

Desta forma, considera-se que os objetivos propostos na Introdução foram satisfatoriamente alcançados, resultando na implementação da metodologia proposta e na identificação das técnicas de rastreamento mais adequadas para o problema em questão.

5.2 Contribuições Principais

Este trabalho oferece as seguintes contribuições para a área de monitoramento animal baseado em visão computacional:

1. **Análise Comparativa de Rastreadores MOT em Contexto Específico:** Fornece uma avaliação quantitativa e qualitativa, comparando múltiplos rastreadores MOT modernos (ByteTrack, BoT-SORT, HybridSort, Deep OC-SORT) no cenário particular e desafiador de rastreamento *top-down* de cães em baia, identificando BoT-SORT e HybridSORT como os mais eficazes e detalhando o *trade-off* entre acurácia geral (MOTA) e consistência de identidade (IDF1);
2. **Validação Metodológica:** Apresenta e valida uma metodologia robusta para avaliação de sistemas de detecção e rastreamento neste contexto, incluindo uma estratégia de divisão de dados por câmera para garantir a avaliação da generalização e também um critério de seleção de detector orientado pelo desempenho na tarefa final de rastreamento;
3. **Insights sobre Anotação de Vídeos:** Incorpora a otimização do processo de anotação de vídeos para *datasets* de rastreamento através do emprego de ferramentas semi-automáticas, como o SAM2. A experiência relatada nesta aplicação prática pode auxiliar trabalhos futuros com tarefas de anotação em larga escala.

5.3 Limitações e Trabalhos Futuros

Apesar dos resultados promissores, este estudo possui limitações que abrem caminhos para pesquisas futuras. A principal limitação reside na manutenção da identidade a longo prazo; mesmo os melhores rastreadores apresentaram scores de IDF1 abaixo de 70% e um número considerável de trocas de identidade ($IDSW > 50$), indicando que a distinção entre cães visualmente similares na perspectiva *top-down* e oclusões/sobreposições frequentes, como observado na análise qualitativa, ainda são desafios significativos. As fragmentações, embora reduzidas por BoT-SORT e HybridSORT (cerca de 150 em aproximadamente 7500 *frames*), indicam que interrupções no rastreamento ainda ocorrem, possivelmente ligadas a falhas momentâneas do detector (baixo contraste, proximidade entre os cães e contato com

a estrutura preta de baixo contraste, como visto nas Figuras 4.6 e 4.7). Outras limitações incluem: o escopo do *dataset*, que, embora variado, poderia ser expandido em duração e diversidade de ambientes para testar a robustez em condições ainda mais amplas, a não otimização dos parâmetros dos algoritmos de rastreamento, que foram utilizados em suas configurações padrão e a ausência de uma análise de eficiência computacional (velocidade de processamento), que seria relevante para aplicações em tempo real.

Com base nisso, sugerem-se as seguintes direções para trabalhos futuros:

- Investigar a integração de módulos de Re-Identificação (Re-ID) específicos para cães ou adaptados à visão *top-down* aos algoritmos de rastreamento, visando melhorar drasticamente as métricas IDF1 e IDSW;
- Realizar uma otimização de hiperparâmetros dedicada para os algoritmos BoT-SORT e HybridSORT no contexto deste *dataset*, potencialmente elevando ainda mais seu desempenho;
- Expandir o *dataset* de rastreamento com vídeos mais longos, maior número de animais simultâneos e, se possível, diferentes layouts de baias;
- Avaliar o custo computacional das melhores combinações detector+tracker em hardware relevante para determinar a viabilidade de implementação em tempo real;
- Aplicar a metodologia desenvolvida para extrair e analisar métricas comportamentais dos cães, validando o potencial prático do sistema;
- Reduzir as fragmentações de trajetória, possivelmente melhorando a interpolação durante oclusões ou a robustez a falhas momentâneas do detector;
- Avaliar o impacto de *datasets* de treinamento maiores e mais diversos para o detector, visando potencialmente melhorar ainda mais o mAP e, conseqüentemente, o rastreamento.

5.4 Considerações Finais

Em suma, este trabalho demonstrou a viabilidade e a eficácia da combinação de detectores de objetos modernos como o YOLO11x com algoritmos de rastreamento MOT estado da arte, validando uma solução de grande aplicabilidade prática e comercial para o monitoramento automático de cães em baias com visão superior. Os resultados comparativos fornecem um guia prático para a seleção do rastreador mais adequado, dependendo das prioridades da aplicação, ao mesmo tempo que as limitações identificadas apontam caminhos claros para a evolução e o aprimoramento dessas tecnologias, com grande potencial para

auxiliar na pesquisa comportamental e na promoção do bem-estar animal em ambientes controlados.

Referências

- AHARON, Nir; ORFAIG, Roy; BOBROVSKY, Ben-Zion. **BoT-SORT: Robust Associations Multi-Pedestrian Tracking**. [S.l.: s.n.], 2022. arXiv: 2206.14651 [cs.CV]. Disponível em: <<https://arxiv.org/abs/2206.14651>>.
- BASHAR, Mk et al. **Multiple Object Tracking in Recent Times: A Literature Review**. [S.l.: s.n.], 2022. arXiv: 2209.04796 [cs.CV]. Disponível em: <<https://arxiv.org/abs/2209.04796>>.
- BERNARDIN, Keni; STIEFELHAGEN, Rainer. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics. **EURASIP Journal on Image and Video Processing**, v. 2008, p. 246309, 2008. DOI: 10.1155/2008/246309.
- BEWLEY, Alex et al. Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP). [S.l.: s.n.], set. 2016. P. 3464–3468. DOI: 10.1109/ICIP.2016.7533003.
- BOCHINSKI, Erik; EISELEIN, Volker; SIKORA, Thomas. High-Speed tracking-by-detection without using image information. In: 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). [S.l.: s.n.], ago. 2017. P. 1–6. DOI: 10.1109/AVSS.2017.8078516.
- CAO, Jinkun et al. Observation-Centric SORT: Rethinking SORT for Robust Multi-Object Tracking. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], jun. 2023. P. 9686–9696. DOI: 10.1109/CVPR52729.2023.00934.
- CHOLLET, Francois. **Deep Learning with Python, Second Edition**. [S.l.]: Manning, 2021. ISBN 9781617296864. Disponível em: <<https://ieeexplore.ieee.org/document/10280481>>.
- DENDORFER, Patrick et al. MOTChallenge: A Benchmark for Single-Camera Multiple Target Tracking. **International Journal of Computer Vision**, v. 129, n. 4, p. 845–881, abr. 2021. ISSN 1573-1405. DOI: 10.1007/s11263-020-01393-0.
- EVERINGHAM, Mark et al. The PASCAL Visual Object Classes (VOC) Challenge. **International Journal of Computer Vision**, v. 88, n. 2, p. 303–338, 2010. DOI: 10.1007/s11263-009-0275-4.

- GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep Learning**. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.
- GURNEY, Kevin. **An Introduction to Neural Networks**. USA: Taylor & Francis, Inc., 1997. ISBN 1857286731.
- HE, Kaiming et al. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], jun. 2016. P. 770–778. DOI: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- JOCHER, Glenn; QIU, Jing; CHAURASIA, Ayush. **Ultralytics YOLO**. [S.l.: s.n.], jan. 2023. <https://github.com/ultralytics/ultralytics>.
- KALMAN, Rudolph Emil. A New Approach to Linear Filtering and Prediction Problems. **Transactions of the ASME–Journal of Basic Engineering**, v. 82, Series D, p. 35–45, 1960.
- KHANAM, Rahima; HUSSAIN, Muhammad. **YOLOv11: An Overview of the Key Architectural Enhancements**. [S.l.: s.n.], 2024. arXiv: [2410.17725](https://arxiv.org/abs/2410.17725) [cs.CV]. Disponível em: <https://arxiv.org/abs/2410.17725>.
- KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. ImageNet classification with deep convolutional neural networks. **Commun. ACM**, Association for Computing Machinery, New York, NY, USA, v. 60, n. 6, p. 84–90, mai. 2017. ISSN 0001-0782. DOI: [10.1145/3065386](https://doi.org/10.1145/3065386).
- KUHN, H. W. The Hungarian method for the assignment problem. **Naval Research Logistics Quarterly**, v. 2, n. 1-2, p. 83–97, 1955. DOI: [10.1002/nav.3800020109](https://doi.org/10.1002/nav.3800020109). eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/nav.3800020109>.
- LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep learning. **Nature**, v. 521, n. 7553, p. 436–444, mai. 2015. ISSN 1476-4687. DOI: [10.1038/nature14539](https://doi.org/10.1038/nature14539).
- LIN, Tsung-Yi et al. Microsoft COCO: Common objects in context. In: COMPUTER Vision – ECCV 2014. [S.l.]: Springer, 2014. P. 740–755. DOI: [10.1007/978-3-319-10602-1_48](https://doi.org/10.1007/978-3-319-10602-1_48).
- LIU, Yeqiang et al. Deep learning in multiple animal tracking: A survey. **Computers and Electronics in Agriculture**, v. 224, 109161, p. 109161, set. 2024. DOI: [10.1016/j.compag.2024.109161](https://doi.org/10.1016/j.compag.2024.109161).
- LOSHCHILOV, Ilya; HUTTER, Frank. Decoupled Weight Decay Regularization. In: INTERNATIONAL Conference on Learning Representations (ICLR). [S.l.: s.n.], 2019. Disponível em: <https://arxiv.org/abs/1711.05101>.
- MAGGIOLINO, Gerard et al. **Deep OC-SORT: Multi-Pedestrian Tracking by Adaptive Re-Identification**. [S.l.: s.n.], 2023. arXiv: [2302.11813](https://arxiv.org/abs/2302.11813) [cs.CV]. Disponível em: <https://arxiv.org/abs/2302.11813>.
- MILAN, A. et al. MOT16: A Benchmark for Multi-Object Tracking. **arXiv:1603.00831** [cs], mar. 2016. arXiv: 1603.00831. Disponível em: <http://arxiv.org/abs/1603.00831>.

MITCHELL, Thomas M. **Machine Learning**. 1. ed. USA: McGraw-Hill, Inc., 1997. ISBN 0070428077.

POLYAK, B.T. Some methods of speeding up the convergence of iteration methods. **USSR Computational Mathematics and Mathematical Physics**, v. 4, n. 5, p. 1–17, 1964. ISSN 0041-5553. DOI: [10.1016/0041-5553\(64\)90137-5](https://doi.org/10.1016/0041-5553(64)90137-5).

RAVI, Nikhila et al. SAM 2: Segment Anything in Images and Videos. In: THE Thirteenth International Conference on Learning Representations. [S.l.: s.n.], 2025. Disponível em: <https://openreview.net/forum?id=Ha6RTeWMd0>.

REDMON, Joseph et al. You Only Look Once: Unified, Real-Time Object Detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], jun. 2016. P. 779–788. DOI: [10.1109/CVPR.2016.91](https://doi.org/10.1109/CVPR.2016.91).

RISTANI, Ergys et al. Performance Measures and a Data Set for Multi-target, Multi-camera Tracking. In: _____. **Computer Vision – ECCV 2016 Workshops**. Cham: Springer International Publishing, 2016. P. 17–35. ISBN 978-3-319-48881-3.

RUMELHART, David E.; HINTON, Geoffrey E.; WILLIAMS, Ronald J. Learning representations by back-propagating errors. **Nature**, Nature Publishing Group UK London, v. 323, n. 6088, p. 533–536, out. 1986. ISSN 1476-4687. DOI: [10.1038/323533a0](https://doi.org/10.1038/323533a0).

RUSSELL, Stuart; NORVIG, Peter. **Artificial Intelligence: A Modern Approach**. 3rd. USA: Prentice Hall Press, 2009. ISBN 0136042597.

SIMONYAN, Karen; ZISSERMAN, Andrew. **Very Deep Convolutional Networks for Large-Scale Image Recognition**. [S.l.: s.n.], 2015. arXiv: [1409.1556](https://arxiv.org/abs/1409.1556) [cs.CV]. Disponível em: <https://arxiv.org/abs/1409.1556>.

SRIVASTAVA, Nitish et al. Dropout: a simple way to prevent neural networks from overfitting. **J. Mach. Learn. Res.**, JMLR.org, v. 15, n. 1, p. 1929–1958, jan. 2014. ISSN 1532-4435.

STIEFELHAGEN, Rainer et al. The CLEAR 2006 evaluation. In: PROCEEDINGS of the 1st International Evaluation Conference on Classification of Events, Activities and Relationships. Southampton, UK: Springer-Verlag, 2006. (CLEAR'06), p. 1–44. ISBN 9783540695677.

SZEGEDY, Christian et al. Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], jun. 2015. P. 1–9. DOI: [10.1109/CVPR.2015.7298594](https://doi.org/10.1109/CVPR.2015.7298594).

WOJKE, Nicolai; BEWLEY, Alex; PAULUS, Dietrich. Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP). [S.l.: s.n.], set. 2017. P. 3645–3649. DOI: [10.1109/ICIP.2017.8296962](https://doi.org/10.1109/ICIP.2017.8296962).

YANG, Mingzhan et al. **Hybrid-SORT: Weak Cues Matter for Online Multi-Object Tracking**. [S.l.: s.n.], 2024. arXiv: 2308.00783 [cs.CV]. Disponível em: <<https://arxiv.org/abs/2308.00783>>.

ZHANG, Libo et al. AnimalTrack: A Benchmark for Multi-Animal Tracking in the Wild. **Int. J. Comput. Vision**, Kluwer Academic Publishers, USA, v. 131, n. 2, p. 496–513, nov. 2022. ISSN 0920-5691. DOI: 10.1007/s11263-022-01711-8. Disponível em: <<https://doi.org/10.1007/s11263-022-01711-8>>.

ZHANG, Yifu et al. **ByteTrack: Multi-Object Tracking by Associating Every Detection Box**. [S.l.: s.n.], 2022. arXiv: 2110.06864 [cs.CV]. Disponível em: <<https://arxiv.org/abs/2110.06864>>.