UNIVERSIDADE FEDERAL DO PARÁ
CAMPUS UNIVERSITÁRIO DE TUCURUÍ
FACULDADE DE ENGENHARIA DE COMPUTAÇÃO

LEONARDO CABRAL DA COSTA

MONITORAMENTO INTELIGENTE DE FADIGA: DETECÇÃO DE SONOLÊNCIA EM MOTORISTAS COM IA E VISÃO COMPUTACIONAL PARA AUMENTAR A SEGURANÇA NO TRÂNSITO

Tucuruí-Pará
2024

LEONARDO CABRAL DA COSTA

MONITORAMENTO INTELIGENTE DE FADIGA: DETECÇÃO DE SONOLÊNCIA EM MOTORISTAS COM IA E VISÃO COMPUTACIONAL PARA AUMENTAR A SEGURANÇA NO TRÂNSITO

Trabalho de Conclusão de Curso apresentado à faculdade de Engenharia de Computação do Campus Universitário de Tucuruí, da Universidade Federal do Pará, como requisito parcial para a obtenção do título de Bacharel em Engenharia de Computação.

Orientador(a): Prof. Dr. Daniel Pinheiro

Tucuruí-Pará

2024

Dados Internacionais de Catalogação na Publicação (CIP) de acordo com ISBD
Sistema de Bibliotecas da Universidade Federal do Pará
Gerada automaticamente pelo módulo Ficat, mediante os dados fornecidos pelo(a) autor(a)

C837m Costa, Leonardo Cabral da.
Monitoramento Inteligente de Fadiga: Detecção de Sonolência em Motoristas com IA e Visão Computacional para Aumentar a Segurança no Trânsito / Leonardo Cabral da Costa. — 2024.
77 f. : il. color.

Orientador(a): Prof. Dr. Daniel da Conceição Pinheiro
Trabalho de Conclusão (Graduação) - Universidade Federal do Pará, Campus Universitário de Tucuruí, Faculdade de Engenharia da Computação, Tucuruí, 2024.

1. Visão Computacional. 2. Fadiga. 3. Yolo. I. Título.

CDD 006.37

UNIVERSIDADE FEDERAL DO PARÁ
CAMPUS UNIVERSITÁRIO DE TUCURUÍ
FACULDADE DE ENGENHARIA DE COMPUTAÇÃO

TÍTULO: MONITORAMENTO INTELIGENTE DE FADIGA: DETECÇÃO DE SONOLÊNCIA EM MOTORISTAS COM IA E VISÃO COMPUTACIONAL PARA AUMENTAR A SEGURANÇA NO TRÂNSITO

DISCENTE: LEONARDO CABRAL DA COSTA
MATRÍCULA: 201733840007

#	BANCA EXAMINADORA	CONDIÇÃO
1	*Prof. Dr. Daniel da Conceição Pinheiro*	*Orientador*
2	*Prof. Eng. Vigner Vieira dos Santos*	*Membro interno*
3	*Eng. Erick Vinícius Damaceno da Silva*	*Membro externo*

Data da Defesa: 28/11/2025	**Hora Início:** 10:04	**Hora Término:** 10:27

Trabalho Escrito (0 a 10 pontos por critério)	**Examinador 1**	**Examinador 2**	**Examinador 3**
Formatação	8,00	10,00	10
Linguagem (gramática e semântica)	8,00	9,50	9,50
Conteúdo técnico	8,00	10,00	10

Defesa Oral (0 a 10 pontos por critério)	**Examinador 1**	**Examinador 2**	**Examinador 3**
Sequência lógica de apresentação	9,00	10,00	9,5
Administração do tempo	9,00	10,00	10
Expressão oral	8,00	10,00	9,5
Domínio do tema	9,00	10,00	9,50

Média por examinador	8,43	9,93	9,71
Média Final			**9,36**
Conceito Final			EXC

gov.br Documento assinado digitalmente
DANIEL DA CONCEICAO PINHEIRO
Data: 28/11/2024 17:13:45-0300
Verifique em https://validar.iti.gov.br

Tucuruí-Pa, 28 de novembro de 2024.

gov.br Documento assinado digitalmente
VIGNER VIEIRA DOS SANTOS
Data: 28/11/2024 17:04:46-0300
Verifique em https://validar.iti.gov.br

Membro interno

gov.br Documento assinado digitalmente
ERICK VINICIUS DAMASCENO DA SILVA
Data: 28/11/2024 17:11:53-0300
Verifique em https://validar.iti.gov.br

Membro externo

Este trabalho e dedicado à minha mãe Aurora, por todo esforço e dedicação.

AGRADECIMENTOS

Quero agradecer a minha mãe Aurora, e minha irmã Caroline, pelo apoio em todos esses anos para chegar até a conclusão dessa etapa da minha vida.

Quero agradecer a minha namorada Steffany pela paciência durante a realização deste trabalho, também pelo apoio em vários momentos durante essa trajetória.

Quero agradecer aos meus colegas de faculdade de Engenharia de Computação da Universidade Federal do Pará da turma de 2017, que sempre pude contar do início ao fim deste curso.

RESUMO

A fadiga que provoca sinais de sonolência em motoristas é uma das principais causas de acidentes de trânsito. Esses sinais se manifestam em diferentes níveis de gravidade, e quanto maior o nível, maior o risco de acidentes. Este trabalho apresenta um método baseado em inteligência artificial, aprendizado de máquina, aprendizado profundo e visão computacional para desenvolver um sistema capaz de identificar sinais de sonolência e emitir alertas proporcionais ao nível de fadiga detectado. Com o uso da ferramenta YOLO (*You Only Look Once*), amplamente reconhecido por sua eficácia em detecção de objetos em tempo real, foi desenvolvido um modelo para reconhecer sinais de sonolência em motoristas. O processo de construção do modelo incluiu etapas essenciais, como a coleta de imagens e o treinamento do modelo. Após o treinamento, o modelo foi submetido a testes, que mostraram sua eficiência em detectar sinais de fadiga e seus resultados foram avaliados por meio de métricas estatísticas, verificando sua precisão na identificação dos diferentes níveis de fadiga. Com base nesses sinais, o sistema pode alertar o motorista em casos de fadiga acentuada, atuando como uma ferramenta preventiva para aumentar a segurança no trânsito. Assim, o sistema contribui para a redução de acidentes relacionados à sonolência.

Palavras chave: Fadiga, YOLO, Sonolência, Visão Computacional.

ABSTRACT

Fatigue that causes signs of drowsiness in drivers is one of the main causes of traffic accidents. These signs manifest themselves in different levels of severity, and the higher the level, the greater the risk of accidents. This work presents a method based on artificial intelligence, machine learning, deep learning and computer vision to develop a system capable of identifying signs of drowsiness and issuing alerts proportional to the level of fatigue detected. Using the YOLO (You Only Look Once) algorithm, widely recognized for its effectiveness in real-time object detection, a model was developed to recognize signs of drowsiness in drivers. The model construction process included essential steps, such as image collection and model training. After training, the model was subjected to tests, which showed its efficiency in detecting signs of fatigue and its results were evaluated through statistical metrics, verifying its accuracy in identifying the different levels of fatigue. Based on these signals, the system can alert the driver in cases of severe fatigue, acting as a preventive tool to increase traffic safety. Thus, the system contributes to the reduction of accidents related to drowsiness.

Keywords*: Fatigue, YOLO, Drowsiness, Computer Vision.*

LISTA DE FIGURAS

LISTA DE ALGORITMOS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

EUA	ESTADOS UNIDOS DA AMÉRICA
YOLO	YOU ONLY LOOK ONCE
EAR	EYE ASPECT RATIO
MAR	MOUTH ASPECT RATIO
ORC	OPTICAL RECOGNITION CHARACTER
IA	INTELIGÊNCIA ARTIFICIAL
RNAs	REDES NEURAIS ARTIFICIAIS
CNN	REDES NEURAIS CONVOLUCIONAL
NMS	NETWORK MANAGEMENT STATION
IoU	INTERSECTION OVER UNION
FLOPs	FLOATING POINT OPERATIONS PER SECOND
PAN	PATH AGGREGATION NETWORK
CSPNet	CROSS STAGE PARTIAL NETWORK
REM	RAPID EYE MOVIMENT
NREM	NON RAPID EYE MOVIMENT
HR	HEART RATE
ECG	ELETROCARDIOGRAMA
EEG	ELETROENCEFALOGRAMA
HRV	HEART RATE VARIABILITY

SUMÁRIO

1 INTRODUÇÃO

Acidentes de trânsito representam a segunda causa de morte não natural evitável no Brasil, levando o país a quarta posição entre os países com mais mortes em acidentes de trânsito no mundo [1]. Em 2018, o país atingiu a marca de 23,4 mortes por 100 mil habitantes, ou seja, 1 pessoa morta a cada 15 minutos. Além disso, 1 pessoa a cada 2 minutos sofre alguma sequela decorrente de acidente de trânsito [2].

Durante a jornada de trabalho, em alguns momentos, o indivíduo pode ter a sensação de estar cansado durante a realização de alguma tarefa específica, assim como se presencia uma perda de desempenho na realização destas tarefas. No entanto, estes estados mentais e físicos não estão muitas vezes presentes na consciência da própria pessoa ou acabam por ser ignorada, uma atitude que pode resultar em falha humana originando acidentes de trabalho, problemas de saúde ocupacional, ou uma diminuição na qualidade da produção [3].

O efeito da fadiga nos acidentes pode variar consideravelmente dependendo da gravidade e das circunstancias dos acidentes. De forma geral, as taxas de porcentagens variam entre 1% e 29%. Estudos na área de acidentes automobilísticos nos Estados Unidos da América – EUA indicam que motoristas sonolentos estão envolvidos em cerca de 100.000 acidentes, 1.357 vítimas fatais e 71.000 vítimas com lesões por ano [4] [5].

O cansaço e a fadiga são fatores que aumentam bastante os riscos de acidentes no trânsito, normalmente a maioria dos motoristas não percebem ou imprudentemente não se importam que se encontram fadigados. Com o avanço de sistemas inteligentes, a detecção automática de fadiga pode ser feita com a análise de características físicas da face de uma pessoa [6].

Diversos trabalhos foram publicados ao longo dos anos sobre a detecção de fadiga do motorista [7]. É um tema bem consolidado e já vem sendo documentado por diversos pesquisadores. Em 2001 foi publicado um artigo que fala sobre a correlação direta entre fadiga e acidentes [10]. Essa correlação não é algo novo para a maioria dos motoristas, porém muitos imprudentemente dirigem em condições de fadiga.

Para detecção do nível de fadiga de um motorista, demonstrou-se diversas técnicas de visão computacional e aprendizado de máquina [8]. [9] Demonstrou que é possível fazer esta tarefa apenas com técnicas de visão computacional, levando em consideração além dos olhos, regiões da boa e posicionamento da cabeça.

1.1 CONTEXTO

O sono compromete a direção em nível semelhante ao provocado pela ingestão de bebida alcoólica, segundo a Associação Brasileira de Medicina de Tráfego (ABRAMET), cerca de 20% dos acidentes de trânsito estão relacionados à sonolência. Essa é ainda uma das principais causas de mortes nas rodovias [11].

Com a evolução das tecnológicas vivenciada nos últimos anos possibilitou sofisticados métodos de detecção de sinais de sonolência. Um exemplo e os sistema implementado em carros da marca Tesla que detecta padrões de sonolência e alerta o condutor pra que reduza a velocidade [12].

1.2 OBJETIVO

1.2.1 Objetivo Geral

Este projeto tem como objetivo treinar um modelo de detecção automática de sinais de sonolência em tempo real, utilizando a arquitetura *You Only Look Once* (YOLO), para identificar e classificar sinais como fechamento dos olhos e bocejo. Além disso, desenvolver um sistema de alerta que utilizará os resultados do modelo para gerar alertas ao motorista, conforme o nível de sonolência detectado, com o intuito de prevenir situações de risco aos motoristas.

1.2.2 Objetivos Específicos

- Preparar um conjunto de dados;
- Treinar um modelo conforme a arquitetura YOLO;
- Analisar e otimizar o treinamento;
- Desenvolver um algoritmo de alerta automático;
- Realizar testes com o modelo em tempo real.

1.3 TRABALHOS CORRELATADOS

A seguir alguns dos trabalhos relacionados a este projeto disponíveis na literatura.

- **Detecção de Sonolência em Motoristas Utilizando Processamento de Imagens** [13]. Esse é um sistema não intrusivo baseado na utilização de um smartphone para a captura de imagens do motorista. Após captura, é realizado um pré-processamento da imagem onde é utilizado o método de interpolação pelo vizinho mais próximo.

- **Monitoramento do Estado de Sonolência de Motoristas de Automóveis Através de Análise de Imagens de Olhos** [8]. Neste trabalho foi desenvolvido uma metodologia que utiliza técnicas de processamento de imagens, visão computacional, aprendizado de máquina e características físicas para a detecção da região dos olhos e a análise de seu comportamento com o objetivo de verificar o nível de desatenção de motoristas de automotores.
- **Detecção de Fadiga Baseada no Monitoramento dos Olhos** [6]. Nesta monografia é apresentado um método para detecção de fadiga em motoristas com a utilização de técnicas de processamento de imagem e aprendizado de máquina.
- **Visão Computacional na Detecção de Fadiga em Operadores de Máquinas em Trabalho Noturno Modelo de Interface Homem Maquina** [14]. Esse trabalho discute o problema e propõe uma solução tecnológica que auxilia na detecção de sinais de fadiga, podendo intervir no funcionamento de maquinário a fim de prevenir acidentes de trabalho.
- **Sistema de Detecção de Sonolência** [15]. Este trabalho desenvolve um dispositivo de baixo custo que detecta sinais de sonolência por meio de técnicas de processamento digital de imagens, essas imagens são capturadas no interior do veículo e são processadas para a obtenção dos parâmetros EAR e MAR que analisam condições fisiológicas do motorista.
- **Sistema Embarcado para Detecção e Alerta de Atitudes de Desatenção ao Dirigir** [16]. O presente trabalho visa a criação de um dispositivo embarcado capaz de alertar o motorista em estado de cansaço e/ou sonolência através de alarmes sonoros e visuais.
- **Reconhecimento de Imagens: Uso do Método YOLO no Reconhecimento de Placas de Trânsito** [17]. Nessa pesquisa buscou-se reconhecer sinais de trânsito brasileiros, tendo como principal ferramenta o uso do método *You Only Look Once* (YOLO), na sua quinta versão. Ele usa de uma única rede neural convolucional que prevê simultaneamente várias caixas delimitadoras e probabilidades de classe para essas caixas, o YOLO treina em imagens completas e otimiza diretamente o desempenho de detecção.
- ***Fatigue Driving Detection Methods Based on Drivers Wearing Sunglasses*** [18]. Este artigo apresenta uma abordagem inovadora que integra YOLOv8-N com aprendizagem por transferência para desenvolver um sistema preciso de detecção de

fadiga feito sob medida para motoristas que usam óculos de sol. Utilizando câmeras infravermelhas integradas, vídeos desses motoristas foram gravados e características faciais essenciais foram extraídas para construir um conjunto de dados especializado.

Os trabalhos relacionados a esta pesquisa abrangem diversos aspectos relevantes para o desenvolvimento do projeto, incluindo o estudo da arquitetura YOLO e sua aplicação em diferentes contextos. Entre os temas abordados, destacam-se a análise de sinais de fadiga em indivíduos, com ênfase na identificação de características como o fechamento dos olhos e o bocejo, além de aplicações práticas como o processamento de imagens para reconhecimento de padrões em olhos e o reconhecimento automatizado de placas de trânsito. Esses estudos oferecem uma base sólida para compreender as capacidades da arquitetura YOLO e seu potencial para aprimorar sistemas de detecção em tempo real, especialmente em cenários caracterizados por situações de fadiga no trânsito.

1.4 JUSTIFICATIVA

De acordo com o que foi mencionado em tópicos anteriores, dormir ao volante está sem dúvida entre as maiores causas de acidentes no trânsito. Fica evidente a necessidade de se monitorar os sinais de sonolência que pode ser apresentado por motorista fadigados.

Diante deste cenário se faz necessário o desenvolvimento de um sistema de prevenção de acidentes provocados por fadiga no trânsito. Capaz de detectar e alerta o motorista que se encontra em um estado de fadiga fora do padrão.

1.5 ORGANIZAÇÃO DO TRABALHO

Este trabalho está organizado em 5 capítulos, o 1º capitulo mostra a introdução sobre os problemas de se dirigir com sono e suas consequências. Também e mostrado o contexto, objetivos gerais e específicos, trabalhos correlatados e a justificativa. No 2º capitulo, apresenta a fundamentação teórica como os conceitos gerais de visão computacional, inteligência artificial, arquitetura YOLO e mostra uma pesquisa mais aprofundada sobre sinais de fadiga e acidentes causados por sonolência. No 3º capitulo, evidencia-se a metodologia do trabalho com a ilustração das fases do desenvolvimento da pesquisa, desde a criação da base de dados, escolha do modelo, até o desenvolvimento do algoritmo. No 4º capitulo, realiza-se a apresentação dos resultados obtidos com os treinamentos do modelo escolhido. Juntamente com os resultados oriundos dos testes em tempo real do algoritmo. No 5º capitulo, aprecia-se as considerações finais sobre o trabalho e os futuros trabalhos.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capitulo será feito o levantamento sistemático de todas as fontes de artigos científicos, para que possa elucidar um analise critica das informações buscadas por este trabalho. Em busca de um panorama geral sobre as principais teorias e resultados encontrados nas pesquisas anteriores.

2.1 VISÃO COMPUTACIONAL

Visão computacional é a ciência que caracteriza informações de interesse numa imagem, a qual é estruturada na definição do problema visual a ser analisado e solucionado, organizado nas etapas sequenciais da aplicação: aquisição, pré-processamento, segmentação, extração de características e reconhecimento. Trata-se da integração de processos e representações gráficas usadas para a percepção visual [19].

Nas percepções humanas encontra-se a estrutura tridimensional do mundo que nos rodeia com aparente facilidade. Pensa-se em como vívida a percepção tridimensional é quando você olha para um vaso de flores na mesa ao lado. Pode-se dizer a forma e translucidez de cada pétala através os padrões sutis de luz e sombras que jogam através de seu segmento de superfície e sem esforço cada flor do fundo da cena. Olhando para um retrato de grupo enquadrado, pode-se facilmente contar todas as pessoas em uma foto e até mesmo adivinhar as emoções de sua aparência facial [20].

> Visão computacional é a transformação de dados a partir de uma câmera fotográfica ou de vídeo. Todas essas transformações são feitas para alcançar algum objetivo particular. Os dados de entrada podem incluir algumas informações contextuais tais como se a câmara estiver montada em um carro ou uma faixa do laser que indica um objeto de 1 metro de distância. A decisão pode ser "há uma pessoa nesta cena" ou "há 14 células tumorais neste slide". A nova representação pode significar transformar uma imagem colorida em uma imagem em tons de cinza ou remover movimento da câmera a partir de uma sequência de imagens [21].

A área de Visão Computacional [22] é compreendida como um conjunto de técnicas usadas para processamento, aquisição, na análise e entendimento de dados relativamente complexos e com alto poder de dimensionalidade, as quais são extraídas do ambiente real para exploração técnica e científica. A meta das visões computacionais é modelar e automatizar o processo de reconhecimento visual [23].

2.1.1 Aplicações com Visão Computacional

Em ambientes computacionais [20], o homem tenta recriar o universo a partir da descrição do mundo que se pode ver em uma ou mais imagens e a reconstruir as suas propriedades, tais como a forma, iluminação, cores e distribuições. É surpreendente que os seres humanos e os animais fazer isso com tamanha facilidade, enquanto algoritmos de visão computacional são tão passivos a erros.

Hoje em uma ampla variedade de aplicações no mundo real já se utilizam de recursos da visão computacional, tais como [20]:

- Reconhecimento óptico de caracteres (ORC);
- Inspeção de partes de máquinas;
- Reconhecimento de objetos para checagem automatizada;
- Modelo 3D totalmente automatizado de fotografias aéreas utilizadas em sistemas como Bing Maps;
- Imagens médicas em registro pré-operatório e imagens intraoperatória ou realizar estudas de longo prazo de pessoas e morfologia do cérebro;
- Segurança automotiva em sistemas de detecção fadiga e auxílio de estacionamento [20].

2.2 INTELIGÊNCIA ARTIFICIAL

A Inteligência Artificial (IA) pode ser definida como o ramo da ciência da computação que se empenha em estudos da automação do desempenho inteligente, tendo como principal interesse realizar de forma efetiva o entendimento e a aplicação inteligente de técnicas para solucionar problemas, para planejamento e a comunicação com inúmeros problemas práticos [24]. Apesar da variedade de problemas tratados pela inteligência artificial, o autor traz oito características importantes que, segundo ele, parecem ser comuns a todas as divisões da área, são elas:

1. O uso do computador para executar raciocínio, reconhecimento de padrões, aprendizado ou outras formas de inferência;

2. Um foco em problemas que não respondem a soluções algorítmicas. Isso implica a utilização de busca heurística como uma técnica de IA para a solução de problemas;

3. Um interesse na solução de problemas utilizando informação inexata, faltante ou insuficientemente definida, e o uso de formalismos representacionais que possibilitem ao programador compensar esses problemas;

4. Raciocínio que utiliza as características qualitativas significativas de uma situação;

5. Uma tentativa de tratar de questões que envolvem tanto significado semântico como forma sintática;

6. Respostas que não são nem exatas, nem ótimas, mas que são “suficientes” em um certo sentido. Isso é resultado de se utilizar essencialmente métodos de solução de problemas baseados em heurísticas em situações em que resultados ótimos, ou exatos, são caros demais ou mesmo impossíveis;

7. O uso de grandes quantidades de conhecimento específico de um domínio para resolver problemas. Essa é a base dos sistemas especialistas;

8. O uso de meta conhecimento para produzir um controle mais sofisticado sobre as estratégias de resolução de problemas. Embora esse seja um problema muito difícil, tratado por relativamente poucos sistemas, ele vem emergindo como uma área essencial de pesquisa.

2.2.1 *Machine Learning*

O aprendizado de máquina usa a teoria da estatística na construção de modelos matemáticos, porque a tarefa principal é fazer inferências a partir de uma amostra. O papel da ciência da computação é duplo: primeiro, no treinamento, é preciso possuir *algoritmos* eficientes para resolver o problema de otimização, bem como para armazenar e processar a grande quantidade de dados que geralmente se possui. Em segundo lugar, uma vez que um modelo é aprendido, sua representação e solução algorítmica para inferência também precisam ser eficientes. Um sistema com raízes em algoritmos de aprendizagem tem capacidade de melhorar resultados à medida que vai sendo realizada a exposição de dados, uma vez que consegue acumular experiências a partir deles – algo parecido ao processo de aprendizagem humana [25].

2.2.2 *Deep Learning*

O aprendizado profundo surgiu a partir de pesquisas em inteligência artificial e aprendizado de máquina. O campo moderno do aprendizado de máquina baseia-se nos dois últimos tópicos: computadores que podem aprender com exemplos e pesquisa de redes neurais. O aprendizado de máquina envolve o desenvolvimento e a avaliação de *algoritmos*

que permitem a um computador extrair (ou aprender) funções de um conjunto de dados (conjuntos de exemplos). Para entender o que significa aprendizado de máquina, precisamos entender três termos: conjunto de dados, algoritmo e função. Em sua forma mais simples, um conjunto de dados é uma tabela em que cada linha contém a descrição de um exemplo de um domínio e cada coluna contém as informações de um dos recursos em um domínio.

Um algoritmo é um processo (ou receita, ou fluxo de instruções) que um computador pode seguir. No contexto do aprendizado de máquina, um algoritmo define um processo para analisar um conjunto de dados e identificar padrões recorrentes nos dados. Uma função é um mapeamento determinístico de um conjunto de valores de entrada para um ou mais valores de saída. O fato de o mapeamento ser determinístico significa que, para qualquer conjunto específico de entradas, uma função sempre retorna as mesmas saídas [21].

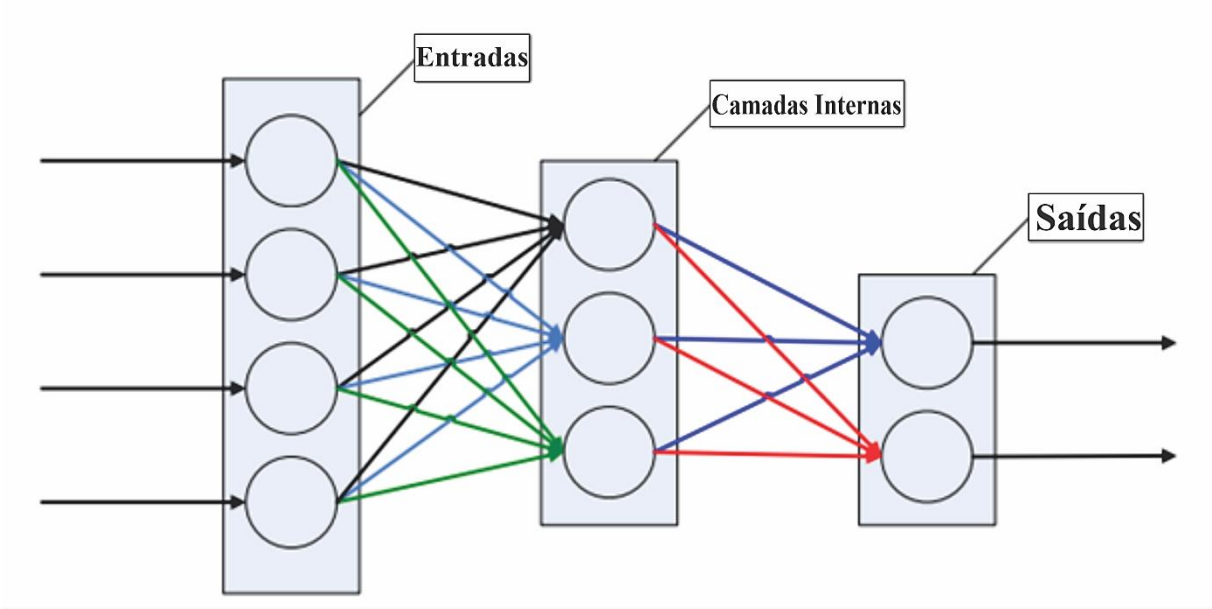
2.2.3 Redes Neurais

Redes Neurais Artificiais – RNAs [26] ou apenas redes neurais, compostos por simples unidades de processamento, denominadas nodos, cujo objetivo é calcular funções matemáticas. Essas unidades de processamento estão distribuídas em uma ou mais camadas interligadas por diversas conexões. Cada conexão conta com um peso, que serve para armazenar o conhecimento representado pelo modelo e encaminhar cada entrada para um neurônio da rede como na Figura 1.

Cada um dos neurônios possuem uma função de ativação, que define se o neurônio vai propagar ou inibir o sinal. Durante o processo de treinamento são atribuídos pesos para cada conexão, que servirão como parte do processo de decisão, de quais neurônios serão ativados na próxima camada, simulando as sinapses de um sistema nervoso animal. A função de ativação define a intensidade da ativação, que neste caso possui diversos tipos, onde a escolha desta determina como devem ser as saídas de cada neurônio.

Para a solução de problemas com uso de RNAs, eles passam inicialmente por um processo de aprendizagem, onde é basicamente apresentado uma série de exemplos para a rede, fazendo com que consiga extrair de forma automática características necessárias para poder representar a informação fornecida. Estas características anteriormente extraídas, são as usadas para gerar a solução para o problema. Com a capacidade natural de armazenar o conhecimento experimental e torná-lo disponível para uso. Assemelhando-se ao cérebro, que o conhecimento adquirido pela rede é através de um processo de aprendizagem e que as forças de conexão entre neurônios são usadas para armazenar o conhecimento adquirido [27].

Figura 1 – Rede Neural Típica



Fonte: LEAL (2006)

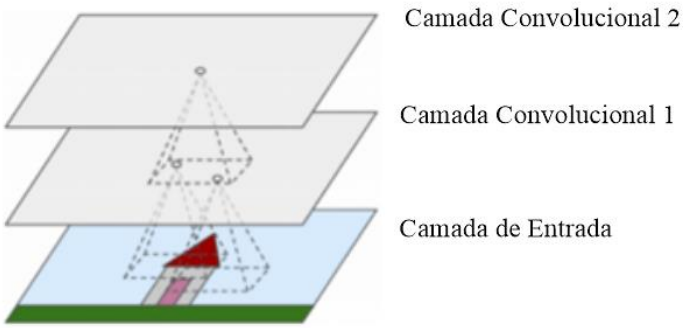
2.2.4 Redes Neurais Convolucionais

Nos últimos anos, graças ao aumento do poder computacional, as *Convolutional Neural Networks* – CNNs conseguiram atingir um desempenho sobre-humano em algumas tarefas visuais complexas. Potencializam os serviços de pesquisa de imagens, carros autônomos, sistemas automáticos de classificação de vídeo, entre outros. As CNNs não se restringem à percepção visual, elas também são bem sucedidas em outras tarefas, como reconhecimento de voz ou processamento de linguagem natural.

Os campos receptivos de diferentes neurônios podem se sobrepor e, juntos, eles revestem todo o campo visual. O bloco de construção mais importante de uma CNN é a camada convolucional, os neurônios na primeira camada de convolução não estão conectados a cada pixel na imagem de entrada, mas apenas aos *pixels* em seus campos receptivos, observado na Figura 2.

Por sua vez, cada neurônio na segunda camada convolucional está conectado apenas a neurônios localizados dentro de um pequeno retângulo na primeira camada. Essa arquitetura permite que a rede se concentre em pequenos recursos de nível inferior na primeira camada oculta, depois os montes em recursos maiores de nível superior na próxima camada ocultam e sucessivamente. Essa hierarquia é comum em imagens reais, um motivo das CNNs funcionarem bem para reconhecimento de imagens [29].

Figura 2 – Rede Neural Convolucional

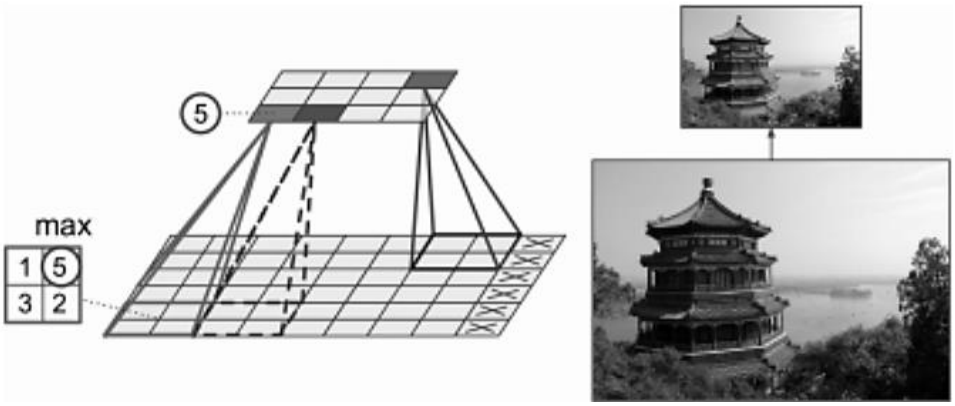


Fonte: Géron (2019)

O *pooling* tem o objetivo de diminuir a imagem de entrada para reduzir a carga computacional, o uso de memória e o número de parâmetros. Assim como nas camadas convolucionais, cada neurônio na camada *pooling* é conectado às saídas de um número limitado de neurônios na camada anterior, localizados dentro de um pequeno campo receptivo retangular. Um neurônio de *pooling* não tem pesos, ele apenas une as entradas, a camada de *pooling* funciona de forma independente em todos os canais de entrada, portanto, a profundidade de saída é a mesma de entrada [29].

O *pooling* ajuda a fazer com que a representação se torne aproximadamente invariável a pequenas traduções da entrada. A invariância para a tradução significa que, se a entrada for traduzida em uma pequena quantidade, os valores da maioria das saídas combinadas não mudam. A invariância para a tradução local pode ser uma propriedade muito útil se nos preocupamos mais se algum recurso está presente ou exatamente onde ele está. O uso de *pooling* pode ser visto como uma adição infinitamente forte de que a função que a camada aprende deve ser invariável para pequenas traduções. Quando essa suposição está correta, ela pode melhorar muito a eficiência estatística da rede.

Figura 3 – Camada *MaxPooling* (*kernel pooling* 2×2, *stride* de 2)



Fonte: Géron (2019)

A Figura 3 mostra que a operação de *MaxPooling* retira o maior elemento de determinada região da matriz, no exemplo é utilizado um filtro de 2x2, um passo de 2 denominado *stride*. Tendo assim que o valor máximo de cada *kernel* chegue para a próxima camada, reduzido em 75% o tamanho da imagem, já que para cada 4 *pixels* é extraído apenas 1 (o de maior valor devido ser *MaxPooling*) para a próxima camada.

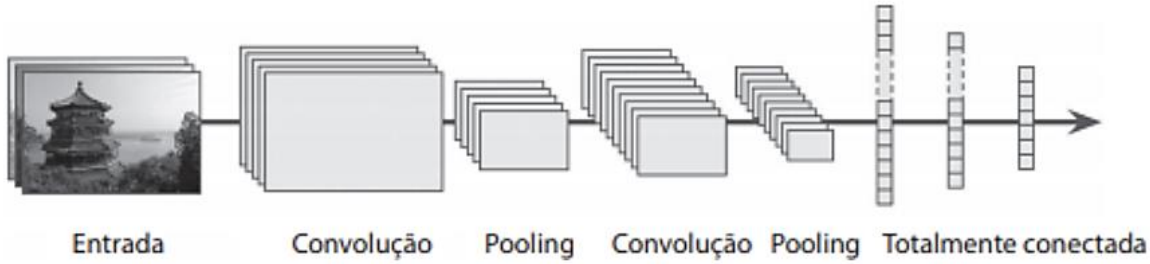
O *pooling* em regiões espaciais produz invariância para a translação, mas se agruparmos as saídas de convoluções parametrizadas separadamente, os recursos podem aprender a quais transformações se tornar invariáveis. Assim, como o *pooling* resume as respostas em toda uma vizinhança, é possível usar menos unidades de *pooling* do que unidades de detector, relatando estatísticas resumidas para regiões de *pool* espaçadas k *pixels* em vez de 1 *pixel*. Isso melhora a eficiência computacional da rede porque a próxima camada tem cerca de k vezes menos entradas para processar.

Quando o número de parâmetros na próxima camada é uma função de seu tamanho de entrada (como quando a próxima camada está totalmente conectada e com base na multiplicação da matriz), essa redução no tamanho de entrada também pode resultar em eficiência estatística aprimorada e requisitos de memória reduzidos para armazenar os parâmetros. Para muitas tarefas, o *pooling* é essencial para lidar com entradas de tamanhos variados. Por exemplo, se quisermos classificar imagens de tamanho variável, a entrada para a camada de classificação deve ter um tamanho fixo.

Isso geralmente é realizado variando-se o tamanho de um deslocamento entre as regiões de *pooling*, de modo que a camada de classificação sempre receba o mesmo número de estatísticas de resumo, independentemente do tamanho da entrada. Por exemplo, a camada

final de *pooling* da rede pode ser definida para produzir quatro conjuntos de estatísticas de resumo, um para cada quadrante de uma imagem, independentemente do tamanho da imagem [30].

Figura 4 – Arquitetura CNN típica



Fonte: Géron (2019)

Na etapa após o último o *pooling* temos a etapa de *flattening* (achatamento), como observamos na Figura 4 onde mostra a arquitetura típica da CNN onde temos a rede totalmente conectada, como o próprio nome sugere, ocorre uma transformação da matriz resultado das operações de convolução e *pooling*, transformando-a num vetor unidimensional. Em seguida, a camada densa recebe como entrada o resultado da etapa de *flattering* sendo composta por neurônios logísticos que calculam os pesos e realizam a predição sobre a saída [31][32].

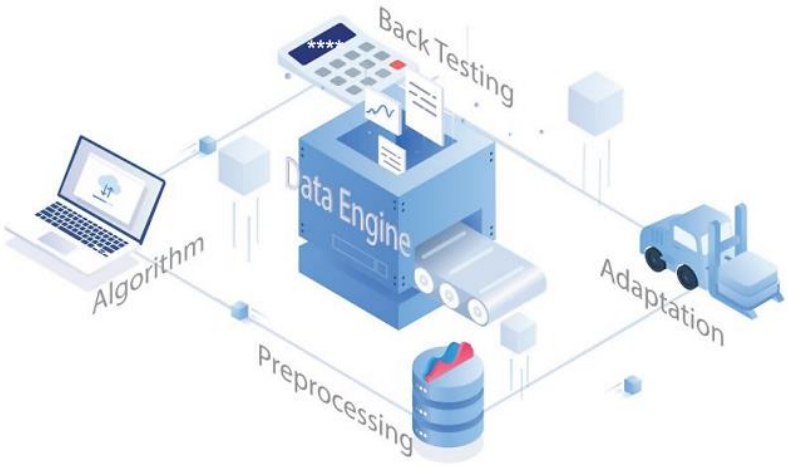
2.3 ROBOFLOW

O *RoboFlow*, um sistema de gerenciamento de fluxo de trabalho baseado em nuvem para desenvolver robôs centrados em dados e aprimorados por IA. Uma visão geral de alto nível do *RoboFlow* é ilustrada na Figura 5. Da forma mais abstrata, o sistema *RoboFlow* divide todo o pipeline de desenvolvimento de robôs aprimorados por IA em 4 módulos de construção (1. processamento de dados, 2. desenvolvimento algorítmico, 3. *back testing* e 4. adaptação de aplicativos) interagindo com um mecanismo de dados centralizado. Especificamente, o mecanismo de dados pode ser visto como um “oráculo” que abstrai todos os detalhes de gerenciamento de dados e está interagindo (por exemplo, sendo consultado ou manipulado) com todos os 4 módulos de construção de forma assíncrona.

Centrado em torno do motor de dados, os 4 módulos de construção seguem um modelo espiral iterativo. Ao contrário do modelo espiral clássico [33] para desenvolvimento de software, esses blocos de construção interagem principalmente com o motor de dados, portanto podem ser desenvolvidos de forma totalmente paralela e cada módulo pode ter

diferentes “versões”. Esse *design* centrado em dados diminui drasticamente as complexidades de desenvolvimento e manutenção, tornando-o especialmente adequado para o desenvolvimento de robôs aprimorados por IA [34].

Figura 5 – Visão Geral do *Roboflow*



Fonte: Lin (2022)

2.4 MÉTODO YOLO

O método YOLO [35] reformula a detecção de objetos ao transformá-la em um problema de regressão, partindo unicamente dos *pixels* de uma imagem, resultando nas probabilidades por classe, nas coordenadas e nas dimensões das predições que delimitam os objetos. Além de simples, já que utiliza uma abordagem fim-a-fim com uma única rede convolucional, também apresenta desempenho competitivo em função do tempo. Diferente de outros métodos de detecção de objetos baseados em redes Convolucionais, YOLO necessita apenas de uma propagação da entrada pela rede para gerar predições, apresentando desempenho compatível com métodos de detecção de objetos em tempo real, mantendo a acurácia preditiva.

2.4.1 Arquitetura YOLOv10

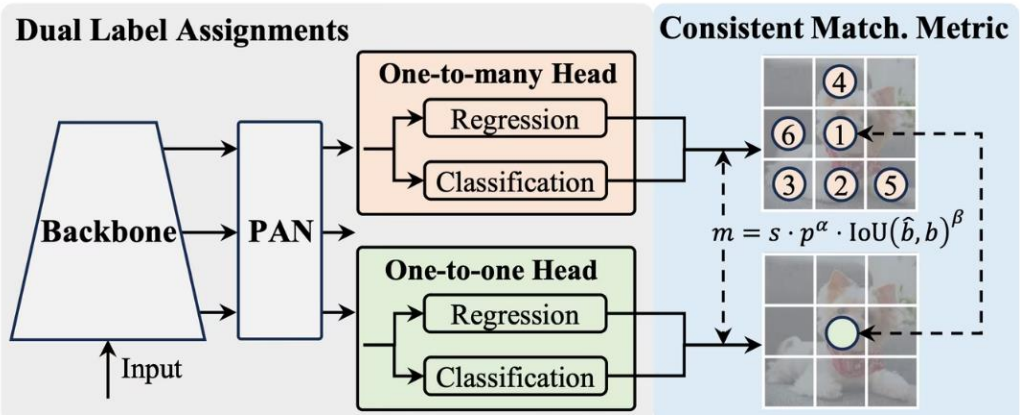
A detecção de objetos em tempo real sempre foi um foco de pesquisa na área de visão computacional, que visa prever com precisão as categorias e posições dos objetos em uma imagem sob baixa latência. É amplamente adotado em diversas aplicações práticas, incluindo direção autônoma, navegação robótica e rastreamento de objetos [36]. Com base nessas abordagens, conseguimos alcançar uma nova família de detectores ponta a ponta em

tempo real com diferentes escalas de modelo, ou seja, [36] YOLOv10-N / S / M / B / L / X. Extensos experimentos em *benchmarks* padrão para detecção de objetos, ou seja, demonstram que o YOLOv10 pode superar significativamente os modelos de última geração anteriores em termos de compensações de precisão computacional em várias escalas de modelo.

O YOLOv10, criado no pacote *Ultralytics Python* por pesquisadores da Universidade Tsinghua, apresenta uma nova abordagem para detecção de objetos em tempo real, abordando as deficiências de pós-processamento e arquitetura de modelo encontradas em versões anteriores do YOLO. Ao eliminar a supressão não máxima (NMS) e otimizar vários componentes do modelo, o YOLOv10 atinge desempenho de última geração com sobrecarga computacional significativamente reduzida. Experimentos extensivos demonstram suas compensações superiores de precisão-latência em várias escalas de modelo [37].

A detecção de objetos em tempo real visa prever com precisão categorias e posições de objetos em imagens com baixa latência. A série YOLO tem estado na vanguarda desta pesquisa devido ao seu equilíbrio entre desempenho e eficiência. No entanto, a dependência do NMS e as ineficiências arquitetônicas têm dificultado o desempenho ideal. O YOLOv10 aborda essas questões introduzindo atribuições duplas consistentes para treinamento sem NMS e uma estratégia de *design* de modelo holística orientada por eficiência-precisão, conforme apresentado na Figura 6.

Figura 6 – Arquitetura YOLOv10



Fonte: Adaptado de Ultralytics (2024)

A arquitetura do YOLOv10 se baseia nos pontos fortes dos modelos YOLO anteriores, ao mesmo tempo em que introduz várias inovações importantes. A arquitetura do modelo consiste nos seguintes componentes:

- *Backbone*: Responsável pela extração de recursos, o *backbone* no YOLOv10 usa uma versão aprimorada do CSPNet (*Cross Stage Partial Network*) para melhorar o fluxo de gradiente e reduzir a redundância computacional;
- *Neck*: O *neck* é projetado para agregar recursos de diferentes escalas e passá-los para a cabeça. Ele inclui camadas PAN (*Path Aggregation Network*) para fusão eficaz de recursos em várias escalas;
- *One-to-Many Head*: Gera várias previsões por objeto durante o treinamento para fornecer sinais de supervisão ricos e melhorar a precisão do aprendizado;
- *One-to-One Head*: Gera uma única melhor previsão por objeto durante a inferência para eliminar a necessidade de NMS, reduzindo assim a latência e melhorando a eficiência [37].

Entre suas principais características estão:

- Treinamento sem NMS: utiliza atribuições duplas consistentes para eliminar a necessidade de NMS, reduzindo a latência de inferência;
- *Design* de modelo holístico: otimização abrangente de vários componentes de perspectivas de eficiência e precisão, incluindo cabeças de classificação leves, amostragem descendente desacoplada de canal espacial e *design* de bloco guiado por classificação;
- Recursos de modelo aprimorados: incorpora convoluções de *kernel* grande e módulos de auto atenção parcial para melhorar o desempenho sem custo computacional significativo.

YOLOv10 foi extensivamente testado em *benchmarks* padrão como COCO, demonstrando desempenho e eficiência superiores. O modelo alcança resultados de última geração em diferentes variantes, apresentando melhorias significativas em latência e precisão em comparação com versões anteriores e outros detectores contemporâneos [37], conforme a Tabela 1.

Tabela 1 – Comparação Detalhada entre as Variantes do YOLOv10 com outros Modelos de Ultima Geração.

Modelo	Parâmetros (M)	FLOPs (G)	mAP^{val} 50-95	Latência (ms)	Avanço na Latência (ms)
YOLOv6-3.0-N	4.7	11.4	37.0	2.69	**1.76**
Gold-YOLO-N	5.6	12.1	**39.6**	2.92	1.82
YOLOv8-N	3.2	8.7	37.3	6.16	1.77
YOLOv10-N	**2.3**	**6.7**	39.5	**1.84**	1.79

Modelo	Parâmetros (M)	FLOPs (G)	mAP^{val} 50-95	Latência (ms)	Avanço na Latência (ms)
YOLOv6-3.0-S	18.5	45.3	44.3	3.42	2.35
Gold-YOLO-S	21.5	46.0	45.4	3.82	2.73
YOLOv8-S	11.2	28.6	44.9	7.07	**2.33**
YOLOv10-S	**7.2**	**21.6**	**46.8**	**2.49**	2.39
RT-DETR-R18	20.0	60.0	46.5	**4.58**	**4.49**
YOLOv6-3.0-M	34.9	85.8	49.1	5.63	4.56
Gold-YOLO-M	41.3	87.5	49.8	6.38	5.45
YOLOv8-M	25.9	78.9	50.6	9.50	5.09
YOLOv10-M	**15.4**	**59.1**	**51.3**	4.74	4.63
YOLOv6-3.0-L	59.6	150.7	51.8	9.02	7.90
Gold-YOLO-L	75.1	151.7	51.8	10.65	9.78
YOLOv8-L	43.7	165.2	52.9	12.39	8.06
RT-DETR-R50	42.0	136.0	53.1	9.20	9.07
YOLOv10-L	**24.4**	**120.3**	**53.4**	**7.28**	**7.21**
YOLOv8-X	68.2	257.8	53.9	16.86	12.83
RT-DETR-R101	76.0	259.0	54.3	13.71	13.58
YOLOv10-X	**29.5**	**160.4**	**54.4**	**10.70**	**10.60**

Fonte: Ultralytics (2024)

O YOLOv10 define um novo padrão em detecção de objetos em tempo real ao abordar as deficiências das versões anteriores do YOLO e incorporar estratégias de *design* inovadoras. Sua capacidade de fornecer alta precisão com baixo custo computacional o torna uma escolha ideal para uma ampla gama de aplicações do mundo real.

2.5 ANÁLISE E MÉTRICAS DE FADIGA

2.5.1 Estágios do Sono

O sono pode ser categorizado em duas fases: Movimento não rápido dos olhos (NREM – *Non Rapid Eye Moviment*) e Movimento rápido dos olhos (REM - *Rapid Eye Moviment*). O estado NREM, que equivale a cerca de 75% do período de sono, pode ser dividido em 4 estágios [38]:

- Estágio 1: É a etapa da sonolência, onde começam os primeiros sinais de fadiga. Neste estágio o indivíduo pode ser acordado facilmente;
- Estágio 2: Com duração de 5 a 25 minutos. A atividade cardíaca diminui, ocorre um relaxamento dos músculos e a temperatura corporal cai. Poder ser denominado de sono leve, pois, já não é possível despertar facilmente;

- Estágio 3: Pode ser caracterizado como um período transitório entre o sono leve e o sono profundo. O ritmo cardíaco diminui e a respiração fica mais lenta;
- Estágio 4: Denominado de sono profundo. Tem duração de cerca de 40 minutos, sendo muito semelhante ao estágio 3, porém, o nível de sonolência está em um nível ainda mais profundo.

O sono REM dura 25% do tempo de repouso, é nesta fase onde os sonhos ocorrem. Pode ser especificado pela ocorrência de movimentos oculares rápidos, contração muscular e a respiração se torna rápida e irregular. Essas características são resultado de uma intensa atividade cerebral [38].

2.6 FADIGA AO DIRIGIR

Os motoristas normalmente passam por uma transição gradual de um estado de alerta para um estado de fadiga. Definir a direção com fadiga geralmente envolve extrair características de uma sequência de expressões faciais do motorista durante um período de tempo e fazer julgamentos sobre fadiga com base na relação entre esses quadros sequenciais. Ao classificar os estados dos olhos e da boca, parâmetros precisam ser selecionados para determinar o estado de fadiga. O cálculo em tempo real destes parâmetros de fadiga é necessário, sendo eles combinados com limites predefinidos para permitir julgamentos sobre o estado de fadiga dos condutores [18].

2.6.1 *Percentage of Eyelid Closure* (PERCLOS)

A detecção de fadiga em tempo real é realizada através do indicador nomeado PERCLOS, cuja composição na língua inglesa é *Percentage of Eyelid Closure*, que significa “Porcentagem de Fechamento dos Olhos ao Longo do Tempo”, é um padrão reconhecido internacionalmente para avaliação de fadiga. Refere-se à porcentagem de tempo gasto com os olhos fechados em um determinado período. No processo de condução real, o fluxo de vídeo capturado pela câmera é convertido em quadros individuais para processamento.

Portanto, o padrão PERCLOS também pode ser traduzido como a porcentagem de quadros de tempo com olhos fechados sobre o número total de quadros dentro de uma unidade de tempo específica [18]. A duração ($D_{i,x}$) é definido como o tempo acumulado em que $y_{i,x}$ é consecutivamente verdadeiro, como mostrado na equação (1). Como o tempo de processamento é inconsistente devido ao atraso de comunicação e perda de pacotes, o nomeado de PERCLOS, tem o tempo predefinido (T), conforme mostrado na equação

(2). $y_{i,x}$ significa i^{th} valor verdadeiro ou falso do quadro de fechamento do olho e direções da cabeça codificadas *one-hot* dentro do total de quadros (L). τ_i significa o tempo de processamento do i^{th} por quadro [39].

$$D_{i,x} = \begin{cases} D_{i-1,x} + \tau_i \text{ , se } y_{i,x} \text{ é } verdadeiro \\ 0, \quad \text{se } y_{i,x} \text{ é } falso \end{cases} \quad (1)$$
$$(i \in \{1,2,3,\dots,L\}, x \in \{fechamento\ dos\ olhos, cabeça\ baxia\})$$

Fonte: Kim (2024)

$$PERCLOS = \frac{1}{N} \sum_{i=n}^{m} y_{i,x} \times \tau_i, \ \ se \ \sum_{j=n}^{m} \tau_j \leq T \quad (2)$$
$$(\text{x} \in \{fechamento\ dos\ olhos\})$$

Fonte: Kim (2024)

2.6.2 Fadiga com Base nas Características dos Olhos (EAR)

A configuração do limite de fadiga do piscar é baseada na proporção de quadros de olhos fechados em relação ao total de quadros em um minuto. Em condições normais, uma pessoa pisca menos de 30 vezes por minuto, com intervalos de aproximadamente 3 a 5 segundos entre as piscadas, e cada piscada dura cerca de 4 a 5 quadros. Quando uma pessoa está fatigada, a frequência de piscar pode exceder 30 vezes por minuto. Segundo pesquisas, se os olhos de uma pessoa permanecerem fechados continuamente por 2 segundos enquanto dirige, isso pode aumentar significativamente o risco de um acidente de trânsito. Através de testes reais, a taxa de quadros do equipamento de teste usado neste estudo pode atingir contínuos 25 quadros por segundo. Portanto, o limite é definido em 30 quadros. Se a duração do fechamento contínuo dos olhos exceder 30 quadros, é considerado um estado de fadiga [18].

2.6.3 Fadiga com Base nas Características da Boca (MAR)

O comportamento de bocejo pode ser considerado, até certo ponto, uma resposta subconsciente à fadiga, servindo como um indicador importante na determinação do estado de direção do motorista. O ato de bocejar é uma característica proeminente na análise de imagens. Se o valor MAR (Proporção de Boca) detectado pelo sistema for significativamente maior que o valor normal em circunstâncias normais e persistir por mais tempo, pode-se considerar que o motorista está bocejando. O limite de fadiga bucal é definido com base na proporção de uma série de quadros, o que mostra um comportamento de bocejo dentro um minuto. O limiar de fadiga da boca é com base na proporção de quadro de bocejo dentro de

um minuto. Uma pessoa boceja menos de 2 vezes por minuto e cada bocejo dura aproximadamente 6 segundos [18].

2.6.4 Sinais de Sonolência

Sinais fisiológicos começam a mudar em estágios iniciais de sonolência. Sendo assim, os mais adequados para detectar a sonolência com poucos falsos positivos tornando possível alertar um motorista sonolento em tempo hábil e, assim, evitar muitos acidentes rodoviários. Muitos pesquisadores consideram que os sinais biomédicos ideias para realizar a detecção de sonolência são eletrocardiograma (ECG), eletroencefalograma (EEG) [40].

A frequência cardíaca (*HR – Heart rate*) varia significativamente entre os diferentes estágios de sonolência, como alerta e fadiga. Portanto, frequência cardíaca, que pode ser facilmente determinada pelo Sinal de ECG, também pode ser usada para detectar a sonolência. Outras formas de mensurar o estágio de sono é utilizando Variabilidade da Frequência Cardíaca (HRV – *Heart Rate Variability*), em que as frequências baixas (LF – *Low Frequencies*) e altas (HF – *High Frequencies*) caem no intervalo de 0.04–0.15 Hz e 0.14–0.4Hz [40].

O sinal EEG possui várias bandas de frequência, incluindo a banda delta (0,5 a 4 Hz), que corresponde à atividade do sono, a banda teta (4 a 8 Hz), que está relacionada à sonolência, a banda alfa (8 a 13 Hz) , que representa relaxamento e criatividade, e a banda beta (13–25 Hz), que corresponde ao estado de alerta. Uma diminuição nas mudanças de poder na banda de frequência alfa e um aumento na banda de frequência teta indica sonolência [40].

O grande problema destes métodos é a natureza intrusiva, já que existe a necessidade de conectar elétrodos ao corpo do indivíduo. Para solucionar este tipo problema já foram elaborados sistemas onde os elétrodos foram colocados no volante ou mesmo no banco do motorista [40].

O termo "sonolento" é sinônimo de sonolento, que significa simplesmente uma inclinação para adormecer. Os estágios do sono podem ser categorizados como acordado, sono sem movimento rápido dos olhos (NREM) e sono de movimento rápido dos olhos (REM). A segunda fase, NREM, pode ser subdividida nas três fases seguintes [41]:

- Estágio 1: Transição de acordado para adormecido (sonolento);
- Estágio 2: Sono leve;
- Estágio 3: Sono profundo.

Para analisar a sonolência do motorista, os pesquisadores estudaram principalmente o estágio 1, que é a fase de sonolência. As colisões que ocorrem devido à sonolência do motorista têm uma série de características [42]:

2.6.5 Acidentes Causados por Sonolência

Para analisar os efeitos da sonolência nos condutores foram realizados estudos, que identificaram que as colisões que ocorrem devido aos efeitos da sonolência têm várias características [40]:

- Ocorre tarde da noite (das 0h às 07h) ou no meio da tarde (das 14h às 16h);

Envolver um único veículo que sai da pista;

- Ocorre em rodovias;
- O motorista está frequentemente sozinho;
- O motorista é geralmente um jovem de 16 a 25 anos;
- Sem marcas de derrapagem ou indicação de freada brusca;
- Em relação a essas características, os bancos de dados da *Southwest England* e da *Midlands Police* usam os seguintes critérios para identificar acidentes causados por sonolência [40];
- Nível de álcool no sangue abaixo do limite legal de direção;
- Veículo não tem defeito mecânico;
- Boas condições meteorológicas e visibilidade clara;
- Eliminação do “excesso de velocidade” ou “dirigir muito perto do veículo na frente” como causas potenciais.

3 METODOLOGIA

Neste capítulo, detalham-se os meios e métodos de desenvolvimento do trabalho, que foi dividido em cinco etapas: criação da base de dados, treinamento do modelo, definição das métricas de avaliação, estudo dos indicadores de fadiga e pós-processamento, conforme apresentado no Quadro 1.

Quadro 1 – Fases do Projeto

FASES DO PROJETO		
Ordem	**Descrição**	**Atividades**
Etapa 1	Criação da Base de Dados	• Plataforma de Desenvolvimento da Base de Dados
		• Levantamento das Imagens
		• Anotações das Classes com *Roboflow* Utilizando *Boxes*
		• Divisão da Base de Dados
Etapa 2	Treinamento do Modelo	• Escolha do Modelo
		• Ambiente de Treinamento
		• Configuração da Rede YOLO
		• Importação da Base de Dados
		• Configuração de Treinamento
Etapa 3	Definição das Métricas de Desempenho para Avaliação	• Matriz de Confusão
		• Precisão (*Precision*)
		• Sensibilidade (*Recall*)
		• F1-*Score*
		• Precisão Média (MAP – *mean Average Precision*)
		• Análise de Perdas e Métricas
Etapa 4	Indicadores de Fadiga	• Julgamento de Nível de Fadiga
Etapa 5	Pós-Processamento	• Ambiente Utilizado
		• Definições do *Algoritmo*

Fonte: Autoria própria

A partir da estrutura estabelecida, cada uma dessas etapas será descrita de forma detalhada nos próximos tópicos, com o objetivo de apresentar de maneira clara e objetiva os procedimentos adotados, bem como as justificativas técnicas que orientaram as decisões ao longo do desenvolvimento deste trabalho.

3.1 CRIAÇÃO DA BASE DE DADOS

3.1.1 Plataforma de Desenvolvimento da Base de Dados

A escolha do *Roboflow* foi motivada por sua interface amigável e facilidade de uso, que aceleram o processo de anotação e pré-processamento das imagens. Além de ser compatível com diversos formatos de dados e *frameworks*, como o YOLO, a plataforma oferece *data augmentation* automática, essencial para aumentar a diversidade da base de dados, para aprimorar a robustez do modelo. O *Roboflow* também possibilita treinamentos na nuvem, reduzindo a necessidade de infraestrutura local, e fornece ferramentas de gerenciamento e visualização, para otimizar o controle do projeto.

3.1.2 Levantamento de Imagens

O banco de dados foi desenvolvido a partir de imagens de duas pessoas, que serviram de modelos para o levantamento inicial. Esse conjunto de dados foi organizado com imagens coloridas e em escala de cinza, visando simular possíveis movimentos indicativos de sobriedade e sonolência. Parte das imagens inclui indivíduos utilizando óculos, as quais foram obtidas por meio do Google Imagens, sendo que cada uma possui tanto a versão colorida quanto a versão em escala de cinza, conforme indicado na Tabela 2.

Tabela 2 – Levantamento das Imagens

Escala de Imagem	Modelo 1	Modelo 2	Google Imagens
Imagens Coloridas	560 19,17%	757 25,91%	176 6,04%
Imagens em Cinza	560 19,17%	757 25,91%	111 3,80%

Fonte: Autoria própria

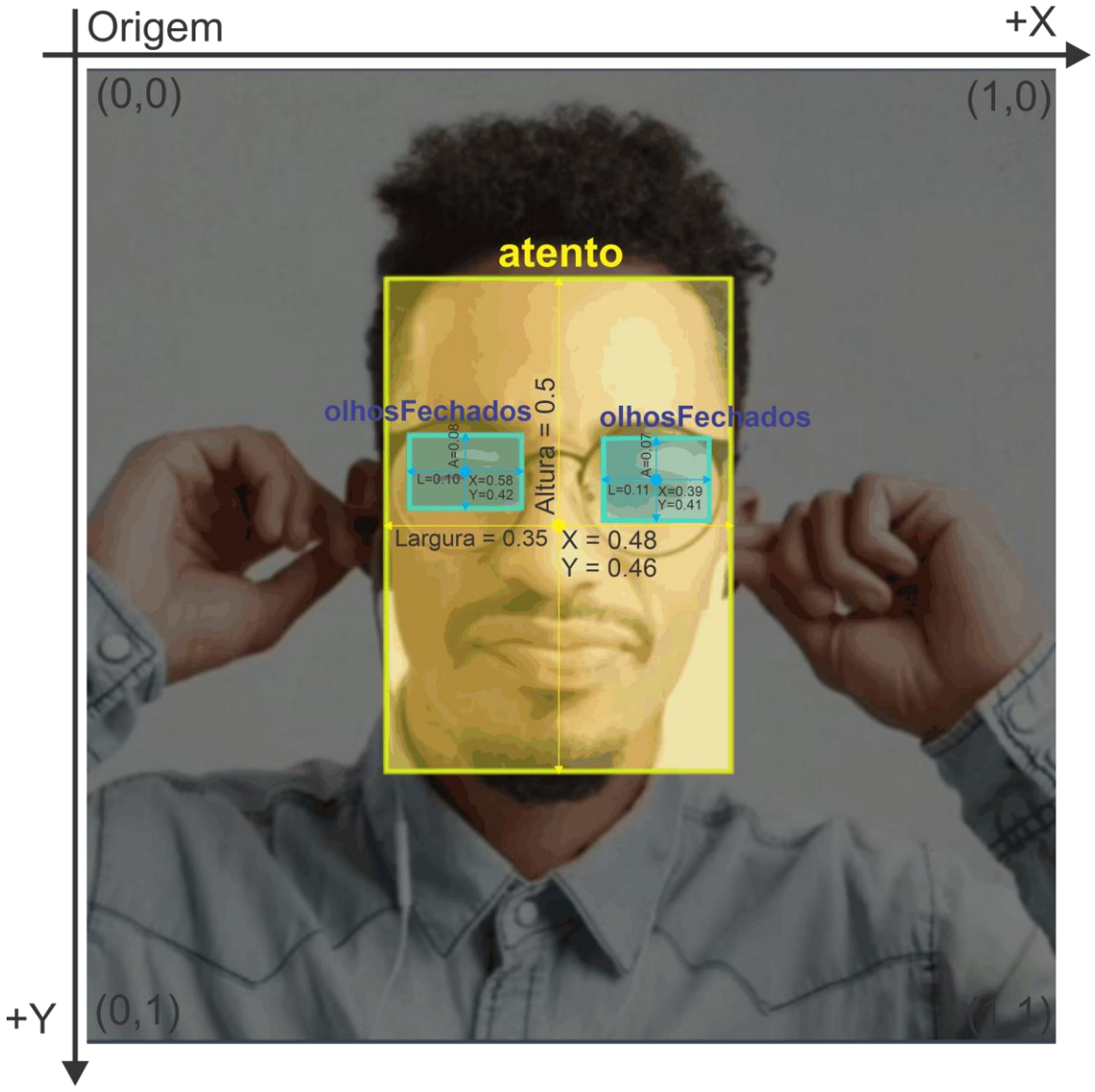
As imagens utilizadas no treinamento foram padronizadas de acordo com os requisitos do modelo YOLO, sendo todas convertidas para o formato compatível, salvas com a extensão “.jpg” e redimensionadas para a resolução de 640x640 *pixels*. Essa padronização

foi essencial para garantir a consistência dos dados de entrada, permitindo que o modelo processe as imagens de forma eficiente.

3.1.3 Anotação das Classes com *Roboflow* Utilizando *Boxes*

As anotações das classes foram realizadas manualmente na plataforma *Roboflow*, com o uso de caixas delimitadoras. Cada caixa foi desenhada e classificada conforme sua respectiva classe. Para este projeto, foram utilizadas 6 classes, variando de 0 a 5: “atento”, “bocejando”, “cabecaBaixa”, “olhosAbertos”, “olhosFechados” e “sonolento”, conforme ilustrado na Figura 7.

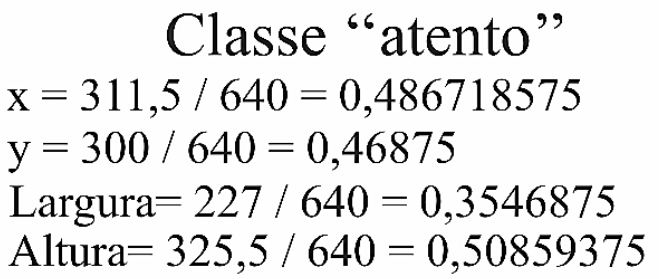
Figura 7 – Modelo de Etiquetas para YOLO



Fonte: Autoria própria

As etiquetas foram exportadas no formato YOLO, gerando um arquivo “.txt” por imagem. O arquivo “.txt” foi formatado com uma linha por objeto, contendo: classe, x, y, largura e altura. As coordenadas das caixas devem estar normalizadas (de 0 a 1). Para isso, as coordenadas em *pixels* devem ser divididas pela largura total da imagem (para x e largura) e pela altura total da imagem (para y e altura), conforme apresentado na Figura 8.

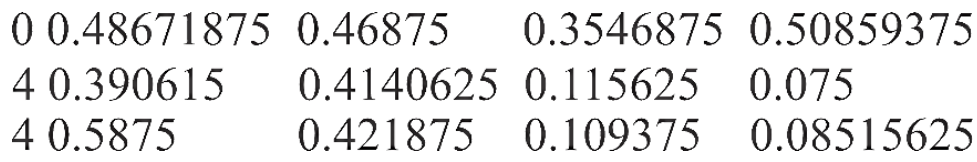
Figura 8 – Exemplo do Formato de Dados para YOLO



Fonte: Autoria própria

As chamadas *labels* são essenciais para o processo de treinamento do modelo YOLO. Elas contêm as informações sobre as classes dos objetos presentes em cada imagem, bem como suas respectivas localizações no formato de coordenadas de *bounding boxes*, conforme apresentado na Figura 9.

Figura 9 – Formato do Arquivo .txt com as *labels*



Fonte: Autoria própria

Essas *labels* são armazenadas em arquivos de texto com a mesma nomenclatura das imagens correspondentes. Esse padrão permite que o modelo YOLO identifique e localize corretamente os objetos durante o treinamento, associando cada região de interesse ao seu rótulo correspondente.

3.1.4 Divisão da Base de Dados

A base de dados consistiu em 2.921 imagens, distribuídas entre treino, validação e teste. Para o treinamento, foram utilizadas 1.860 imagens (64%); para a validação, 619 imagens (21%); e para os testes, 442 imagens (15%). Essa divisão reflete um equilíbrio entre o volume necessário para aprendizado e o volume necessário para validação e teste, assegurando que o modelo seja treinado de forma eficiente.

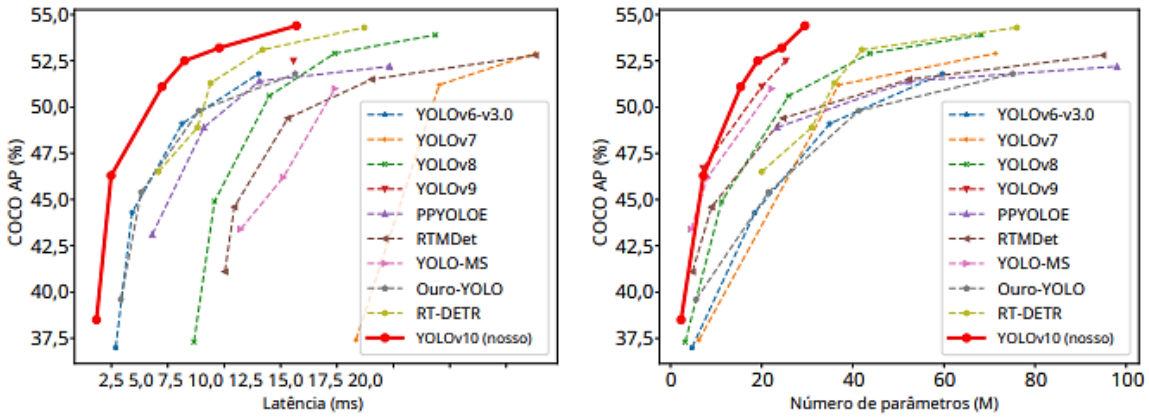
3.2 TREINAMENTO DO MODELO

O treinamento do modelo é parte fundamental de um projeto de visão computacional que envolve detecção de imagens, além de incluir a escolha do modelo que oferece maior desempenho e melhor precisão.

3.2.1 Escolha do Modelo

O modelo escolhido para o treinamento foi o YOLOv10-S, a versão mais recente e aprimorada da família YOLO. Essa escolha foi baseada em suas melhorias significativas, como alto desempenho, velocidade superior, requisitos de hardware reduzidos e baixa latência, indicados na Figura 10. Durante os testes realizados com imagens, o YOLOv10-S demonstrou maior eficiência e precisão em comparação com outras versões.

Figura 10 – Comparação em Termos de Compensações de Latência e Número de Parâmetros.



Fonte: Adaptado de (AO WANG, 2024)

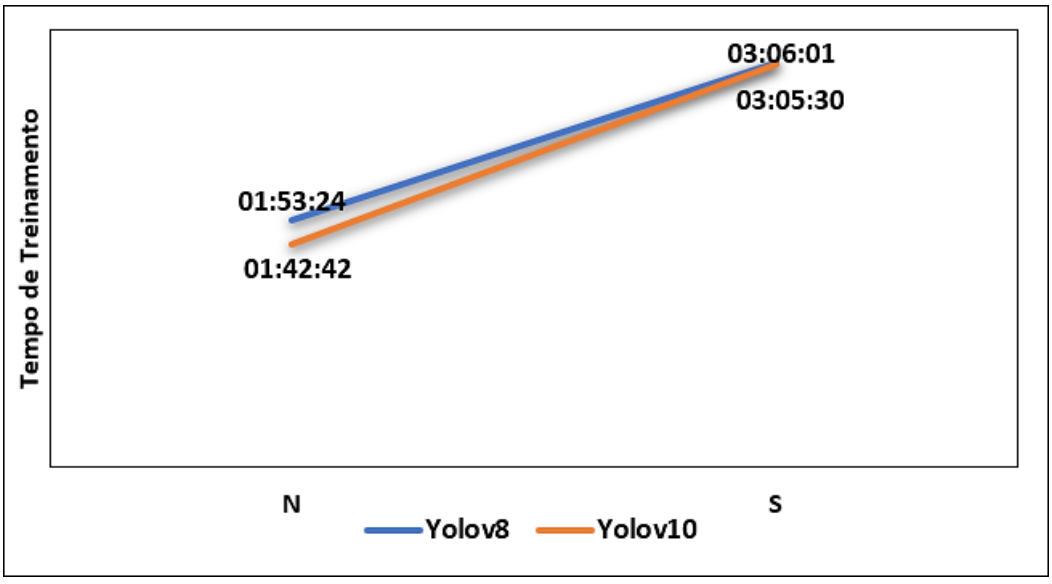
A versão YOLOv8-N apresentou problemas relacionados ao IoU (Intersection over Union), exibindo múltiplas caixas sobrepostas para a mesma detecção conforme a. Já a versões YOLOv8-S e YOLOv10-N não conseguiram identificar todas as expressões na imagem testada na Figura 11, o que reforçou a escolha do YOLOv10-S como o modelo ideal para o projeto.

Figura 11 – Comparação dos Modelos Treinados

Fonte: Autoria própria

A Figura 12 teve como objetivo comparar o desempenho em termos de velocidade de treinamento dos modelos YOLOv8 e YOLOv10, utilizando as variantes N e S. Para tanto, ambos os modelos foram treinados com o mesmo conjunto de dados, juntamente com os mesmos hiper parâmetros apresentados a seguir na Tabela 3.

Figura 12 – Tempo de Treinamento dos Modelos YOLO



Fonte: Autoria própria

Os resultados obtidos, indicam que o modelo YOLOv10 apresentou um tempo de treinamento menor em comparação ao YOLOv8, tanto na variante N quanto na S.

3.2.2 Ambiente de Treinamento

Neste projeto foi utilizado a plataforma *Kaggle*[1], uma plataforma online para desenvolvimento de algoritmos, análise de dados e *Machine Mearning*. *Kaggle*[1] oferece um ambiente baseado em *Notebooks Jupyter* para executar códigos em *Python*. A plataforma nos disponibiliza o uso de uma GPU (Nvidia Tesla P100) de forma gratuita. O *Kaggle*[1] oferece um processador Intel (R) Xeon (R) CPU @ 2.20GHz, 32GB de memória RAM e CPU NVIDIA Tesla P100-PCIE 16GB.

3.2.3 Configuração da Rede YOLO

Para treinar o modelo YOLOv10 no *Kaggle*[1], primeiro e necessário clonar seu repositório da *Ultralytics* no *GitHub*, onde está contido todas as ferramentas necessárias para o treinamento. Em seguida e feito a importação do modelo YOLOv10 da biblioteca *Ultralytics* com as seguintes linhas de códigos:

Algoritmo 1 – Clonando a Biblioteca *Ultralytics* e Importando o Modelo Yolov10

```
!pip install -q git+https://github.com/THU-MIG/yolov10.git
from ultralytics import YOLOv10
```

3.2.4 Importação da Base de Dados

A importação do base de dados foi realizada diretamente da plataforma do *Roboflow*, seguindo o formato de dados do YOLOv8, que é compatível com a versão do YOLOv10. Cada usuário tem sua chave de acesso para a importação da base de dados. O processo foi executado com as seguintes linhas de códigos:

Algoritmo 2 – *Download* da Base de Dados por Código

```
!pip install roboflow
from roboflow import Roboflow
rf = Roboflow(api_key = " chave de acesso ")
project = rf.workspace("leonardo-6asee").project("deteccao-de-sono")
version = project.version(2)
dataset = version.download(“yolov8”)
```

[1] Disponível em: <https://www.kaggle.com/>. Acesso em: 25 de junho de 2024.

3.2.5 Configuração de Treinamento

Para o treinamento dos modelos utilizados neste projeto, e feito o *download* do modelo pré-treinado desejado, que se encontra no repositório do *GitHub*, com o uso do código abaixo:

Algoritmo 3 – *Download* do Repositório com o Modelo a ser Treinado

```
!wget -P /kaggle/working -q https://github.com/THU-MIG/yolov10/releases/download/v1.1/yolov10n.pt
```

Foram utilizadas duas distribuições do modelo YOLOv10, (YOLOv10-N e YOLOv10-S) com o intuído de realizar um comparativo e escolher a melhor distribuição. O mesmo processo e repetido para utilizar cada uma das distribuições.

Em seguida se utilizou uma ferramenta chamada *WandB*[2], que serviu para o monitoramento e gerenciamento dos dados gerados durante o treinamento, *WandB*[2] permite que se registre cada execução de um modelo com todos os detalhes relevantes, como hiper parâmetros, arquitetura do modelo e métricas de desempenho. Esta ferramenta foi escolhida devido a sua integração com outras ferramentas, neste caso, utilizamos para integrar ao *Jupyter Notebooks* da ferramenta *Kaggle*[1]. Ao se criar uma conta pelo site da ferramenta *WandB*[2] e criada uma chave de acesso para cada usuário, utilizada durante a importação no ambiente de execução do projeto. Sua importação e feita antes do treinamento se iniciar, com os códigos abaixo:

Algoritmo 4 – Importação do Banco de Dados Wandb

```
import wandb
wandb.login(key=" chave de acesso ")
wandb.init(
   project='YOLOV10',
   name="Yolov10s-trainv1",
)
```

O processo de treinamento foi iniciado com o ajuste dos hiper parâmetros, etapa essencial para otimizar o desempenho de cada modelo avaliado. Esses ajustes são fundamentais para garantir que o modelo seja capaz de extrair as características mais relevantes dos dados de entrada. Os mais comuns estão destacados na Tabela 3:

[2] Disponível em: <https://wandb.ai/>. Acesso em: 25 de junho de 2024.

Tabela 3 – Hiper Parâmetros de Treinamento

Hiper Parâmetros	YOLOv10-N	YOLOv10-S	YOLOv8-N	YOLOv8-S
Epochs	300	300	300	300
Batch	80	32	80	32
Workers	8	8	8	8
Patience	100	100	100	100

Fonte: Autoria própria

Epochs – Número total de épocas de treinamento. Cada época representa uma passagem completa por todo o conjunto de dados. Ajustar esse valor pode afetar a duração do treinamento e o desempenho do modelo;

Batch – Número de amostras de dados passadas através do modelo antes de atualizar os dados, influencia a estabilidade do treinamento e o uso da memória GPU;

Workers – Número de threads de trabalho para carregamento de dados. Influencia a velocidade de pré-processamento de dados e alimentação no modelo, especialmente útil em configurações multi-GPU;

Patience – Número de épocas para esperar sem melhoria nas métricas de validação antes de para o treinamento antecipadamente. Ajuda a evitar *overfitting* ao para o treinamento quando o desempenho atinge um platô. Além de outros diversos parâmetros que podem ser ajustados de acordo com a necessidade de cada modelo. Segue abaixo o código que inicia o treinamento.

Algoritmo 5 – Parâmentros de Início de Treinamento

```
!yolo task=detect mode=train epochs=300 batch=80 patience=100 workers = 8 \
model=/kaggle/working/yolov10n.pt \
data=/kaggle/working/Detecção-de-Sono-2/data.yaml
```

3.3 DEFINIÇÃO DAS METRICAS DE DESEMPENHO PARA AVALIAÇÃO

Nesta etapa, buscou-se definir as métricas utilizadas para avaliar o desempenho do modelo treinado. O meio mais comum para representar o desempenho de classificação de um modelo é conhecido como matriz de confusão. No entanto, a matriz de confusão registra apenas contagens, e ao lidar com uma grande quantidade de dados, avaliar o desempenho do

modelo apenas com base nessas contagens pode ser desafiador. Consequentemente, o uso de métricas como precisão, sensibilidade e pontuação F1 torna-se uma abordagem padrão para avaliar o sistema proposto. Além disso, outras métricas, como a precisão média (MAP) e a análise de perdas, foram utilizadas como complementos para proporcionar uma avaliação mais precisa do modelo.

3.3.1 Matriz de Confusão

Utilizou-se a matriz de confusão para apresentar os resultados quantitativos das previsões. Existem quatro resultados de classificação mutuamente exclusivos nas previsões, conforme a Figura 13. A estrutura da matriz de confusão pode ser facilmente estendida para lidar com problemas de classificação multiclasse.

Figura 13 – Sistema de Matriz de Confusão

		Valor Predito	
		Sim	**Não**
Real	**Sim**	Verdadeiro Positivo (TP)	Falso Negativo (FN)
	Não	Falso Positivo (FP)	Verdadeiro Negativo (TN)

Fonte: Nogare, 2020

T, F (Verdadeiro e Falso) e P, N (Positivo e Negativo). T e F indicam se o resultado da classificação corresponde ao valor verdadeiro, enquanto P e N representam os valores que estão sendo classificados.

3.3.2 Precisão (*Precision*)

Para melhorar a precisão dos resultados a métrica de precisão, a equação (3) foi utilizada para mostrar como a precisão das previsões do modelo varia em diferentes níveis de confiança e reflete na capacidade do modelo de diferenciar amostras negativas.

$$Precision = \frac{TP(Verdadeiro\ Positivo)}{TP(Verdadeiro\ Positivo) + FP(Falso\ Positivo)} \quad (3)$$

Fonte: Tang (2024)

3.3.3 Sensibilidade (*Recall*)

A curva de Sensibilidade (*Recall*) ilustrada na equação (4), mede a capacidade do modelo de reconhecer amostras positivas, que foi utilizada para demonstrar a variação nas taxas de recall em diferentes limites de confiança. Reflete as variações nas taxas de *recall* à medida que os níveis de confiança flutuam.

$$Recall = \frac{TP(Verdadeiro\ Positivo)}{TP(Verdadeiro\ Positivo) + FP(Falso\ Negativo)} \quad (4)$$

Fonte: Tang (2024)

3.3.4 F1-*Score*

A pontuação F1 serviu como uma medida da média ponderada de precisão e sensibilidade. Sua expressão é como a equação (5).

$$F1 - Score = \frac{2x\ Precision\ x\ Recall}{Precision + Recall} \quad (5)$$

Fonte: Tang (2024)

onde *Precision* a representa a proporção de amostras positivas previstas corretamente entre todas as amostras previstas como positivas e *Recall* representa a proporção de amostras positivas previstas corretamente entre todas as amostras positivas verdadeiras.

3.3.5 Precisão Média (MAP – *Mean Average Precision*)

Foi usada para avaliar o desempenho de modelos de detecção de objetos em várias categorias. Ele calcula a precisão média (AP) para cada categoria e, em seguida, calcula a média desses valores de AP para avaliar o desempenho do modelo, conforme a equação (6).

$$mAP = \frac{1}{C}\sum_{i=1}^{c} AP_i \quad (6)$$

Fonte: Tang (2024)

onde C representa o número de categorias no conjunto de dados. Quanto maior o valor de mAP, melhor será o desempenho do modelo. AP é a área sob a curva de recuperação de precisão e representa a precisão média do modelo em diferentes taxas de recuperação. É definido como na equação (7):

$$AP = \int_0^1 PRdr \tag{7}$$

Fonte: Tang (2024)

À medida que o limite de classificação muda, a precisão e a recuperação também mudam.

3.3.6 Análise de Perdas e Métricas

Os gráficos com as perdas (caixas delimitadoras, objeto, classes e *Dual Focal Loss* (DFL)) e métricas de qualidade (precisão, sensibilidade e MAP) do modelo através das épocas de treinamento para conjuntos de treinamento e validão. Mostram como as métricas de desempenho do modelo aumentam com as épocas e as perdas do modelo decaem com as épocas.

3.4 INDICADORES DE FADIGA

O sistema de detecção de sonolência do motorista foi desenvolvido utilizando quatro indicadores principais, que auxiliaram na avaliação do estado de fadiga do condutor:

- **PERCLOS**: Esse indicador mede o percentual de tempo em que os olhos permanecem fechados acima de um determinado limite.
- **EAR**: Esse indicador é essencial para detectar tanto piscadas prolongadas quanto olhos completamente fechados, o que sinaliza fadiga.
- **MAR**: Esse índice é utilizado para detectar bocejos, um comportamento comum em motoristas cansados.
- **Direção da Cabeça**: A orientação da cabeça é monitorada para identificar inclinações na posição da cabeça, como inclinações frequentes ou quedas bruscas, podem ser indicativos de sonolência ou distração.

3.4.1 Julgamento de Nível de Fadiga

Para a avaliação do nível de fadiga do motorista, foram estabelecidos diversos parâmetros que permitem detectar e classificar diferentes graus de fadiga. Esses parâmetros são baseados em previsões geradas a partir das caixas delimitadoras associadas a diferentes rótulos, como “atento”, “bocejando”, “cabeça baixa”, “olhos abertos”, “olhos fechados” e “sonolento”. As detecções são realizadas em tempo real, e as anotações dessas etiquetas servem para classificar o estado do motorista após um determinado intervalo de tempo. Dessa

forma, o sistema é capaz de analisar continuamente o comportamento do motorista e atribuir um nível de fadiga de acordo com os padrões detectados ao longo do tempo.

As configurações de limite para a sinalização de um estado de alerta de fadiga são mostradas na Tabela 4, definiu-se os limites para a determinação do estado de sonolência baseados nos indicadores EAR e MAR.

Tabela 4 – Limites Baseados em EAR e MAR.

Um minuto	Condição A Frequência de Piscadas	Condição B Frequência de Bocejos	Determinação de Fadiga
Todas Satisfeitas	< 30	< 2	Normal
Condição A ou B	≥ 30	≥ 2	Fadiga Leve
Todas Satisfeitas	≥ 30	≥ 2	Fadiga Grave

Fonte: Adaptado de Tang (2024)

A cada piscada detectada, o contador é incrementado, a contagem é usada como um indicador para a detecção de fadiga ao dirigir. O comportamento normal de um motorista que não está cansado e inferior a 30 piscadas por minuto, o que não aciona o alerta. O mesmo acontece quando o indivíduo boceja menos de duas vezes por minuto, o que é considerado um estado normal. Quando a contagem de piscadas e superior a 30 ou número de bocejos e superior a 2 é classificado como fadiga leve. Caso o número de piscadas for superior a 30 e o número de bocejos for superior a 2, é classificado como fadiga grave.

A Figura 14 ilustra os níveis de fadiga baseados no indicador PERCLOS. Para cada intervalo de tempo, é atribuído um nível de fadiga de acordo com a duração dos eventos de piscar ou de inclinação da cabeça. Os níveis variam de 0 a 4, onde o nível 0 não foi considerado neste trabalho, pois representa o estado normal do motorista e não aciona alertas de sonolência.

Figura 14 – Definição do Nível de Fadiga por Tempo de Olhos Fechados.

$$X_{eu} = \begin{cases} T_0 \in [T, \infty) \\ T_1 \in [2{,}0\text{s}, \infty) \\ T_2 \in [0{,}5\text{s}, 1{,}0\text{s}) \\ T_3 \in [1{,}0\text{s}, 1{,}5\text{s}) \\ T_4 \in [1{,}5\text{s}, 2{,}0\text{s}) \end{cases}$$

Fonte: Kim (2019)

Os demais níveis de fadiga (de 1 a 4) são registrados a cada detecção dos eventos de piscar ou baixar a cabeça, refletindo a intensidade da fadiga do motorista. Esses registros em tempo real são utilizados para classificar o nível de alerta necessário, permitindo a intervenção preventiva quando o risco de sonolência aumenta.

3.5 PÓS-PROCESSAMENTO

Nesta etapa, foi desenvolvido um algoritmo que integra todos os parâmetros estabelecidos neste estudo. O modelo de teste utilizado foi o YOLOv10-S, que demonstrou ser adequado para atender às necessidades do projeto, considerando o poder de processamento do ambiente utilizado durante os testes. O sistema é capaz de determinar o estado do motorista, registrando as detecções, o nível de fadiga, a duração de cada evento (como piscar, baixar a cabeça e bocejar), bem como a contagem desses eventos.

Com base nessas informações, o sistema exibe alertas em tempo real quando as condições predefinidas são atendidas. Os alertas gerados são: "alerta de fadiga leve", "alerta de fadiga grave" e "alerta dormindo", conforme o nível de sonolência detectado. Para os testes, foi utilizada uma *webcam* (USB-303 1080P HD, 2Mpx) para simular diferentes níveis de sonolência, permitindo avaliar o desempenho do sistema e obter resultados mais precisos.

3.5.1 Ambiente Utilizado

Para a realização dos testes com o *algoritmo* de detecção, foi utilizado um notebook Dell Inspiron 5567, equipado com um processador Intel® Core™ i5 de 2,50 GHz, 8 GB de memória RAM e um SSD de 360 GB. As simulações foram conduzidas no *Visual Studio Code* (VS Code), um editor de código-fonte de fácil uso desenvolvido pela *Microsoft*. O *algoritmo* foi implementado utilizando a linguagem *Python*, na versão 3.11.4. As demais bibliotecas e ferramentas utilizadas no desenvolvimento do sistema estão detalhadas no Quadro 2.

Quadro 2 – Bibliotecas Utilizadas no Algoritmo

Nome	Função	Versão
OPEN-CV	OpenCV é uma biblioteca de código aberto que inclui várias centenas de algoritmos de visão computacional. Inclui estimativa de movimento, subtração em segundo plano e algoritmos de rastreamento de objetos.	4.9.0.80
TIME	Time é uma biblioteca padrão do Python. Ela permite manipular, formatar e exibir datas, horários e intervalos de tempo. Com a biblioteca Time, é possível realizar operações como cálculos de duração de tempo, além de obter informações detalhadas sobre o tempo atual.	3.11.4
ULTRALYTICS	Ultralytics fornece um sistema de gestão para permitir um controle fino das experiências com Yolo. Estas são armazenadas num ficheiro YAML e podem ser visualizadas ou modificadas diretamente no ambiente Python.	8.1.34
PANDAS	O Pandas é uma ferramenta essencial para trabalhar com dados em Python, sendo fundamental nas áreas de ciência de dados e análise de dados.	2.2.2

Fonte: Autoria própria

Após a definição das ferramentas e bibliotecas utilizadas, é possível compreender como cada uma delas contribui para o desenvolvimento do sistema de detecção de sonolência.

3.5.2 Definições do *Algoritmo*

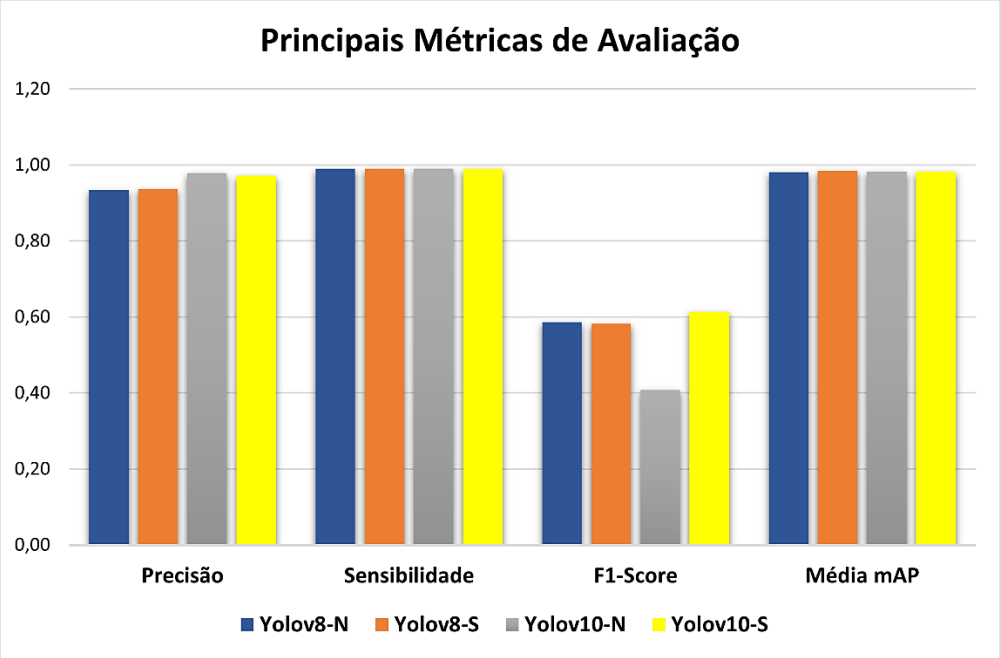
O *algoritmo* deve utilizar as detecções geradas pelo modelo escolhido para realizar a telemetria dos sinais identificados, permitindo a coleta de dados estatísticos que servirão de base para a geração dos alertas implementados. As principais funções do *algoritmo* incluem:

- Contar o número de piscadas;
- Contar o número de vezes que baixa e levanta a cabeça;
- Contar o número de vezes que há um bocejo;
- Mostrar um alerta de fadiga leve;
- Mostrar um alerta de fadiga grave;
- Mostrar um alerta que o motorista está dormindo com base no estado dos olhos;
- Mostrar um alerta que o motorista está dormindo com base no estado da cabeça;
- Contar o tempo de cada piscada;
- Contar o tempo quando baixa e levanta a cabeça;
- Validar se há um bocejo;
- Realizar ciclos de 60 segundos para o monitoramento dos alertas de sonolência;
- Salvar todos os dados das detecções em uma planilha no formato de Excel.

4 RESULTADOS E DISCUSSÕES

A Figura 15 apresenta uma comparação entre algumas das principais métricas de avaliação dos resultados obtidos durante os treinamentos dos modelos alvos deste estudo.

Figura 15 – Comparação entre os Resultados das Versões Avaliadas



Fonte: Autoria própria

Os resultados obtidos deste comparativo não são significativos para determinar qual é a melhor versão, a escolha da versão YOLOv10-S e baseada em pequenos detalhes mostrados anteriormente, como um menor tempo de treino, e uma melhor visualização das caixas delimitadoras durante testes inicias apresentados previamente na Figura 11.

4.1 ANALISE DE DESEMPENHO DO TREINAMENTO

A Figura 16 apresenta os resultados preliminares das detecções realizadas pelo sistema proposto, com o uso de imagens selecionadas do conjunto de validação. As caixas delimitadoras destacam as regiões de interesse nas imagens, onde diferentes estados de sonolência e cansaço do motorista foram detectados.

Figura 16 – Validação de Lote 2 do Modelo YOLOv10-S

Fonte: Autoria própria

Cada caixa está associada a uma etiqueta que indica o comportamento identificado, como "cabeça baixa", "olhos fechados", "bocejando" ou "sonolento", acompanhada do valor de confiança da predição. Esses valores de confiança indicam a certeza do modelo em relação à detecção feita.

Na Figura 17, são exibidos exemplos adicionais de predições realizadas pelo sistema de detecção de estados de atenção e sonolência. As imagens do conjunto de validação mostram indivíduos em diferentes situações, com as caixas delimitadoras indicando as características identificadas, como "atento" e "olhos fechados", além dos valores de confiança associados.

Figura 17 – Validação de Lote 0 do Modelo YOLOv10-S

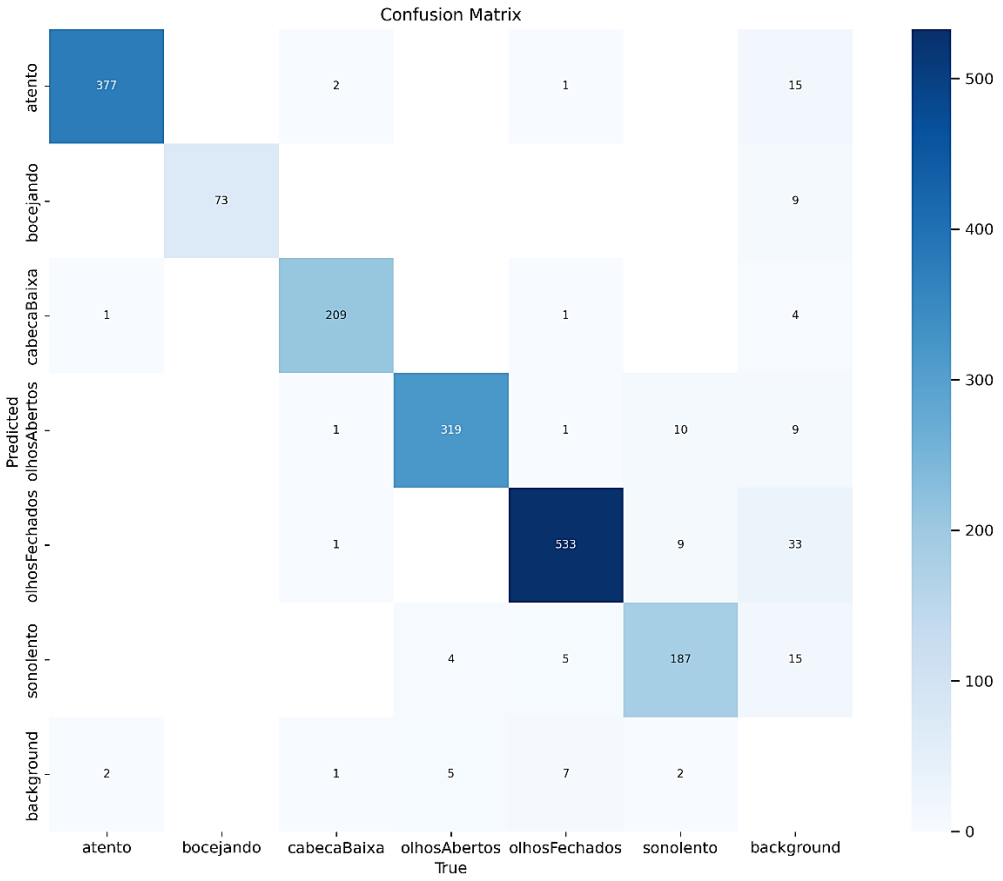
Fonte: Autoria própria

Esses resultados demonstram a capacidade do sistema em detectar múltiplos sinais de sonolência simultaneamente, mesmo em situações onde as expressões faciais e posturais são complexas. A análise desses comportamentos em conjunto permite uma detecção mais dos estados de fadiga do motorista, contribuindo para a eficácia do sistema de monitoramento. Onde evidencia a robustez do sistema em identificar diferenças no comportamento do motorista.

4.1.1 Análise por Matriz de Confusão

A matriz de confusão ilustrada na Figura 18 é utilizada para avaliar o desempenho do modelo de classificação em termos de suas classes reais e classes previstas. As previsões são organizadas em uma tabela que apresenta as contagens de acertos e erros de classificação, separando os verdadeiros positivos, falsos positivos, verdadeiros negativos e falsos negativos. Essa representação quantitativa pode revelar grandes diferenças na quantidade absoluta entre as detecções de cada classe, destacando a necessidade de uma análise cuidadosa para entender as nuances do desempenho do modelo.

Figura 18 – Matriz de Confusão do Modelo YOLOv10-S



Fonte: Autoria própria

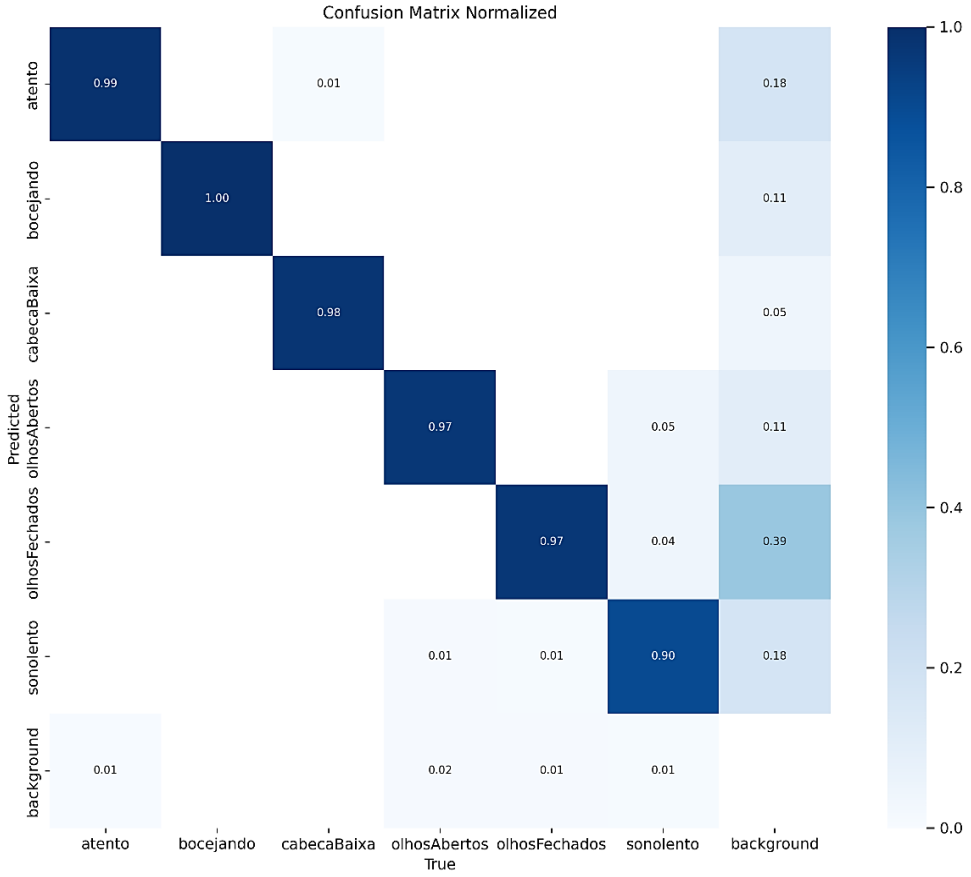
De acordo com a matriz de confusão do modelo YOLOv10-S, a classe “olhosFechados” teve o maior número de detecções. Das 548 amostras, 533 foram previstas corretamente, 5 como “sonolento”, 1 como “olhosAbertos”, 1 como “cabecaBaixa”, 1 como “atento” e 7 como *background*. A classe “bocejando” se destaca pelo fato no qual das 73 amostras, todas foram previstas corretamente. Ouras classes como “atento” das 380 amostras, 377 foram preditas corretas, 1 foi predita como “cabecaBaixa” e 2 como *background*. Classe

“cabecaBaixa”, das 213 amostras, 209 corretas, 2 como “atento”, 1 como ”olhosFechados”, 1 “olhosAbertos" e 1 como *background*. Classe “olhosAbertos”, das 328 amostras, 319 foram preditas corretamente, 4 como “sonolento” e 5 como *background*. Já a classe “sonolento”, das 208 amostras, 187 foram preditas corretamente, 10 como “olhosAbertos”, 9 como “olhosFechados” e 2 como *background*.

A Figura 19 apresenta a matriz de confusão normalizada, uma forma alternativa de demonstrar os resultados de classificação das classes. Essa técnica ajusta os valores para refletir proporções em vez de contagens absolutas, exibindo-os como percentuais em relação ao total de cada classe. Tal abordagem é particularmente útil em cenários de desbalanceamento entre as classes, ou seja, quando há uma quantidade significativamente maior de exemplos de uma classe em comparação às demais. A matriz de confusão normalizada oferece uma interpretação mais clara do desempenho em cada classe, proporcionando uma análise mais precisa dos resultados de classificação.

Os resultados de precisão para cada classe foram os seguintes: a classe "Atento" obteve uma precisão de 99%, enquanto a classe "Bocejando" apresentou 100% de precisão. A classe "Cabeça Baixa" alcançou uma precisão de 98%, seguida pelas classes "Olhos Abertos" e "Olhos Fechados", ambas com 97% de precisão. Por fim, a classe "Sonolento" registrou uma precisão de 90%.

Figura 19 – Matriz de Confusão Normalizada do Modelo YOLOv10-S



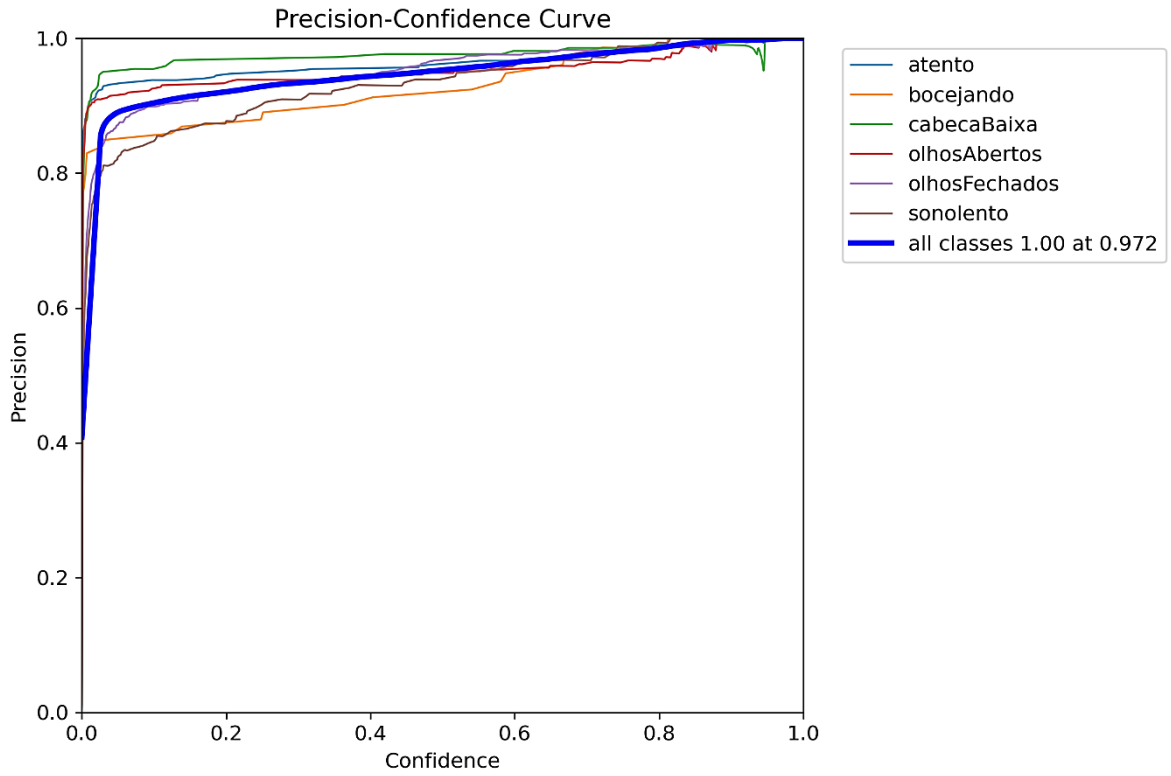
Fonte: Autoria própria

A classe “bocejando” destacou-se positivamente, apresentando 100% das amostras preditas corretamente. Isso se deve ao fato de que, para essa classe, não há uma classe de contraste que indique se a pessoa está ou não bocejando. Em contrapartida, a classe “sonolento” obteve o pior índice de predições corretas, pois está entre as classes “olhosAbertos” e “olhosFechados”, o que pode resultar em detecções incorretas. No entanto, o índice ainda é satisfatório, com um percentual de acertos de 90%.

4.1.2 Análise por Curva de Precisão (*Precision*)

Curva de precisão apresentada na Figura 20 mede a proporção de detecções corretas em relação a detecções feitas. Isso indica a confiança do modelo em suas predições e retrata a variação na precisão da previsão do modelo em vários níveis de confiança.

Figura 20 – Curva de Precisão-Confiança do Modelo YOLOv10-S



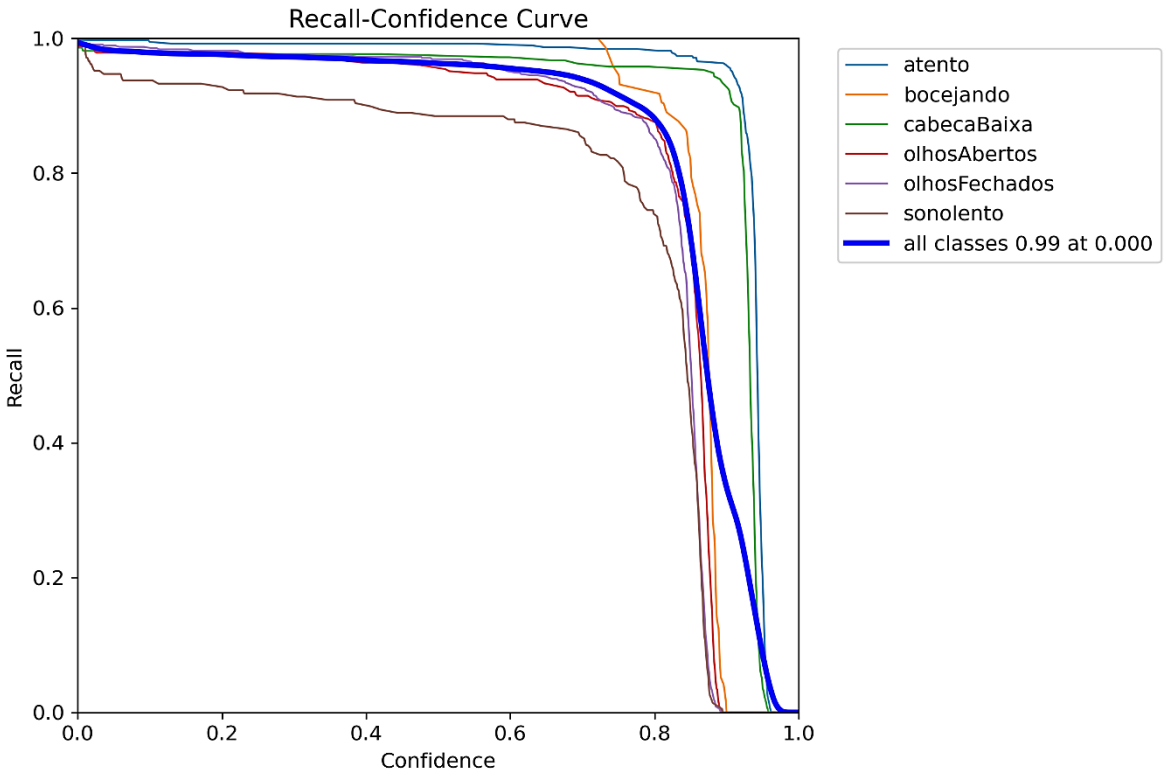
Fonte: Autoria própria

À medida que a confiança aumenta a precisão das detecções também aumenta, devido a maior confiança do modelo em fazer previsões confiáveis. Notavelmente, quando o limite de confiança atinge 0,972, a precisão para todas as categorias atinge 1,0. Quando o nível de confiança está entre e aproximadamente de 0,8 a 0,9, a precisão da previsão atinge um pico antes de apresentar uma tendência de queda.

4.1.3 Análise por Curva de Sensibilidade (*Recall*)

Curva de sensibilidade mede a proporção de objetos verdadeiros corretamente detectados pelo modelo e demostrar a variação nas taxas de sensibilidade em diferentes limites de confiança. Essa métrica indica a capacidade do modelo de detectar todos os objetos presentes na imagem. Reflete as variações nas taxas de recall à medida que os níveis de confiança flutuam.

Figura 21 – Curva de Sensibilidade-Confiança do Modelo YOLOv10-S



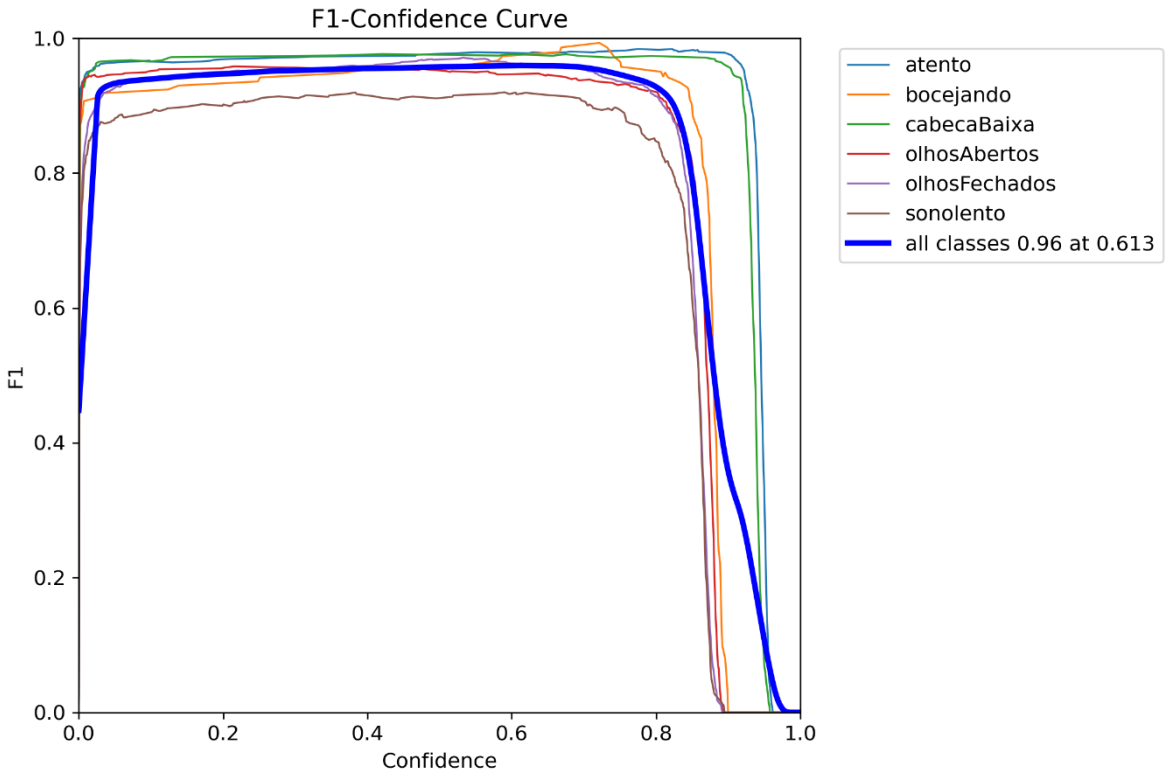
Fonte: Autoria própria

Conforme representado na Figura 21, as taxas de sensibilidade para várias categorias com um nível de confiança zero no treinamento do modelo atual são de 0,99. Com o aumento dos níveis de confiança, as taxas de sensibilidade diminuem gradualmente. Quando a curva se aproxima do canto superior direito, significa que o modelo pode manter alto sensibilidade e alta precisão. Isso indica melhor desempenho do modelo, pois ele pode manter alto sensibilidade e ao mesmo tempo manter maior precisão. Uma taxa de sensibilidade mais alta implica uma capacidade mais forte do modelo para identificar amostras positivas. Isto destaca o bom desempenho deste modelo nas tarefas de identificação de sinais de sonolência.

4.1.4 Análise por F1-*Score*

O F1-*Score* é a média harmônica entre a precisão e a sensibilidade, proporcionando um único valor que balanceia as duas métricas. Ele avalia o desempenho do modelo equilibrando *recall* e precisão. Uma pontuação F1 mais alta indica um modelo mais robusto. Como mostrado na Figura 22.

Figura 22 – Curva de Pontuação F1 do Modelo YOLOv10-S



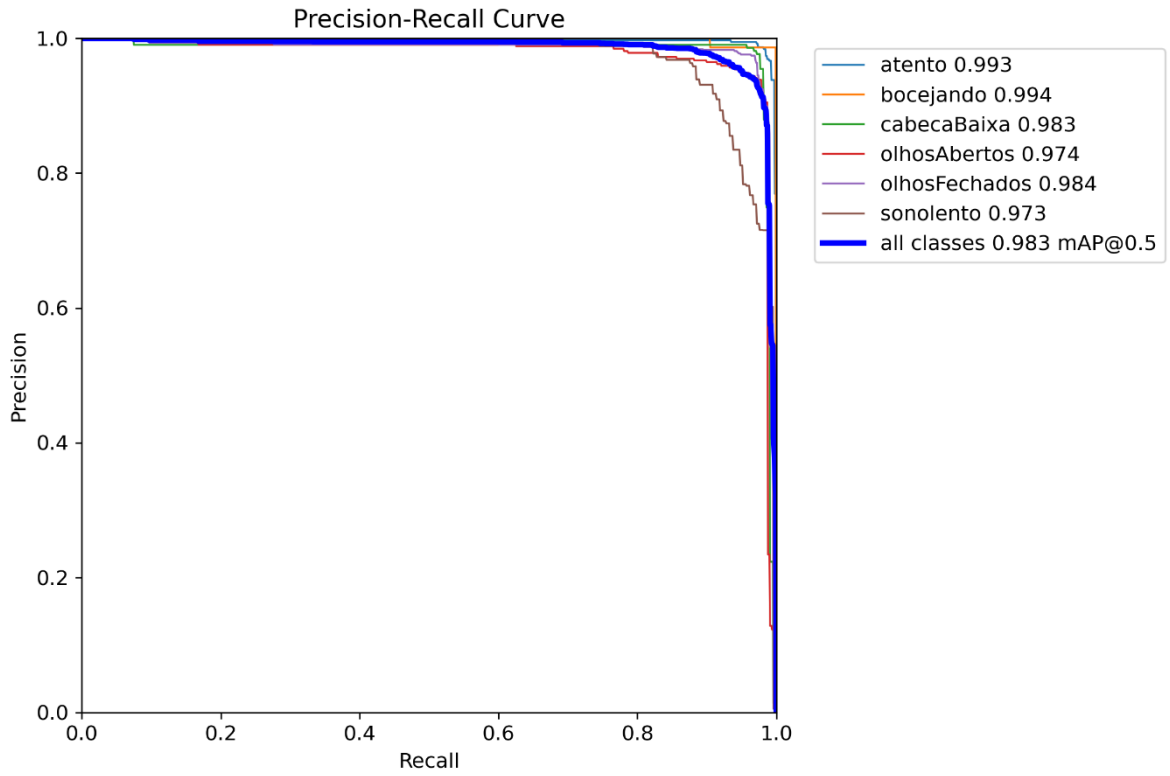
Fonte: Autoria própria

A pontuação F1 atinge seu valor mais alto de 0,96 quando o limite de confiança para o treinamento deste modelo é definido em 0,613, o que indica uma confiança relativamente boas. As curvas com picos mais altos e mais à direita indicam um melhor desempenho do modelo, pois combinam alto F1-*Score* com maior confiança.

4.1.5 Análise por Precisão Média (MAP – *Mean Average Precision*)

A MAP é a principal métrica usada para avaliar modelos de detecção de objetos. Ela representa a média das precisões para diferentes valores de sensibilidade em várias classes de objetos. À medida que o limite de classificação muda, a precisão e a recuperação também mudam. A Figura 23 ilustra as curvas PR do modelo proposto. A análise mostra que as curvas PR do modelo proposto, estão próximas ao canto superior direito, indicando a alta recuperação e precisão do modelo.

Figura 23 – Curva de Precisão-Sensibilidade do Modelo YOLOv10-S



Fonte: Autoria própria

A área sob as curvas PR, por ser relativamente grande, sugere um bom desempenho do modelo. O valor médio de precisão deste modelo no limite mAP@0,5 é 0,983, indica que o modelo pode garantir simultaneamente alta precisão e recuperação neste limite. Ele pode prever com precisão todas as classes, fornecendo assim resultados de previsão bastante precisos.

4.1.6 Análise de Perdas e Métricas

A Figura 24 mostra diferentes componentes de perdas para o modelo YOLOv10-S, significam perdas de caixas, perda de aprendizado de recurso dinâmico e perda de classificação. O eixo horizontal significa a progressão do treinamento e o eixo vertical mostra os valores de perdas acumuladas ao longo de todo o período de treinamento.

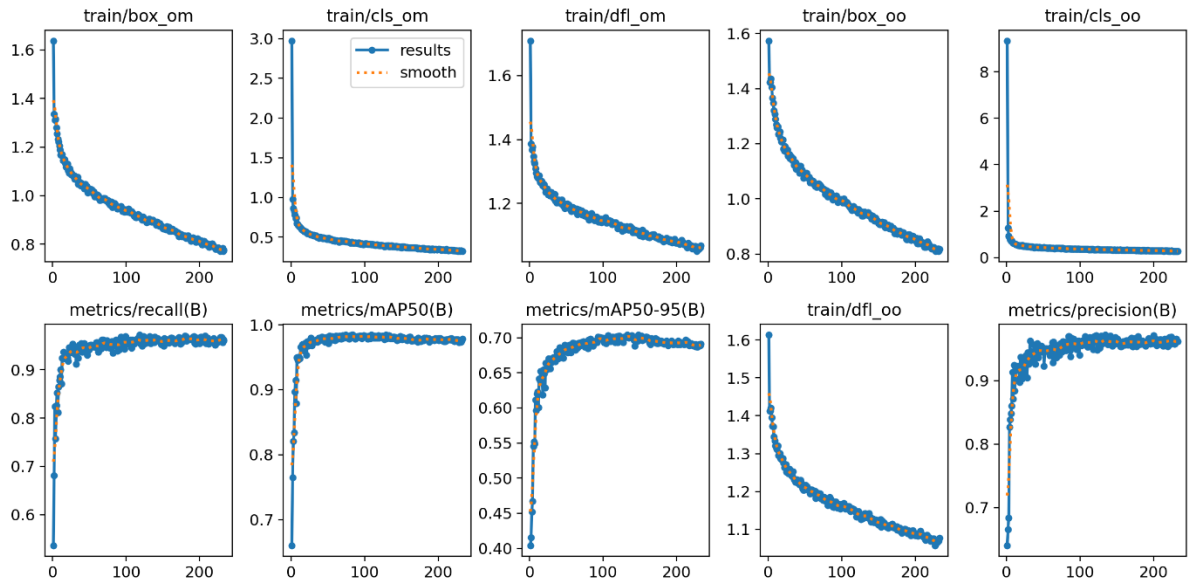
A curva *train/box_om* mostra como a perda de regressão da caixa delimitadora acumulada muda ao longo do treinamento do modelo, está ligado a perda da qualidade da predição das caixas delimitadoras. Mede o desvio entre a caixa delimitadora prevista com a caixa real. O gráfico com tendência decrescente mostra que o modelo está aprendendo

e melhorando ao longo do tempo, alcançando uma boa performance na predição das caixas delimitadoras.

A curva *train/cls_loss* representa a perda de classificação do modelo durante o treinamento, refletindo o nível de erro na classificação da categoria predita. A curva decrescente observada demonstra uma diminuição rápida dos erros de classificação para as categorias preditas durante a fase de treino. Indica uma melhora na sua capacidade de classificar corretamente os objetos preditos melhorando progressivamente ao longo do processo de aprendizagem.

A curva *train/dfl_loss* refere-se a perda de aprendizado de recursos dinâmicos ao longo do processo de treinamento do modelo. Essa métrica mede o erro na previsão das coordenadas das caixas delimitadoras. A tendência decrescente desta curva sugere que o modelo se torna mais eficaz em prever caixas delimitadoras, reduz o erro nas coordenadas das caixas, minimizando os erros de forma consistente.

Figura 24 – Gráficos de Perdas e Métricas do Modelo YOLOv10-S



Fonte: Autoria própria

A curva *train/box_oo* mede a perda de caixa delimitadora com foco em *outliers*. Isso pode ocorrer quando há predições que são significativamente distantes das caixas reais ou quando há anomalias nas coordenadas da caixa. No início do treinamento o modelo faz previsões imprecisas, resultado em perdas altas, durante o treinamento as perdas começam a cair, melhorando o desempenho em prever as caixas mais precisamente, no final do

treinamento a curva atinge um platô baixo, o que significa que o modelo está conseguindo prever caixas delimitadoras com alta precisão.

A curva *train/cls_oo* refere-se à perda de classificação em relação às previsões incorretas, associadas a *outliers*. Isso ocorre quando as classes previstas são erradas em comparação com as classes reais. O gráfico se mostra estável durante todo o treinamento, isso mostra que o desempenho em fazer classificações corretas e bastante preciso, mesmo em casos de *outliers*.

A curva *train/dfl_oo* está relacionada à perda de distribuição focalizada das caixas delimitadoras, principalmente em *outliers*. A tendência decrescente do gráfico mostra que o modelo evolui durante o tempo, diminuindo gradualmente os erros até atingir uma estabilização, chegando na sua capacidade máxima de previsão das caixas delimitadoras.

A curva *metrics/recall*(B) é a capacidade do modelo de detectar corretamente todos os objetos reais presentes em cada *batch*. O gráfico mostra uma curva crescente com rápida estabilização e valores com média acima de 0.9, isso indica que o modelo detecta com precisão os sinais de sonolência me cada *batch.*

A curva *metrics/*mAP50(B) é a precisão média calculada considerando que uma predição correta entre a caixa predita e a caixa real é maior que 0.5, em cada *batch*. O gráfico indica valores elevados em cada *batch*, isso indica que o modelo fez boas predições em termos de localização dos sinais de sonolência.

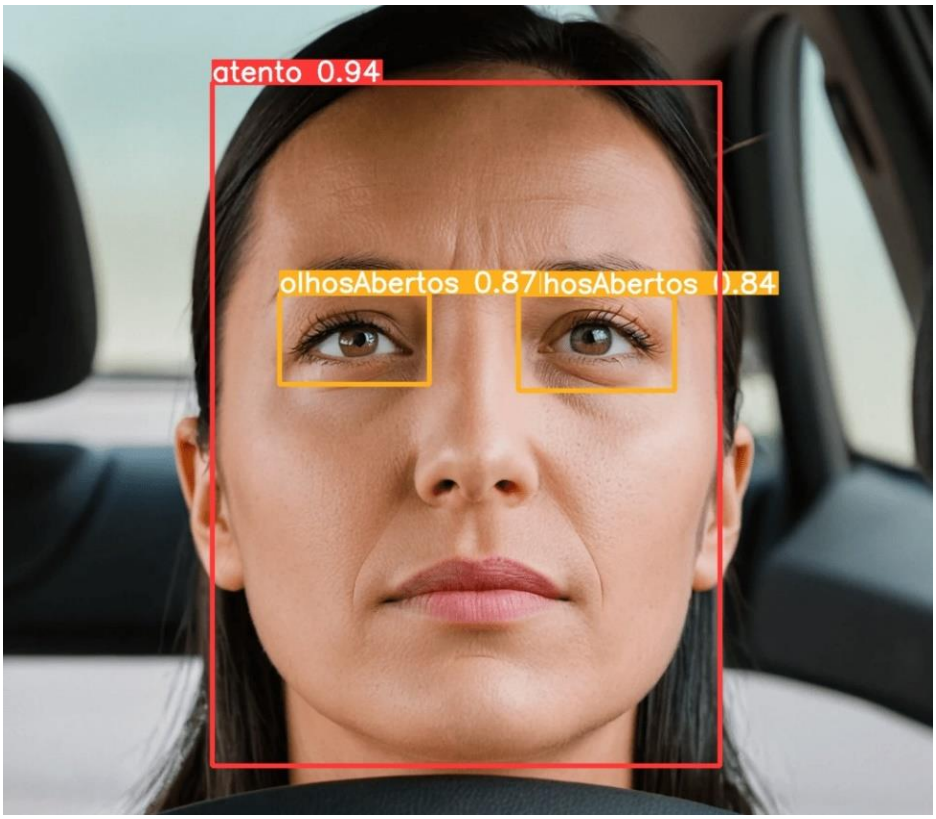
A curva *metrics*/mAP50-95(B) refere-se à média da precisão média *(Mean Average Precision* - mAP) calculada em diferentes limiares, variando de 0.5 a 0.95, em incrementos de 0.05. E leva em consideração tanto a localização quanto a precisão das predições das caixas delimitadoras. Sendo um limiar mais rigoroso, onde a caixa predita deve ter ao menos 95% de sobreposição, para ser considerada uma predição correta. O gráfico mostra um bom desempenho em localização dos sinais de sonolência, equilíbrio entre precisão e sensibilidade, e uma certa robustez do modelo, funcionando bem em diferentes situações de tamanho e complexidade das imagens.

A curva *metrics/Precision*(B) mede a proporção de detecções corretas (verdadeiros positivos) em relação ao total de detecções (verdadeiros positivos + falsos positivos) para cada batch. O gráfico indica que o modelo está fazendo boas predições por *batch*, com poucas detecções incorretas.

4.2 RESULTADO DOS TESTES DE DETECÇÃO DE SINAIS DE SONOLÊNCIA

Os primeiros testes foram realizados inicialmente com imagens, para demostrar a eficácia do treinamento. As detecções mostram a caixa delimitadora, e seus respectivos níveis de confiança para aquela detecção. As classes são diferenciadas por cores, facilita a visualização dos resultados. Na Figura 25 as detecções são “atento” com confiança de 0.94, os olhos são detectados e classificados individualmente, isso nos dá diferentes níveis de confiança, ainda na Figura 25 os olhos da esquerda e da direita foram classificados como “olhosAbertos” e com confiança de 0.87 e 0.84, respectivamente. Na Figura 26 a quatro classificações, “atento” com confiança de 0.93, “bocejando” com confiança de 0.66 e olhos esquerdo e direito como “sonolento” onde só e possível visualizar o rotulo dos olhos esquerdo com confiança de 0.78, isso se dá devido as diferentes dimensões das imagens.

Figura 25 – Teste 1 de Detecção de Sinais de Sonolência

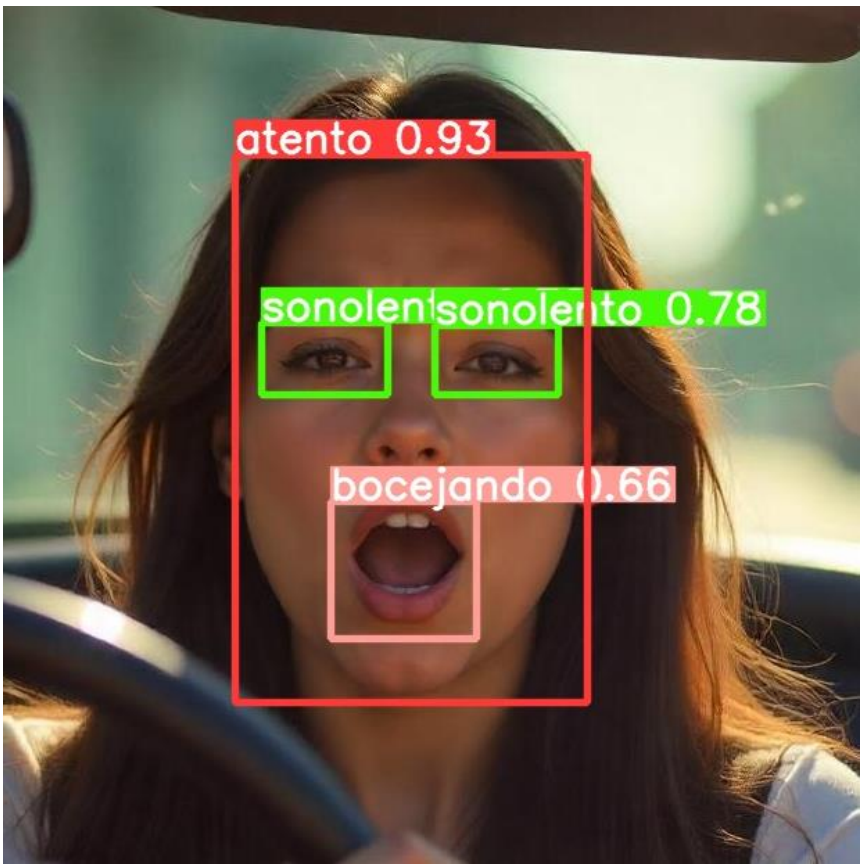


Fonte: Autoria própria

Na Figura 26 a quatro classificações, “atento” com confiança de 0.93, “bocejando” com confiança de 0.66 e olhos esquerdo e direito como “sonolento” onde só e possível visualizar o rotulo dos olhos esquerdo com confiança de 0.78, isso se dá devido as diferentes

dimensões das imagens. Figura 25 com dimensões de 654x572 *pixels* e Figura 26 com dimensões de 642x644 *pixels*.

Figura 26 – Teste 2 de Detecção de Sinais de Sonolência



Fonte: Autoria própria

Os testes realizados demonstraram altos níveis de confiança do modelo na predição de sinais de sonolência. Esses resultados indicam que o modelo é capaz de identificar com precisão os momentos em que o indivíduo apresenta sinais de fadiga, permitindo a implementação de medidas preventivas para evitar acidentes.

4.3 ANÁLISE DOS RESULTADOS COM O ALGORITMO EM TEMPO REAL

O algoritmo desenvolvido para a detecção de sonolência realiza uma análise quantitativa detalhada dos sinais de fadiga, levando em consideração tanto o domínio da frequência quanto o do tempo. Ele monitora a frequência de piscadas, bocejos e movimentos da cabeça. Cruzando esses dados com parâmetros previamente estabelecidos com base em estudos científicos sobre os níveis de fadiga em motoristas.

Através dessa análise, o sistema consegue identificar padrões que indicam diferentes estágios de sonolência, desde sinais iniciais até situações de alto risco, em tempo real. Com isso, o algoritmo gera dois tipos de alertas principais, que são **Alerta de Fadiga** (dividido em leve e grave) e **Alerta Dormindo**, cada um indicando diferentes níveis de risco.

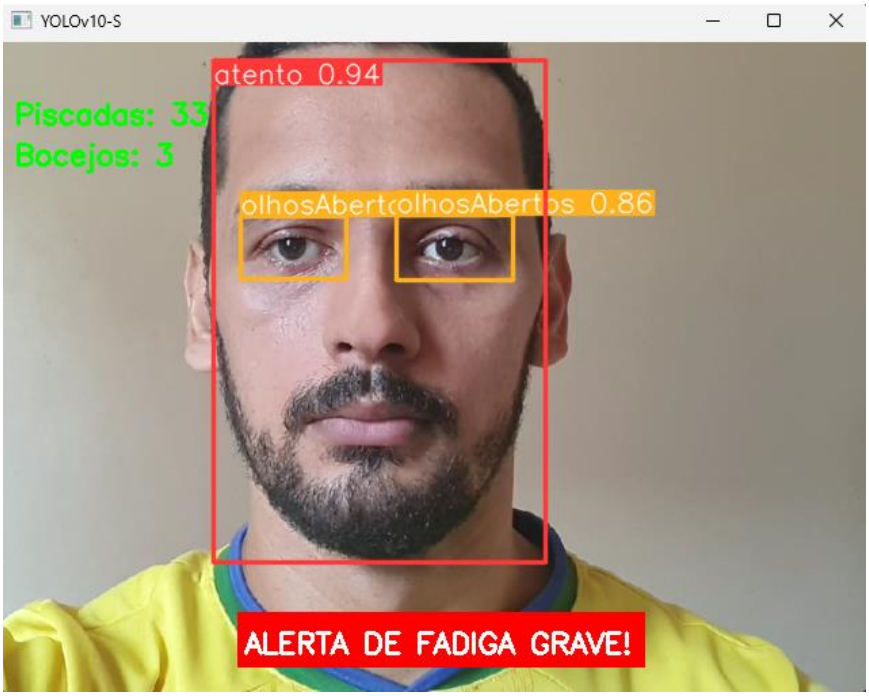
A Figura 27 ilustra o **Alerta de Fadiga Leve**, que é ativado quando o sistema detecta que os sinais iniciais de fadiga foram alcançados. Esse alerta ocorre quando um dos indicadores de sonolência, como o número de piscadas ou bocejos, ultrapassa os limites estabelecidos de menos de 30 piscadas e menos de 2 bocejos dentro de um intervalo de 60 segundos. Esse nível de alerta indica que o motorista começa a apresentar cansaço, mas ainda está em um estágio leve de fadiga, requer atenção para evitar uma progressão para níveis mais graves [18].

Figura 27 – Alerta de Fadiga Leve

Fonte: Autoria própria

A Figura 28 ilustra o **Alerta de Fadiga Grave**, que é ativado quando o sistema detecta que ambos os indicadores de fadiga, o número de piscadas e de bocejos, ultrapassam os limites estabelecidos. Esse alerta é acionado quando há mais de 30 piscadas e mais de 2 bocejos dentro de um intervalo de 60 segundos. Essa situação indica que o motorista já está em um nível avançado de fadiga, apresenta um risco elevado de adormecer ao volante, exigindo uma ação imediata para evitar acidentes [18].

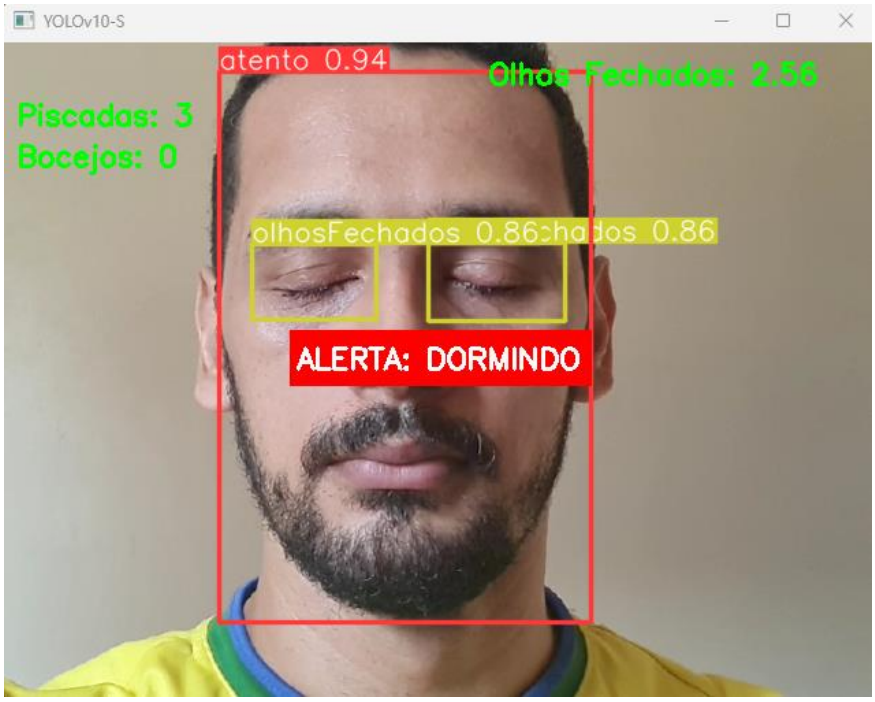
Figura 28 – Alerta de Fadiga Grave



Fonte: Autoria própria

Na Figura 29, o sistema dispara o **Alerta: Dormindo** após detectar que os olhos do motorista permaneceram fechados por mais de 2 segundos consecutivos. Este alerta indica que o motorista está adormecendo ao volante, o que representa um risco crítico à segurança.

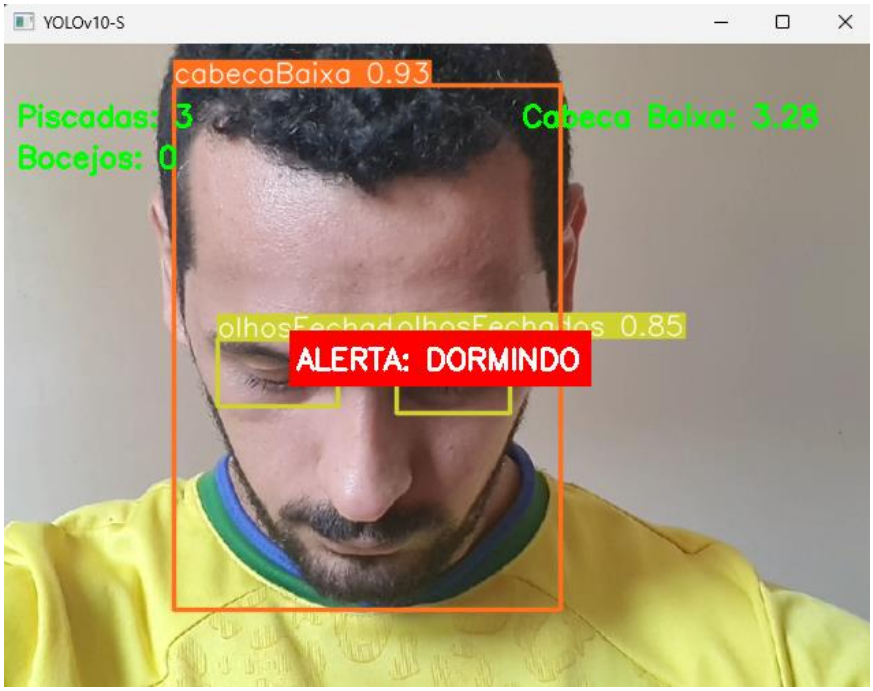
Figura 29 – Alerta Dormindo com Base no Estado dos Olhos



Fonte: Autoria própria

A Figura 30 mostra o **Alerta: Dormindo,** que é acionado quando a cabeça do motorista permanece abaixada por mais de 2 segundos. Essa postura é um forte indicativo de sonolência profunda ou até de perda de consciência momentânea. Assim como no caso dos olhos fechados, este alerta exige uma ação imediata, o recomendado seria a parada do veículo.

Figura 30 – Alerta Dormindo com Base no Estado da Cabeça



Fonte: Autoria própria

Após a execução do código, um documento é gerado automaticamente, conforme exemplificado na Tabela 5. Esses eventos foram capturados em tempo real, porém, em um ambiente de teste controlado, permite a avaliação precisa do sistema sem influências externas. Esse documento contém informações cruciais para o monitoramento e análise dos eventos de sonolência detectados. Cada linha do relatório inclui:

- **Nome do arquivo**: Referente ao vídeo, para fins de identificação de cada inferência .
- **Evento detectado**: O tipo de evento identificado, como piscadas, bocejos, olhos fechados ou movimentos da cabeça.
- **Data e hora**: O registro exato de quando o evento foi detectado, permitindo um acompanhamento temporal detalhado.
- **Duração (em segundos)**: O tempo total em que o evento foi detectado, indicando o nível de sonolência. Isso é especialmente relevante em casos de eventos críticos, como "olhos fechados" ou "cabeça baixa".
- **Nível de fadiga**: A classificação do estado de fadiga do motorista, que pode variar entre **leve**, **média**, **forte** ou **dormiu**, conforme os parâmetros definidos pelo sistema.

Esses dados são organizados em formato de tabela para facilitar a análise posterior, permitindo que seja feita uma avaliação do comportamento do motorista durante o período monitorado.

Tabela 5 – Dados Gerados em Simulação de Tempo Real

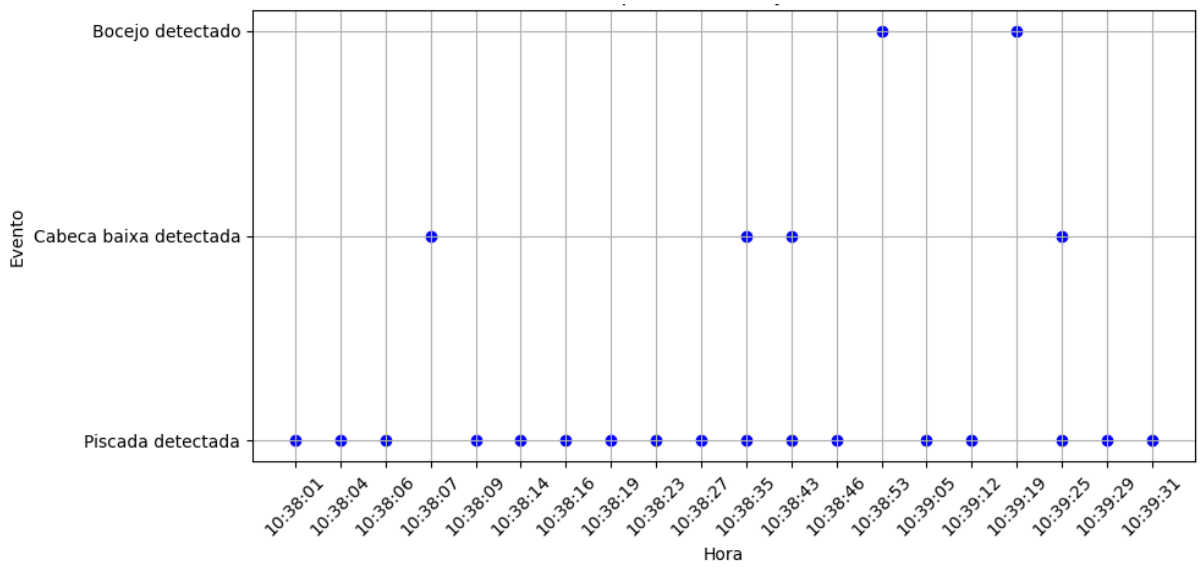
Arquivo	Evento	Data	Hora	Duração	Nível
teste02	Piscada detectada	05/09/2024	10:38:01	0.70	Leve
teste02	Piscada detectada	05/09/2024	10:38:04	1.22	Media
teste02	Piscada detectada	05/09/2024	10:38:06	0.63	Leve
teste02	Cabeça baixa detectada	05/09/2024	10:38:07	1.89	Leve
teste02	Piscada detectada	05/09/2024	10:38:09	0.65	Leve
teste02	Piscada detectada	05/09/2024	10:38:14	0.63	Leve
teste02	Piscada detectada	05/09/2024	10:38:16	0.61	Leve
teste02	Piscada detectada	05/09/2024	10:38:19	1.71	Forte
teste02	Piscada detectada	05/09/2024	10:38:23	1.73	Forte
teste02	Piscada detectada	05/09/2024	10:38:27	2.31	Dormiu
teste02	Piscada detectada	05/09/2024	10:38:35	3.12	Dormiu
teste02	Cabeça baixa detectada	05/09/2024	10:38:35	3.12	Dormiu
teste02	Piscada detectada	05/09/2024	10:38:43	2.47	Dormiu
teste02	Cabeça baixa detectada	05/09/2024	10:38:43	2.47	Forte
teste02	Piscada detectada	05/09/2024	10:38:46	0.67	Leve
teste02	Bocejo detectado	05/09/2024	10:38:53	5.62	Leve
teste02	Piscada detectada	05/09/2024	10:39:05	0.67	Leve
teste02	Piscada detectada	05/09/2024	10:39:12	0.61	Leve
teste02	Bocejo detectado	05/09/2024	10:39:19	5.20	Leve
teste02	Piscada detectada	05/09/2024	10:39:25	3.10	Dormiu

Fonte: Autoria própria

Esse registro detalhado de cada evento detectado pelo sistema permite um acompanhamento em tempo real e também facilita auditorias e revisões futuras, garante que cada situação de risco seja devidamente documentada e tratada de acordo com a gravidade do alerta.

Na Figura 31 o gráfico exibe a detecção de três tipos de eventos relacionados à sonolência do motorista ao longo do tempo: **bocejos**, **cabeça baixa** e **piscadas**. A análise é feita no eixo do tempo (hora), com os eventos plotados em intervalos específicos. A quantidade significativa de piscadas pode ser um indicador inicial de fadiga leve, o que está de acordo com os critérios de alerta do sistema. A detecção de bocejos, ainda que espaçada, pode indicar que o motorista está caminhando para um estado de sonolência mais avançado.

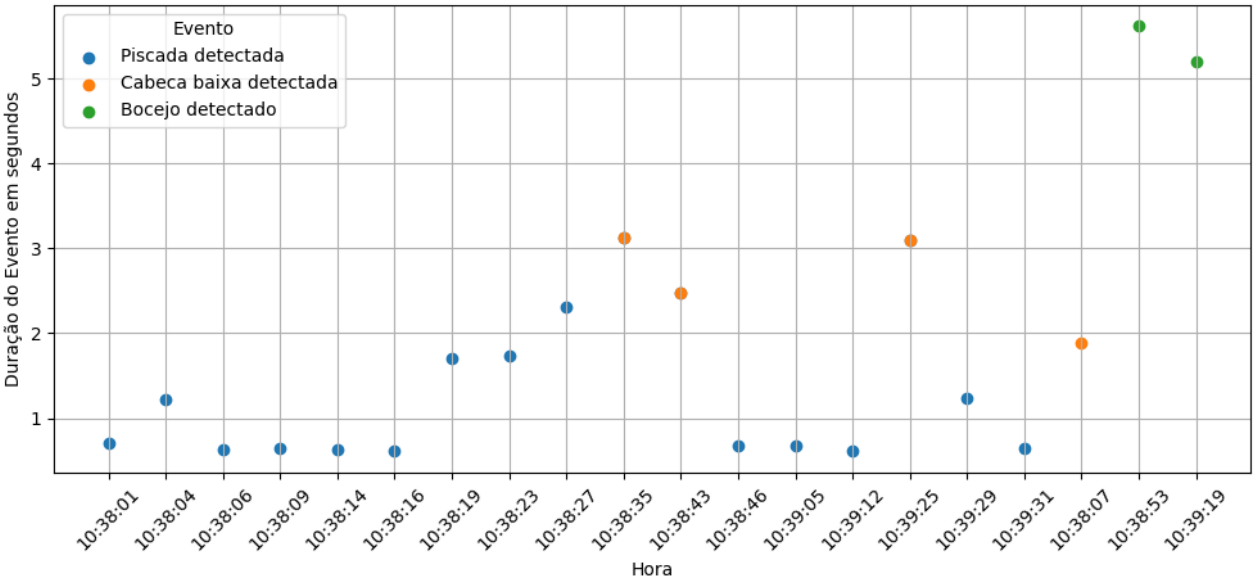
Figura 31 – Linha do Tempo com a Detecção de Eventos



Fonte: Autoria própria

A Figura 32 apresenta uma análise da duração de cada evento detectado. Observa-se que os eventos de piscadas geralmente têm uma duração mais curta. Por outro lado, eventos de cabeça baixa podem ser um sinal significativo de sonolência. Além disso, a duração dos bocejos é naturalmente maior em comparação às demais detecções, uma vez que estudos indicam que um bocejo é considerado quando sua duração atinge aproximadamente 5 segundos.

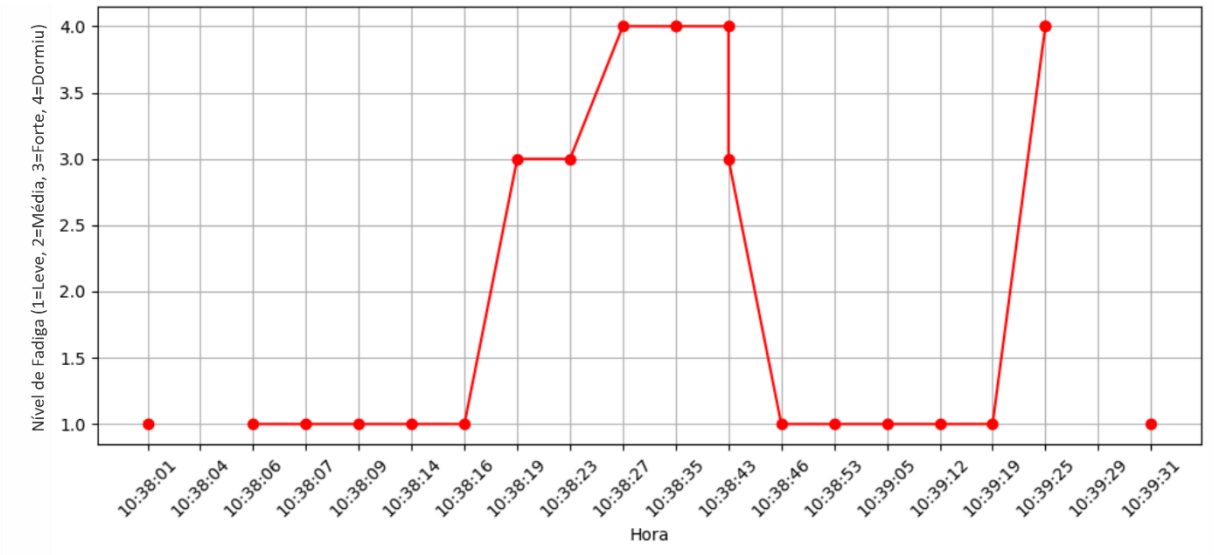
Figura 32 – Duração dos Eventos ao Longo da Linha do Tempo



Fonte: Autoria própria

A Figura 33 apresenta uma análise dos níveis de fadiga ao longo da série temporal, categorizados em quatro níveis distintos. Cada nível é determinado com base na duração dos eventos detectados, o que permite uma avaliação mais precisa do estado de alerta do motorista.

Figura 33 – Linha do Tempo com os Níveis de Fadiga



Fonte: Autoria própria

Essa análise permite não apenas identificar momentos de atenção plena, mas também reconhecer padrões de fadiga que podem comprometer a segurança durante a condução. Essa abordagem quantitativa proporciona uma compreensão mais profunda do comportamento do motorista ao longo do tempo, facilitando a implementação de intervenções apropriadas para mitigar riscos associados à sonolência.

5 CONCLUSÃO

O presente trabalho alcançou com sucesso seus objetivos. Embora o modelo não tenha sido testado em condições reais, os testes simulados apresentaram resultados bastante satisfatórios, demonstrou a eficiência da aprendizagem de máquina na detecção precisa de sinais de sonolência. O sistema desenvolvido mostrou-se robusto, sendo capaz de identificar sinais de fadiga com alta precisão e eficiência, permitiu a criação de um mecanismo de alerta automático. Este sistema detecta e avalia os sinais de sonolência do motorista, gerando alertas proporcionais à gravidade dos sinais identificados, contribuindo diretamente para a redução de acidentes de trânsito.

Os parâmetros relacionados aos sinais de sonolência, como o tempo de ocorrência de uma ação ou a repetição em intervalos específicos, ainda são temas de estudo contínuo. Foi realizado um ajuste detalhado desses parâmetros, visa maximizar a precisão na detecção de sonolência, dada a importância do sistema para salvar vidas e aumentar a segurança no trânsito. Mesmo com a possibilidade de alterações nos parâmetros, o sistema mantém grande flexibilidade e pode ser facilmente ajustado em sua programação para novas condições.

O sistema desenvolvido demonstrou alta precisão na identificação de sinais de fadiga em motoristas durante os testes simulados, validando a eficácia do modelo de aprendizagem de máquina utilizado. Foi implementado um mecanismo de alerta automático, capaz de gerar notificações proporcionais à gravidade dos sinais detectados, contribuindo para a prevenção de acidentes de trânsito.

Portanto, fica evidente a necessidade de aprofundar os estudos sobre os estágios do sono durante a condução. Isso permitirá o fortalecimento contínuo do sistema de alertas, garantindo um comportamento mais preciso do modelo de detecção em diversas situações de sonolência no trânsito.

TRABALHOS FUTUROS

Como continuidade deste trabalho, propõe-se a expansão e robustez do modelo de treinamento por meio de um banco de dados maior e mais diversificado, além de ajustes finos nos parâmetros de detecção. Após essas melhorias, sugere-se a implementação prática do sistema em veículos, com o uso de câmeras para capturar os sinais de fadiga em tempo real e emitir alertas sonoros ou luminosos para os motoristas. A automatização desse processo pode ser expandida para incluir o controle do veículo, como a redução automática de velocidade em situações críticas.

REFERÊNCIAS

[1] SARAGIOTTO, D. **Mortes no Trânsito: Tráfego brasileiro mata 1 pessoa a cada 15 minutos**. Estadão. 15 de setembro de 2020.

[2] MURIALDO, M. **Em 2020, 80 pessoas morreram por dia em consequência de acidente de trânsito no país. Portal do Trânsito**. 25 de dezembro de 2020.

[3] MELLO, M.T. **Trabalhador em turno: fadiga**: São Paulo, 2013.

[4] LYZNICKI, J.M., et al., 2008. **Sleepiness, driving, and motor vehicle crashes**. Jornal of the Americam Médical Association 279, 1908-1913.

[5] HORNE, J., REUNER, L., **falling asleep as the whel, report TRL, 168**. Transport Research Laboratory, Crowthorne. 2010.

[6] RODRIGUES, Danilo Nogueira. **Detecção de Fadiga Baseada no Monitoramento dos Olhos**. 2018. 34 f. Mongrafia - Universidade Federal do Maranhão, São Luiz, 2018. Disponível em: <http://hdl.handle.net/123456789/3574>.

[7] GARCÍA, I. et al. **Vision-based drowsiness detector for a realistic driving simulator**. In: IEEE. Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on. [S.l.], 2010. p. 887–894.

[8] SILVA, Leonardo Dorneles Figueiredo. **Monitoramento do estado de sonolência de motoristas de automóveis através de análise de imagens de olhos** . 2012. 74 f. Dissertação (Mestrado em Engenharia) - Universidade Federal do Maranhão, São Luís, 2012. Disponível em: <http://tedebc.ufma.br:8080/jspui/handle/tede/478>

[9] ROQUE, P. M. d. S. **Técnicas de visão computacional para a deteção automática de padrões de fadiga**. Dissertação (Mestrado) — Universidade de Évora, 2013.

[10] FATIGUE, D. **Road accidents: A literature review and position paper**. The Royal Society for the Prevention of Accidents, 2001.

[11] ABRAMET. **Sono e Direção não combinam**. 2019. Associação Brasileira de Disponível em: <https://www.abramet.com.br/repo/public/commons/Jornal%20Medicina%20de%20Tr%C3%A1fego%20-%20Abril.pdf>. Acesso em 26 de setembro de 2024

[12] TESLA. **Aviso de Sonolência do Condutor**. 2024. Disponivel em: <https://www.tesla.com/ownersmanual/model3/pt_pt/GUID-65BF21B8-50C5-4FA5-86A4-DA363DCD0484.html>. Acesso em 26 de setembro de 2024.

[13] FORMIGHERI, Samuel. **Detecção de sonolência em motoristas utilizando processamento de imagens.** Trabalho de Conclusão de Curso. Engenharia de Computação na Área do Conhecimento de Ciências Exatas e Engenharias da Universidade de Caxias do Sul. 2022. Disponível em: <https://repositorio.pucgoias.edu.br/jspui/handle/123456789/1667>.

[14] JÚNIOR, Wagner José dos Santos. **Visão Computacional na Detecção de Fadiga em Operadores de Máquinas em Trabalho Noturno Modelo de Interface Homem Maquina**. 2015. 47 f. Dissertação (Mestrado em Tecnologia da Informação aplicada a Biologia Computacional). Faculdade Promove de Tecnologia, Belo Horizonte, 2015.

[15] PARANHOS, Vinícius Kerber. **Sistema de detecção de sonolência**. 2019. 66 f. Trabalho final de graduação (Engenheiro Eletricista). Curso de Engenharia Elétrica. Universidade de Passo Fundo, Passo Fundo, RS, 2019.

[16] Lima, Mariana Ruivo Rabello de. **Sistema Embarcado para Detecção e Alerta de Atitudes de Desatenção ao Dirigir**. 2021. Trabalho de Conclusão de Curso. Escola de Engenharia (EE).Universidade Presbiteriana Mackenzie. 2021. Disponível em: <https://dspace.mackenzie.br/handle/10899/29107>

[17] CIMIRRO, Jean Lucas da Silva. **Reconhecimento de imagens: Uso do método Yolo no reconhecimento de placas de trânsito**. 64p. 2022. Trabalho de Conclusão de Curso (Bacharel em Engenharia da Computação) – Universidade Federal do Pampa, Curso de Ciência da Computação, Bagé, 2022. Disponível em: <https://repositorio.unipampa.edu.br/jspui/handle/riu/7527>.

[18] X. -X. Tang and P. -Y. Guo, "**Fatigue Driving Detection Methods Based on Drivers Wearing Sunglasses**," in IEEE Access, vol. 12, p. 70946-70962, 2024, doi: 10.1109/ACCESS.2024.3394218.

[19] GONZAGA, A. (2010). **Notas da aula em Visão Computacional** (Disciplina Pós-Graduação). Universidade de São Paulo, Escoa de Engenharia de São Carlos, Despertamento de Engenharia Elétrica - Laboratório de visão computacional (USO/EESC/SEL/LAVI) - São Carlos, São Paulo - Brasil.

[20] SZELISKI, Richard. **Computer Vision: Algorithms and Applications**. New York, 2010.

[21] KELLEHER, J**. Deep Learning**. MIT Press, 2019. (MIT Press Essential Knowledge series). ISBN 9780262537551. Disponível em: <https://books.google.com.br/books?id=b06qDwAAQBAJ>.

[22] Jähne, B., Haubecker, H., **Computer Vision and Applications**, Academic Press, 2000.

[23] Forsyth, D.A. and Ponce, J. (2002) **Computer Vision: A Modern Approach**. Prentice Hall.

[24] LUGER, G. **Inteligência Artificial.** PEARSON BRASIL, 2013. ISBN 9788581435503.
Disponível em:<https://books.google.com.br/books?id=UNEKvQEACAAJ>.

[25] ALPAYDIN, E. **Introduction to Machine Learning**. MIT Press, 2020. (Adaptive Computation and Machine Learning series). ISBN 9780262043793. Disponível em: <https://books.google.com.br/books?id=tZnSDwAAQBAJ>.

[26] BRAGA, A. de P.; CARVALHO, A. de L. F.; LUDERMIR, T. **Redes neurais artificiais: teoria e aplicações**. Livros Técnicos e Científicos, 2000. ISBN 9788521612186.
Disponível em: <https://books.google.com.br/books?id=cUgEaAEACAAJ>.

[27] HAYKIN, S. **Redes Neurais: Princípios e Prática**. 2. ed. Artmed, 2007. ISBN 9788577800865. Disponível em: <https://books.google.com.br/books?id=bhMwDwAAQBAJ>.

[28] LEAL, Adriano Galindo. **Sistema para determinação de perdas em redes de distribuição de energia elétrica utilizando curvas de demanda típicas de consumidores e redes neurais artificiais**. 2006. Tese (Doutorado em Sistemas de Potência) - Escola Politécnica, University of São Paulo, São Paulo, 2006. doi:10.11606/T.3.2006.tde-18042007-153132. Acesso em: 2024-10-15.

[29] GÉRON, A. **Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems**. O'Reilly Media, 2019. ISBN 9781492032618. Disponível em: <https://books.google.com.br/books?id=HHetDwAAQBAJ>.

[30] GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. MIT Press, 2016. (Adaptive Computation and Machine Learning series). ISBN 9780262035613. Disponível em: <https://books.google.com.br/books?id=Np9SDQAAQBAJ>.

[31] BARROS, J. G. R. D. **Um estudo sobre redes neurais fractais**. 2021.

[32] DU, K. et al. **Sar atr based on displacement-and rotation-insensitive cnn**. Remote Sensing Letters, Taylor & Francis, v. 7, n. 9, p. 895–904, 2016.

[33] B. W. Boehm, "**A spiral model of software development and enhancement**," in Computer, vol. 21, no. 5, pp. 61-72, May 1988, doi: 10.1109/2.59.

[34] LIN, Qinjie et al. **RoboFlow: a data-centric workflow management system for developing AI-enhanced Robots.** In: Conference on Robot Learning. PMLR, 2022. p. 1789-1794.

[35] REDMON, J. et al. **You only look once: Unified, real-time object detection**. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], 2016.

[36] Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., & Ding, G. (2024**). Yolov10: Real-time end-to-end object detection**. arXiv preprint arXiv:2405.14458.

[37] ULTRALYTICS. **YOLOv10**. Disponível em: <https://docs.ultralytics.com/models/yolov10/>. Acesso em: 30 de setembro de 2024.

[38] DANTAS, Tiago. **Fases do Sono**. virtual book, 2018. Disponível em <https://brasilescola.uol.com.br/curiosidades/fases-sono.htm>. Acesso em 07 de maio de 2018.

[39] Kim, W.; Jung, W.-S.; Choi, H.K. **Lightweight Driver Monitoring System Based on Multi-Task Mobilenets**. Sensors 2019, 19, 3200. https://doi.org/10.3390/s19143200

[40] SAHAYADHAS, Arun, KENNETH Sundaraj. **Detecting Driver Drowsiness Based on Sensors**. A Review: Virtual Book, 2012, Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3571819/>. Acesso em 25 de setembro de 2024.

[41] Brodbeck, V.; Kuhn, A.; von Wegner, F.; Oliveira, A.; Tagliazucchi, E.; Oliveira, S.; Michel, C.M.; Laufs, H**. Microestados EEG de vigília e sono NREM**. NeuroImagem 2012, 62, 2129–2139.

[42] **Direção sonolenta e acidentes automobilísticos**; Centro Nacional de Pesquisa de Distúrbios do Sono e Administração Nacional de Segurança no Tráfego Rodoviário: Howe, TX, EUA, 1998.

[43] KAGGLE**. Kaggle: Your home for data science.** Disponível em: <https://www.kaggle.com/>. Acesso em: 25 de junho de 2024.

[44] **Weights & Biases – Developer tools for ML**. Disponível em: <https://wandb.ai>. Acesso em: 25 de junho de 2024.

[45] Diego Nogare. **Performance de Machine Learning: Matriz de Confusão**. Acesso em: 25 jul. 2024. 2016. url: https : / / diegonogare . net / 2020 / 04 / performance-de-machine-learning-matriz-de-confusao.

APÊNDICE A – CÓDIGO DO PROGRAMA

https://github.com/Leonardocosta3/TCC-apendice.git