Universidade Federal do ABC
Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas
Trabalho de Graduação em Engenharia de Informação

Desenvolvimento de um detector e rastreador de jogadores e bola em jogos de futebol

Alex Zaneratto Cavalcante

Santo André

Julho de 2025

Alex Zaneratto Cavalcante

Desenvolvimento de um detector e rastreador de jogadores e bola em jogos de futebol

Trabalho de Graduação apresentado ao concluir a Graduação em Engenharia de Informação, como parte dos requisitos necessários para a obtenção do Título Bacharel em Engenharia de Informação.

Universidade Federal do ABC

Orientador: Claudio José Bordin Júnior

Santo André
Julho de 2025

Agradecimentos

A conclusão desta monografia representa não apenas o fim de uma etapa acadêmica, mas também o resultado do apoio incondicional de pessoas que foram fundamentais nesta jornada. Primeiramente, gostaria de expressar minha profunda gratidão à minha família e minha esposa, que sempre acreditou em meu potencial e me proporcionou o suporte necessário para que eu pudesse me dedicar integralmente aos estudos. Suas palavras de incentivo nos momentos de dificuldade e a compreensão durante os períodos de ausência foram essenciais para que eu chegasse até aqui.

Estendo meus agradecimentos aos meus amigos, que compartilharam comigo não apenas os desafios acadêmicos, mas também os momentos de alegria e descoberta ao longo do curso. Suas contribuições através de discussões enriquecedoras, troca de ideias e apoio mútuo tornaram esta caminhada mais leve e significativa. A amizade verdadeira que construímos transcende os muros da universidade e certamente perdurará por toda a vida.

Aos meus professores, registro aqui meu reconhecimento pelo conhecimento compartilhado e pela dedicação em formar não apenas profissionais competentes, mas também cidadãos conscientes de seu papel na sociedade. Em especial, agradeço ao meu orientador, cuja paciência, orientação criteriosa e expertise foram fundamentais para o desenvolvimento deste trabalho. Suas sugestões valiosas e acompanhamento constante contribuíram significativamente para a qualidade desta pesquisa.

Por fim, reconheço que esta conquista é fruto de um esforço coletivo, onde cada pessoa mencionada teve sua parcela de contribuição. A todos que, direta ou indiretamente, fizeram parte desta trajetória acadêmica, meu sincero obrigado. Este trabalho representa não apenas meu crescimento pessoal e profissional, mas também o reflexo do apoio e confiança de todos vocês.

"O cosmos está dentro de nós. Somos feitos de material estelar. Somos uma forma do universo conhecer a si mesmo."

Carl Sagan

Resumo

Este trabalho apresenta o desenvolvimento de um sistema automatizado para detecção e rastreamento de jogadores e bola em vídeos de futebol utilizando técnicas de visão computacional e aprendizado profundo. A pesquisa emprega a arquitetura YOLOv8 em diferentes configurações para avaliar o impacto de parâmetros como resolução e capacidade do modelo na qualidade da detecção. Foram realizados três experimentos sistemáticos: YOLOv8n com resolução 640px (baseline), YOLOv8l com resolução 640px (alta capacidade) e YOLOv8n com resolução 1920px (alta resolução). Os resultados demonstram que o aumento da resolução proporciona ganhos superiores ao aumento da capacidade do modelo, especialmente para objetos pequenos como a bola, com melhorias de até 191,3% na revocação. O sistema implementa ainda módulos de rastreamento baseado em ByteTrack, refinamento de trajetórias por interpolação e identificação automática de equipes através de análise de cores. As contribuições incluem o estabelecimento de benchmarks para avaliação de modelos YOLO em contextos esportivos e diretrizes práticas para implementação em diferentes cenários de aplicação.

Palavras-chaves: Visão Computacional. Detecção de Objetos. YOLOv8. Futebol. Rastreamento.

Abstract

This work presents the development of an automated system for detecting and tracking players and ball in soccer videos using computer vision and deep learning techniques. The research employs the YOLOv8 architecture in different configurations to evaluate the impact of parameters such as resolution and model capacity on detection quality. Three systematic experiments were conducted: YOLOv8n with 640px resolution (baseline), YOLOv8l with 640px resolution (high capacity), and YOLOv8n with 1920px resolution (high resolution). Results demonstrate that increasing resolution provides superior gains compared to increasing model capacity, especially for small objects like the ball, with improvements up to 191.3% in recall. The system also implements tracking modules based on ByteTrack, trajectory refinement through interpolation, and automatic team identification through color analysis. Contributions include establishing benchmarks for evaluating YOLO models in sports contexts and practical guidelines for implementation in different application scenarios. **Keywords**: Computer Vision. Object Detection. YOLOv8.

Soccer. Tracking.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

AP	Average Precision (Precisão Média)
BSV	Broadcast Soccer Videos (Vídeos de Transmissão de Futebol)
CNN	Convolutional Neural Network (Rede Neural Convolucional)
COCO	Common Objects in Context (Objetos Comuns em Contexto)
CSP	Cross Stage Partial (Parcial entre Estágios)
CSPDarknet	Cross Stage Partial Darknet
CUDA	Compute Unified Device Architecture
DFL	Distribution Focal Loss (Perda Focal de Distribuição)
FN	False Negative (Falso Negativo)
FP	False Positive (Falso Positivo)
FPS	Frames Per Second (Quadros por Segundo)
GB	Gigabyte
GFLOPS	Giga Floating Point Operations Per Second (Giga Operações de Ponto Flutuante por Segundo)
GPU	Graphics Processing Unit (Unidade de Processamento Gráfico)
HSV	Hue, Saturation, Value (Matiz, Saturação, Valor)
ID	Identifier (Identificador)
IFAB	International Football Association Board
IoU	Intersection over Union (Interseção sobre União)
mAP	mean Average Precision (Precisão Média Geral)
MB	Megabyte
Mbps	Megabits per second (Megabits por segundo)
MLP	Multilayer Perceptron (Perceptron Multicamadas)
MOT	Multi-Object Tracking (Rastreamento de Múltiplos Objetos)

MP	Megapixel
ms	milliseconds (milissegundos)
PAN	Path Aggregation Network (Rede de Agregação de Caminho)
RAM	Random Access Memory (Memória de Acesso Aleatório)
RGB	Red, Green, Blue (Vermelho, Verde, Azul)
RNA	Rede Neural Artificial
ROI	Region of Interest (Região de Interesse)
SGD	Stochastic Gradient Descent (Gradiente Descendente Estocástico)
SORT	Simple Online and Realtime Tracking (Rastreamento Online e em Tempo Real Simples)
SSD	Solid State Drive (Unidade de Estado Sólido)
TFLOPS	Tera Floating Point Operations Per Second (Tera Operações de Ponto Flutuante por Segundo)
VFL	Varifocal Loss (Perda Varifocal)
VP	Verdadeiro Positivo
YOLO	You Only Look Once
YOLOv8l	You Only Look Once version 8 large
YOLOv8n	You Only Look Once version 8 nano

Sumário

1 Introdução

O desenvolvimento tecnológico nas áreas de visão computacional e inteligência artificial tem transformado significativamente a forma como analisamos e compreendemos eventos esportivos. No contexto do futebol, esporte mais popular do mundo, a capacidade de extrair informações automáticas e precisas de vídeos de partidas tornou-se não apenas uma ferramenta valiosa para análise técnica, mas também um diferencial competitivo crucial para equipes profissionais. A análise de dados no futebol evoluiu de anotações manuais simples para sistemas complexos de captura e processamento automático, revolucionando a compreensão tática do esporte e influenciando decisões que podem determinar o resultado de competições.

Pesquisas recentes demonstram que a aplicação de tecnologias de visão computacional em esportes tem crescido exponencialmente [1]. Entre 2023 e 2024, houve uma transição significativa na área, com pesquisadores migrando de técnicas fundamentais de detecção para objetivos de nível superior, incluindo predição de movimento, análise tática e detecção de eventos. Este crescimento reflete a maturidade da área e a demanda crescente por soluções mais sofisticadas de análise esportiva automatizada [2].

A análise manual de partidas de futebol, tradicionalmente realizada por observadores técnicos, é um processo laborioso que pode consumir várias horas para uma única partida, com taxas de erro consideráveis devido à subjetividade e limitações da observação humana. Estudos atuais na área de análise de desempenho esportivo evidenciam que a aplicação de inteligência artificial e machine learning continuam impulsionando este campo, com impactos significativos tanto para treinamento atlético quanto para sistemas de pontuação competitiva [3]. Neste cenário, sistemas automatizados de detecção e rastreamento surgem como uma solução promissora para democratizar o acesso a análises profissionais e aumentar significativamente a precisão das informações extraídas.

O problema central abordado neste trabalho reside na complexidade técnica de detectar e rastrear múltiplos objetos em movimento em vídeos de futebol. Diferentemente de ambientes controlados, as transmissões esportivas apresentam desafios únicos identificados pela literatura recente: imagens de baixa resolução de jogadores e bolas distantes, oclusão quando a uma aglomeração de jogadores, motion blur durante movimentos rápidos, e a falta de conjuntos de dados anotados em larga escala especificamente adaptados para cenários dinâmicos de futebol [1]. Especialmente desafiante é a detecção da bola, que ocupa uma fração mínima da área total do quadro em câmeras de transmissão padrão.

A motivação para este trabalho emerge da convergência entre a demanda crescente por análise tática quantitativa no futebol profissional e os avanços recentes em arquiteturas

de detecção de objetos. Clubes ao redor do mundo estão investindo recursos substanciais em departamentos de análise de desempenho, utilizando dados de rastreamento totalmente automatizados e escaláveis para todos os 22 jogadores [4]. Paralelamente, o desenvolvimento da família YOLO (You Only Look Once), especialmente com o YOLOv8, oferece um equilíbrio sem precedentes entre velocidade e precisão [5].

1.1 Trabalhos Relacionados

Diversos trabalhos têm contribuído significativamente para o avanço da área de detecção e rastreamento em vídeos esportivos. O projeto FootyVision [6] desenvolveu um sistema robusto de rastreamento multi-objeto, localização e augmentação de jogadores e bola em vídeos de futebol, demonstrando capacidades robustas com pontuação IDF1 de aproximadamente 93% nos datasets ISSIA e SoccerNet. Este trabalho estabeleceu novos padrões de precisão na área, alcançando MOTA de 94,04% para o SoccerNet e HOTA geral de 72,16%.

Na detecção de jogadores, análises comparativas recentes entre YOLOv5, YOLOv8 e YOLOv9 para detecção, rastreamento e reconhecimento de ações humanas em esportes de combate revelaram que o YOLOv8 oferece o melhor equilíbrio entre precisão e recall, tornando-se particularmente adequado para aplicações em tempo real na análise esportiva [3]. A detecção da bola, por sua vez, permanece como um dos maiores desafios. Trabalhos recentes focaram em melhorias específicas do YOLOv8 para detecção de alvos pequenos [7], introduzindo módulos como GAM (Global Attention Mechanism) para combinar características globais e locais, melhorando significativamente a percepção de informações chave.

No contexto de rastreamento multi-objeto, o algoritmo ByteTrack [8] estabeleceu as bases para rastreamento eficiente. O ByteTrack apresenta uma abordagem simples mas efetiva, rastreando através da associação de quase todas as caixas de detecção em vez de apenas as de alta pontuação, utilizando similaridades com tracklets para recuperar objetos verdadeiros e filtrar detecções de fundo. Desenvolvimentos recentes propuseram melhorias no ByteTrack para rastreamento de veículos [9, 10], incluindo aprimoramentos no filtro de Kalman para melhor lidar com movimentos não-lineares e introdução de Gaussian Smoothed Interpolation (GSI) para preencher lacunas de trajetória.

O dataset SoccerNet tem sido fundamental para o avanço da pesquisa na área. O SoccerNet-Tracking [11], introduzido em 2022, fornece um dataset novo para rastreamento de múltiplos objetos composto por 200 sequências de 30 segundos cada, representativas de cenários desafiadores de futebol, além de um tempo completo de 45 minutos para rastreamento de longo prazo. A evolução para SoccerNet-v3 [12] incluiu anotações espaciais extensivas e correspondências cross-view, fornecendo informações locais exaustivas para

análises automatizadas mais refinadas.

A identificação automática de equipes evoluiu significativamente, com sistemas IoT-based recentes como o VAR-YOLOv8 [13] demonstrando capacidades melhoradas de detecção de faltas em partidas de futebol, combinando tecnologias avançadas como MPDIoU, redes de características locais residuais (RLFN) e módulos de sistema de árbitro assistente de vídeo. Sistemas integrados recentes [14] têm demonstrado a aplicação prática de detecção de objetos em partidas de futebol para análise de posicionamento de jogadores, rastreamento de bola e detecção de eventos-chave baseados em movimentos e proximidade dos jogadores.

Apesar desses avanços, a maioria dos trabalhos foca em aspectos isolados do problema, com poucos sistemas completamente integrados que abordem detecção, rastreamento e análise de forma unificada. A literatura atual indica uma lacuna significativa na integração de múltiplas modalidades de análise esportiva em sistemas práticos e escaláveis.

1.2 Contribuições

Este trabalho apresenta três contribuições principais para o campo de visão computacional aplicada a esportes. Primeiro, através de experimentos controlados, demonstra-se quantitativamente que o aumento de resolução pode ser mais efetivo que o aumento de capacidade do modelo para detecção de objetos pequenos, com melhorias de até 191% na detecção da bola. Segundo, desenvolve-se um sistema integrado completo que combina detecção por YOLOv8, rastreamento por ByteTrack, e módulos especializados de pós-processamento incluindo refinamento de trajetórias e identificação automática de equipes. Terceiro, disponibiliza-se publicamente um conjunto de dados com 4.116 imagens anotadas de partidas de futebol, totalizando 87.684 anotações, contribuindo para a reprodutibilidade e avanço da pesquisa na área.

1.3 Estrutura do Trabalho

O restante deste trabalho está organizado da seguinte forma: o Capítulo 2 apresenta a fundamentação teórica, cobrindo conceitos de redes neurais convolucionais, a evolução da família YOLO, algoritmos de rastreamento e os desafios específicos do domínio esportivo. O Capítulo 3 detalha a metodologia empregada, incluindo a preparação da base de dados, configuração do ambiente computacional, métricas de avaliação e a arquitetura do sistema desenvolvido. O Capítulo 4 apresenta e discute os resultados obtidos nos experimentos realizados, incluindo análises comparativas detalhadas entre diferentes configurações e avaliação qualitativa do sistema. Por fim, o Capítulo 5 conclui o trabalho, resumindo as principais descobertas, discutindo limitações identificadas e propondo direções para

pesquisas futuras.

2 Fundamentação Teórica

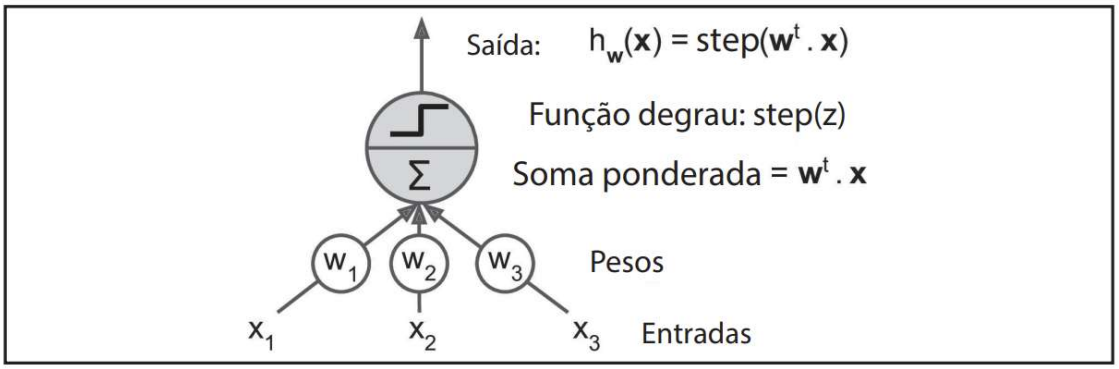
Este capítulo apresenta os conceitos fundamentais necessários para compreensão das técnicas empregadas no desenvolvimento do sistema de detecção e rastreamento. A exposição inicia com os princípios básicos de redes neurais artificiais, evoluindo progressivamente para arquiteturas mais complexas até alcançar o estado da arte em detecção de objetos. Posteriormente, são apresentados os fundamentos matemáticos dos algoritmos de agrupamento e interpolação utilizados no pós-processamento dos resultados.

2.1 Redes Neurais Artificiais

As redes neurais artificiais representam uma classe de modelos computacionais inspirados no funcionamento do sistema nervoso biológico. Conforme descrito por Zhang et al. [15], uma rede neural pode ser compreendida como um grafo direcionado onde os nós representam unidades de processamento (neurônios artificiais) e as arestas representam conexões ponderadas entre essas unidades. O processo de aprendizado consiste fundamentalmente em ajustar os pesos dessas conexões através da exposição a exemplos, permitindo que a rede aprenda padrões complexos nos dados.

O perceptron, desenvolvido por Rosenblatt [16] no final da década de 1950, constitui a unidade fundamental das redes neurais modernas. Este modelo matemático simples recebe um vetor de entrada $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ e produz uma saída escalar através de uma transformação linear seguida de uma função de ativação. A Figura 1 ilustra esta arquitetura básica, onde cada entrada x_i é multiplicada por um peso correspondente w_i, os produtos são somados juntamente com um termo de bias b, e o resultado passa por uma função de ativação ϕ.

Figura 1 – Modelo do perceptron mostrando as entradas x_i, pesos w_i, soma ponderada e função de ativação



Fonte: Adaptado de Géron [17]

Matematicamente, a saída y do perceptron é calculada através da Equação 2.1:

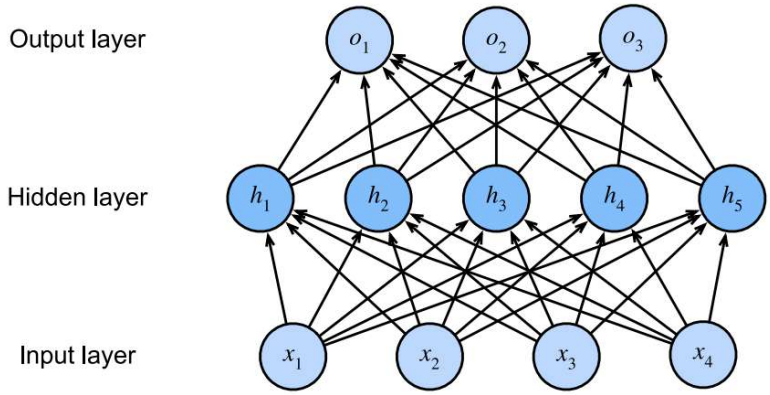
$$y = \phi\left(\sum_{i=1}^{n} w_i x_i + b\right) = \phi(\mathbf{w}^T\mathbf{x} + b) \tag{2.1}$$

onde $\mathbf{w} = [w_1, w_2, \ldots, w_n]^T$ representa o vetor de pesos que determina a importância relativa de cada entrada, b é o termo de bias que permite deslocar o hiperplano de decisão, e ϕ é a função de ativação. No perceptron clássico, utiliza-se a função sinal como ativação, que produz saídas binárias (+1 ou -1) dependendo do sinal da soma ponderada. Esta simplicidade, embora elegante, limita o perceptron a resolver apenas problemas linearmente separáveis, ou seja, aqueles onde as classes podem ser separadas por um único hiperplano no espaço de características.

A limitação do perceptron simples foi superada através do desenvolvimento do Perceptron Multicamadas (MLP), que empilha múltiplas camadas de neurônios com funções de ativação não-lineares entre elas. A Figura 2 apresenta a arquitetura de um MLP com uma camada oculta, demonstrando como os sinais fluem da entrada através das camadas intermediárias até a saída. Cada camada realiza uma transformação dos dados, e a composição dessas transformações permite ao modelo aprender representações

hierárquicas complexas.

Figura 2 – Arquitetura de um perceptron multicamadas com camada de entrada, uma camada oculta e camada de saída



Fonte: Adaptado de Zhang et al. [15]

Em um MLP, a computação ocorre através de propagação direta (forward propagation), onde cada camada ℓ (onde $\ell = 1, 2, \ldots, L$ e L é o número total de camadas) aplica uma transformação aos dados recebidos da camada anterior. Para a ℓ-ésima camada, a saída $\mathbf{h}^{(\ell)}$ é calculada como:

$$\mathbf{z}^{(\ell)} = \mathbf{W}^{(\ell)}\mathbf{h}^{(\ell-1)} + \mathbf{b}^{(\ell)} \tag{2.2}$$

$$\mathbf{h}^{(\ell)} = \phi^{(\ell)}(\mathbf{z}^{(\ell)}) \tag{2.3}$$

onde $\mathbf{h}^{(0)} = \mathbf{x}$ representa a entrada, $\mathbf{W}^{(\ell)} \in \mathbb{R}^{m^{(\ell)} \times m^{(\ell-1)}}$ é a matriz de pesos conectando a camada $\ell - 1$ à camada ℓ, $\mathbf{b}^{(\ell)} \in \mathbb{R}^{m^{(\ell)}}$ é o vetor de bias, e $\phi^{(\ell)}$ é a função de ativação. Note que $m^{(\ell)}$ representa o número de neurônios na camada ℓ.

As funções de ativação são cruciais para introduzir não-linearidade no modelo. A função ReLU (Rectified Linear Unit), definida como $\phi(z) = \max(0, z)$, tornou-se a escolha padrão em redes profundas devido à sua simplicidade computacional e eficácia em mitigar o problema do gradiente desvanecente. Outras opções incluem a sigmoid $\phi(z) = 1/(1 + e^{-z})$, que mapeia valores para o intervalo (0,1), e a tangente hiperbólica $\phi(z) = \tanh(z)$, que produz saídas no intervalo (-1,1).

O treinamento de redes neurais é realizado através de otimização iterativa, tipicamente usando variantes do algoritmo de gradiente descendente. O objetivo é minimizar uma função de perda $\mathcal{L}(\boldsymbol{\theta})$ que quantifica o erro entre as predições da rede e os valores verdadeiros, onde $\boldsymbol{\theta}$ representa todos os parâmetros treináveis (pesos e bias). A atualização dos parâmetros segue a regra:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_t) \tag{2.4}$$

onde t indica a iteração atual, η é a taxa de aprendizado que controla o tamanho do passo de atualização, e $\nabla_{\boldsymbol{\theta}} \mathcal{L}$ é o gradiente da função de perda em relação aos parâmetros.

Na prática, conforme explicado por Zhang et al. [15], utiliza-se predominantemente o gradiente descendente estocástico com mini-lotes, que calcula o gradiente usando pequenos subconjuntos dos dados (tipicamente 32, 64 ou 128 exemplos). Esta abordagem equilibra a estabilidade das atualizações com a eficiência computacional, aproveitando o paralelismo das GPUs modernas enquanto mantém estimativas razoavelmente precisas do gradiente verdadeiro.

2.2 Redes Neurais Convolucionais

As Redes Neurais Convolucionais (CNNs) representam uma especialização das redes neurais projetada especificamente para processar dados com estrutura de grade, como imagens. O sucesso da AlexNet em 2012 no desafio ImageNet marcou o início da era moderna de visão computacional baseada em aprendizado profundo. As CNNs exploram três princípios fundamentais que as tornam particularmente eficazes para processamento de imagens: conectividade local (cada neurônio conecta-se apenas a uma pequena região da entrada), compartilhamento de parâmetros (os mesmos pesos são usados em diferentes posições espaciais), e invariância à translação (capacidade de detectar padrões independentemente de sua posição na imagem).

A operação central das CNNs é a convolução discreta bidimensional. Para uma imagem de entrada $\mathbf{I}$ e um kernel (ou filtro) $\mathbf{K}$ de dimensões $k_h \times k_w$, a operação de convolução produz um mapa de características $\mathbf{F}$ onde cada elemento é calculado como:

$$F[i,j] = \sum_{m=0}^{k_h-1} \sum_{n=0}^{k_w-1} I[i+m, j+n] \cdot K[m,n] \tag{2.5}$$

Esta operação, ilustrada na Equação 2.5, essencialmente desliza o kernel sobre a imagem, computando produtos internos locais que detectam a presença de padrões específicos. Quando trabalhamos com imagens coloridas ou mapas de características com múltiplos canais, a convolução se estende para três dimensões:

$$F[i,j,d] = \sum_{c=0}^{C_{in}-1} \sum_{m=0}^{k_h-1} \sum_{n=0}^{k_w-1} I[i+m, j+n, c] \cdot K[m,n,c,d] + b_d \tag{2.6}$$

onde C_{in} é o número de canais de entrada, d indexa os diferentes filtros (cada um produzindo um canal de saída), e b_d é o termo de bias para o d-ésimo filtro.

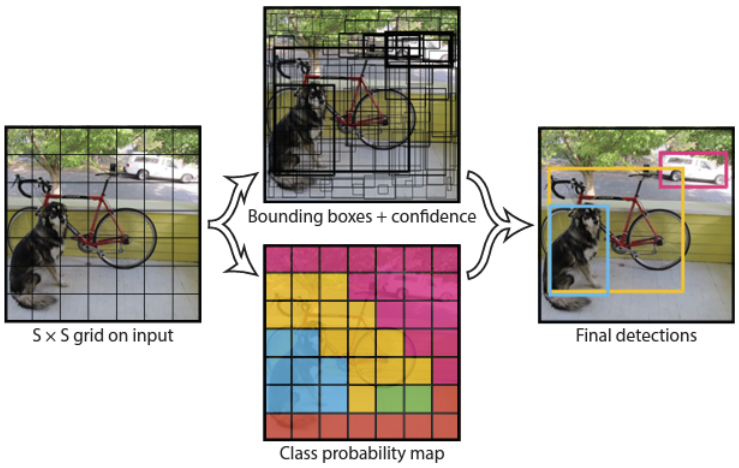
Uma camada convolucional completa aplica múltiplos filtros à entrada, cada um especializado em detectar diferentes características. Por exemplo, em camadas iniciais, os filtros podem detectar bordas e texturas simples, enquanto camadas mais profundas aprendem a reconhecer padrões complexos como partes de objetos. Esta hierarquia de características é aprendida automaticamente durante o treinamento, eliminando a necessidade de engenharia manual de características que dominava a visão computacional clássica.

2.3 Detecção de Objetos com YOLO

A detecção de objetos representa um desafio fundamental em visão computacional que vai além da simples classificação de imagens. Enquanto a classificação identifica o que está presente em uma imagem, a detecção deve responder não apenas "o que"mas também "onde", fornecendo as coordenadas precisas de cada objeto através de caixas delimitadoras (bounding boxes). Esta tarefa dupla de localização e classificação torna a detecção significativamente mais complexa.

A família YOLO (You Only Look Once) revolucionou este campo ao reformular a detecção como um problema de regressão unificado, processando a imagem completa em uma única passada pela rede. A Figura 3 ilustra o pipeline de detecção do YOLO original, demonstrando como a imagem é dividida em uma grade e cada célula prediz simultaneamente múltiplas caixas delimitadoras e probabilidades de classe.

Figura 3 – Pipeline de detecção do YOLO: a imagem é dividida em grade, cada célula prediz caixas e classes, resultando nas detecções finais



Fonte: Adaptado de Redmon e Farhadi [18]

O YOLO original, proposto por Redmon et al. [18], divide a imagem de entrada em uma grade $S \times S$. Cada célula desta grade é responsável por detectar objetos cujo centro caia dentro de seus limites. Para cada célula, o modelo prediz B caixas delimitadoras, onde cada caixa consiste em cinco valores: as coordenadas do centro (x, y) relativas à célula, as dimensões (w, h) normalizadas pela imagem total, e um score de confiança. Este score de confiança é definido como:

$$\text{Confiança} = P(\text{Objeto}) \times \text{IoU}_{\text{pred}}^{\text{truth}} \tag{2.7}$$

onde $P(\text{Objeto})$ indica a probabilidade de existir um objeto na célula e IoU (Intersection over Union) mede a sobreposição entre a caixa predita e a verdadeira durante o treinamento.

A evolução da arquitetura YOLO ao longo dos anos trouxe melhorias significativas. YOLOv2 [19] introduziu âncoras pré-definidas e normalização em lote, melhorando substancialmente a precisão. YOLOv3 [20] adicionou predições em múltiplas escalas, crucial para detectar objetos de tamanhos variados. As versões subsequentes (v4-v7) continuaram refinando a arquitetura com inovações como CSPNet, PANet e várias técnicas de augmentação de dados.

O YOLOv8, lançado em 2023 pela Ultralytics [5] e utilizado neste trabalho, representa uma das mais atuais versões da arquitetura na data desse trabalho. Suas principais inovações incluem a eliminação de âncoras pré-definidas (**anchor-free**), permitindo predição direta das coordenadas dos objetos, um backbone baseado em CSPNet com módulos C2f (**Cross Stage Partial with 2 convolutions**) que melhoram o fluxo de gradiente, e

um head de detecção desacoplado que separa as tarefas de classificação e regressão em ramos especializados. Esta separação permite que cada tarefa seja otimizada independentemente, resultando em melhor precisão sem comprometer a velocidade.

Para o contexto específico de detecção em vídeos esportivos, particularmente a detecção da bola, surgem desafios únicos. A bola tipicamente ocupa menos de 0,1% dos pixels totais da imagem, tornando sua detecção extremamente desafiadora. Além disso, o movimento rápido causa desfoque significativo, as oclusões por jogadores são frequentes, e existem muitos objetos visualmente similares no campo (marcações circulares, elementos gráficos na cor da bola). Estes fatores explicam por que, conforme observado nos experimentos, a detecção da bola apresenta métricas significativamente inferiores às de detecção de jogadores.

2.4 Algoritmos de Agrupamento e Interpolação

Os algoritmos de pós-processamento são fundamentais para refinar os resultados brutos da detecção e extrair informações de alto nível. Dois métodos matemáticos são particularmente relevantes para este trabalho: o algoritmo K-means para agrupamento de cores na identificação de equipes, e técnicas de interpolação para suavização de trajetórias.

2.4.1 Algoritmo K-means

O K-means[17] é um algoritmo de aprendizado não-supervisionado amplamente utilizado para particionar dados em grupos (clusters) baseados em similaridade. Dado um conjunto de pontos $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ onde cada $\mathbf{x}_i \in \mathbb{R}^D$, o objetivo é particionar estes pontos em K clusters de forma a minimizar a soma dos quadrados das distâncias intra-cluster.

Formalmente, o K-means busca encontrar K centróides $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_K\}$ que minimizem a função objetivo:

$$J = \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \mathbf{c}_k||^2 \tag{2.8}$$

onde $r_{ik} \in \{0, 1\}$ é uma variável indicadora que vale 1 se o ponto $\mathbf{x}_i$ pertence ao cluster k e 0 caso contrário. A norma $|| \cdot ||$ representa tipicamente a distância Euclidiana.

O algoritmo opera iterativamente através de dois passos principais até convergência:

Passo de Atribuição: Cada ponto é atribuído ao cluster cujo centróide está mais próximo:

$$r_{ik} = \begin{cases} 1 & \text{se } k = \arg\min_j ||\mathbf{x}_i - \mathbf{c}_j||^2 \\ 0 & \text{caso contrário} \end{cases} \tag{2.9}$$

Passo de Atualização: Os centróides são recalculados como a média dos pontos atribuídos a cada cluster:

$$\mathbf{c}_k = \frac{\sum_{i=1}^{N} r_{ik}\mathbf{x}_i}{\sum_{i=1}^{N} r_{ik}} \tag{2.10}$$

No contexto deste trabalho, o K-means é aplicado no espaço de cores RGB para identificar automaticamente as duas equipes em campo. Cada jogador detectado contribui com um vetor de cor $\mathbf{x}_i = [R_i, G_i, B_i]^T$ extraído da região do uniforme. Com $K = 2$, o algoritmo particiona estas cores em dois grupos correspondentes às duas equipes. Uma modificação importante implementada é a ponderação por saturação, onde cores mais saturadas (mais vibrantes) recebem maior peso no cálculo dos centróides, pois uniformes esportivos tipicamente apresentam cores vivas que os distinguem melhor.

2.4.2 Interpolação de Trajetórias

A interpolação é uma técnica matemática fundamental para estimar valores desconhecidos entre pontos de dados conhecidos. No contexto de rastreamento da bola, a interpolação é crucial para preencher lacunas quando a detecção falha temporariamente devido a oclusões ou blur de movimento.

Dados n pontos conhecidos $(t_0, y_0), (t_1, y_1), ..., (t_{n-1}, y_{n-1})$ onde t_i representa o tempo (frame) e y_i a posição, o objetivo é construir uma função $f(t)$ tal que $f(t_i) = y_i$ para todos os pontos conhecidos e que forneça estimativas razoáveis para tempos intermediários.

Interpolação Linear: A forma mais simples conecta pontos consecutivos com segmentos de reta. Entre dois pontos (t_i, y_i) e (t_{i+1}, y_{i+1}), a interpolação linear é:

$$f(t) = y_i + \frac{y_{i+1} - y_i}{t_{i+1} - t_i}(t - t_i), \quad t_i \leq t \leq t_{i+1} \tag{2.11}$$

Embora simples e rápida, a interpolação linear produz trajetórias com mudanças abruptas de direção nos pontos de dados, não representando adequadamente o movimento suave de uma bola.

Interpolação por Spline Cúbica: Para trajetórias mais suaves, utiliza-se interpolação por splines cúbicas, que constrói polinômios de terceiro grau entre cada par de pontos consecutivos. A spline cúbica $S(t)$ satisfaz:

1. $S(t_i) = y_i$ para todos os pontos de dados (interpolação exata)
2. $S(t)$ é um polinômio cúbico em cada intervalo $[t_i, t_{i+1}]$
3. $S(t)$, $S'(t)$ e $S''(t)$ são contínuas em todo o domínio

Em cada intervalo $[t_i, t_{i+1}]$, a spline tem a forma:

$$S_i(t) = a_i + b_i(t - t_i) + c_i(t - t_i)^2 + d_i(t - t_i)^3 \tag{2.12}$$

Os coeficientes a_i, b_i, c_i, d_i são determinados resolvendo um sistema de equações que garante as condições de continuidade. Para splines naturais, utilizadas neste trabalho, impõe-se adicionalmente que $S''(t_0) = S''(t_{n-1}) = 0$, resultando em curvas que minimizam a curvatura total.

No sistema implementado, quando há quatro ou mais detecções válidas da bola, aplica-se interpolação por spline cúbica para gerar uma trajetória suave. Para casos com menos pontos, utiliza-se interpolação linear como **fallback**. As coordenadas x e y do centro da bola são interpoladas independentemente, e posteriormente as bounding boxes são reconstruídas usando o raio médio observado nas detecções válidas. Esta abordagem garante continuidade visual no rastreamento mesmo quando a detecção falha temporariamente, melhorando significativamente a qualidade da análise final.

3 Materiais e Métodos

Este capítulo apresenta a metodologia desenvolvida para construção de um sistema de detecção e rastreamento automático de jogadores e bola em vídeos de futebol. O trabalho utilizou técnicas de visão computacional e aprendizado de máquina para processar sequências de vídeo e extrair informações relevantes sobre o posicionamento e movimentação dos elementos em campo. A estrutura metodológica foi organizada em etapas sequenciais, desde a preparação dos dados até a implementação do sistema completo de análise.

3.1 Base de Dados e Preparação

A escolha da base de dados constitui um elemento crucial para o desenvolvimento e validação do sistema proposto, impactando diretamente na qualidade dos modelos treinados e na capacidade de generalização para diferentes cenários de partidas. Embora existam datasets públicos para análise de futebol, como o SoccerNet [11], a maioria foca em tarefas específicas como spotting de ações ou análise tática, não fornecendo as anotações detalhadas de bounding boxes necessárias para treinamento de modelos de detecção de objetos.

Para suprir esta lacuna, optou-se por utilizar os vídeos do conjunto DFL-Bundesliga-Data-Shootout [21], disponibilizado através da plataforma Kaggle, como fonte de dados brutos para criação de um dataset customizado. É importante destacar que os vídeos originais não continham anotações de objetos, sendo necessário desenvolver um processo completo de extração, curadoria e anotação manual para criar um dataset adequado às necessidades específicas deste trabalho. A seleção desta fonte de vídeos fundamentou-se em características técnicas específicas: gravações de alta qualidade de partidas da Bundesliga alemã incluindo diversas situações de jogo, anonimização de elementos identificadores dos times garantindo aspectos éticos, estabilidade das câmeras com variações angulares mínimas, e qualidade de imagem adequada para extração de características visuais discriminativas.

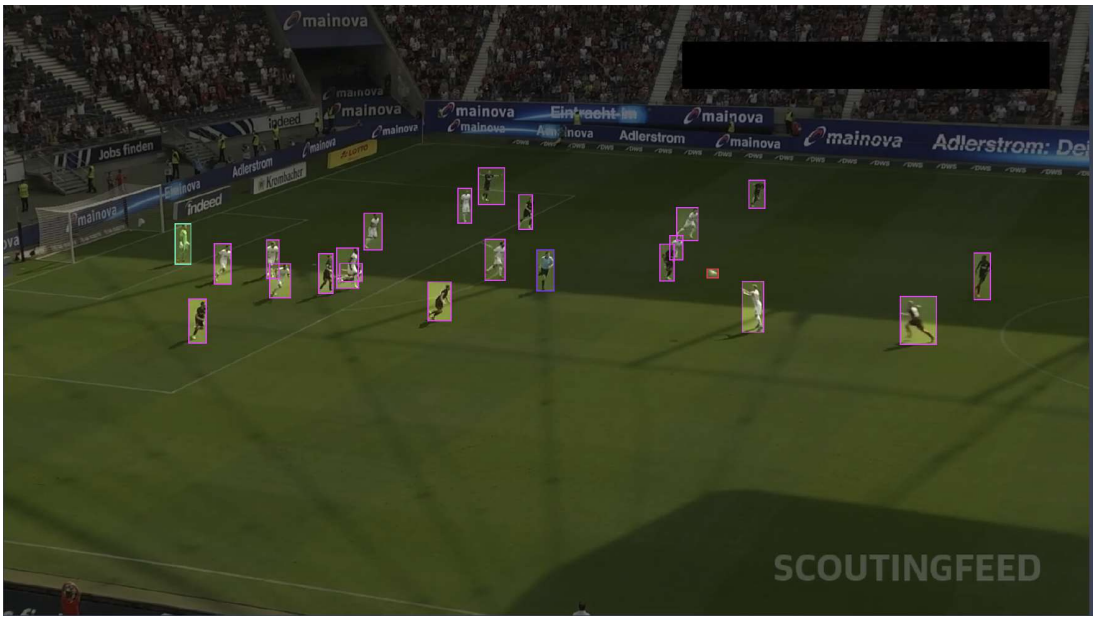
O desenvolvimento do dataset customizado envolveu múltiplas etapas cuidadosamente planejadas, desde a extração de quadros dos vídeos originais até a anotação manual completa. Esta abordagem foi necessária pois os vídeos brutos do DFL-Bundesliga, embora de alta qualidade, não possuíam as anotações de bounding boxes essenciais para treinamento de algoritmos de detecção de objetos. Inicialmente, os múltiplos clipes disponíveis foram concatenados em um único vídeo longo, facilitando o processo de amostragem uniforme e garantindo que diferentes momentos e situações de jogo fossem representados proporcionalmente no dataset final. Esta abordagem preveniu vieses que poderiam comprometer o treinamento dos modelos, assegurando que situações táticas variadas estivessem

adequadamente representadas.

A extração de quadros foi realizada utilizando a ferramenta FFmpeg [22], configurada para extrair um frame a cada 3 segundos através do comando `ffmpeg -i video.mp4 -vf "fps=1/3"frame_%04d.png`. Esta taxa de amostragem foi cuidadosamente escolhada para capturar adequadamente a dinâmica do jogo sem introduzir redundância excessiva entre quadros consecutivos, resultando em um conjunto inicial de imagens que passou por rigorosa curadoria manual. Durante a curadoria, foram removidos frames de início e fim de partida que frequentemente contêm informações não relevantes para detecção em jogo, além de imagens com qualidade comprometida por blur excessivo ou momentos de pausa prolongada. Este processo criterioso resultou na obtenção de 4.116 imagens de alta qualidade, todas mantidas com a resolução padrão de 1920×1080 pixels, mantendo a proporção original através de preenchimento quando necessário.

O processo de anotação manual foi conduzido utilizando a plataforma Roboflow [23], escolhida por sua interface intuitiva e capacidade de exportação em múltiplos formatos. A Figura 4 ilustra a interface utilizada durante o processo, demonstrando a precisão necessária na delimitação dos objetos de interesse. Durante a anotação, foram definidas quatro categorias principais baseadas na relevância para análise tática: Jogador (todos os jogadores de linha, independente da equipe), Árbitro principal (anotado separadamente devido à sua influência no posicionamento dos jogadores), Goleiros e Bola (elemento central cuja detecção precisa é fundamental para análises de posse e movimentação).

Figura 4 – Interface de anotação manual utilizando a plataforma Roboflow, demonstrando a delimitação precisa de jogadores, árbitros e bola em uma cena típica de jogo



Fonte:

Elaboração própria.

O conjunto de dados anotado desenvolvido neste trabalho, disponibilizado publicamente através da plataforma Roboflow Universe [24], representa uma contribuição significativa para a comunidade de pesquisa, totalizando 87.684 anotações distribuídas nas 4.116 imagens, resultando em uma média de 21,3 anotações por imagem. Esta densidade reflete adequadamente a complexidade típica de cenas de futebol, onde múltiplos jogadores interagem simultaneamente. A Tabela 1 apresenta as características quantitativas detalhadas da base de dados criada, revelando a distribuição natural das classes com predominância de jogadores de campo (77.618 instâncias), seguidos por árbitros (4.053), goleiros (3.249) e bolas (2.764).

Tabela 1 – Características quantitativas do dataset customizado desenvolvido, demonstrando a distribuição de classes e densidade de anotações

Parâmetro	**Valor**
Resolução padrão das imagens	1920×1080
Total de imagens processadas	4.116
Total de anotações realizadas	87.684
Média de anotações por imagem	21,3
Distribuição por Categoria	
Jogadores de campo	77.618
Árbitros	4.053
Goleiros	3.249
Bolas	2.764

Para assegurar uma avaliação robusta e imparcial do desempenho do modelo, o dataset customizado desenvolvido foi particionado seguindo a proporção 70-15-15, amplamente reconhecida como eficaz para conjuntos de dados de visão computacional [25]. O conjunto de treinamento, com 2.987 imagens (72,6% do total), fornece volume suficiente para aprendizado das características visuais complexas presentes em cenas de futebol. O conjunto de validação, contendo 564 imagens (13,7%), permite monitoramento do desempenho durante o treinamento e ajuste de hiperparâmetros, enquanto o conjunto de teste, com 565 imagens (13,7%), proporciona avaliação final imparcial com dados não vistos durante o treinamento. A criação deste dataset customizado representa uma das principais contribuições deste trabalho para a comunidade científica, fornecendo um recurso valioso para pesquisas futuras em detecção de objetos em vídeos de futebol.

3.2 Ambiente Computacional e Configuração Experimental

O desenvolvimento e treinamento dos modelos foram conduzidos em ambiente híbrido, combinando recursos de computação em nuvem para treinamento intensivo e desenvolvimento local para prototipação e testes. Esta abordagem permitiu aproveitar hardware especializado para treinamento mantendo flexibilidade para desenvolvimento iterativo.

Para o treinamento dos modelos, utilizou-se o ambiente Google Colab Pro com GPU NVIDIA A100-SXM4-40GB, proporcionando 432 Tensor Cores de terceira geração otimizados para operações de aprendizado profundo. A configuração incluiu memória HBM2 de 40GB com bandwidth de 1555GB/s, eliminando gargalos de transferência de dados, e desempenho FP32 de 19,5 TFLOPS, assegurando processamento eficiente dos modelos YOLO. O ambiente local de desenvolvimento utilizou processador AMD Ryzen 7

3700X com 8 núcleos e GPU ASUS RTX 3060 Ti, oferecendo capacidade suficiente para testes e prototipação com 64GB DDR4 de memória principal.

O desenvolvimento foi realizado em Python, utilizando o framework Ultralytics YOLOv8 [5] versão 8.3.151 como base para o sistema de detecção. Esta versão oferece implementações otimizadas com suporte nativo a GPU e técnicas avançadas de aceleração. Para o componente de rastreamento, implementou-se o algoritmo ByteTrack [8] através da biblioteca Supervision, demonstrando robustez excepcional na identificação e rastreamento simultâneo de múltiplos objetos em cenários dinâmicos. O processamento de imagens e vídeos utilizou OpenCV [26], enquanto a identificação automática de equipes foi implementada através do algoritmo K-Means [17] via Scikit-learn.

3.3 Sistema de Avaliação e Métricas

A avaliação de modelos de detecção de objetos em vídeos esportivos requer métricas que quantifiquem tanto a precisão da localização quanto a correção da classificação. Estas métricas são fundamentais para determinar se o modelo consegue identificar corretamente jogadores, bola e outros elementos relevantes para análise tática posterior.

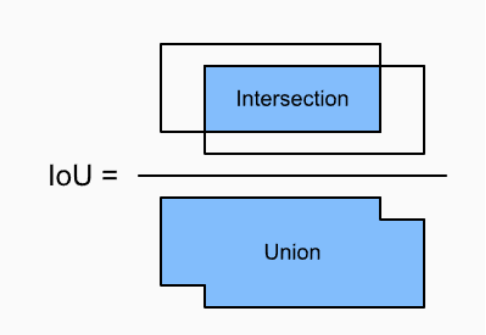
No contexto de detecção de objetos, os *bounding boxes* representam o conceito fundamental, podendo ser expressos em formato corner $(x_{min}, y_{min}, x_{max}, y_{max})$ ou formato center $(x_{center}, y_{center}, width, height)$. Para vídeos esportivos, estes devem capturar com precisão os contornos dos jogadores mesmo em situações desafiadoras como oclusões parciais e movimento rápido, sendo fundamentais para aplicações posteriores de rastreamento e análise tática [27].

A métrica fundamental para avaliar a qualidade dos *bounding boxes* é o Intersection over Union (IoU), também conhecido como índice de Jaccard. Conforme estabelecido por Everingham et al. [28], o IoU é calculado como a razão entre a área de intersecção e a área de união de dois *bounding boxes*, conforme a Equação 3.1:

$$\text{IoU} = \frac{\text{Área}(B_{pred} \cap B_{gt})}{\text{Área}(B_{pred} \cup B_{gt})} \tag{3.1}$$

onde B_{pred} representa o *bounding box* predito pelo modelo e B_{gt} o *bounding box* verdadeiro (*ground truth*). O IoU varia entre 0 (sem sobreposição) e 1 (sobreposição perfeita). A Figura 5 ilustra visualmente este conceito, demonstrando como a métrica quantifica o grau de sobreposição entre a detecção predita e a anotação verdadeira.

Figura 5 – Representação visual do cálculo de IoU entre duas *bounding boxes*, ilustrando a área de intersecção e a área de união utilizada no cálculo



Fonte: Adaptado de Zhang et al. [15]

Para classificar uma detecção como verdadeiro positivo (TP), falso positivo (FP) ou falso negativo (FN), utiliza-se um limiar de IoU. Tipicamente, uma detecção é considerada correta quando IoU ≥ 0.5, embora aplicações que requerem maior precisão possam usar limiares mais rigorosos. Com base nestas classificações, calculam-se as métricas de precisão e revocação conforme definido por Padilla et al. [29]:

$$\text{Precisão} = \frac{TP}{TP + FP} \tag{3.2}$$

$$\text{Revocação} = \frac{TP}{TP + FN} \tag{3.3}$$

A precisão representa a proporção de detecções corretas entre todas as detecções realizadas, enquanto a revocação indica a proporção de objetos verdadeiros que foram detectados. Para integrar o desempenho em diferentes níveis de confiança, calcula-se a Precisão Média (AP) como a área sob a curva precisão-revocação:

$$AP = \int_0^1 P(R)\, dR \tag{3.4}$$

onde $P(R)$ representa a precisão como função da revocação. Na prática, esta integral é aproximada através da interpolação de 11 pontos conforme estabelecido no desafio PASCAL VOC:

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, 0.2, \ldots, 1.0\}} P_{interp}(r) \tag{3.5}$$

onde $P_{interp}(r) = \max_{\tilde{r} \geq r} P(\tilde{r})$ representa a precisão interpolada.

O mAP (**mean Average Precision**) consolida o desempenho multi-classe através da média das APs calculadas para cada categoria:

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{3.6}$$

onde N representa o número total de classes e AP_i a precisão média da classe i.

Duas variantes principais foram utilizadas neste trabalho: mAP@0.5, que utiliza IoU fixo de 0.5 estabelecido no desafio PASCAL VOC [28], e mAP@[0.5:0.95], que fornece avaliação mais rigorosa através da média do mAP calculado em múltiplos limiares de IoU:

$$mAP@[0.5 : 0.95] = \frac{1}{10} \sum_{t=0.5}^{0.95} mAP@t \tag{3.7}$$

onde t varia de 0.5 a 0.95 em incrementos de 0.05, seguindo o protocolo estabelecido pelo dataset COCO.

A escolha destas métricas baseia-se em evidências empíricas de estudos recentes em detecção esportiva, onde Kumar et al. [30] demonstraram que incrementos em mAP@0.5 correlacionam diretamente com melhorias perceptíveis na qualidade da detecção. No contexto específico deste trabalho, a métrica mAP@0.5 assume importância particular devido à presença da bola como objeto de interesse. Estudos em detecção de objetos pequenos demonstram que limiares de IoU muito elevados (>0.7) podem ser inadequados para objetos que ocupam área reduzida no quadro, como a bola de futebol em transmissões de câmera padrão [8]. A natureza física da bola, combinada com fatores como movimento rápido, oclusões frequentes e baixa resolução relativa, resulta em detecções que, mesmo sendo visualmente aceitáveis para aplicações de rastreamento, podem não atingir limiares de IoU superiores a 0.8 ou 0.9.

Portanto, embora o mAP@[0.5:0.95] forneça uma avaliação abrangente para objetos grandes como jogadores, a métrica mAP@0.5 constitui um indicador mais apropriado e prático para avaliar o desempenho geral do sistema, especialmente considerando que uma detecção com IoU 0.5 já fornece informação espacial suficiente para algoritmos de rastreamento subsequentes. Esta abordagem alinha-se com recomendações da literatura especializada em detecção de objetos pequenos em vídeos esportivos.

3.4 Configuração dos Experimentos

Foram realizados três experimentos sistemáticos para avaliar diferentes configurações de modelo e parâmetros, conforme apresentado na Tabela 2. O Experimento 1 estabeleceu um ***baseline*** utilizando YOLOv8n com resolução de 640px, priorizando velocidade de inferência e menor uso de recursos computacionais. O Experimento 2 avaliou o impacto do aumento da capacidade do modelo através do YOLOv8l mantendo a mesma resolução, explorando o compromisso entre complexidade e precisão. O Experimento 3 in-

vestigou o efeito da alta resolução (1920px) na detecção de objetos pequenos, especialmente a bola, mantendo arquitetura leve para viabilizar o processamento.

Tabela 2 – Configuração completa dos experimentos de treinamento, destacando as diferenças entre cada abordagem

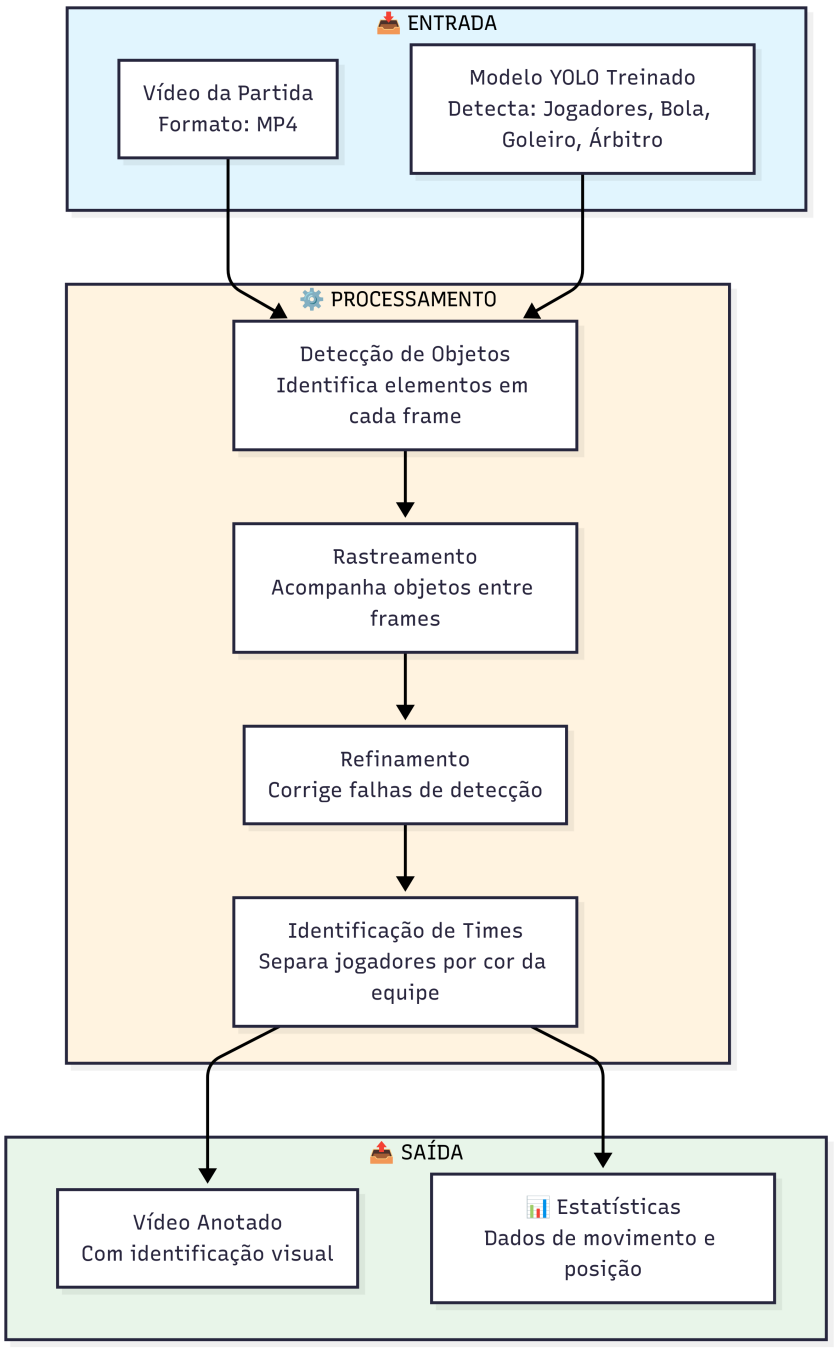
Parâmetro	**Exp. 1**	**Exp. 2**	**Exp. 3**
Modelo base	YOLOv8n	YOLOv8l	YOLOv8n
Parâmetros (M)	3.2	43.7	3.2
FLOPs (G)	8.1	164.8	8.1
Resolução (imgsz)	640px	640px	1920px
Batch size	16	16	16
Épocas	200	200	200
Paciência	300	300	300
Otimizador	AdamW	AdamW	AdamW
Learning rate inicial	0.01	0.01	0.01

Todos os experimentos utilizaram configurações de treinamento consistentes para garantir comparabilidade. Não foi utilizado o critério de early stopping porque para obter um comparativo de referência optou-se para sempre rodar 200 épocas em cada experimento, salvando automaticamente o melhor modelo baseado na métrica de mAP@0.5.

3.5 Arquitetura do Sistema Desenvolvido

O sistema implementado segue uma arquitetura modular organizada em etapas sequenciais de processamento, onde cada módulo foi projetado para abordar desafios específicos na análise automatizada dos vídeos de futebol. O pipeline completo de análise integra diferentes componentes que processam o vídeo desde sua entrada até a geração de análises completas, passando por etapas de detecção, rastreamento, refinamento e identificação de equipes. A Figura 6 apresenta o fluxograma geral desta arquitetura.

Figura 6 – Fluxograma geral da arquitetura do sistema, ilustrando o fluxo de processamento desde a entrada do vídeo até a saída com análises completas



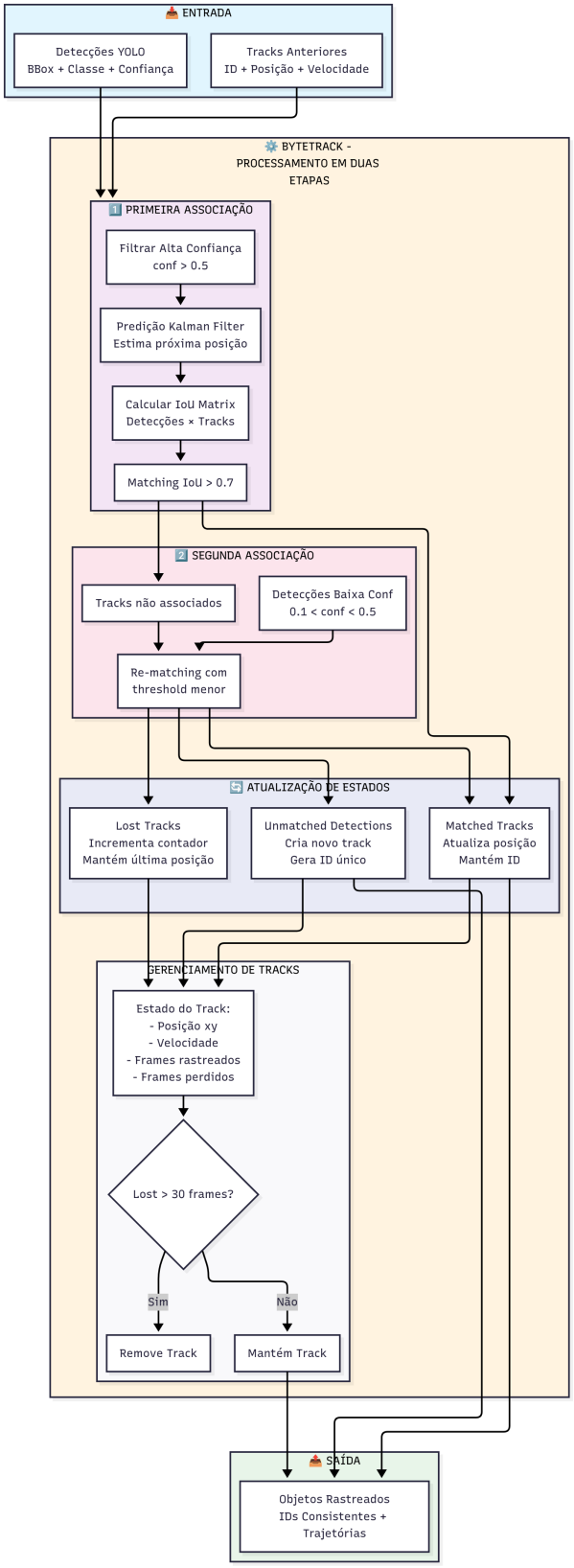
Fonte: Elaboração própria.

3.5.1 Módulo de Detecção e Rastreamento

O núcleo do sistema baseia-se na detecção através do modelo YOLO treinado, seguido pelo rastreamento utilizando o algoritmo ByteTrack [8]. Este algoritmo foi escolhido por sua robustez no rastreamento de múltiplos objetos em cenários dinâmicos, características essenciais para análise esportiva.

O ByteTrack implementa uma estratégia de associação em dois níveis: primeiro utiliza detecções de alta confiança (0.5) para manter tracks estáveis, depois emprega detecções de baixa confiança (0.1 a 0.5) para recuperar tracks temporariamente perdidas. O processo interno utiliza o Filtro Kalman para predição de movimento e o algoritmo Hungarian para associação ótima entre detecções e rastreamentos existentes. A Figura 7 detalha o fluxo completo deste algoritmo.

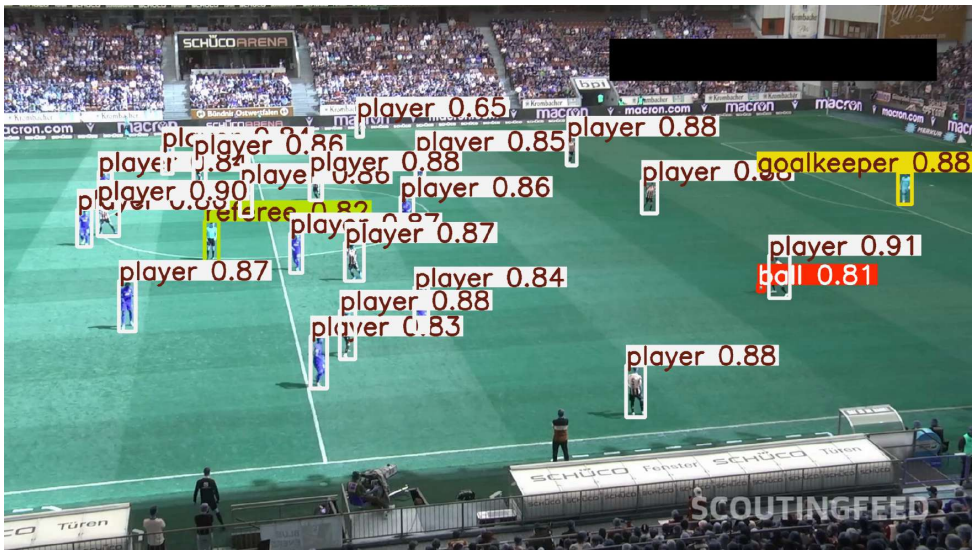
Figura 7 – Fluxograma detalhado do algoritmo de rastreamento ByteTrack, ilustrando o processo de associação em dois níveis



Fonte: Elaboração própria.

O resultado prático deste módulo pode ser observado nas etapas de processamento. Inicialmente, o modelo treinado realiza a detecção dos elementos, identificando jogadores, árbitro e bola com suas respectivas *bounding boxes*, como demonstrado na Figura 8.

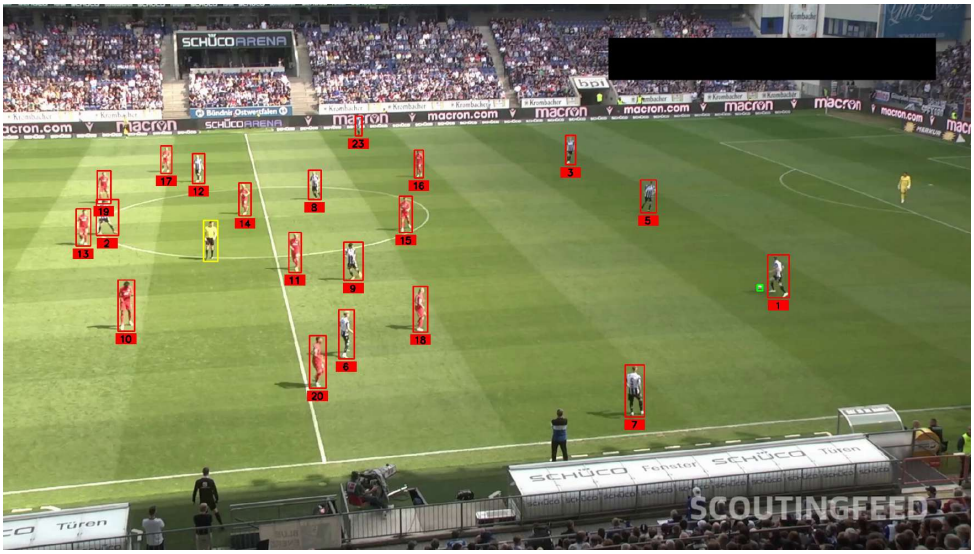
Figura 8 – Imagem de entrada processada pelo modelo treinado, mostrando as detecções iniciais de jogadores, árbitro e bola



Fonte: Elaboração própria.

Após a detecção inicial, o sistema de rastreamento atribui IDs únicos para cada elemento detectado, permitindo o acompanhamento individual ao longo do vídeo. É importante notar que o sistema mantém os IDs do árbitro e bola ocultos para simplificar a visualização, enquanto o goleiro não é rastreado nesta implementação inicial. A Figura 9 apresenta o resultado com os IDs atribuídos.

Figura 9 – Imagem de saída com o rastreamento ativo, exibindo IDs únicos para cada jogador detectado



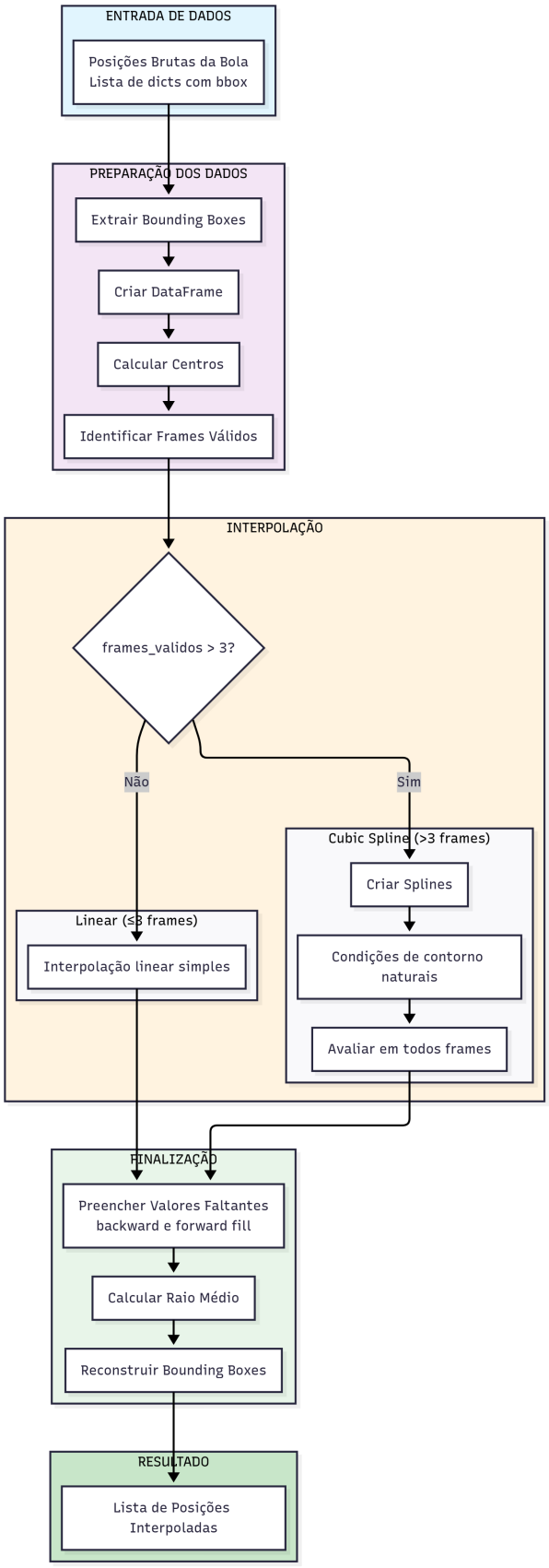
Fonte: Elaboração própria.

3.5.2 Módulo de Refinamento de Trajetórias

O refinamento de trajetórias constitui uma etapa fundamental para correção de inconsistências na detecção da bola, que frequentemente apresenta falhas devido ao seu tamanho reduzido e movimento rápido. O processo implementado utiliza técnicas de interpolação matemática para suavizar e completar trajetórias incompletas.

O processo de refinamento opera em cinco etapas sequenciais. Primeiro, são extraídas as coordenadas centrais das **bounding boxes** detectadas para a bola em cada quadro. Em seguida, identifica-se os quadros onde a bola foi detectada com sucesso, criando um índice de quadros válidos. A terceira etapa implementa interpolação adaptativa conforme explicado na fundamentação teórica: utiliza-se spline cúbica quando há quatro ou mais pontos válidos (proporcionando suavização superior através de curvas contínuas), ou interpolação linear como alternativa quando há menos pontos disponíveis. Para quadros nas extremidades do vídeo onde não há dados suficientes, aplica-se preenchimento utilizando os valores válidos mais próximos. Finalmente, as coordenadas interpoladas são reconvertidas para o formato de **bounding box**, calculando-se o raio médio da bola e reconstruindo as caixas delimitadoras com dimensões consistentes. A Figura 10 ilustra este processo completo.

Figura 10 – Fluxograma do algoritmo de refinamento de posições da bola, demonstrando as etapas de interpolação e reconstrução



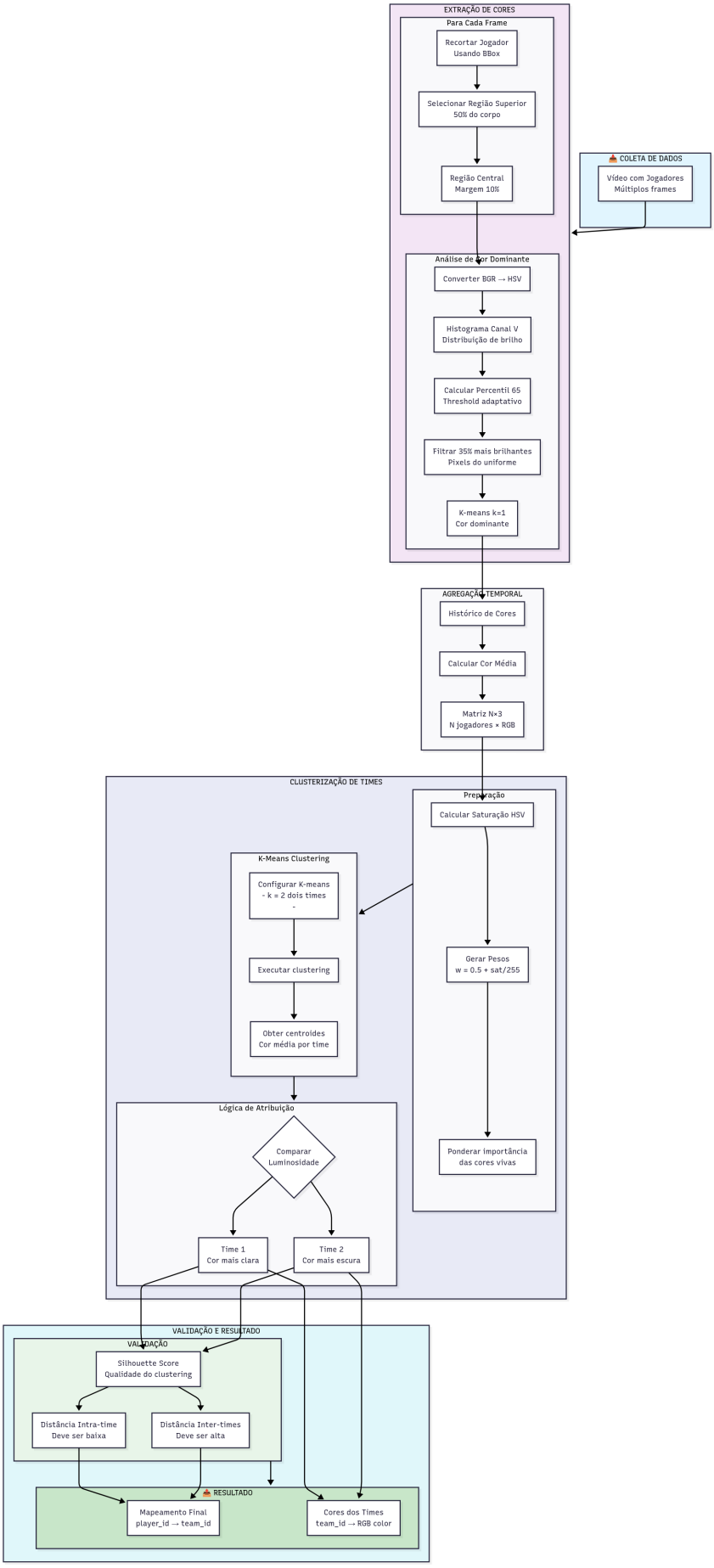
Fonte: Elaboração própria.

3.5.3 Módulo de Identificação Automática de Equipes

O sistema incorpora um módulo inovador para identificação automática das equipes baseada nas cores dos uniformes, utilizando clusterização e técnicas de visão computacional. O algoritmo processa as detecções de jogadores para extrair e analisar as cores predominantes dos uniformes através de um processo em múltiplas etapas.

O processo de extração de características implementa uma abordagem em três estágios. Inicialmente, para cada jogador detectado, define-se uma Região de Interesse (ROI) focada no torso superior, correspondendo aos 30% superiores da **bounding box**. Esta área apresenta maior consistência cromática em uniformes esportivos, minimizando interferências de números, patrocínios e shorts que poderiam comprometer a análise. Após a definição da ROI, realiza-se o pré-processamento de cores com conversão para o espaço HSV (Matiz, Saturação, Valor) para melhor discriminação de tonalidades, seguido por filtragem de pixels com saturação inferior a 50 para remoção de tons acinzentados. A seleção dos 35% pixels com maior valor de brilho minimiza interferências de sombreamento, resultando na identificação precisa da cor dominante do uniforme. A Figura 11 apresenta o fluxograma completo deste algoritmo.

Figura 11 – Fluxograma do algoritmo de identificação de equipes, mostrando o processo de extração e clusterização de cores



Fonte: Elaboração própria.

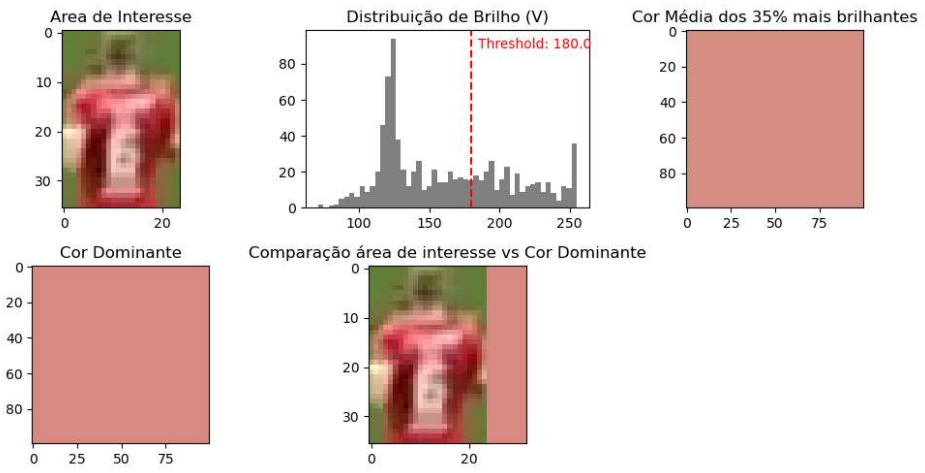
A definição precisa da Região de Interesse é fundamental para o sucesso da identificação. A área selecionada, correspondente aos 30% superiores da **bounding box**, é ilustrada na Figura 12, onde o destaque em vermelho indica a região utilizada para análise.

Figura 12 – Definição da Região de Interesse (ROI) para extração de cor do uniforme, com destaque em vermelho para a área selecionada

Fonte: Elaboração própria.

O processo de filtragem no espaço HSV é crucial para a correta identificação das cores. Através da conversão para este espaço de cores e aplicação de filtros baseados em saturação e brilho, consegue-se isolar as cores mais representativas do uniforme, eliminando interferências de sombras e variações de iluminação. A Figura 13 demonstra este processo em detalhes, mostrando cada etapa desde o recorte da área de interesse até a identificação da cor dominante.

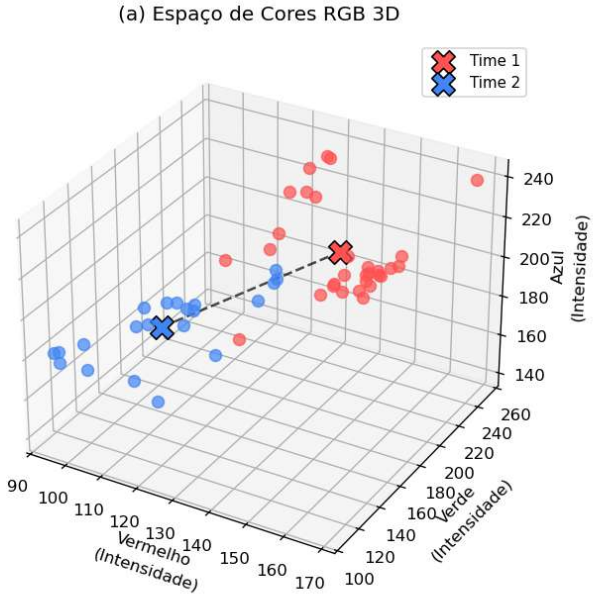
Figura 13 – Processo de filtragem no espaço HSV demonstrando: (a) recorte da área de interesse, (b) distribuição do brilho, (c) cor média dos pixels mais brilhantes, (d) cor dominante identificada, e (e) comparação com a imagem original



Fonte: Elaboração própria.

Para a distinção automática das equipes, implementou-se uma versão modificada do algoritmo K-Means [17] com ponderação baseada em saturação. Esta ponderação atribui maior importância a cores mais vibrantes, característica típica de uniformes esportivos. O algoritmo agrupa as cores extraídas em dois clusters distintos, cada um correspondendo a uma equipe. A distribuição das cores no espaço RGB tridimensional demonstra a separação clara entre as duas equipes através dos centróides calculados, como apresentado na Figura 14.

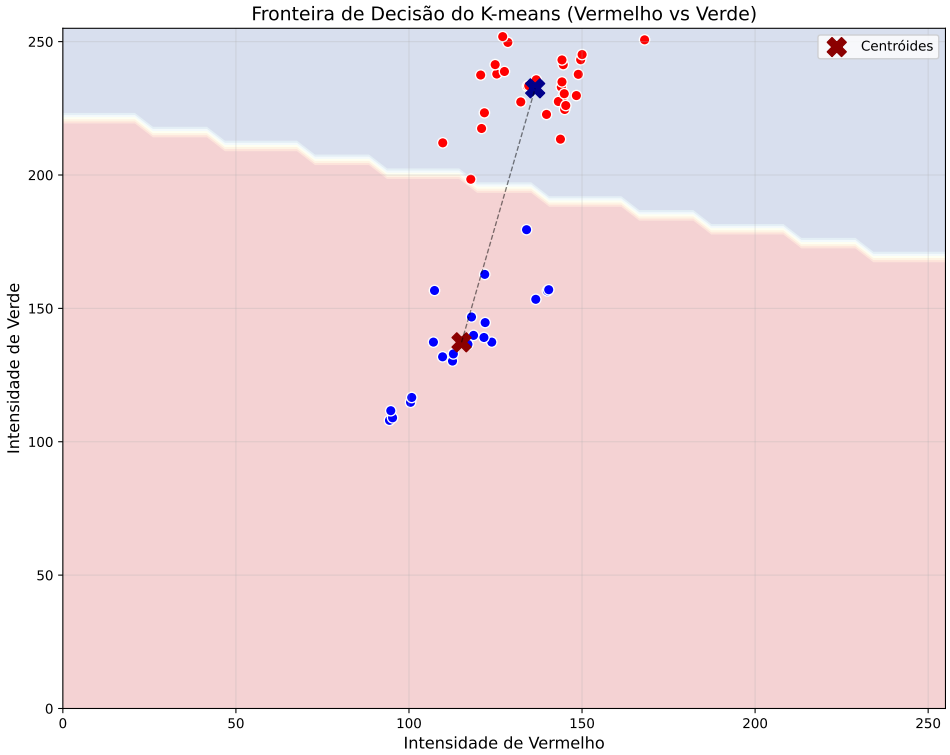
Figura 14 – Espaço de cores RGB mostrando os centróides identificados para cada time, demonstrando a separabilidade das equipes



Fonte: Elaboração própria.

Os critérios de validação implementados garantem a qualidade da classificação através de verificações de distância mínima entre centróides (limiar de 50 unidades RGB), distribuição equilibrada (proporção entre 40-60% de jogadores por equipe), e consistência temporal (manutenção da classificação por pelo menos 10 quadros consecutivos). O resultado da clusterização através do algoritmo K-Means é ilustrado na Figura 15, onde cada ponto representa a cor extraída de um jogador e as cores são agrupadas em dois clusters distintos.

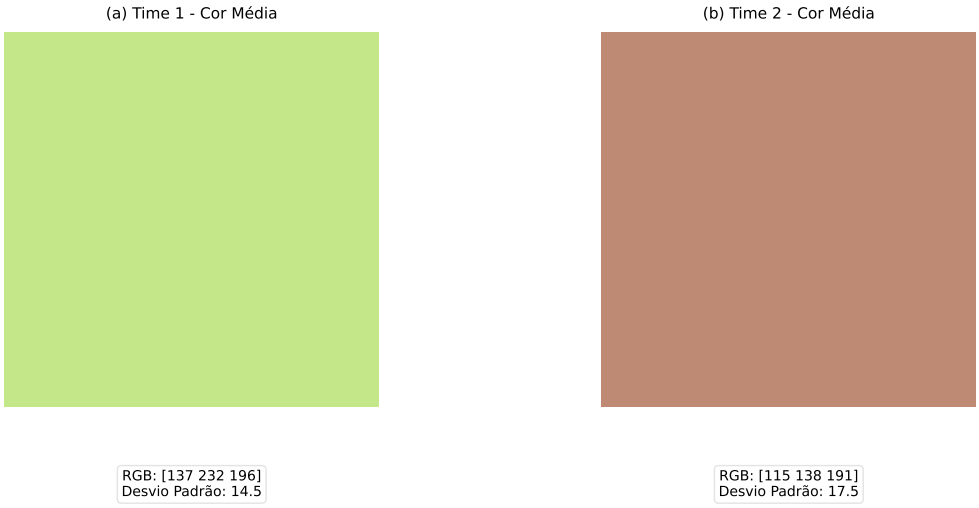
Figura 15 – Resultado da clusterização K-Means mostrando a separação das cores em dois grupos distintos correspondentes às equipes



Fonte: Elaboração própria.

O sistema produz como saída final a identificação visual dos jogadores com cores correspondentes às suas equipes, mantendo consistência dos IDs através dos quadros. As cores dominantes detectadas para cada time são extraídas automaticamente dos uniformes, como exemplificado na Figura 16.

Figura 16 – Exemplo das cores dominantes detectadas para o Time 1 (verde) e Time 2 (azul), extraídas automaticamente dos uniformes



Fonte: Elaboração própria.

O resultado final da identificação automática apresenta os jogadores devidamente identificados e coloridos de acordo com suas equipes, com o Time 1 em verde (RGB: 137, 232, 196) e Time 2 em vermelho (RGB: 115, 138, 191), como demonstrado na Figura 17.

Figura 17 – Resultado final da identificação automática mostrando Time 1 em verde (RGB: 137, 232, 196) e Time 2 em vermelho (RGB: 115, 138, 191)

Fonte: Elaboração própria.

4 Resultados e Discussão

Este capítulo apresenta uma análise detalhada dos resultados obtidos através dos três experimentos de treinamento propostos na metodologia. A avaliação foi conduzida seguindo métricas estabelecidas na literatura de detecção de objetos, permitindo uma análise comparativa entre as diferentes configurações testadas e identificação das abordagens mais eficazes para o problema proposto.

4.1 Visão Geral dos Resultados

Os três experimentos realizados tiveram como objetivo avaliar o impacto de diferentes arquiteturas e resoluções na qualidade de detecção dos elementos de interesse em vídeos de futebol. A Tabela 3 apresenta um resumo comparativo dos principais resultados, onde o Experimento 1 (YOLOv8n-640) foi estabelecido como ***baseline*** para comparações subsequentes. Os percentuais indicados representam a variação em relação a este ***baseline***, permitindo uma compreensão clara do impacto de cada configuração.

Tabela 3 – Resumo comparativo dos experimentos realizados com variações percentuais em relação ao baseline

Métrica	**Exp. 1 (Baseline) YOLOv8n-640**	**Exp. 2 YOLOv8l-640**	**Exp. 3 YOLOv8n-1920**	**Melhor Resultado**
mAP@0.5	0,824	0,870 (+5,6%)	**0,940 (+14,1%)**	Exp. 3
mAP@0.5:0.95	0,546	0,601 (+10,1%)	**0,649 (+18,9%)**	Exp. 3
Precisão	0,892	0,916 (+2,7%)	**0,964 (+8,1%)**	Exp. 3
Revocação	0,785	0,825 (+5,1%)	**0,910 (+15,9%)**	Exp. 3
Tempo (ms)	**0,3**	1,7 (+467%)	1,2 (+300%)	Exp. 1
Parâmetros (M)	**3,0**	43,6 (+1353%)	**3,0 (0%)**	Exp. 1/3

Os resultados evidenciam que o Experimento 3, utilizando YOLOv8n com alta resolução, alcançou o melhor desempenho geral com mAP@0.5 de 0,940, representando uma melhoria de 14,1% sobre o **baseline**. Notavelmente, esta configuração manteve a mesma quantidade de parâmetros do modelo **baseline** (3,0M), demonstrando que o aumento de resolução pode ser mais eficaz que o aumento de complexidade arquitetural para este domínio específico.

4.2 Análise Detalhada do Experimento 1 - Baseline

O primeiro experimento estabeleceu a linha de base utilizando a arquitetura YOLOv8n com resolução de entrada de 640 pixels, representando uma configuração padrão amplamente utilizada na literatura. A análise de convergência, apresentada na Figura 18, demonstra que o modelo alcançou estabilidade após aproximadamente 150 épocas, com as curvas de perda de treinamento e validação convergindo de forma paralela, indicando ausência de overfitting significativo.

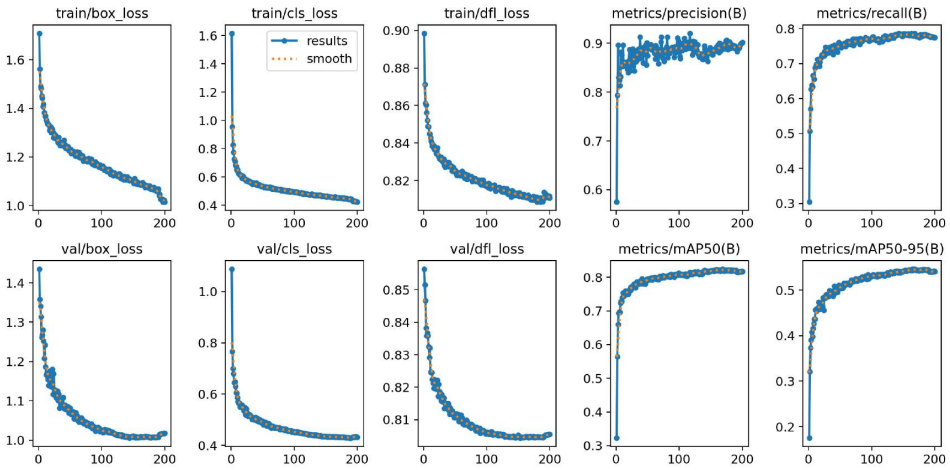


Figura 18 – Evolução da perda para YOLOv8n com resolução 640px (Baseline), mostrando convergência estável sem sinais de overfitting

A matriz de confusão normalizada do modelo **baseline**, ilustrada na Figura 19, revela padrões importantes sobre o desempenho por classe. O modelo demonstrou excelente capacidade de distinção entre as classes de pessoas (jogador, árbitro e goleiro), com taxas de classificação correta superiores a 95% para estas categorias. Entretanto, a detecção da bola apresentou desafios significativos, com apenas 24,2% de taxa de detecção correta, estabelecendo este como o principal ponto de melhoria para os experimentos subsequentes.

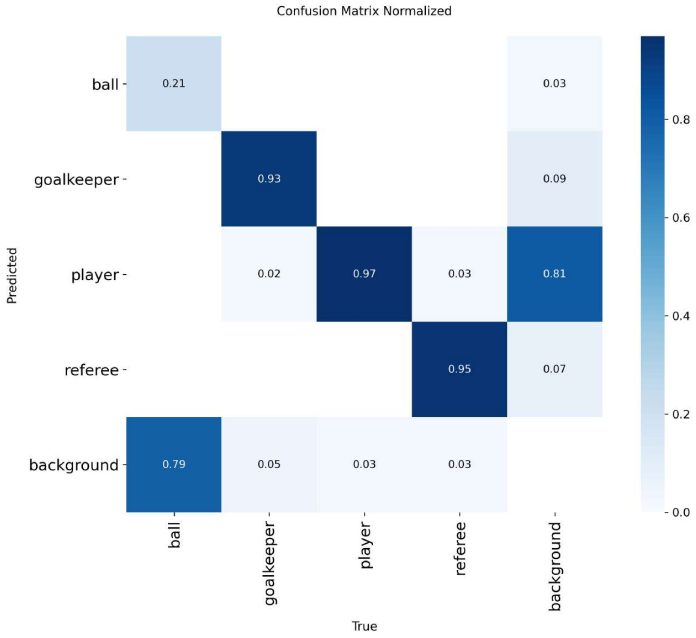


Figura 19 – Matriz de confusão normalizada para YOLOv8n-640 (Baseline), evidenciando alta precisão para pessoas mas limitações na detecção da bola

A análise detalhada das métricas por classe, apresentada na Tabela 4, confirma esta discrepância de desempenho. Enquanto as classes relacionadas a pessoas alcançaram valores de mAP@0.5 superiores a 96%, a bola obteve apenas 35,7%, com revocação particularmente baixa de 24,2%. Esta limitação pode ser atribuída ao tamanho reduzido da bola na resolução de 640px, onde o objeto ocupa tipicamente menos de 0,1% da área total da imagem, resultando em densidade insuficiente de pixels para extração eficaz de características.

Tabela 4 – Métricas de desempenho por classe - Experimento 1 (Baseline)

Classe	**Precisão**	**Revocação**	**mAP@0.5**	**mAP@0.5:0.95**
Jogador	0,955	0,975	0,989	0,716
Árbitro	0,939	0,965	0,982	0,702
Goleiro	0,903	0,958	0,966	0,634
Bola	0,774	0,242	0,357	0,133
Média	**0,892**	**0,785**	**0,824**	**0,546**

4.3 Análise do Experimento 2 - Alta Capacidade

O segundo experimento avaliou o impacto do aumento da capacidade do modelo através da utilização da arquitetura YOLOv8l, que possui 43,7 milhões de parâmetros comparados aos 3,2 milhões do YOLOv8n. Esta configuração manteve a resolução de entrada em 640 pixels, permitindo isolar o efeito da complexidade arquitetural no desempenho.

A evolução das perdas, apresentada na Figura 20, demonstra convergência mais eficiente que o **baseline**, com as perdas estabilizando em valores menores devido à maior capacidade de representação do modelo. O modelo convergiu mais rapidamente, alcançando estabilidade por volta da época 120, sugerindo que a maior capacidade permite aprendizado mais eficiente das características visuais.

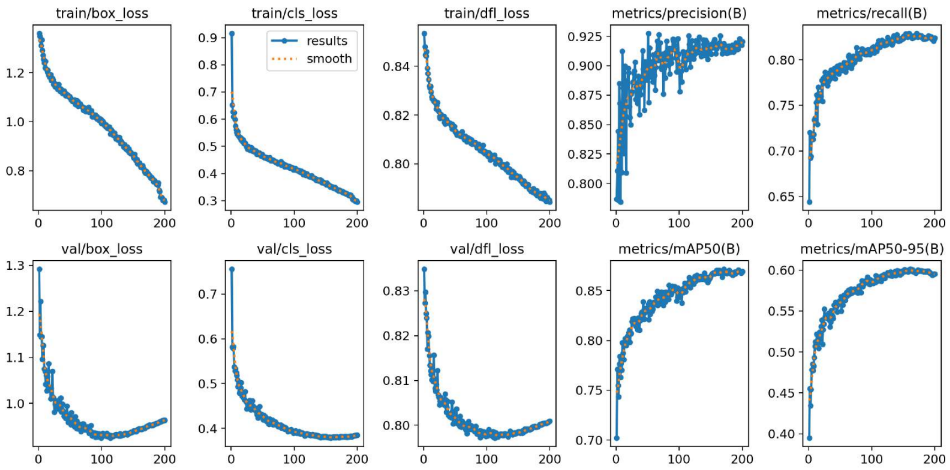


Figura 20 – Evolução da perda para YOLOv8l com resolução 640px, demonstrando convergência mais rápida e valores finais menores que o baseline

A matriz de confusão do YOLOv8l, ilustrada na Figura 21, revela melhorias notáveis na detecção da bola, com a taxa de detecção correta aumentando de 24,2% para 35,9%. Este aumento de aproximadamente 48% na capacidade de detecção da bola demonstra que modelos com maior capacidade conseguem aprender representações mais complexas mesmo com informação limitada de entrada.

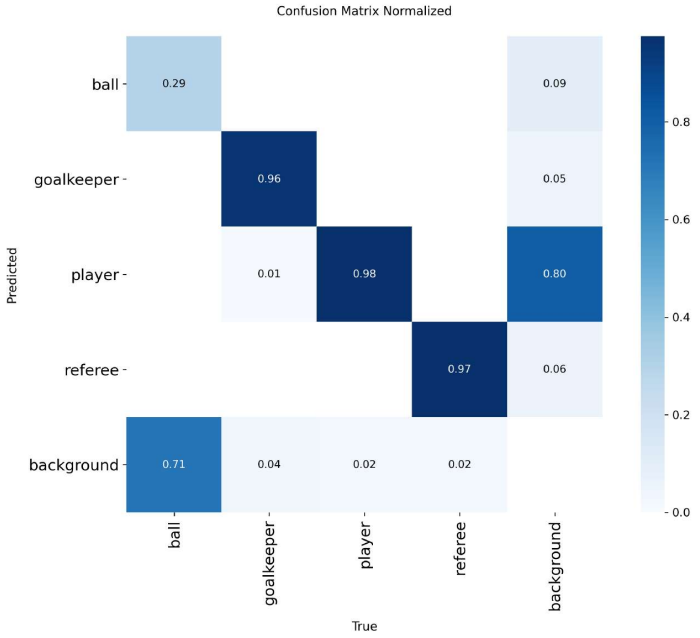


Figura 21 – Matriz de confusão normalizada para YOLOv8l-640, mostrando melhoria significativa na detecção da bola mantendo alta precisão nas demais classes

A comparação detalhada entre o Experimento 2 e o **baseline**, apresentada na Tabela 5, quantifica as melhorias obtidas. O aumento de 45,4% no mAP@0.5 da bola representa um avanço significativo, embora ainda insuficiente para aplicações práticas que demandam alta confiabilidade. É importante notar que este ganho de desempenho veio com custo computacional substancial: o tempo de inferência aumentou 467%, de 0,3ms para 1,7ms por frame, potencialmente limitando aplicações em tempo real.

Tabela 5 – Comparação detalhada entre Experimento 2 e Baseline por classe

Classe	**Métrica**	**Baseline**	**Exp. 2**	**Diferença**	**Variação (%)**
Jogador	Precisão	0,955	0,975	+0,020	+2,1
	Revocação	0,975	0,983	+0,008	+0,8
	mAP@0.5	0,989	0,992	+0,003	+0,3
	mAP@0.5:0.95	0,716	0,751	+0,035	+4,9
Árbitro	Precisão	0,939	0,966	+0,027	+2,9
	Revocação	0,965	0,984	+0,019	+2,0
	mAP@0.5	0,982	0,991	+0,009	+0,9
	mAP@0.5:0.95	0,702	0,749	+0,047	+6,7
Goleiro	Precisão	0,903	0,957	+0,054	+6,0
	Revocação	0,958	0,974	+0,016	+1,7
	mAP@0.5	0,966	0,976	+0,010	+1,0
	mAP@0.5:0.95	0,634	0,695	+0,061	+9,6
Bola	Precisão	0,774	0,766	-0,008	-1,0
	Revocação	0,242	0,359	**+0,117**	**+48,3**
	mAP@0.5	0,357	0,519	**+0,162**	**+45,4**
	mAP@0.5:0.95	0,133	0,209	+0,076	+57,1

4.4 Análise do Experimento 3 - Alta Resolução

O terceiro experimento investigou o impacto da alta resolução mantendo a arquitetura leve do YOLOv8n. Utilizando resolução de entrada de 1920 pixels, esta configuração aumentou em nove vezes a densidade de pixels disponíveis para cada objeto comparado ao **baseline**, permitindo extração mais detalhada de características visuais.

A curva de convergência, apresentada na Figura 22, demonstra padrão de aprendizado estável com declínio consistente nas perdas. O modelo requereu menos épocas para convergir (aproximadamente 100) comparado ao **baseline**, sugerindo que a maior quantidade de informação visual demandou menos tempo de treinamento para se obter um resultado estável.

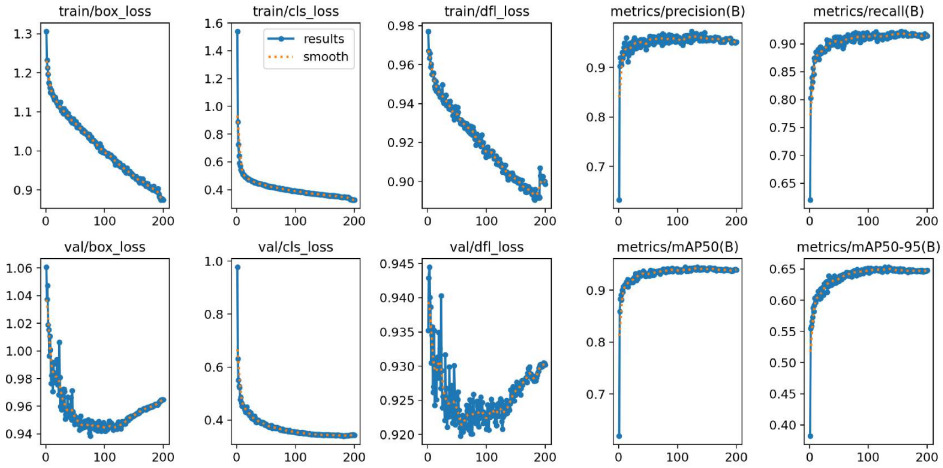


Figura 22 – Evolução da perda para YOLOv8n com resolução 1920px, mostrando convergência gradual e estável

A matriz de confusão do Experimento 3, ilustrada na Figura 23, revela transformação dramática no desempenho de detecção da bola. A taxa de detecção correta saltou de 24,2% no **baseline** para 70,5%, representando melhoria de 191,3%. Esta melhoria substancial confirma a hipótese de que a densidade de pixels é fator crítico para detecção de objetos pequenos em vídeos esportivos.

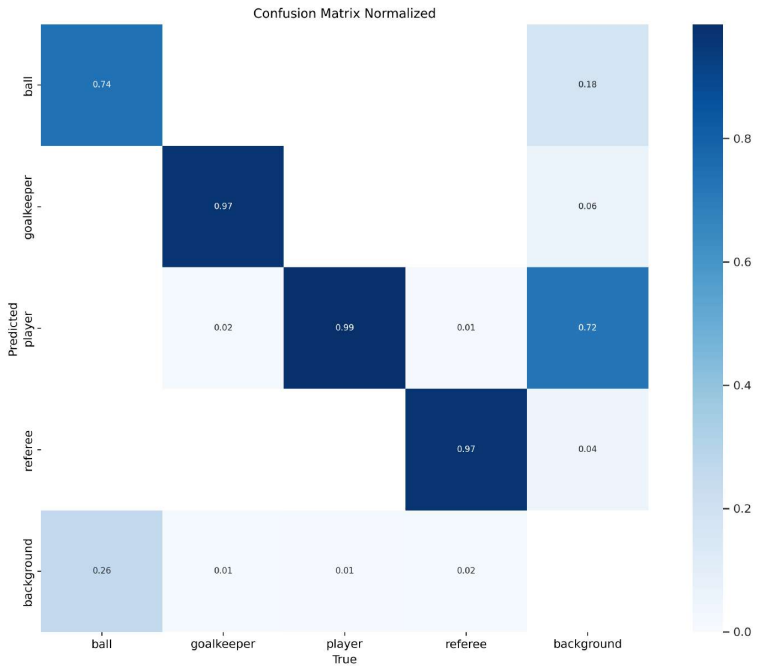


Figura 23 – Matriz de confusão normalizada para YOLOv8n-1920, demonstrando melhoria dramática na detecção da bola

A análise comparativa detalhada, apresentada na Tabela 6, quantifica os ganhos obtidos através do aumento de resolução. O mAP@0.5 da bola aumentou de 0,357 para 0,805, uma melhoria de 125,5% que torna o sistema viável para aplicações práticas. Notavelmente, esta configuração também melhorou o desempenho das demais classes, com aumentos de 3,2% a 7,1% na precisão, sugerindo que a maior resolução beneficia a detecção de todos os objetos, não apenas os menores.

Tabela 6 – Comparação detalhada entre Experimento 3 e Baseline por classe

Classe	**Métrica**	**Baseline**	**Exp. 3**	**Diferença**	**Variação (%)**
Jogador	Precisão	0,955	0,986	+0,031	+3,2
	Revocação	0,975	0,983	+0,008	+0,8
	mAP@0.5	0,989	0,990	+0,001	+0,1
	mAP@0.5:0.95	0,716	0,763	+0,047	+6,6
Árbitro	Precisão	0,939	0,991	+0,052	+5,5
	Revocação	0,965	0,978	+0,013	+1,3
	mAP@0.5	0,982	0,988	+0,006	+0,6
	mAP@0.5:0.95	0,702	0,750	+0,048	+6,8
Goleiro	Precisão	0,903	0,967	+0,064	+7,1
	Revocação	0,958	0,974	+0,016	+1,7
	mAP@0.5	0,966	0,976	+0,010	+1,0
	mAP@0.5:0.95	0,634	0,718	+0,084	+13,2
Bola	Precisão	0,774	0,912	+0,138	+17,8
	Revocação	0,242	0,705	**+0,463**	**+191,3**
	mAP@0.5	0,357	0,805	**+0,448**	**+125,5**
	mAP@0.5:0.95	0,133	0,366	+0,233	+175,2

%

4.5 Análise Comparativa e Discussão

A análise consolidada dos três experimentos revela padrões importantes sobre os fatores que determinam o desempenho em detecção de objetos esportivos. A Tabela 7 sintetiza a evolução das métricas principais para cada classe, destacando as configurações que alcançaram melhor desempenho.

Tabela 7 – Análise consolidada da evolução das métricas principais por classe

Classe	Métrica	Baseline YOLOv8n-640	Exp. 2 YOLOv8l-640	Exp. 3 YOLOv8n-1920	Melhor Resultado	Melhoria vs. Baseline
Jogador	mAP@0.5	0,989	0,992	0,990	0,992	+0,3%
	Revocação	0,975	**0,983**	**0,983**	0,983	+0,8%
Árbitro	mAP@0.5	0,982	**0,991**	0,988	0,991	+0,9%
	Revocação	0,965	**0,984**	0,978	0,984	+2,0%
Goleiro	mAP@0.5	0,966	**0,976**	**0,976**	0,976	+1,0%
	Revocação	0,958	**0,974**	**0,974**	0,974	+1,7%
Bola	mAP@0.5	0,357	0,519	**0,805**	**0,805**	**+125,5%**
	Revocação	0,242	0,359	**0,705**	**0,705**	**+191,3%**

Os resultados demonstram claramente que a densidade de pixels emerge como fator crítico para detecção de objetos pequenos. No Experimento 3, o aumento da resolução de 640px para 1920px resultou em aproximadamente nove vezes mais pixels por objeto. Para a bola especificamente, isso significa aumentar de uma média de 100-400 pixels (em 640px) para 900-3600 pixels (em 1920px). Esta densidade adicional permite que a rede neural extraia características mais discriminativas, explicando o aumento dramático de 191,3% na revocação.

A relação entre capacidade do modelo e eficiência também merece análise cuidadosa. O Experimento 2 demonstrou que aumentar a capacidade do modelo em 14 vezes (de 3,2M para 43,7M parâmetros) sem aumentar a resolução resulta em melhorias marginais para objetos grandes mas significativas para objetos pequenos. Entretanto, quando comparamos o custo-benefício, o YOLOv8n-1920 alcança mAP@0.5 14,1% superior ao **baseline** com tempo de inferência 4 vezes maior, enquanto o YOLOv8l-640 alcança apenas 5,6% de melhoria com tempo 5,7 vezes maior. Isso sugere que, para cenários com objetos pequenos, investir em maior resolução oferece retorno superior ao aumento de complexidade arquitetural.

4.6 Análise Qualitativa do Sistema Completo

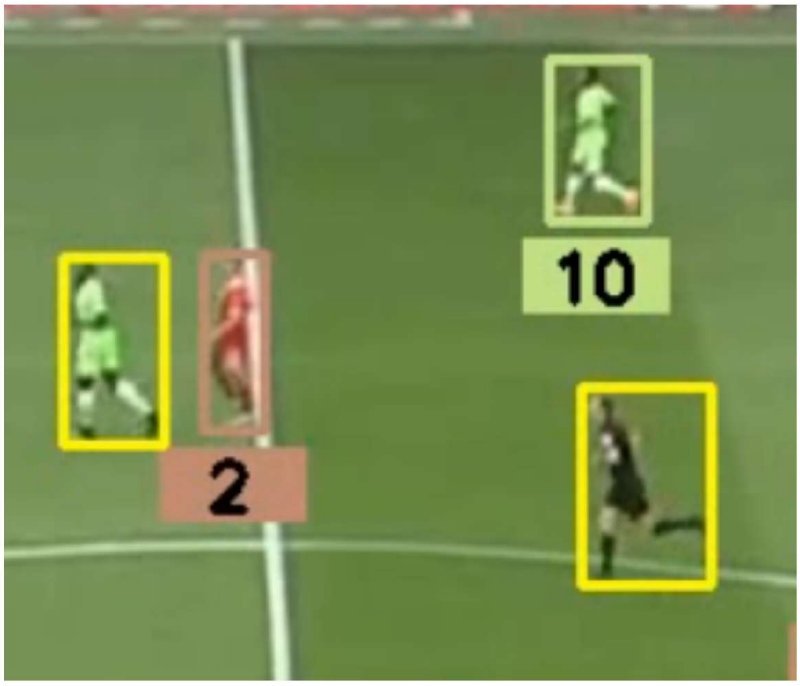
Além das métricas quantitativas, a avaliação qualitativa do sistema de rastreamento e atribuição de equipes revelou características importantes sobre sua aplicabilidade prática. O sistema de rastreamento baseado em ByteTrack demonstrou robustez satisfatória na maioria dos cenários, mantendo continuidade de IDs mesmo em situações de sobreposição parcial entre jogadores. A Figura 24 exemplifica um caso bem-sucedido onde o sistema mantém corretamente a distinção entre jogador e árbitro mesmo com sobreposição significativa das bounding boxes.

Figura 24 – Exemplo de detecção robusta mantendo distinção correta entre classes mesmo com sobreposição

Fonte: Elaboração própria.

Entretanto, o sistema apresenta limitações em cenários específicos. A Figura 25 ilustra uma falha onde um jogador é temporariamente classificado como árbitro, demonstrando que variações de postura ou oclusões podem confundir o classificador. Estas trocas de classe, indicam necessidade de técnicas de validação temporal para manter consistência de classificação.

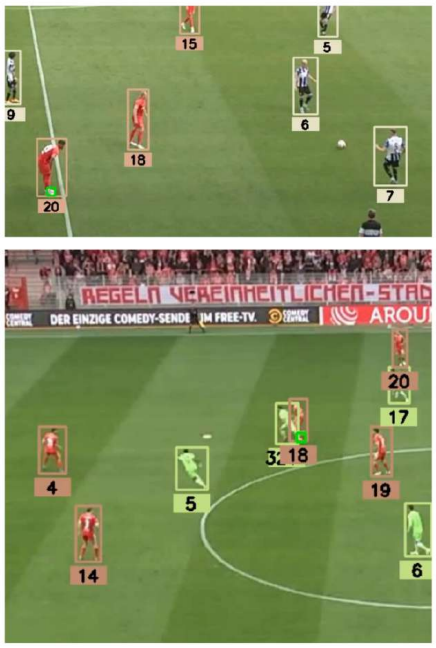
Figura 25 – Exemplo de falha temporária na classificação, com jogador sendo detectado incorretamente como árbitro



Fonte: Elaboração própria.

A detecção da bola permanece como o desafio mais significativo, mesmo com as melhorias substanciais obtidas. A Figura 26 apresenta dois cenários típicos de falha: no primeiro, os pés de jogadores são confundidos com a bola devido à similaridade visual na cor; Estas situações evidenciam que, apesar do progresso alcançado, técnicas complementares como informação temporal e contexto de jogo podem ser necessárias para alcançar robustez completa.

Figura 26 – Exemplos de falhas na detecção da bola: (a) confusão com a chuteira do jogador e (b) blur devido a movimento rápido



Fonte: Elaboração própria.

O sistema de atribuição automática de equipes demonstrou eficácia variável dependente das condições específicas de cada partida. Em situações com uniformes contrastantes, o algoritmo baseado em K-Means consegue separação clara e estável das equipes. Contudo, uniformes com cores similares ou variações significativas de iluminação ao longo do campo podem comprometer a precisão da classificação. A decisão de não rastrear goleiros inicialmente simplificou o desenvolvimento mas representa uma limitação para análises táticas completas, requerendo extensão futura do algoritmo para incorporar esta classe especial considerando suas características únicas de uniforme e comportamento.

4.7 Implicações Práticas e Recomendações

Os resultados obtidos fornecem diretrizes claras para implementação prática do sistema em diferentes cenários. Para análises offline detalhadas, onde o tempo de processamento não é crítico, recomenda-se utilizar a configuração YOLOv8n-1920, que oferece a melhor precisão geral com mAP@0.5 de 0,940. Esta configuração é ideal para análise pós-jogo, geração de estatísticas detalhadas e estudos táticos aprofundados.

Para aplicações em tempo real, como transmissões ao vivo ou análise durante o jogo, o YOLOv8n-640 oferece o melhor compromisso entre velocidade e precisão. Com tempo de inferência de apenas 0,3ms por frame, esta configuração permite processamento a 30fps mesmo em hardware modesto. Embora a detecção da bola seja limitada, técnicas de pós-processamento como interpolação temporal podem compensar parcialmente esta deficiência.

O desenvolvimento de técnicas de pós-processamento especializadas emerge como necessidade clara. A implementação de validação temporal para manter consistência de classificação, interpolação inteligente para trajetórias da bola, e fusão de informações contextuais (como posicionamento relativo e padrões de movimento) podem significativamente melhorar a robustez do sistema sem aumentar substancialmente o custo computacional.

5 Conclusões e Trabalhos Futuros

Este trabalho apresentou um sistema de detecção e rastreamento para análise de vídeos de futebol usando modelos YOLO. Os experimentos realizados mostraram que a escolha da resolução de entrada tem impacto maior que a complexidade do modelo na detecção de objetos pequenos como a bola.

5.1 Principais Contribuições

O trabalho teve três contribuições principais. Primeiro, a comparação entre diferentes versões do YOLO mostrou que o YOLOv8n com resolução de 1920px detecta a bola melhor que o YOLOv8l (que tem 14 vezes mais parâmetros) em resoluções menores. Isso contradiz a ideia comum de que modelos maiores sempre performam melhor.

Segundo, foi criado um dataset com 4.116 imagens anotadas contendo 87.684 objetos marcados. O dataset está disponível publicamente para outros pesquisadores usarem em seus trabalhos.

Terceiro, o sistema desenvolvido integra detecção, rastreamento, correção de trajetórias e identificação de times em um pipeline único. O código também foi disponibilizado no GitHub.

5.2 Limitações e Desafios

A detecção da bola ainda falha em cerca de 30% dos casos difíceis. Os principais problemas acontecem quando a bola está parcialmente escondida por jogadores, quando o movimento rápido causa blur na imagem, ou quando elementos do campo como linhas e marcações confundem o detector. Essas situações são comuns durante o jogo e representam um desafio técnico considerável que ainda precisa ser resolvido.

O rastreamento usando ByteTrack apresenta problemas em lances de bola parada, onde muitos jogadores ficam próximos. Nesses casos, o sistema às vezes troca os IDs dos jogadores ou perde o rastreamento completamente. A alta densidade de objetos em áreas pequenas dificulta a associação correta entre detecções consecutivas, especialmente em escanteios e faltas próximas à área.

A identificação de times por cor funciona bem quando os uniformes são visualmente distintos, mas encontra dificuldades em situações específicas. Cores similares como azul escuro e preto são frequentemente confundidas pelo algoritmo. Além disso, mudanças na iluminação durante o jogo afetam a percepção das cores, e quando um time troca de

uniforme no intervalo, o sistema não consegue se adaptar automaticamente. Os goleiros não foram incluídos no rastreamento inicial, o que limita análises táticas que precisam considerar todos os jogadores em campo.

O uso de alta resolução melhora significativamente a detecção mas aumenta em 4 vezes o tempo de processamento. Isso pode ser um problema para equipes que não têm acesso a GPUs potentes ou que precisam processar vídeos em tempo real durante as partidas. Esse trade-off entre precisão e velocidade é uma questão central para a aplicação prática do sistema.

5.3 Trabalhos Futuros

Com o sistema base de detecção e rastreamento funcionando, o próximo passo natural é desenvolver métricas e análises que gerem valor real para treinadores e analistas. A capacidade de identificar jogadores individuais e suas equipes abre possibilidades para extrair estatísticas que antes exigiam análise manual demorada.

A implementação de detecção de passes seria uma das features mais valiosas. Combinando a posição da bola com a identificação dos jogadores, é possível determinar quando a posse muda e registrar automaticamente origem e destino de cada passe. Isso permitiria gerar estatísticas como precisão de passe por jogador, mapas de conexões entre jogadores mostrando quem passa mais para quem, e identificação de padrões de construção de jogadas de cada time. A análise temporal desses dados revelaria como o estilo de jogo muda durante a partida.

Mapas de calor individuais e coletivos são outra aplicação direta do rastreamento implementado. Acumulando as posições de cada jogador ao longo do jogo, seria possível visualizar zonas de maior atuação, identificar se um lateral está subindo muito ou se um meia está cobrindo bem o campo. Para o time completo, o mapa de calor mostraria o posicionamento médio da equipe, ajudando a entender se o time joga mais centralizado ou pelas pontas, se mantém linha alta ou baixa.

A análise de distância percorrida e velocidade de cada jogador forneceria dados sobre condicionamento físico e intensidade. Identificar momentos de sprint versus caminhada ajudaria a entender o perfil físico de cada atleta. Cruzando esses dados com o momento do jogo, seria possível detectar queda de rendimento físico no segundo tempo ou em momentos específicos da partida.

Para incluir os goleiros nessas análises, seria necessário adaptar o algoritmo de identificação de times para reconhecer que o jogador mais recuado de cada lado, mesmo com uniforme diferente, pertence àquela equipe. Com o goleiro rastreado, novas métricas surgem: distância média da linha do gol (indicando se joga adiantado), velocidade de saída

em contra-ataques, e participação na construção de jogadas com os pés.

A detecção de eventos táticos específicos agregaria ainda mais valor. Identificar quando um time está pressionando a saída de bola adversária analisando a proximidade e velocidade dos jogadores em direção ao portador da bola. Detectar linhas de impedimento automáticas verificando o alinhamento dos defensores. Reconhecer transições defesa-ataque medindo quanto tempo leva desde recuperar a bola até chegar ao campo de ataque.

Todas essas métricas poderiam ser exportadas em relatórios automáticos pós-jogo, com gráficos e visualizações que facilitam a interpretação. Um dashboard interativo permitiria filtrar análises por período do jogo, por jogador específico, ou por situação de jogo (apenas contra-ataques, apenas bolas paradas, etc). A comparação entre jogos criaria uma base de dados valiosa para acompanhar evolução de desempenho ao longo da temporada.

O desenvolvimento dessas features transformaria o sistema de uma ferramenta de visão computacional em uma plataforma completa de análise tática. Times menores teriam acesso a estatísticas profissionais usando apenas vídeos de suas partidas, democratizando a análise de desempenho no futebol.

5.4 Considerações Finais

O trabalho mostrou que é possível criar um sistema funcional de análise de futebol usando técnicas de visão computacional. O resultado mais interessante foi descobrir que aumentar a resolução de 640px para 1920px melhorou a detecção da bola em 191%, enquanto usar um modelo 14 vezes maior teve impacto menor. Essa descoberta sugere que, para problemas específicos como detecção de objetos pequenos, otimizações no pré-processamento podem ser mais efetivas que aumentar a complexidade do modelo.

O dataset criado e o código disponibilizado representam recursos valiosos para a comunidade de pesquisa. Times com menos recursos financeiros poderiam usar sistemas como este para fazer análises táticas básicas sem precisar de equipamentos especializados caros. A possibilidade de extrair informações táticas de vídeos comuns pode democratizar o acesso a ferramentas de análise que antes eram exclusivas de grandes clubes.

Ainda existem problemas importantes a resolver, principalmente na detecção da bola em situações complexas e no rastreamento em lances com alta densidade de jogadores. No entanto, os resultados obtidos mostram que é viável desenvolver ferramentas de análise acessíveis para diferentes níveis do futebol. Com o avanço contínuo das técnicas de deep learning e o barateamento de hardware, sistemas de análise em tempo real devem se tornar realidade para todos nos próximos anos. Este trabalho contribui com uma peça importante desse desenvolvimento, estabelecendo bases técnicas e identificando os principais desafios

a serem superados.

ANEXO A – código fonte

https://github.com/AZaneratto/soccer-player-track

Referências

1 ZHENG F., A.-H. A review of computer vision technology for football videos. *Information*, MDPI, v. 16, n. 5, p. 355, 2025. Disponível em: https://www.mdpi.com/2078-2489/16/5/355. Citado na página 1.

2 COMPUTER vision in sports: applications and challenges. *SuperAnnotate Blog*, March 2023. Disponível em: https://www.superannotate.com/blog/computer-vision-in-sports. Citado na página 1.

3 QUINN, E.; CORCORAN, N. The comparative analysis of yolov5/v8/v9 for object detection, tracking, and human action recognition in combat sports. In: *International Conference on AI Research*. [S.l.: s.n.], 2024. v. 4, n. 1. Proceedings of the International Conference on AI Research, ICAIR 2024. Citado 2 vezes nas páginas 1 e 2.

4 SKILLCORNER. *XY Tracking Data - Fully automated scalable tracking data for all 22 players*. 2025. Disponível em: https://skillcorner.com/tracking-data. Citado na página 2.

5 JOCHER, G.; CHAURASIA, A.; QIU, J. *Ultralytics YOLOv8*. [S.l.]: GitHub, 2025. <https://github.com/autogyro/yolo-V8>. Citado 3 vezes nas páginas 2, 10 e 18.

6 ANDREWS, P.; BORCH, N.; FJELD, M. Footyvision: Multi-object tracking, localisation, and augmentation of players and ball in football video. In: *Proceedings of the 9th International Conference on Multimedia and Image Processing*. [S.l.: s.n.], 2024. p. 1–11. Citado na página 2.

7 LEI, R. et al. An improved yolov8 target detection algorithm and its application in tennis ball picking robot. In: *Advances in Neural Networks – ISNN 2024*. [S.l.]: Springer, 2024. Disponível em: https://dl.acm.org/doi/10.1007/978-981-97-4399-5_18. Citado na página 2.

8 ZHANG, Y. et al. ByteTrack: Multi-object tracking by associating every detection box. In: *European Conference on Computer Vision (ECCV)*. [S.l.]: Springer, 2022. p. 1–21. Citado 4 vezes nas páginas 2, 18, 20 e 22.

9 ZHENG F., A.-H. D. Z. C. P. H. J. Y. C. . L. X. J. Multi-object vehicle detection and tracking algorithm based on improved yolov8 and bytetrack. *Electronics*, MDPI, v. 13, n. 15, p. 3033, 2024. Citado na página 2.

10 LIU J., X.-Y. Z. Y. . L. H. Vehicle flow detection and tracking based on an improved yolov8n and bytetrack framework. *World Electric Vehicle Journal*, MDPI, v. 16, n. 1, p. 13, 2024. Citado na página 2.

11 CIOPPA, A. et al. Soccernet-tracking: Multiple object tracking dataset and benchmark in soccer videos. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2022. p. 3491–3502. Citado 2 vezes nas páginas 2 e 14.

12 CIOPPA, A. et al. Scaling up soccernet with multi-view spatial localization and re-identification. *Scientific Data*, Nature Publishing Group, v. 9, n. 1, p. 355, 2022. Citado na página 2.

13 SHAO, Z. H. Y. Var-yolov8s: Iot-based automatic foul detection in soccer matches. *Alexandria Engineering Journal*, Elsevier, October 2024. Citado na página 3.

14 BLOGGER, T. T. Sports analytics — a data-centric approach to computer vision. *Medium*, January 2024. Disponível em: https://medium.com/@tenyks_blogger/sports-analytics-a-data-centric-approach-to-computer-vision-246f19cf8a04. Citado na página 3.

15 ZHANG, A. et al. *Dive into deep learning*. [S.l.]: arXiv preprint arXiv:2106.11342, 2021. Citado 4 vezes nas páginas 5, 7, 8 e 19.

16 ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, n. 6, p. 386–408, 1958. Citado na página 5.

17 GéRON, A. *Mãos à Obra: Aprendizado de Máquina com Scikit-Learn, Keras e TensorFlow*. 2. ed. Rio de Janeiro: Alta Books, 2019. ISBN 978-85-508-0725-0. Citado 4 vezes nas páginas 6, 11, 18 e 31.

18 REDMON, J. et al. You only look once: Unified, real-time object detection. In: *CVPR*. [S.l.: s.n.], 2016. p. 779–788. Citado na página 10.

19 REDMON, J.; FARHADI, A. Yolo9000: better, faster, stronger. In: *CVPR*. [S.l.: s.n.], 2017. p. 7263–7271. Citado na página 10.

20 REDMON, J.; FARHADI, A. *YOLOv3: An Incremental Improvement*. 2018. Disponível em: <https://arxiv.org/abs/1804.02767>. Citado na página 10.

21 MICHALCZYK, J. et al. *DFL - Bundesliga Data Shootout*. 2022. Kaggle. <https://kaggle.com/competitions/dfl-bundesliga-data-shootout>. Citado na página 14.

22 FFmpeg Developers. *FFmpeg: A complete, cross-platform solution to record, convert and stream audio and video*. 2021. <https://ffmpeg.org/>. Version 4.4. Citado na página 15.

23 Roboflow Inc. *Roboflow: Computer Vision Tools for Developers and Enterprises*. 2024. <https://roboflow.com>. Citado na página 15.

24 alexzaneratto. *football-player-detect Dataset*. [S.l.]: Roboflow, 2024. <https://universe.roboflow.com/alexzaneratto/football-player-detect-zjoxy>. Citado na página 16.

25 ROBOFLOW. *Train, Validation, Test Split for Machine Learning*. 2024. Accessed: 2025-01-07. Disponível em: <https://blog.roboflow.com/train-test-split/>. Citado na página 17.

26 OpenCV Team. *OpenCV: Open Source Computer Vision Library*. [S.l.]: GitHub, 2024. <https://github.com/opencv/opencv>. Citado na página 18.

27 ZHAO, Z.-Q. et al. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, v. 30, n. 11, p. 3212–3232, 2019. Citado na página 18.

28 EVERINGHAM, M. et al. The pascal visual object classes (voc) challenge. In: *International journal of computer vision*. [S.l.]: Springer, 2010. v. 88, n. 2, p. 303–338. Citado 2 vezes nas páginas 18 e 20.

29 PADILLA, R. et al. A comparative analysis of object detection metrics with a companion open-source toolkit. *Electronics*, MDPI, v. 10, n. 3, p. 279, 2021. Citado na página 19.

30 KUMAR, R.; SINGH, A.; SHARMA, P. Enhancing the performance and accuracy in real-time football and player detection using upgraded yolov5 architecture. In: *International Journal of Computational Intelligence Systems*. [S.l.: s.n.], 2024. v. 17, n. 1, p. 1–15. Citado na página 20.