Universidade Federal de Juiz de Fora
Instituto de ciencias exatas - ICE
Bacharelado em Ciência da Computação

Avaliação de Confiança de Classificadores de ML para Reconhecimento de Semáforos em Veículos Autônomos

Sávio Chermont Warol Teixeira

JUIZ DE FORA
JANEIRO, 2026

Avaliação de Confiança de Classificadores de ML para Reconhecimento de Semáforos em Veículos Autônomos

Sávio Chermont Warol Teixeira

Universidade Federal de Juiz de Fora

Instituto de ciencias exatas - ICE

Departamento de Ciência da Computação - DCC

Bacharelado em Ciência da Computação

Orientador: André Luiz de Oliveira

JUIZ DE FORA

JANEIRO, 2026

AVALIAÇÃO DE CONFIANÇA DE CLASSIFICADORES DE ML PARA RECONHECIMENTO DE SEMÁFOROS EM VEÍCULOS AUTÔNOMOS

Sávio Chermont Warol Teixeira

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIENCIAS EXATAS - ICE DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

André Luiz de Oliveira
Doutor em Ciência da Computação

Luciana Conceição Dias Campos
Doutora em Engenharia Elétrica

Carlos Cristiano Hasenclever Borges
Doutor em Engenharia Civil

JUIZ DE FORA

09 DE JANEIRO, 2026

Resumo

A adoção de veículos autônomos e de sistemas avançados de assistência ao condutor depende diretamente da capacidade dos algoritmos embarcados de operar com segurança em ambientes dinâmicos. Entre esses componentes, o reconhecimento automático de semáforos desempenha um papel essencial para evitar colisões e garantir decisões adequadas em vias urbanas. No entanto, modelos de *Machine Learning*, mesmo quando apresentam alta acurácia, podem falhar ao enfrentar condições inesperadas, tornando necessária a existência de mecanismos capazes de avaliar não apenas a previsão, mas também sua confiabilidade. Este trabalho investiga o uso da abordagem SafeML como ferramenta de análise de confiança aplicada a classificadores de semáforos treinados sobre o dataset LISA. Dois modelos — uma SVM (*Support Vector Machine*) e uma CNN (*Convolutional Neural Network*) — são avaliados a fim de identificar padrões que indiquem quando uma previsão pode ser considerada segura ou potencialmente arriscada. A análise realizada permite discutir como desvios entre dados de treino e de aplicação podem influenciar o comportamento dos modelos e em que medida abordagens estatísticas podem auxiliar na interpretação desses cenários. Os resultados obtidos contribuem para avaliar a viabilidade do uso dessa metodologia como apoio à análise de confiabilidade em sistemas de Visão Computacional embarcados.

Palavras-chave: Confiabilidade, SafeML, Visão Computacional, Aprendizado de Máquina, Veículos Autônomos, Semáforos.

Abstract

The adoption of autonomous vehicles and advanced driver assistance systems depends directly on the ability of onboard algorithms to operate safely in dynamic environments. Among these components, automatic traffic-light recognition plays a key role in preventing collisions and ensuring appropriate decision-making in urban scenarios. However, machine learning models, even when achieving high accuracy, may fail when exposed to unexpected conditions, reinforcing the need for mechanisms capable of assessing not only the prediction itself but also its level of reliability. This work investigates the use of the SafeML approach as a confidence analysis tool applied to traffic-light classifiers trained on the LISA dataset. Two models — a Support Vector Machine (SVM) and a Convolutional Neural Network (CNN) — are evaluated in order to identify patterns that indicate when a prediction may be considered safe or potentially risky. The analysis enables discussion on how deviations between training and application data can influence model behavior and how statistical approaches may assist in interpreting such scenarios. The results obtained contribute to evaluating the feasibility of using this methodology to support reliability assessment in embedded computer vision systems.

Keywords: Reliability, SafeML, Computer Vision, Machine Learning, Autonomous Vehicles, Traffic Lights.

Agradecimentos

Agradeço profundamente aos meus pais, Marcelo e Aline, e à minha irmã Alice, pelo apoio constante e por estarem sempre ao meu lado, mesmo à distância. Sou imensamente grato pela compreensão nos momentos e datas importantes em que não pude estar presente, como aniversários, comemorações e encontros familiares, em razão da dedicação exigida por esta etapa. O suporte de vocês foi fundamental ao longo de toda essa jornada, proporcionando-me força, incentivo e motivação para seguir em frente.

À minha companheira Maria Clara, pelo amor, paciência e compreensão nos momentos em que precisei abdicar de sair ou passar tempo com ela para me dedicar aos estudos. Seu apoio diário, incentivo constante e presença ao meu lado foram essenciais para tornar essa fase mais leve e tranquila.

Aos meus amigos e familiares, pelas risadas, pelas conversas e pela presença que sempre trouxeram leveza ao meu dia a dia.

Ao professor André, agradeço pela orientação e conselhos durante a elaboração deste trabalho.

Agradeço também aos professores e funcionários da UFJF, pela contribuição ao meu aprendizado e à minha experiência ao longo do curso.

Por fim, agradeço ao Clube de Regatas do Flamengo, fonte constante de alegria e motivação ao longo da minha vida. Durante esta trajetória, não foi diferente, pois acompanhar os jogos proporcionou alívio nos momentos de maior pressão e ajudou a manter o ânimo diante dos desafios do percurso.

"O aspecto mais triste da vida atualmente é que a ciência acumula conhecimento mais rápido do que a sociedade acumula sabedoria."

Isaac Asimov

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

ANN	Artificial Neural Network
CNN	Convolutional Neural Network
CVM	Cramér–Von Mises Distance
DCC	Departamento de Ciência da Computação
ECDF	Empirical Cumulative Distribution Function
IA	Inteligência Artificial
IPEA	Instituto de Pesquisa Econômica Aplicada
KS	Kolmogorov–Smirnov Distance
LIME	Local Interpretable Model-agnostic Explanations
ML	Machine Learning
MLP	Multi-Layer Perceptron
NHTSA	National Highway Traffic Safety Administration
OMS	Organização Mundial da Saúde
RGB	Red, Green, Blue (canais de cor)
SHAP	SHapley Additive exPlanations
SVM	Support Vector Machine
UFJF	Universidade Federal de Juiz de Fora
WD	Wasserstein Distance
XAI	Explainable Artificial Intelligence

1 Introdução

O avanço das técnicas de *Machine Learning* tem impulsionado o desenvolvimento de sistemas capazes de interpretar cenários complexos e tomar decisões de forma autônoma. Modelos de classificação, detecção e reconhecimento visual tornaram-se centrais em diversas aplicações, incluindo veículos autônomos, inspeção industrial e vigilância inteligente. No entanto, apesar de seu alto desempenho em ambientes controlados, esses modelos frequentemente carecem de mecanismos para indicar quando suas previsões podem estar incorretas ou quando estão operando fora de seu domínio de treinamento. Essa limitação torna a análise de confiança um componente essencial para aumentar a segurança e a transparência desses sistemas, especialmente em aplicações críticas.

A evolução de sistemas inteligentes aplicados ao trânsito tem desempenhado papel central na busca por maior segurança, eficiência e autonomia na mobilidade urbana. Entre esses avanços, destacam-se os sistemas de Visão Computacional embarcados, capazes de interpretar o ambiente ao redor do veículo e apoiar processos decisórios críticos. Tarefas como detecção de obstáculos, reconhecimento de placas e identificação do estado de semáforos tornaram-se componentes essenciais de veículos autônomos e de sistemas avançados de assistência ao condutor.

Apesar dos progressos obtidos por modelos de aprendizado de máquina, sua utilização em cenários reais ainda apresenta desafios relevantes. É comum que classificadores tenham bom desempenho em ambientes controlados, mas sofram degradação quando expostos a condições distintas das presentes nos dados de treinamento. Variações de iluminação, mudanças climáticas, ruído no sensor ou alterações no contexto urbano podem comprometer a robustez dos modelos e conduzir a erros potencialmente perigosos. Nesse cenário, torna-se indispensável não apenas obter uma previsão, mas também estimar o quão confiável é essa previsão.

Entre as abordagens que buscam enfrentar esse desafio, destaca-se o framework SafeML, proposto por Aslansefat et al. (2020), que utiliza técnicas estatísticas para monitorar desvios entre distribuições de dados de treino e de aplicação, sinalizando situações

de maior risco. Este trabalho aplica essa abordagem ao reconhecimento automático de semáforos, avaliando como diferentes modelos de Visão Computacional se comportam do ponto de vista de confiabilidade.

1.1 Descrição do Problema

Os acidentes de trânsito configuram-se como um dos principais problemas de saúde pública em escala global. De acordo com a OMS, aproximadamente 1,19 milhão de pessoas morrem anualmente em decorrência de acidentes de trânsito, além de dezenas de milhões de feridos, muitos dos quais sofrem sequelas permanentes (World Health Organization, 2018). A gravidade desse cenário torna os sinistros viários uma das principais causas de morte entre crianças, adolescentes e jovens adultos.

No contexto brasileiro, a situação apresenta números igualmente alarmantes. Segundo dados do IPEA, entre os anos de 2010 e 2019 o Brasil registrou, em média, cerca de 40 mil mortes por ano no trânsito, além de mais de 300 mil pessoas gravemente feridas anualmente (Instituto de Pesquisa Econômica Aplicada, 2021). Esses dados evidenciam não apenas as perdas humanas, mas também impactos sociais e econômicos significativos.

Diante desse cenário, veículos autônomos e sistemas avançados de assistência ao condutor surgem como alternativas promissoras para mitigar a ocorrência de acidentes, ao reduzir a influência de fatores humanos como desatenção, fadiga e erro de julgamento. Entretanto, a adoção dessas tecnologias também introduz novos desafios relacionados à segurança dos sistemas computacionais responsáveis pela percepção e tomada de decisão.

Investigações recentes demonstram que falhas em sistemas de direção automatizada já resultaram em acidentes fatais. Uma apuração conduzida pela NHTSA (Administração Nacional de Segurança no Tráfego Rodoviário dos EUA), concluída em 2024, identificou ao menos 13 mortes associadas ao uso do sistema Autopilot da Tesla, destacando uma lacuna crítica de segurança relacionada ao uso indevido do sistema e à ausência de monitoramento eficaz do condutor. Desde 2016, mais de 40 investigações especiais foram abertas envolvendo tecnologias similares, totalizando ao menos 23 mortes reportadas. Como resposta, a Tesla realizou um *recall* de cerca de 2 milhões de veículos, com o objetivo de atualizar o software e reforçar os alertas de atenção ao motorista (The Guardian,

2024).

Por outro lado, empresas do setor argumentam que sistemas autônomos podem reduzir significativamente o número de acidentes quando comparados à condução humana. A Waymo, subsidiária da Google, reportou ter percorrido mais de 127 milhões de milhas em modo totalmente autônomo até 2025, afirmando reduções de até 90% em acidentes com ferimentos graves e 82% no acionamento de airbags em relação à média de motoristas humanos (Waymo LLC, 2025). Esses dados ilustram que, embora promissora, a tecnologia ainda apresenta resultados contrastantes e depende fortemente de mecanismos de segurança adicionais.

No Brasil, a adoção de sistemas automatizados também tem avançado. Estima-se que cerca de 40% dos veículos novos vendidos em 2025 já contem com algum nível de automação. Paralelamente, casos judiciais recentes evidenciam que falhas nesses sistemas já têm gerado repercussões legais concretas, com decisões que responsabilizam montadoras por acidentes envolvendo piloto automático. Além disso, juristas apontam a dificuldade de apurar responsabilidades em tais incidentes, devido à falta de transparência e de acesso aos dados internos dos algoritmos — o que caracteriza o problema da chamada "caixa-preta" dos modelos de ML. (Senna Martins Advogados, 2025).

Nesse contexto, a ausência de mecanismos capazes de indicar quando uma decisão automatizada pode não ser confiável representa um obstáculo relevante para a adoção segura dessas tecnologias. Em particular, falhas no reconhecimento do estado de semáforos podem resultar em decisões críticas incorretas em ambientes urbanos, reforçando a necessidade de abordagens que avaliem não apenas a acurácia, mas também a confiabilidade das previsões.

1.2 Objetivos

O objetivo deste trabalho é analisar a confiabilidade de classificadores de aprendizado de máquina aplicados ao reconhecimento automático de semáforos, considerando o contexto de segurança em veículos autônomos e sistemas avançados de assistência ao condutor. Para isso, busca-se analisar se as técnicas de análise estatística disponibilizadas pelo framework SafeML são adequadas e eficazes para identificar indícios de classificações

potencialmente não confiáveis, mesmo em modelos que apresentam bom desempenho em métricas tradicionais de avaliação.

Como objetivos específicos, este trabalho propõe:

- treinar e avaliar dois modelos de classificação de semáforos, baseados em uma SVM e em uma CNN, utilizando o conjunto de dados *LISA Traffic Light Dataset*;
- analisar o comportamento desses modelos a partir da comparação entre amostras corretamente e incorretamente classificadas;
- aplicar a abordagem SafeML para a análise de confiança das previsões, investigando a presença de desvios estatísticos entre os dados de treino e os dados utilizados na avaliação;
- explorar, por meio de análises quantitativas e visuais, padrões associados a situações em que os modelos apresentam maior propensão ao erro;
- discutir a viabilidade do uso dessa abordagem como apoio à avaliação de confiabilidade em sistemas de Visão Computacional embarcados.

1.3 Justificativa

A crescente adoção de técnicas de ML em sistemas críticos, como veículos autônomos e sistemas avançados de assistência ao condutor, torna indispensável a discussão sobre segurança e confiabilidade das decisões tomadas por esses modelos. Embora avanços recentes tenham permitido alcançar elevados níveis de desempenho em tarefas de Visão Computacional, casos reais de acidentes demonstram que métricas tradicionais, como acurácia, não são suficientes para garantir comportamento seguro em ambientes reais e dinâmicos.

Os dados recentes de investigações, *recalls* e disputas judiciais evidenciam que falhas em sistemas automatizados já resultam em consequências humanas, econômicas e legais. Ao mesmo tempo, a dificuldade de auditoria e explicação das decisões desses sistemas reforça a necessidade de mecanismos independentes de monitoramento e avaliação de risco, capazes de atuar como uma camada adicional de segurança.

No contexto do reconhecimento automático de semáforos, erros de classificação podem acarretar consequências graves, uma vez que esse componente está diretamente associado à tomada de decisões em cruzamentos e vias urbanas. Dessa forma, torna-se relevante investigar abordagens que permitam avaliar o grau de confiança associado às classificações produzidas pelos modelos, contribuindo para uma compreensão mais ampla de seus limites e potenciais riscos de uso.

A escolha do framework SafeML justifica-se por sua proposta de fornecer mecanismos estatísticos para a análise de confiabilidade de classificadores, sem a necessidade de modificar o modelo original ou acessar informações internas de sua arquitetura. Essa característica torna a abordagem especialmente atrativa em cenários práticos e jurídicos, nos quais a transparência e a independência da análise são requisitos importantes.

Assim, este trabalho se justifica pela relevância social do tema, pela necessidade técnica de avaliação de confiabilidade em aprendizado de máquina e pela possibilidade de contribuir para o desenvolvimento de sistemas de Visão Computacional mais seguros, auditáveis e responsáveis no contexto do tráfego urbano.

1.4 Organização do Trabalho

Este trabalho está organizado em seis capítulos. No Capítulo 1, é apresentada a introdução, na qual são descritos o contexto do problema, os objetivos do estudo, sua justificativa e a estrutura geral do trabalho.

O Capítulo 2 apresenta a fundamentação teórica, abordando conceitos relacionados a veículos autônomos, *Machine Learning*, reconhecimento de semáforos e confiabilidade em sistemas de Visão Computacional, além de introduzir os princípios do framework SafeML.

No Capítulo 3 são discutidos os trabalhos relacionados, destacando abordagens existentes para análise de confiabilidade e monitoramento de segurança em modelos de aprendizado de máquina, bem como suas principais características e limitações.

O Capítulo 4 descreve a metodologia adotada, incluindo o conjunto de dados utilizado, o pré-processamento das imagens, os modelos de classificação empregados e o pipeline de análise baseado no SafeML.

No Capítulo 5 são apresentados e analisados os resultados experimentais obtidos, contemplando o desempenho dos classificadores e a análise de confiabilidade por meio das métricas e visualizações geradas.

Por fim, o Capítulo 6 apresenta as conclusões do trabalho, destacando as principais contribuições, limitações do estudo e perspectivas para trabalhos futuros.

2 Fundamentação Teórica

Este capítulo apresenta os principais conceitos que embasam o desenvolvimento deste trabalho, fornecendo a base necessária para compreender tanto a metodologia adotada quanto a interpretação dos resultados obtidos.

2.1 Machine Learning e Visão Computacional

Para analisar a confiabilidade de sistemas inteligentes, é fundamental compreender primeiro os componentes que constituem essa inteligência. Frequentemente confundida com uma tecnologia única, a Inteligência Artificial é, na verdade, uma área ampla da Ciência da Computação. O Departamento de Educação dos EUA define IA como um "termo guarda-chuva" que abrange um conjunto crescente de capacidades de modelagem (Office of Educational Technology, 2023). Conforme ilustrado na Figura 2.1, a IA se ramifica em diversas subáreas, como robótica, processamento de linguagem natural e, especificamente para este trabalho, duas especialidades centrais: o *Machine Learning* e a Visão Computacional, conforme nomeadas na figura original.

O *Machine Learning* representa uma mudança de paradigma na forma como criamos software. Na programação tradicional, um desenvolvedor escreve regras explícitas para o computador seguir. Porém, segundo Hurwitz e Kirsch (2018), o ML foca na capacidade do sistema de aprender e adaptar um modelo baseando-se em dados, e não em regras pré-codificadas. O algoritmo "ingere" exemplos — como dados históricos ou imagens — para identificar padrões e criar um modelo capaz de prever resultados futuros.

Dentro deste universo, existem diferentes abordagens para ensinar o computador. Hurwitz e Kirsch (2018) categorizam o ML em quatro subconjuntos principais:

1. **Aprendizado Supervisionado:** O método mais comum, onde o algoritmo é treinado com dados que já contêm a "resposta correta" (dados rotulados). É como ensinar uma criança mostrando a ela várias fotos e dizendo "isto é um gato" ou "isto é um cachorro", até que ela possa distinguir sozinha.

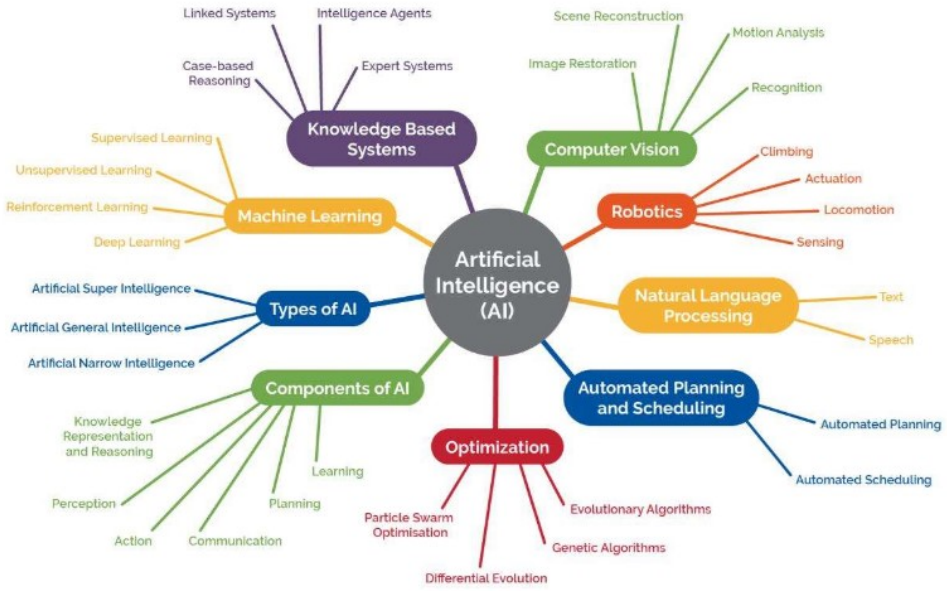

Figura 2.1: Componentes, tipos e subáreas da IA. Fonte: Adaptado de Regona et al. (2022) apud Office of Educational Technology (2023).

2. **Aprendizado Não Supervisionado:** Utilizado quando os dados não possuem rótulos. O algoritmo explora os dados para encontrar estruturas ocultas ou agrupar itens semelhantes (clusters) por conta própria.

3. **Aprendizado por Reforço:** Um modelo comportamental onde o sistema aprende por tentativa e erro, recebendo "recompensas" por decisões corretas e penalidades por erros, reforçando o comportamento desejado.

4. ***Deep Learning*:** Uma técnica avançada que utiliza redes neurais artificiais para modelar padrões complexos em grandes volumes de dados.

Enquanto o ML atua como o cérebro que processa padrões, a Visão Computacional funciona como os olhos do sistema. No entanto, enxergar digitalmente é um desafio matemático. Szeliski (2021) descreve como um "problema inverso": o computador recebe uma imagem 2D plana e precisa deduzir as propriedades do mundo real 3D, como forma, distância e identidade dos objetos. O que para um humano é intuitivo — diferenciar uma sombra de um objeto real — para a máquina exige algoritmos robustos para superar a ambiguidade e a falta de informações de profundidade inerentes a uma fotografia.

A intersecção entre essas áreas ocorre porque a Visão Computacional moderna depende do ML para funcionar com precisão. Como modelar matematicamente todas as

variações de luz e forma do mundo é impossível, utiliza-se o ML para treinar o computador a reconhecer visualmente os objetos com base em probabilidades (SZELISKI, 2021). No contexto deste TCC, essa união é o que permite a um veículo autônomo analisar imagens da via e classificar, com determinado grau de confiança, se um semáforo indica permissão para avançar ou ordem de parada.

2.2 SVM

Dentre os diversos algoritmos de aprendizado supervisionado disponíveis para classificação, o *Support Vector Machines* destaca-se por uma abordagem geométrica e intuitiva. Segundo Coutinho (2019), o objetivo fundamental do SVM é encontrar uma linha de separação (ou fronteira) entre duas classes distintas de dados. Em um exemplo bidimensional simples, como pontos vermelhos e azuis em uma folha de papel, o algoritmo busca traçar a reta que melhor separa esses dois grupos.

No entanto, em problemas reais com múltiplas características, essa "linha" torna-se um conceito multidimensional chamado de **hiperplano**. Coutinho (2019) explica que podem existir infinitas retas capazes de separar dois grupos de dados, mas o SVM não escolhe uma aleatoriamente. Ele busca o **hiperplano ótimo**: aquele que maximiza a distância perpendicular entre a fronteira de decisão e os pontos mais próximos de cada classe. Essa distância de segurança é tecnicamente denominada margem.

Os pontos de dados que estão localizados exatamente na borda dessa margem são os elementos mais críticos para o algoritmo e recebem o nome de **Vetores de Suporte** (*Support Vector*). Conforme detalham Szeliski (2021) e Coutinho (2019), são esses pontos que "seguram" ou definem a posição do hiperplano. Se movermos os pontos que estão distantes da fronteira, a linha de classificação não muda; porém, qualquer alteração nos vetores de suporte redefine todo o modelo. A Figura 2.2 ilustra visualmente esses componentes, destacando como o hiperplano central é equidistante dos vetores de suporte de ambas as classes.

Embora o conceito original do SVM tenha sido desenhado para separar duas classes (classificação binária), problemas como o reconhecimento de semáforos envolvem múltiplas categorias. Para lidar com isso, utiliza-se a estratégia conhecida como **One-vs-**

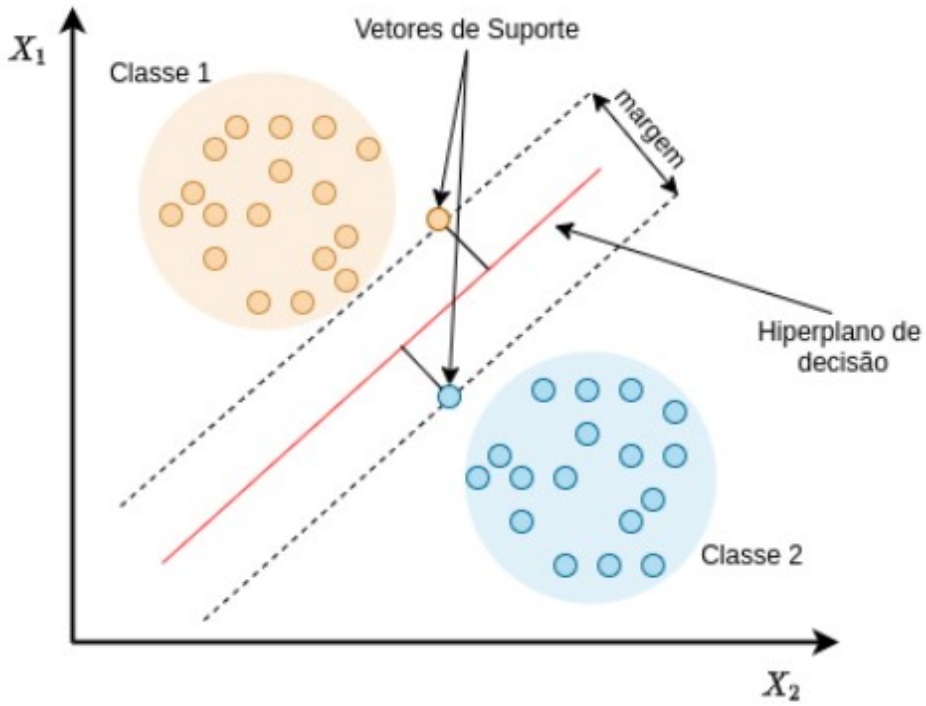

Figura 2.2: Estrutura geral do algoritmo SVM: vetores de suporte definindo o hiperplano de separação. Fonte: Barbosa et al. (2021).

All (ou *One-vs-Rest*). De acordo com Coutinho (2019), essa técnica divide um problema multiclasse em várias classificações binárias. Para classificar três cores de semáforo, o algoritmo treina: "Vermelho contra o resto", "Verde contra o resto", e assim por diante. Ao receber um novo dado, o sistema verifica qual desses classificadores apresenta a maior confiança para decidir a classe final.

Na implementação prática deste trabalho, optou-se pelo uso do classificador linear (`LinearSVC` do pacote *scikit-learn*). Isso significa que o modelo busca separar as classes usando hiperplanos retos. Embora existam variações do SVM não-linear que transformam o espaço para criar curvas complexas de separação — úteis quando os dados estão muito misturados —, Coutinho (2019) ressalta que o modelo linear é frequentemente preferível pela simplicidade e menor custo computacional.

Em suma, a escolha do SVM Linear com a estratégia *One-vs-All* oferece uma base robusta para a avaliação de confiança proposta neste TCC. Ele permite não apenas decidir qual a cor do semáforo, mas, através da análise da margem (a distância dos dados em relação ao hiperplano mostrada na Figura 2.2), inferir o quão segura é aquela classificação.

2.3 CNN

Para compreender a tecnologia que permite a um veículo autônomo "ver", é necessário primeiro entender a unidade fundamental de inteligência computacional: a *Artificial Neural Network*. Segundo Jorgecardete (2024), a base dessas redes é o neurônio artificial, inspirado biologicamente. Quando organizados em camadas sequenciais (*Input Layer*, *Hidden Layer* e *Output Layer*), eles formam o chamado *Multi-Layer Perceptron*, capaz de aprender relações complexas entre dados (vide Figura 2.3).

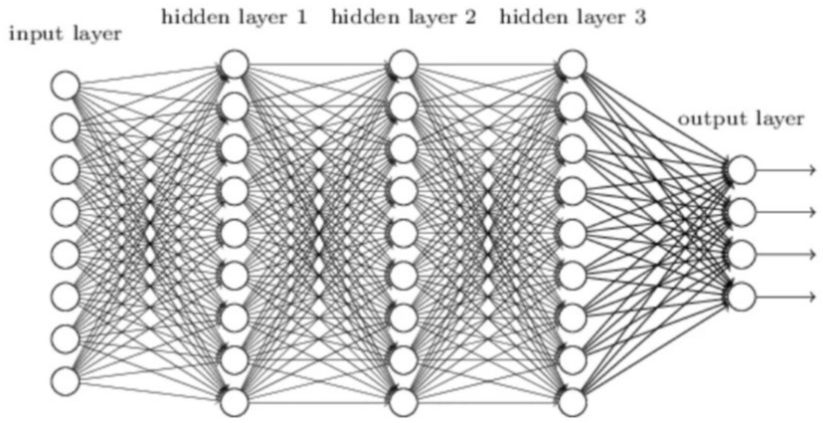

Figura 2.3: Estrutura básica de um MLP. Fonte: Dustin (2025).

No entanto, aplicar MLPs diretamente em imagens apresenta um problema crítico. Dustin (2025) explica que uma MLP trata a entrada como uma lista plana de números. Para processar uma imagem, a rede precisa "achatá-la" em um vetor único, destruindo a estrutura espacial. A rede perde a noção de que um pixel é vizinho do outro, ignorando formas e contornos. Para solucionar isso, utiliza-se a Convolutional Neural Network, que preserva a estrutura tridimensional da imagem (largura, altura e cores) e processa os dados em duas grandes etapas: a extração de características e a classificação.

A extração de características é realizada pela operação de Convolução. Ao contrário da conexão total da MLP, a convolução utiliza filtros (ou *kernels*) — pequenas matrizes de números — que deslizam sobre a imagem realizando operações matemáticas locais (vide Figura 2.4). Alves (2018) compara esse processo a uma "lanterna" escaneando a cena: cada filtro aprende a detectar um padrão visual específico, como bordas verticais, curvas ou texturas, gerando os chamados mapas de características.

Para tornar o processamento mais eficiente, aplicam-se camadas de Pooling (Su-

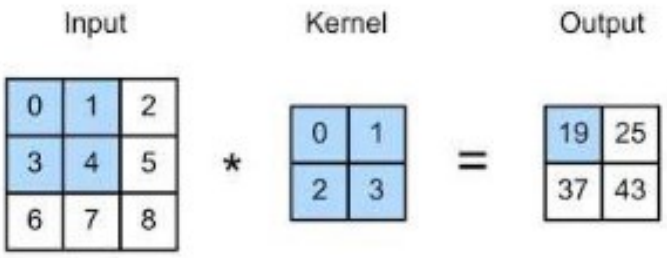

Figura 2.4: Operação de Convolução: um filtro desliza sobre a imagem para gerar um mapa de características. Fonte: Jorgecardete (2024).

bamostragem) entre as convoluções. O objetivo, segundo Jorgecardete (2024), é reduzir a dimensão da imagem mantendo apenas as informações mais relevantes. O método mais comum, o *Max Pooling*, seleciona apenas o valor mais alto dentro de uma pequena região (vide Figura 2.5), descartando ruídos e tornando a rede mais robusta a pequenas variações de posição do objeto.

Após as camadas de convolução e pooling extraírem os padrões visuais, ocorre a transição para a etapa de Classificação. Jorgecardete (2024) detalha que os mapas de características resultantes são submetidos a um processo de *Flattening* (achatamento), transformando os dados 3D em um vetor linear. Diferente do início do processo, agora esse achatamento é útil, pois alimenta camadas densas (*Fully Connected*) que interpretam esses padrões de alto nível para tomar uma decisão final, conforme a visão geral na Figura 2.6. Adicionalmente, podem ser empregadas camadas de *Dropout* com o objetivo de reduzir o sobreajuste (*overfitting*) durante o treinamento, desativando aleatoriamente neurônios ao longo do processo de aprendizagem.

Na camada de saída, arquiteturas modernas frequentemente utilizam funções como a **Softmax** para converter os valores numéricos produzidos pela rede em escores normalizados, comumente interpretados como probabilidades associadas a cada classe (JORGECARDETE, 2024). No contexto deste trabalho, isso permite que o sistema produza pontuações para rótulos acionáveis no trânsito, como *go*, *stop* ou *warning*, indicando a preferência do modelo por cada classe.

Por fim, é crucial entender que a CNN não é programada com regras fixas, mas sim treinada via Aprendizado Supervisionado. Segundo Szeliski (2021) e Jorgecardete (2024),

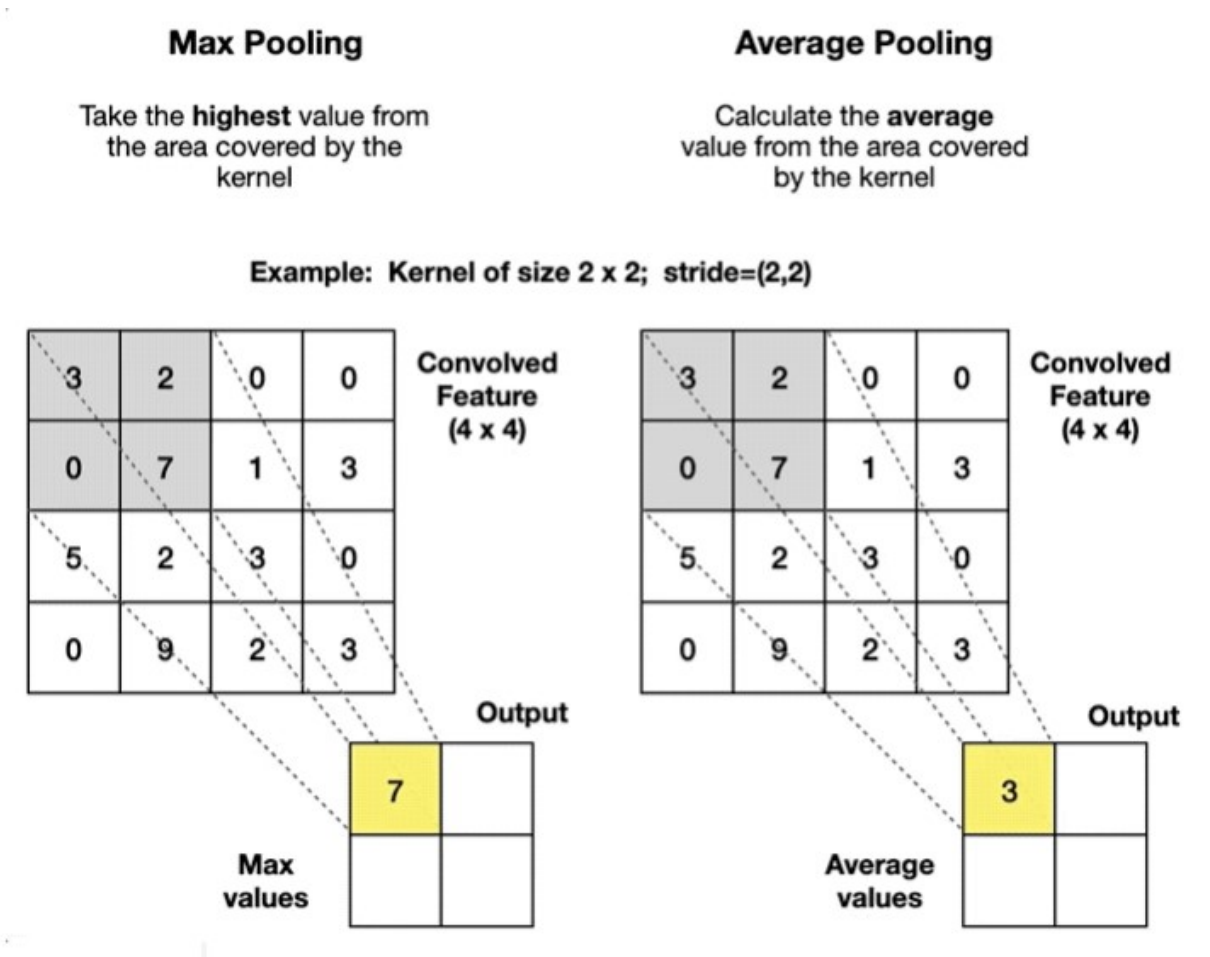

Figura 2.5: Max Pooling: redução da dimensão mantendo apenas os valores máximos. Fonte: Jorgecardete (2024).

a rede é exposta a milhares de imagens previamente rotuladas e realiza predições iniciais. O erro associado a essas predições é calculado por uma função de perda (*Loss Function*) e, por meio do algoritmo de *Backpropagation* (retropropagação), esse erro é propagado de volta pela rede para ajustar iterativamente os pesos dos filtros e das camadas densas. Esse ciclo iterativo permite que o modelo aprenda representações visuais cada vez mais adequadas para a tarefa de classificação de semáforos.

2.4 Reconhecimento de Semáforos

O reconhecimento de semáforos desempenha um papel vital na segurança e eficiência dos sistemas de transporte modernos. Enquanto humanos ocasionalmente falham em perceber ou obedecer à sinalização — seja por distração, fadiga ou imprudência —, os Sistemas

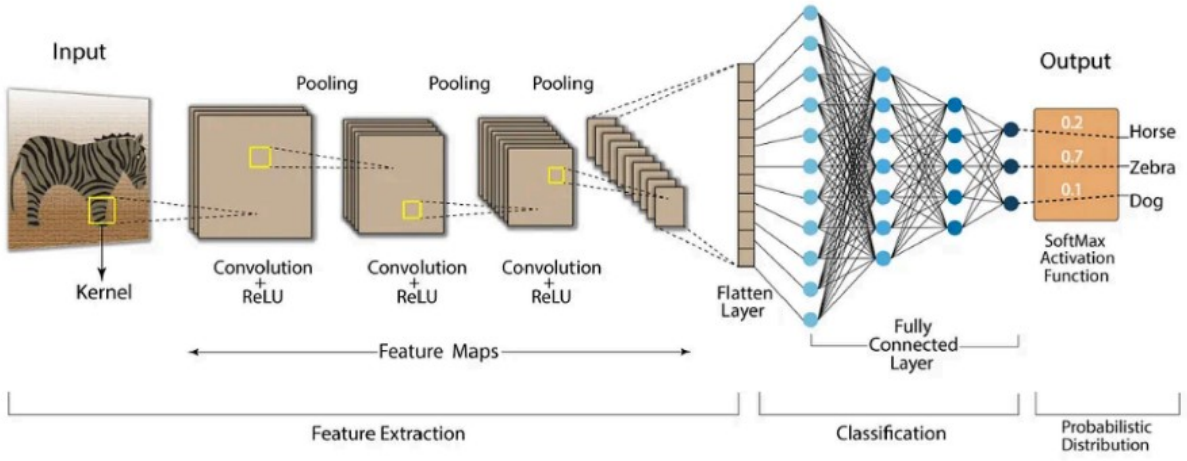

Figura 2.6: Visão completa de uma CNN: da extração de características à classificação. Fonte: Jorgecardete (2024).

Avançados de Assistência ao Condutor (ADAS) e os veículos autônomos prometem mitigar esses erros através de monitoramento contínuo. Segundo Jensen et al. (2016), a importância dessa tecnologia é sublinhada pelas estatísticas de acidentes: apenas nos EUA, em 2012, centenas de pessoas morreram e milhares ficaram feridas em colisões causadas especificamente pelo desrespeito ao sinal vermelho.

Idealmente, a comunicação entre a infraestrutura e o veículo (I2V) via rádio seria a solução definitiva para informar o estado do semáforo. No entanto, Philipsen et al. (2015) argumentam que, devido ao alto custo de implementação dessa infraestrutura em larga escala, a solução imediata e mais viável reside na Visão Computacional. O veículo deve ser capaz de "ver"e interpretar a sinalização existente da mesma forma que um humano faz, utilizando câmeras e algoritmos inteligentes para tomar decisões críticas, como parar em um cruzamento ou atravessar com segurança durante a "zona de dilema"da luz amarela (JENSEN et al., 2016).

Contudo, replicar a percepção humana via software apresenta desafios técnicos formidáveis. O ambiente urbano é caótico e visualmente poluído. Jensen et al. (2016) destacam que um sistema robusto precisa lidar com variações extremas de iluminação: durante o dia, o sol pode causar reflexos ou "fantasmas"na lente, lavando a cor da luz; à noite, o alto contraste pode saturar o sensor da câmera ou gerar halos de luz que distorcem a forma do semáforo. Além disso, existe o problema recorrente das falsas

detecções, onde luzes de freio de outros carros, letreiros de neon ou reflexos em janelas podem ser confundidos com um sinal de trânsito (PHILIPSEN et al., 2015).

Outro obstáculo significativo é a variabilidade geométrica e posicional. Conforme ilustrado por Jensen et al. (2016), os semáforos não são padronizados globalmente: eles podem estar dispostos verticalmente ou horizontalmente, suspensos por cabos (balançando com o vento) ou fixos em postes laterais. Além das tradicionais luzes circulares, existem setas direcionais, contadores regressivos e ícones específicos para pedestres ou bicicletas. O sistema de visão computacional deve ser capaz de generalizar o aprendizado para reconhecer todas essas variantes, mesmo quando o objeto está parcialmente oculto por um caminhão, uma árvore ou afetado por condições climáticas adversas como chuva e neblina (JENSEN et al., 2016).

Para enfrentar essa complexidade, a literatura divide o problema em etapas sequenciais: detecção (encontrar o objeto na imagem), classificação (determinar se é verde, amarelo ou vermelho) e rastreamento (acompanhar o objeto ao longo do tempo). Enquanto abordagens antigas tentavam resolver isso com regras manuais baseadas em cor e formato (métodos heurísticos), Philipsen et al. (2015) apontam que o estado da arte migrou para abordagens baseadas em aprendizado de máquina (*learning-based*), que demonstraram ser ordens de magnitude superiores em precisão e recuperação (*recall*), especialmente quando treinadas com grandes volumes de dados.

É nesse contexto de aprendizado baseado em dados que surge a necessidade de bancos de imagens robustos para treinamento e validação. Philipsen et al. (2015) introduziram o *LISA Traffic Light Dataset*, desenvolvido pela Universidade da Califórnia em San Diego, para suprir a carência de bases de dados públicas e desafiadoras. Diferente de conjuntos de dados pequenos e controlados, o LISA contém mais de 112 mil anotações de semáforos capturadas em cenários reais de condução urbana nos EUA.

O diferencial do dataset LISA, segundo Jensen et al. (2016), reside na sua diversidade: as sequências de vídeo capturam transições de iluminação, variando desde a manhã clara até o crepúsculo e a noite completa. Ele inclui situações complexas com múltiplos semáforos visíveis simultaneamente, oclusões e distâncias variadas. Essa variabilidade o torna uma ferramenta essencial para testar a confiabilidade de algoritmos, servindo como

a base de dados escolhida para o desenvolvimento e avaliação dos modelos de IA propostos neste trabalho.

2.5 Segurança e Confiabilidade em ML

A adoção de modelos de ML em sistemas críticos exige uma mudança de paradigma na engenharia de software. Diferente da programação tradicional, onde a lógica é codificada explicitamente por humanos através de regras determinísticas, sistemas de ML induzem seu comportamento a partir de dados (HURWITZ; KIRSCH, 2018). Isso implica que a confiabilidade do sistema não depende apenas da ausência de *bugs* no código, mas fundamentalmente da qualidade, veracidade e representatividade dos dados utilizados no treinamento. Conforme alertam Hurwitz e Kirsch (2018), um modelo é apenas uma "aproximação da realidade"; se os dados contiverem vieses ou não cobrirem cenários raros, a "inteligência" do sistema pode falhar silenciosamente.

Do ponto de vista da engenharia de sistemas, Sculley et al. (2015) introduzem o conceito de "Dívida Técnica Oculta" em ML. Eles argumentam que, embora desenvolver um modelo inicial seja relativamente rápido, mantê-lo confiável ao longo do tempo é complexo devido ao princípio CACE (*Changing Anything Changes Everything*). Em sistemas de ML, não existe isolamento estrito: uma simples alteração no sinal de entrada, como a troca de um sensor ou uma mudança na iluminação do ambiente, pode alterar o comportamento de toda a rede de predição. Sculley et al. (2015) destacam que a maior parte do código em um sistema real não é o algoritmo de ML em si, mas a vasta infraestrutura de coleta, preparação e monitoramento de dados, onde a erosão de fronteiras pode criar dependências frágeis e potencialmente perigosas.

Além da fragilidade sistêmica, existe o desafio estatístico da validação. Koopman e Wagner (2016) explicam que é inviável garantir a segurança de um veículo autônomo apenas através de testes extensivos de rodagem, pois eventos catastróficos são estatisticamente raros — os chamados "Cisnes Negros". Um sistema pode funcionar corretamente para milhões de situações comuns e ainda assim falhar em um cenário inédito que não estava presente no conjunto de treinamento. Por esse motivo, a confiabilidade deve ser considerada como um requisito de projeto, utilizando estratégias arquiteturais como o

par Monitor/Atuador, no qual um sistema de segurança mais simples e determinístico supervisiona as decisões de um modelo de aprendizado complexo (KOOPMAN; WAGNER, 2016).

Esse cenário evidencia uma limitação fundamental das práticas tradicionais de Engenharia de Software. Técnicas clássicas de verificação e validação realizadas em tempo de projeto (*design-time*), como testes unitários, testes de integração e validação baseada em requisitos, pressupõem sistemas com comportamento determinístico e totalmente especificável. Em sistemas baseados em *Machine Learning*, cujo comportamento é induzido a partir de dados e pode variar conforme o ambiente de operação, tais abordagens não são suficientes para garantir níveis adequados de confiança, especialmente em aplicações de segurança crítica como veículos autônomos. Dessa forma, torna-se necessária a adoção de mecanismos complementares capazes de avaliar o comportamento do modelo durante a execução, monitorando possíveis desvios e situações de risco que não puderam ser antecipadas na fase de desenvolvimento.

Para avaliar o desempenho de classificadores, métricas quantitativas são amplamente utilizadas. Entretanto, métricas isoladas podem levar a interpretações equivocadas. A literatura técnica (PHILIPSEN et al., 2015; JENSEN et al., 2016) destaca três conceitos fundamentais para a avaliação de modelos de classificação:

- **Acurácia (Accuracy):** Métrica intuitiva que expressa a proporção total de classificações corretas realizadas pelo modelo. Embora amplamente utilizada, pode ser enganosa em cenários com dados desbalanceados. Em um contexto urbano onde a maioria dos semáforos esteja verde, um modelo que sempre indique esse estado pode apresentar alta acurácia, apesar de ser inadequado e inseguro.

- **Precisão (Precision):** Indica o quão confiáveis são as decisões positivas do modelo, ou seja, com que frequência uma detecção realizada corresponde de fato a um evento real. Em sistemas veiculares, alta precisão reduz ações desnecessárias, como frenagens indevidas.

- **Revocação (Recall):** Mede a capacidade do sistema de identificar todos os eventos relevantes presentes no ambiente. Em aplicações de segurança, essa métrica é

particularmente crítica, pois valores baixos indicam que situações perigosas podem não ser detectadas.

A aplicação desses conceitos ao domínio do trânsito evidencia a assimetria do risco. Conforme discutido por Jensen et al. (2016), em sistemas de assistência ao condutor, os erros não possuem o mesmo impacto. A não detecção de um sinal vermelho pode resultar em acidentes graves, enquanto uma detecção incorreta tende a gerar apenas uma interrupção desnecessária da condução. Dessa forma, um sistema de ML confiável, no contexto deste Trabalho de Conclusão de Curso, não é necessariamente aquele que apresenta a maior acurácia geral, mas sim aquele capaz de minimizar falhas críticas, mantendo desempenho estável mesmo diante das variações do ambiente real.

2.6 SafeML

Para garantir que os modelos operem dentro de parâmetros seguros, mesmo diante de dados desconhecidos, este trabalho utiliza a abordagem **SafeML**. Proposta originalmente por Aslansefat et al. (2020) e aprimorada em sua segunda versão (SafeML II) por Aslansefat et al. (2021), essa técnica atua como um monitor de segurança em tempo de execução (*runtime safety monitor*), detectando quando os dados de entrada divergem estatisticamente dos dados conhecidos pelo modelo durante o treinamento.

Arquitetura e Funcionamento: Fases Offline e Online

A implementação do SafeML não se resume a uma métrica isolada, mas sim a um framework operacional dividido em duas fases distintas: a Fase de Treinamento (Offline) e a Fase de Aplicação (Online), conforme ilustrado na estrutura geral da Figura 2.7.

Na **Fase Offline**, utiliza-se um *dataset* confiável e rotulado para treinar o classificador. Após o treinamento, o sistema não apenas salva os pesos do modelo, mas também extrai o perfil estatístico de cada classe correta. Segundo Aslansefat et al. (2021), isso é feito armazenando a Função de Distribuição Cumulativa Empírica (ECDF) das características extraídas das imagens de treino, como valores de pixels ou ativações internas do modelo, dependendo do classificador utilizado. Esses perfis estatísticos servem como a

assinatura de referência do que é considerado um dado seguro e conhecido.

Na **Fase Online**, o sistema opera em tempo de execução recebendo imagens não rotuladas. Um *buffer* temporal é utilizado para coletar uma pequena sequência de amostras (por exemplo, 15 quadros consecutivos). O SafeML então calcula a ECDF dessas novas amostras e mede a distância estatística em relação à assinatura de referência armazenada durante a fase offline.

Distâncias Estatísticas e a Evolução com P-Value

Para quantificar a divergência entre os dados de treino e os dados observados em execução, o framework SafeML emprega um conjunto de métricas de distância estatística. As principais implementadas são:

1. **Distância de Kolmogorov-Smirnov (KS):** Calcula a maior distância vertical absoluta entre duas funções de distribuição acumulada. É eficaz para detectar mudanças abruptas, mas apresenta menor sensibilidade às caudas da distribuição.

2. **Distância de Kuiper (K):** Variação da KS que considera conjuntamente os desvios positivos e negativos, sendo mais sensível a alterações nos extremos dos dados.

3. **Distância de Anderson-Darling (AD):** Atribui maior peso às diferenças observadas nas caudas da distribuição, sendo particularmente útil para detectar eventos raros e potencialmente críticos em sistemas de segurança.

4. **Distância de Cramer–Von Mises (CVM):** Calcula a soma das diferenças quadráticas ao longo de toda a distribuição, oferecendo uma visão global da divergência entre os conjuntos de dados.

5. **Distância de Wasserstein (WD):** Mede o esforço necessário para transformar uma distribuição na outra. Conforme discutido por Aslansefat et al. (2021), a WD é especialmente adequada para aplicações de visão computacional, pois captura a geometria das alterações entre distribuições (vide Figura 2.8).

Embora o SafeML em sua versão inicial utilizasse essas métricas diretamente, o método apresentava sensibilidade a variações irrelevantes dos dados. A principal inovação

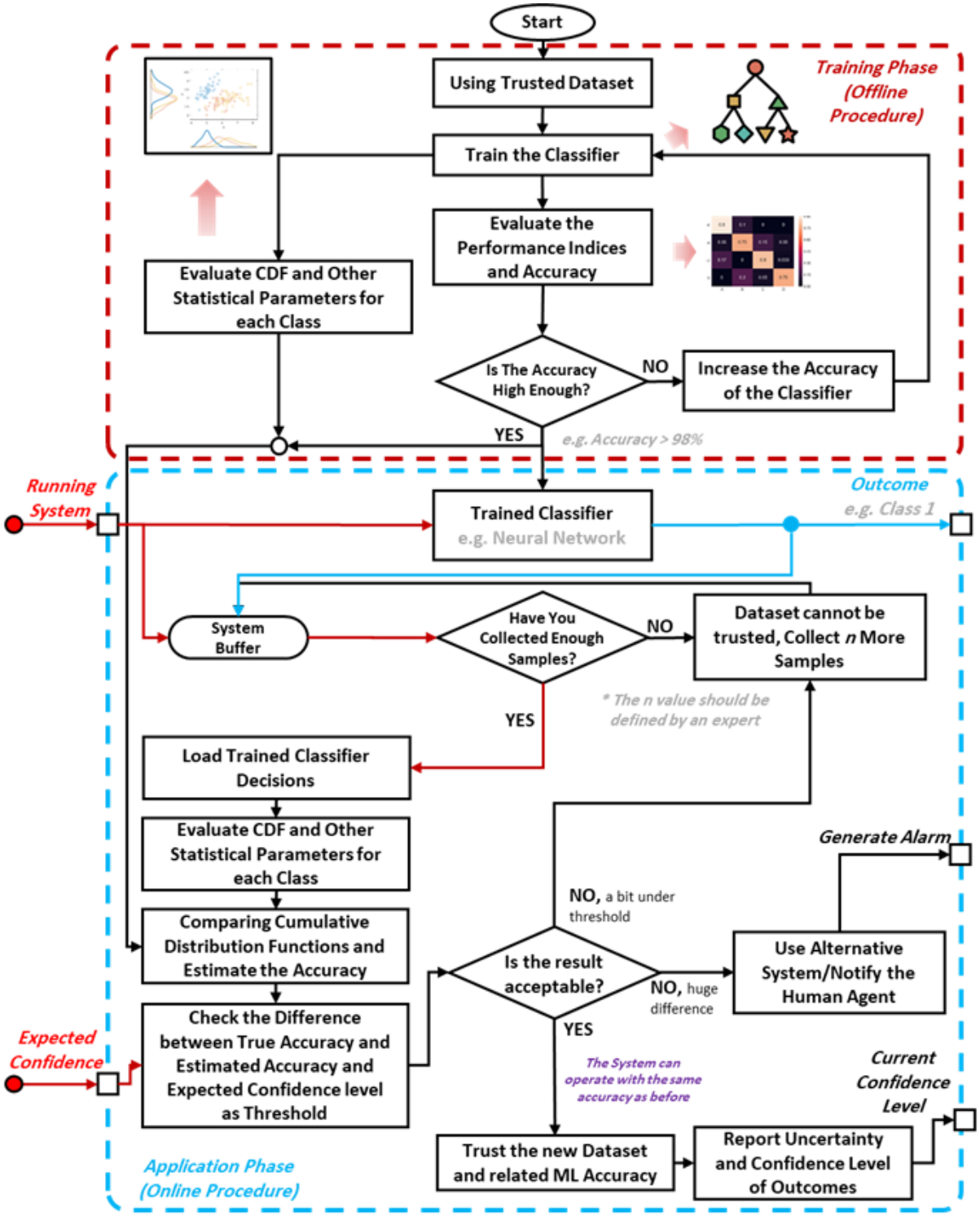

Figura 2.7: Fluxograma da abordagem SafeML: distinção entre a fase de treinamento (extração de parâmetros) e a fase de aplicação (monitoramento e decisão). Fonte: Aslansefat et al. (2020).

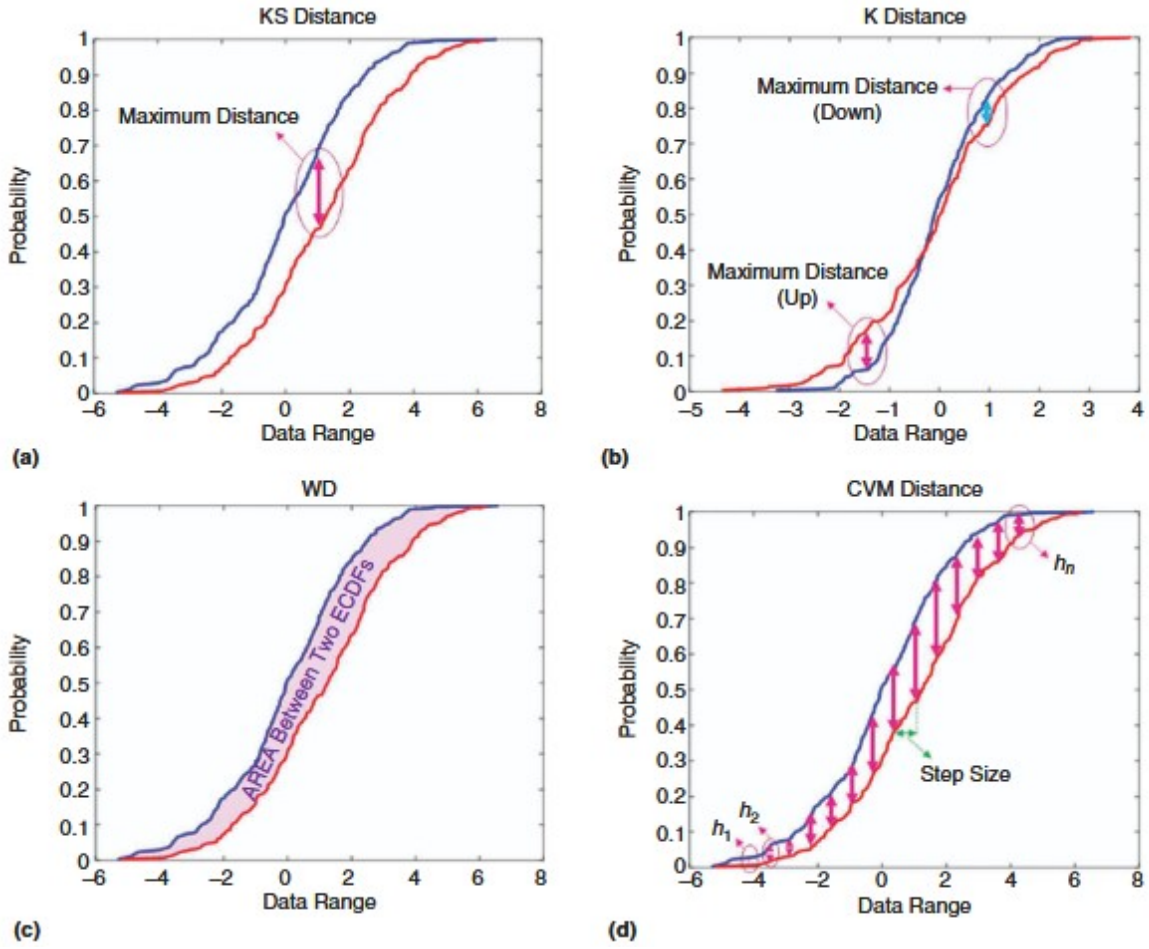

Figura 2.8: Comparação visual das distâncias estatísticas: (a) KS foca no desvio máximo; (b) Kuiper considera desvios positivos e negativos; (c) Wasserstein mede a área entre as curvas. Fonte: Aslansefat et al. (2021).

do **SafeML II** foi a introdução de um processo de validação estatística baseado em **p-value**, obtido por meio de técnicas de *bootstrapping*. O sistema realiza múltiplas reamostragens para verificar se a distância observada é estatisticamente significativa. Distâncias associadas a valores elevados de *p-value* são consideradas estatisticamente não significativas e tratadas como ruído. Conforme ilustrado na Figura 2.9, essa filtragem permite que o monitor ignore variações de fundo e concentre a análise nas características estruturais relevantes do semáforo.

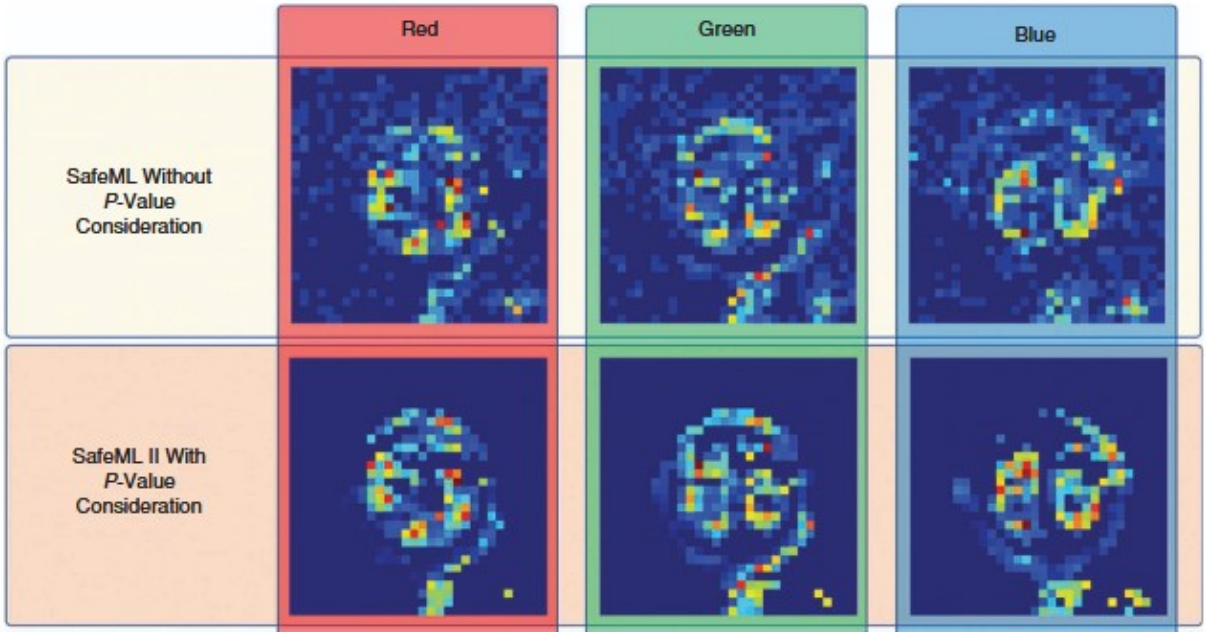

Figura 2.9: Impacto do p-value: (Topo) Sem validação estatística, o sistema é sensível a ruídos no fundo. (Baixo) Com p-value, o foco se restringe às características relevantes do objeto. Fonte: Aslansefat et al. (2021).

Protocolo de Decisão e *Human-in-the-Loop*

O objetivo final do monitoramento estatístico é apoiar a tomada de decisão segura. Aslansefat et al. (2021) definem três cenários de resposta com base no nível de confiança estimado pelo monitor:

- **Alta Confiança:** Quando a distância estatística validada é baixa, o sistema aceita automaticamente a decisão produzida pelo modelo de aprendizado.
- **Incerteza Moderada:** Quando a confiança encontra-se próxima ao limiar, o sistema pode optar por coletar mais dados, aumentar o tamanho do *buffer* ou consultar

sensores redundantes antes de tomar uma decisão definitiva.

- **Baixa Confiança (*Human-in-the-Loop*):** Quando a divergência estatística é elevada, o cenário é considerado desconhecido ou potencialmente adversarial. Nesses casos, o SafeML rejeita a decisão automática e aciona um protocolo *Human-in-the-Loop*, solicitando a intervenção do condutor ou a transição do veículo para um modo seguro.

Essa abordagem foi originalmente validada no dataset GTSRB, no qual uma CNN com elevada acurácia foi monitorada. Os resultados apresentados por Aslansefat et al. (2021) indicam que o SafeML II, utilizando a Distância de Wasserstein combinada com validação via *p-value*, é capaz de antecipar a degradação do desempenho do classificador de forma mais eficaz do que métodos baseados exclusivamente na incerteza da saída Softmax, indicando seu potencial como um mecanismo robusto de alerta para sistemas de direção autônoma.

3 Trabalhos Relacionados

A pesquisa em sistemas de percepção para veículos autônomos evoluiu significativamente na última década. A literatura, contudo, evidencia uma dicotomia clara: de um lado, trabalhos voltados à maximização de métricas de desempenho, como acurácia e precisão, no reconhecimento automático de semáforos; de outro, estudos que abordam a segurança funcional e os desafios relacionados à confiabilidade em sistemas baseados exclusivamente em *Machine Learning*. Esta seção analisa essas abordagens de forma comparativa e posiciona a proposta do SafeML nesse contexto.

No domínio específico do reconhecimento de semáforos, Jensen et al. (2016) apresentam um levantamento abrangente que descreve a evolução das abordagens empregadas na área. Trabalhos iniciais baseavam-se predominantemente em métodos heurísticos, utilizando regras manuais associadas à cor e à geometria dos semáforos. Embora eficientes sob condições controladas, tais abordagens mostraram-se frágeis diante de variações de iluminação, oclusões e cenários urbanos complexos. Como consequência, a literatura passou a adotar métodos baseados em *Machine Learning*, incluindo classificadores SVM e, mais recentemente, redes neurais convolucionais (CNNs). Os autores destacam, entretanto, que a comparação entre soluções é dificultada pela utilização recorrente de bases de dados privadas, o que motivou a criação do *LISA Traffic Light Dataset* como referência pública para avaliação de desempenho.

Apesar dos avanços obtidos em termos de acurácia, a validação da segurança desses sistemas permanece um desafio em aberto. Koopman e Wagner (2016) argumentam que abordagens tradicionais baseadas exclusivamente em testes extensivos são inadequadas para sistemas autônomos, devido à raridade estatística de eventos críticos. Nesse contexto, os autores defendem que a confiabilidade deve ser tratada como um requisito arquitetural, propondo mecanismos de monitoramento em tempo de execução capazes de supervisionar modelos de *Machine Learning* durante sua operação.

É nesse cenário que se insere o framework **SafeML**. Em sua proposta original, Aslansefat et al. (2020) introduzem um mecanismo de monitoramento estatístico em tempo

de execução, baseado em medidas de distância entre distribuições de dados, com o objetivo de identificar desvios entre os dados de treinamento e os dados observados durante a aplicação. A abordagem permite estimar a confiabilidade do classificador sem acesso à sua lógica interna, tratando o modelo como uma *caixa-preta*. Posteriormente, o SafeML II (ASLANSEFAT et al., 2021) aprimora essa estratégia ao incorporar validação estatística por meio de *p-value* e técnicas de *bootstrapping*, reduzindo a sensibilidade a ruídos irrelevantes.

A versão aprimorada do SafeML foi validada por Aslansefat et al. (2021) no contexto de veículos autônomos, utilizando o reconhecimento de placas de trânsito no dataset GTSRB. Os resultados demonstram que a Distância de Wasserstein, quando combinada com filtragem estatística via *p-value*, é capaz de antecipar a degradação do desempenho do classificador com maior precisão do que métodos baseados apenas na incerteza da saída da rede neural. Contudo, esse estudo concentra-se em objetos estáticos, cujas características visuais diferem substancialmente das encontradas em semáforos.

A análise da literatura revela, portanto, uma lacuna específica. Enquanto trabalhos como o de Jensen et al. (2016) priorizam a maximização do desempenho no reconhecimento de semáforos, e estudos como o de Aslansefat et al. (2021) investigam mecanismos de monitoramento de confiabilidade em objetos estáticos, há uma ausência de investigações que apliquem a metodologia SafeML II ao reconhecimento de semáforos. Esses objetos apresentam desafios particulares, como mudança dinâmica de estado, menor área em pixels e alta variabilidade de iluminação. O presente trabalho busca preencher essa lacuna ao aplicar e analisar o uso do SafeML II em classificadores de semáforos treinados com o dataset LISA, deslocando o foco da simples maximização de métricas de acurácia para a avaliação da confiabilidade em tempo de execução.

Além do monitoramento estatístico, uma linha de pesquisa complementar busca aumentar a confiança em sistemas de percepção por meio de técnicas de explicabilidade, inseridas no escopo da *Explainable Artificial Intelligence* (XAI). Duas das abordagens mais proeminentes na literatura são o LIME (*Local Interpretable Model-agnostic Explanations*) e o SHAP (*SHapley Additive exPlanations*). O LIME, proposto por Ribeiro et al., atua aproximando localmente o comportamento de um modelo complexo por meio de

modelos lineares mais simples e interpretáveis, permitindo entender quais características de uma imagem específica influenciaram a classificação (RIBEIRO; SINGH; GUESTRIN, 2016; LIMA, 2024). Já o SHAP, fundamentado na teoria dos jogos cooperativos, atribui valores de importância a cada atributo de entrada, oferecendo uma medida unificada de contribuição de características (LUNDBERG; LEE, 2017; NAZAT; ABDALLAH, 2024).

Embora essas ferramentas ofereçam insights valiosos sobre o funcionamento interno dos modelos, elas não operam como mecanismos ativos de detecção de anomalias ou de monitoramento estatístico em tempo real — papel desempenhado pelo SafeML II nesta pesquisa. Além disso, estudos recentes apontam que métodos como LIME e SHAP podem gerar explicações conflitantes entre si ou instáveis diante de pequenas perturbações nos dados, não garantindo a *faithfulness* necessária para aplicações críticas de segurança (JIA et al., 2022; NAZAT; ABDALLAH, 2024). A simples visualização de quais pixels foram "ativados"não é suficiente para garantir que o modelo não falhará em um cenário de trânsito inédito.

Para lidar com essa lacuna entre o desenvolvimento de modelos e a garantia formal de segurança, metodologias baseadas em *Safety Cases* têm sido propostas. O framework AMLAS (*Assurance of Machine Learning for use in Autonomous Systems*), desenvolvido por Paterson et al. (2025), estabelece um processo sistemático para justificar a segurança de componentes baseados em ML. Os autores ilustram essa metodologia com um estudo de caso aplicado a um detector de placas de pare em veículos autônomos. Eles demonstram que, para garantir a segurança, não basta apenas treinar o modelo; é necessário construir um argumento estruturado que vincule os dados de treinamento, os processos de verificação e os monitores de tempo de execução aos requisitos de segurança do sistema, assegurando que o veículo opere dentro de parâmetros aceitáveis de risco.

4 Metodologia

Este capítulo apresenta a metodologia adotada neste Trabalho de Conclusão de Curso, descrevendo o fluxo completo empregado para a análise de confiabilidade de classificadores de semáforos. Inicialmente, são apresentados os recursos computacionais e as ferramentas utilizadas para a implementação do pipeline experimental. Em seguida, detalha-se a base de dados empregada, bem como os procedimentos de preparação e organização das amostras.

Na sequência, são descritos os modelos utilizados, incluindo uma SVM linear e uma rede neural convolucional (CNN), bem como o processo de treinamento e a avaliação de desempenho básica. Por fim, apresenta-se a integração do monitor estatístico SafeML II ao pipeline proposto, detalhando como as medidas de distância são calculadas, analisadas e utilizadas para avaliar a confiabilidade das decisões dos modelos em tempo de execução. Cada uma dessas etapas é aprofundada nas seções subsequentes deste capítulo. A Figura 4.1 ilustra, de forma consolidada, o diagrama que representa o fluxo completo da metodologia adotada neste trabalho.

4.1 Tecnologias e Ferramentas

A implementação do pipeline experimental deste trabalho foi realizada integralmente em **Python 3.x**[1], linguagem amplamente utilizada em pesquisa científica e desenvolvimento de sistemas baseados em aprendizado de máquina. A escolha do Python se deve à sua

[1] ⟨https://www.python.org⟩

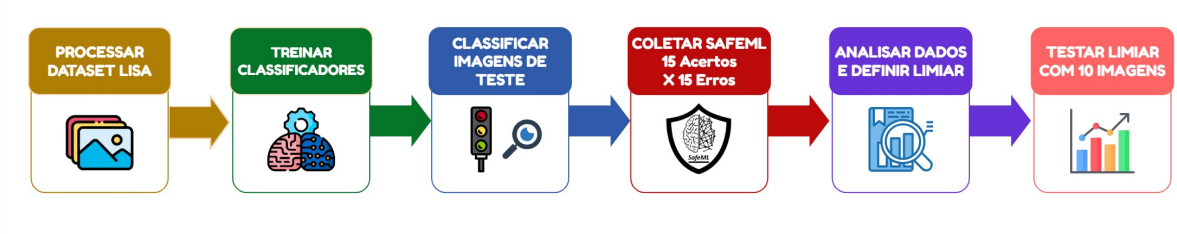

Figura 4.1: Diagrama que ilustra o fluxo da metodologia proposta neste trabalho.

vasta disponibilidade de bibliotecas consolidadas, facilidade de integração entre diferentes etapas do fluxo experimental e forte adoção na comunidade acadêmica e industrial.

Para a manipulação de imagens e processamento dos recortes provenientes do *dataset* LISA, foi utilizada a biblioteca **Pillow (PIL)**[2], que oferece suporte eficiente para leitura, conversão e salvamento de imagens. Operações numéricas e manipulação de matrizes foram realizadas com o auxílio da biblioteca **NumPy**[3], fundamental para o tratamento de dados em aplicações de visão computacional e aprendizado de máquina.

Os modelos avaliados neste trabalho foram implementados com duas bibliotecas amplamente reconhecidas. Para o classificador baseado em Máquinas de Vetores de Suporte, utilizou-se a biblioteca **scikit-learn**[4], que fornece implementações robustas de algoritmos clássicos de aprendizado supervisionado, além de ferramentas para avaliação de desempenho. Para o classificador baseado em redes neurais, empregou-se o módulo **TensorFlow/Keras**[5], escolhido por sua flexibilidade, clareza na definição de modelos e ampla utilização em aplicações de aprendizado profundo.

A análise de confiabilidade dos classificadores foi realizada por meio do framework **SafeML II**[6], seguindo a abordagem proposta por seus autores. Essa metodologia foi integrada ao pipeline por meio de módulos responsáveis pelo cálculo de distâncias estatísticas, geração de mapas de calor e avaliação de limiares de confiança. A escolha do SafeML II se deve à sua capacidade de atuar como um monitor de segurança independente do modelo, permitindo a análise de confiabilidade sem acesso à lógica interna do classificador.

Os artefatos gerados durante os experimentos, como modelos treinados e resultados intermediários, são armazenados de forma persistente para garantir reprodutibilidade. Para serialização de modelos e organização da execução dos scripts, foram utilizadas bibliotecas auxiliares como **joblib**[7] e **argparse**[8]. A estrutura de diretórios e a padronização dos caminhos do projeto são centralizadas em um arquivo de configuração, facilitando a manutenção e execução do código.

[2] ⟨https://pillow.readthedocs.io/en/stable/⟩
[3] ⟨https://numpy.org⟩
[4] ⟨https://scikit-learn.org⟩
[5] ⟨https://www.tensorflow.org⟩
[6] ⟨https://github.com/koroshAslansefat/SafeML⟩
[7] ⟨https://joblib.readthedocs.io⟩
[8] ⟨https://docs.python.org/3/library/argparse.html⟩

O versionamento do código-fonte é realizado por meio da plataforma **GitHub**[9], permitindo o controle de versões, rastreamento de alterações e organização do desenvolvimento ao longo do trabalho. Essa prática contribui para a transparência e reprodutibilidade dos experimentos apresentados neste Trabalho de Conclusão de Curso.

Todos os experimentos foram executados em ambiente local, utilizando um computador com processador **AMD Ryzen 7 7700** de 8 núcleos a 3.80 GHz, **16 GB de RAM**, unidade SSD de 932 GB (Kingston SNV2S1000G) e placa de vídeo dedicada **NVIDIA GeForce RTX 3060** com 12 GB de memória. Essa configuração permitiu realizar as etapas de treinamento, coleta e avaliação com desempenho satisfatório, embora o tempo de execução tenha sido monitorado como um fator relevante, especialmente nas análises com o framework SafeML II.

4.2 Base de Dados

Os experimentos deste trabalho utilizam o LISA Traffic Light Dataset, um conjunto de dados público amplamente empregado em pesquisas sobre reconhecimento automático de semáforos em ambientes urbanos. O dataset é composto por sequências de vídeo capturadas em condições reais de condução, abrangendo períodos diurnos e noturnos, com grande variabilidade de iluminação, distância, oclusões parciais e múltiplos semáforos visíveis simultaneamente (PHILIPSEN et al., 2015; JENSEN et al., 2016). As imagens são acompanhadas de anotações manuais que indicam, quadro a quadro, a posição da lâmpada ativa do semáforo por meio de caixas delimitadoras (*bounding boxes*), bem como o respectivo estado do sinal.

O LISA Traffic Light Dataset contempla múltiplos estados de semáforos, incluindo não apenas as classes tradicionais *go*, *warning* e *stop*, mas também variações direcionais como *goLeft*, *goForward*, *warningLeft* e *stopLeft*. Essa granularidade possibilita análises mais detalhadas sobre sinalização direcional, porém resulta em uma distribuição fortemente desbalanceada entre as classes.

Neste trabalho, todas as imagens anotadas no *dataset* são inicialmente processadas por meio do recorte automático das regiões de interesse correspondentes aos semáforos,

[9] ⟨https://github.com/SavioChermont/tcc-safeml⟩

conforme definido nas anotações originais. Esse procedimento é realizado pelo script source/lisa_dataset_builder.py, responsável por ler os arquivos de anotação, extrair as regiões delimitadas e associar cada recorte ao seu rótulo. Essa estratégia permite isolar a tarefa de **classificação do estado do semáforo**, abstraindo o problema de detecção e concentrando a análise no comportamento dos classificadores e na confiabilidade estimada pelo monitor estatístico.

A organização dos dados segue a separação entre conjuntos de treinamento e teste, respeitando a estrutura original do dataset. O conjunto de treinamento é composto por amostras provenientes das sequências `dayTrain` e `nightTrain`, enquanto o conjunto de teste utiliza imagens das sequências `daySequence1`, `daySequence2`, `nightSequence1` e `nightSequence2`. Essa separação assegura que as imagens utilizadas para avaliação não sejam vistas durante o processo de treinamento dos modelos.

A Tabela 4.1 apresenta a distribuição das classes após a etapa de curadoria. Observa-se uma forte assimetria na quantidade de amostras entre as classes principais e as classes direcionais, especialmente no conjunto de teste. Em particular, classes como *goForward* e *warningLeft* apresentam número reduzido de exemplos, o que comprometeria a robustez estatística das análises de confiabilidade baseadas na comparação entre distribuições.

Tabela 4.1: Distribuição das classes do LISA Traffic Light Dataset após a etapa de curadoria

Classe	**Treinamento**	**Teste**
go	22 946	23 777
goForward	–	205
goLeft	1 236	1 240
stop	18 382	25 936
stopLeft	7 707	5 027
warning	1 258	1 411
warningLeft	297	53

Diante dessa distribuição desigual, optou-se por restringir os experimentos às três classes principais — *go*, *warning* e *stop*. Essa decisão baseia-se na constatação de que as classes direcionais apresentam número insuficiente de amostras, sobretudo no conjunto de teste, inviabilizando uma análise estatística robusta. Além disso, as três classes selecionadas correspondem diretamente às decisões fundamentais de condução veicular — seguir,

atenção e parar — que constituem o foco deste Trabalho de Conclusão de Curso.

Além da distribuição por classe, o *dataset* contempla imagens capturadas em diferentes condições de iluminação, incluindo períodos diurnos e noturnos. Essa distinção não se apresenta de forma rigidamente separada, mas reflete variações naturais do cenário de aquisição, uma vez que fatores como iluminação, contraste, reflexos e saturação luminosa podem modificar significativamente a aparência das imagens. A Tabela 4.2 apresenta a distribuição dessas imagens nos conjuntos de treinamento e teste, evidenciando a coexistência de ambas as condições em diferentes proporções.

Tabela 4.2: Distribuição das classes do LISA Traffic Light Dataset segundo o período de captura (dia/noite)

Classe	**Treinamento**		**Teste**	
	Dia	**Noite**	**Dia**	**Noite**
go	13 830	9 116	6 959	16 818
goLeft	851	385	553	687
goForward	0	0	205	0
stop	15 113	3 269	6 962	18 974
stopLeft	6 971	736	2 884	2 143
warning	755	503	404	1 007
warningLeft	290	7	53	0

Cabe destacar que, neste trabalho, optou-se por utilizar o *dataset* conforme a divisão originalmente disponibilizada, a qual contém uma quantidade maior de imagens no conjunto de teste em relação ao conjunto de treinamento, diferindo da prática mais comum na área, que geralmente adota uma divisão aproximada de 80% para treinamento e 20% para teste. Essa escolha metodológica é intencional e está alinhada aos objetivos do estudo, uma vez que o foco principal não é maximizar o desempenho preditivo dos modelos, mas avaliar a capacidade do SafeML II em identificar cenários nos quais os classificadores operam fora de seu domínio de treinamento, mesmo quando apresentam níveis elevados de acurácia. Dessa forma, a adoção dessa separação considerada atípica fornece um cenário apropriado para a análise da confiabilidade dos modelos em condições adversas e de maior variabilidade visual, aspecto explorado nos capítulos de resultados e conclusão.

4.3 Modelos de Classificação

Esta seção descreve as etapas comuns envolvidas na preparação dos dados e no fluxo de treinamento dos classificadores utilizados neste trabalho. São apresentados os procedimentos de carregamento, pré-processamento e organização das amostras, bem como as estratégias adotadas para garantir consistência entre os experimentos. Esses passos constituem a base sobre a qual as abordagens específicas de cada modelo são posteriormente detalhadas nas subseções seguintes.

O carregamento dos dados é realizado por meio da função `load_split`, definida no módulo `data_utils.py`. Essa função percorre os diretórios correspondentes aos conjuntos de treinamento ou teste, lendo as imagens previamente recortadas. Durante essa etapa, são consideradas apenas as classes definidas na lista de permissões (*whitelist*), configurada no arquivo `config.py` como {*go*, *stop*, *warning*}, garantindo consistência entre os experimentos e foco nas classes principais analisadas neste trabalho.

Cada imagem é carregada no formato RGB, convertida para representação em ponto flutuante e normalizada para o intervalo $[0, 1]$. As imagens possuem tamanho fixo de 30×30 pixels, conforme definido na configuração global do projeto. Essa padronização assegura que todos os exemplos de entrada possuam a mesma dimensionalidade, independentemente do modelo utilizado.

No caso do classificador SVM, as imagens normalizadas são achatadas em vetores unidimensionais de 2 700 posições ($30 \times 30 \times 3$), formando o conjunto de características de entrada do modelo. Para a CNN, as imagens são mantidas inicialmente em sua estrutura bidimensional com canais de cor, sendo reorganizadas posteriormente no formato $30 \times 30 \times 3$ durante a etapa específica de treinamento da rede neural.

Com o objetivo de mitigar o desbalanceamento entre classes no conjunto de treinamento, é aplicado um limite máximo de amostras por classe, definido pela constante `MAX_PER_CLASS_TRAIN`. Neste trabalho, esse valor foi fixado em 3 000 exemplos por classe. Dessa forma, classes majoritárias como *go* e *stop* têm seu volume reduzido, enquanto a classe *warning* é utilizada em sua totalidade, uma vez que possui menor número de amostras disponíveis. O conjunto de teste, por sua vez, é carregado sem restrições, utilizando todas as imagens disponíveis para cada classe.

Após o carregamento e pré-processamento, os dados de treinamento são armazenados em arquivos de cache no formato `.npz`, contendo as amostras de entrada, os rótulos correspondentes e a lista de classes consideradas. Esses arquivos são salvos em diretórios específicos para cada modelo e utilizados tanto na etapa de treinamento quanto nas análises posteriores de confiabilidade com o SafeML II. Essa estratégia evita o reprocessamento repetido do dataset e garante que as mesmas representações de dados sejam utilizadas de forma consistente ao longo de todo o pipeline experimental.

4.3.1 SVM

Classificador SVM Linear

O classificador baseado em Máquinas de Vetores de Suporte (SVM) foi implementado utilizando uma fronteira de decisão linear por meio da biblioteca `scikit-learn`. A opção por um modelo linear é adequada ao contexto deste trabalho, pois as imagens são representadas em um espaço de características de alta dimensionalidade, no qual separações lineares costumam apresentar bom desempenho com menor custo computacional.

A implementação é realizada por meio de um *pipeline* que combina uma etapa de padronização das características com o algoritmo de classificação. A padronização é feita utilizando o `StandardScaler`, cuja função é ajustar a escala das variáveis de entrada para que todas apresentem variância comparável. Essa etapa é importante porque o modelo é sensível à escala das características: valores muito grandes podem dominar o processo de otimização, prejudicando a qualidade da fronteira de decisão.

O `StandardScaler` é configurado com a opção *with_mean=False*, o que significa que os dados não são centralizados em torno da média. Essa escolha é necessária porque as imagens são representadas como vetores de pixels não negativos, normalizados no intervalo $[0, 1]$. Subtrair a média desses vetores poderia alterar a interpretação direta dos valores de intensidade dos pixels, além de não trazer benefícios práticos para esse tipo de representação. Assim, mantém-se a distribuição original dos dados, ao mesmo tempo em que se controla sua escala.

Para a etapa de classificação, utiliza-se o algoritmo `LinearSVC`. Diferentemente do `SVC` com kernel linear, o `LinearSVC` é otimizado especificamente para classificadores

lineares em espaços de alta dimensionalidade, oferecendo maior eficiência computacional e menor consumo de memória. Essa característica é especialmente relevante quando se trabalha com vetores de entrada longos, como no caso das imagens achatadas utilizadas neste trabalho.

Além disso, o `LinearSVC` lida automaticamente com o problema multiclasse por meio da estratégia *one-vs-rest*, treinando um classificador binário para cada classe em relação às demais. Essa abordagem é simples, eficiente e amplamente utilizada em aplicações práticas de classificação.

Após o treinamento, o modelo é avaliado no conjunto de teste utilizando métricas de desempenho padrão, permitindo verificar o comportamento do classificador antes da aplicação das técnicas de análise de confiabilidade. O modelo treinado é então persistido em disco, possibilitando sua reutilização nas etapas posteriores do pipeline experimental.

4.3.2 CNN

O segundo modelo avaliado neste trabalho é um classificador baseado em Rede Neural Convolucional (CNN), implementado com o objetivo de explorar explicitamente a estrutura espacial das imagens de semáforos. Diferentemente do SVM, que opera sobre vetores de características achatados, a CNN processa as imagens preservando a organização bidimensional dos pixels e seus canais de cor, característica particularmente relevante para tarefas de visão computacional.

As imagens de entrada, inicialmente representadas como vetores unidimensionais, são reorganizadas no formato $30 \times 30 \times 3$ antes do treinamento da rede neural. Os rótulos, originalmente representados como cadeias de caracteres, são convertidos para índices numéricos e, em seguida, codificados no formato *one-hot*. Essa representação é necessária para o treinamento supervisionado da rede, uma vez que a função de perda utilizada assume uma codificação categórica das classes.

A arquitetura da CNN adotada neste trabalho é propositalmente simples, visando reduzir o risco de sobreajuste e facilitar a análise de confiabilidade. O modelo é composto por três camadas convolucionais empilhadas, intercaladas com camadas de subamostragem (*max pooling*), permitindo a extração progressiva de características visuais de baixo

para alto nível. Após a etapa convolucional, os mapas de características são transformados em um vetor unidimensional e submetidos a uma camada de *dropout*, utilizada como mecanismo de regularização para reduzir a dependência excessiva de neurônios específicos durante o treinamento.

Na fase final, uma camada densa intermediária é empregada para combinar as características extraídas, seguida por uma camada de saída com função de ativação *softmax*. Essa configuração permite que o modelo produza uma distribuição de probabilidades sobre as classes consideradas, possibilitando a classificação do estado do semáforo em *go*, *stop* ou *warning*.

O treinamento da CNN é realizado utilizando o otimizador Adam, escolhido por sua eficiência e ampla adoção em aplicações de aprendizado profundo. A função de perda empregada é a *categorical cross-entropy*, adequada para problemas de classificação multiclasse com rótulos categóricos. Os hiperparâmetros de treinamento foram definidos de forma conservadora, priorizando a estabilidade do aprendizado e a redução do risco de sobreajuste. Em particular, adotou-se um número limitado de épocas de treinamento (15 épocas), evitando ajustes excessivos aos dados de treinamento, bem como um tamanho de lote intermediário (32 amostras), que favorece um compromisso entre estabilidade na estimativa do gradiente e capacidade de generalização. Além disso, a arquitetura da rede foi mantida relativamente simples, com três camadas convolucionais contendo um número moderado de filtros (32, 64 e 128), seguidas de uma camada de *dropout* com taxa de 0,5, utilizada como mecanismo de regularização. Essas escolhas refletem uma abordagem conservadora ao evitar arquiteturas excessivamente profundas ou longos períodos de treinamento, especialmente considerando o tamanho reduzido do conjunto de treinamento e o objetivo deste trabalho de analisar o comportamento e a confiabilidade dos modelos, em vez de maximizar exclusivamente o desempenho preditivo.

Após o treinamento, o desempenho da rede é avaliado no conjunto de teste por meio de métricas padrão de classificação, permitindo analisar o comportamento da CNN antes da aplicação das técnicas de monitoramento de confiabilidade. O modelo treinado é então persistido em disco para uso posterior.

Além do salvamento do modelo, os dados de treinamento são armazenados em

formato compatível com as etapas de análise do SafeML II. Essa decisão garante que a análise estatística de confiabilidade seja realizada sobre a mesma representação de dados utilizada durante o treinamento da rede, assegurando consistência entre os modelos avaliados e permitindo uma comparação direta entre os resultados obtidos com a CNN e com o classificador SVM.

4.4 Predição e Coleta de Dados com SafeML

Após o treinamento dos classificadores, a etapa seguinte do pipeline consiste na aplicação do monitor estatístico SafeML II para a análise de confiabilidade das predições realizadas pelos modelos. Essa etapa tem como objetivo quantificar o grau de divergência entre os dados utilizados no treinamento e as amostras observadas durante a fase de aplicação, permitindo avaliar se o classificador está operando em um regime conhecido ou potencialmente inseguro.

A coleta das métricas de confiabilidade é realizada por meio do *script* source/-safeml_collect.py, que opera de forma independente do modelo avaliado. O *script* recebe como parâmetro o tipo de classificador analisado (*svm* ou *cnn*) e utiliza automaticamente os caminhos padrão definidos no arquivo `config.py` para localizar o modelo treinado e o cache de dados de treinamento. Opcionalmente, podem ser aplicados filtros para restringir a análise a imagens capturadas exclusivamente durante o dia ou durante a noite, possibilitando investigações específicas sobre o impacto das condições de iluminação.

Inicialmente, carrega o conjunto de dados de treinamento previamente armazenado em cache, contendo as amostras de entrada, os rótulos correspondentes e a lista de classes consideradas. Esse cache utiliza exatamente a mesma representação dos dados empregada no treinamento dos classificadores, garantindo consistência entre as etapas do pipeline. Em seguida, o conjunto de teste é carregado e utilizado como entrada para o modelo treinado, que realiza a predição do estado do semáforo para cada imagem.

Com base nos rótulos reais e nas predições do modelo, as amostras do conjunto de teste são separadas, para cada classe, em dois subconjuntos distintos: amostras classificadas corretamente e amostras classificadas incorretamente. Essa separação é fundamental para a análise proposta, pois permite comparar o comportamento estatístico do modelo

em situações nas quais a decisão foi correta e em situações nas quais ocorreu erro de classificação.

Para viabilizar o cálculo das métricas estatísticas, é realizada uma amostragem limitada dos dados, controlada por um parâmetro de configuração. Para cada classe, são selecionadas até um número máximo de amostras do conjunto de treinamento, bem como das amostras corretas e incorretas do conjunto de teste. Essa estratégia reduz o custo computacional da análise e permite manter a comparação estatística equilibrada entre os conjuntos.

A partir dessas amostras, o SafeML II calcula, para cada classe, a Distância de Wasserstein (WD) entre a distribuição dos pixels do conjunto de treinamento e a distribuição dos pixels das amostras classificadas incorretamente. Essa implementação de WD foi incorporada diretamente da biblioteca do framework SafeML, com base na formulação original proposta pelos autores. A implementação adaptada para Python está localizada no módulo `Wasserstein_Dist_PVal.py`.

Em paralelo, é aplicado um procedimento de validação estatística baseado em *bootstrap* e *p-value*, permitindo identificar quais diferenças são estatisticamente significativas. O valor de p é calculado através de 1.000 reamostragens dos dados, onde distâncias maiores que a original são contadas para determinar a significância. Distâncias com p-value maior que 0.05 são consideradas como ruído e descartadas da análise. Essa filtragem ajuda a reduzir a influência de variações irrelevantes e foca apenas nas diferenças estatisticamente significativas.

Além dos mapas de distância, o *script* gera estatísticas resumidas por canal de cor, incluindo a média das distâncias estatisticamente significativas e o número de *pixels* que apresentam divergência relevante. Essas estatísticas são calculadas tanto para amostras classificadas incorretamente quanto para amostras classificadas corretamente, sendo estas últimas utilizadas como uma referência de comportamento esperado do modelo.

Todos os resultados da coleta SafeML são organizados em uma estrutura de dados que inclui contagens de amostras, estatísticas por classe e por canal, mapas de distância e os caminhos das imagens utilizadas nas amostras incorretas. Essas informações são armazenadas em arquivos persistentes no diretório de artefatos do projeto, possibilitando

sua reutilização nas etapas subsequentes de análise visual e avaliação experimental de limiares de confiança.

Essa etapa estabelece, portanto, a ponte entre as predições realizadas pelos classificadores e a análise de confiabilidade proposta neste trabalho, fornecendo tanto métricas quantitativas quanto subsídios visuais para a interpretação do comportamento dos modelos em cenários corretos e incorretos.

4.5 Análise dos Dados

Após a coleta das métricas estatísticas de confiabilidade por meio do SafeML II, torna-se necessário interpretar esses dados de forma estruturada. Nesta etapa, a análise é conduzida sob duas perspectivas complementares: uma análise visual, baseada em mapas de calor que evidenciam regiões críticas das imagens, e uma análise experimental baseada em limiares, que avalia o comportamento do monitor estatístico em um cenário simplificado de decisão.

4.5.1 Análise Visual por Mapas de Calor

A análise visual tem como objetivo fornecer uma interpretação intuitiva dos valores calculados pelo SafeML II, auxiliando na compreensão de quais regiões da imagem mais contribuem para a divergência estatística observada nos casos de erro de classificação. Essa etapa é realizada pelo código `source/safeml_heatmaps.py`, que opera exclusivamente sobre os resultados previamente salvos durante a fase de coleta, sem recalcular quaisquer métricas estatísticas.

O *script* lê o arquivo `safeml_results.npz`, que contém, para cada classe, os mapas bidimensionais da WD calculados pixel a pixel e por canal de cor (RGB). Dois tipos de mapas são considerados: o mapa completo de distâncias, que representa a intensidade relativa da divergência estatística em cada *pixel*, e o mapa filtrado por significância estatística, no qual valores associados a *pixels* com *p-value* maior ou igual a 0,05 são zerados, de acordo com o limiar adotado neste trabalho.

A partir desses dados, são geradas visualizações em forma de mapas de calor no

formato 30×30, correspondentes à resolução das imagens utilizadas nos experimentos. Os mapas são produzidos individualmente para cada canal de cor, bem como em uma versão agregada, obtida pela média dos três canais, permitindo observar tanto padrões cromáticos específicos quanto tendências espaciais globais.

É importante destacar que os mapas de calor utilizam uma escala de cores normalizada para fins de visualização. Dessa forma, regiões exibidas em tons mais claros (como amarelo ou branco) não indicam valores absolutos da Distância de Wasserstein próximos de 1, mas sim os pixels que apresentam maior divergência estatística relativa em comparação aos demais pixels da mesma imagem. Os valores absolutos de WD permanecem moderados, refletindo o fato de que, mesmo nos casos de erro, as imagens analisadas ainda pertencem ao mesmo domínio visual do conjunto de treinamento.

Os mapas, que serão apresentados no próximo capítulo deste trabalho, não têm finalidade preditiva, mas desempenham um papel explicativo importante. Eles permitem identificar regiões da imagem onde pequenas variações nos valores dos pixels resultam em diferenças estatísticas mais pronunciadas em relação ao perfil aprendido durante o treinamento, indicando potenciais fontes de incerteza para o classificador. Dessa forma, a análise visual complementa as métricas numéricas ao fornecer subsídios qualitativos para a interpretação do comportamento do modelo sob condições adversas.

4.5.2 Avaliação Experimental por Definição de Limiar

Além da análise visual, é realizada uma avaliação experimental simplificada com base na definição de limiares de aceitação para as métricas de confiabilidade calculadas pelo SafeML II. Essa etapa tem caráter exploratório e visa simular, de forma controlada, como um monitor estatístico poderia ser utilizado para aceitar ou rejeitar decisões de um classificador em tempo de execução.

Essa avaliação é conduzida pelo código `source/safeml_eval_threshold.py`. Diferentemente da etapa de coleta, esse *script* não utiliza diretamente o modelo classificador treinado. Em vez disso, ele opera exclusivamente sobre o conjunto de treinamento previamente armazenado em cache e sobre amostras do conjunto de teste, focando apenas no comportamento estatístico dos dados.

O procedimento consiste na seleção de pequenos *buffers* de imagens do conjunto de teste, agrupados por classe, contendo um número fixo de amostras sequenciais. Para cada classe, é calculada a média da WD estatisticamente significativa entre as distribuições dos *pixels* do conjunto de treinamento e as distribuições dos *pixels* presentes no *buffer*. Esse cálculo é realizado por canal de cor e posteriormente agregado em um único valor representativo por classe.

Os valores obtidos são então comparados a limiares previamente definidos para cada classe. Caso a distância média fique abaixo do limiar estabelecido, o *buffer* é considerado estatisticamente consistente com os dados de treinamento, sendo aceito pelo monitor. Caso contrário, o *buffer* é rejeitado, indicando um possível cenário fora do domínio conhecido pelo modelo.

Para reduzir o custo computacional, são aplicadas subamostragens tanto no número de imagens do conjunto de treinamento quanto no número de pixels utilizados no cálculo, sem comprometer o objetivo exploratório da análise. O *script* permite ainda a aplicação de filtros para avaliar separadamente imagens capturadas durante o dia ou durante a noite, possibilitando investigações específicas sobre o impacto das condições de iluminação.

Essa etapa não tem como objetivo medir desempenho preditivo, mas sim avaliar a viabilidade prática do uso de métricas estatísticas como critério auxiliar de decisão. Os resultados obtidos servem como base para discutir o potencial do SafeML II como ferramenta de apoio à confiabilidade em sistemas de Visão Computacional embarcados.

5 Resultados

5.1 Resultados do Treinamento dos Classificadores

Esta seção apresenta os resultados obtidos com o treinamento e a avaliação dos classificadores, considerando métricas tradicionais de desempenho. Conforme descrito no Capítulo 4, o conjunto de treinamento foi balanceado por meio da limitação do número máximo de amostras por classe, enquanto o conjunto de teste foi utilizado em sua totalidade, sem qualquer restrição, refletindo de forma mais fiel a distribuição real dos dados.

As métricas apresentadas nas tabelas a seguir são amplamente utilizadas na avaliação de modelos de classificação. Além da acurácia, precisão e revocação — já explicadas no Capítulo 2 (Fundamentação Teórica) —, utilizamos uma métrica complementar chamada *F1-score*.

O *F1-score* representa a média harmônica entre a precisão e a revocação, sendo especialmente útil em cenários com classes desbalanceadas, pois oferece uma medida equilibrada entre ambos os aspectos. Por fim, a coluna *Quantidade* indica o número de amostras reais de cada classe no conjunto de teste, o que permite contextualizar a representatividade estatística dos resultados.

5.1.1 Resultados da CNN

A CNN apresentou acurácia de **98,39%** no conjunto de teste. A Tabela 5.1 apresenta o desempenho do modelo por classe.

Tabela 5.1: Desempenho da CNN no conjunto de teste

Classe	**Precisão**	**Revocação**	**F1-score**	**Quantidade**
go	1.00	0.98	0.99	23 777
stop	0.99	0.99	0.99	25 936
warning	0.76	0.95	0.84	1 411
Acurácia geral		0.9839		

Observa-se que as classes *go* e *stop* apresentaram valores elevados de precisão e revocação, indicando que o modelo foi capaz de aprender de forma eficaz os padrões visuais

associados a esses estados do semáforo. Em contraste, a classe *warning* apresentou alta revocação e menor precisão, o que indica que o modelo tende a identificar a maioria dos semáforos amarelos, porém ao custo de um maior número de falsas detecções.

Esse comportamento é particularmente relevante em um contexto de segurança, pois indica que o modelo prefere errar por excesso — classificando imagens como *warning* mesmo quando não são — em vez de deixar de detectar um semáforo amarelo existente.

5.1.2 Resultados do Classificador SVM Linear

O classificador SVM Linear alcançou acurácia de **98,11%** no conjunto de teste. A Tabela 5.2 apresenta o desempenho do modelo.

Tabela 5.2: Desempenho do SVM Linear no conjunto de teste

Classe	**Precisão**	**Revocação**	**F1-score**	**Quantidade**
go	0.98	0.99	0.99	23 777
stop	0.99	0.98	0.98	25 936
warning	0.83	0.88	0.85	1 411
Acurácia geral		0.9811		

Assim como observado na CNN, o SVM apresentou desempenho elevado nas classes *go* e *stop*. Para a classe *warning*, o modelo apresentou valores mais equilibrados entre precisão e revocação, ainda que com desempenho inferior às classes majoritárias.

5.1.3 Análise Comparativa

A comparação entre os dois modelos mostra que tanto ambos atingiram níveis elevados de desempenho segundo métricas tradicionais. A CNN apresenta ligeira vantagem em termos de acurácia geral, enquanto o SVM Linear demonstra desempenho competitivo considerando sua simplicidade estrutural.

Entretanto, a análise por classe evidencia que métricas globais, como a acurácia, podem mascarar fragilidades relevantes. Em ambos os modelos, a classe *warning* apresenta desempenho inferior às demais, reforçando que avaliações baseadas apenas em acertos globais não são suficientes para caracterizar a segurança do sistema. Essa limitação motiva a análise de confiabilidade apresentada nas seções seguintes, por meio da aplicação do monitor estatístico SafeML II.

5.2 Resultados da Coleta SafeML II para o SVM

Nesta seção são apresentados os resultados da aplicação do SafeML II ao classificador SVM, considerando separadamente cenários diurnos e noturnos. A análise é conduzida sobre as três classes utilizadas neste trabalho (*go*, *stop* e *warning*). Para cada classe, são analisadas amostras corretamente e incorretamente classificadas, sendo selecionadas até 15 imagens por grupo, conforme a disponibilidade no conjunto de teste. Os resultados são avaliados por meio da Distância de Wasserstein, com o objetivo de caracterizar o comportamento estatístico do classificador e definir limiares de aceitação específicos para cada cenário.

O tempo médio necessário para a execução completa dessa etapa de coleta, incluindo a seleção das amostras, o cálculo da Distância de Wasserstein pixel a pixel e a aplicação do critério estatístico de significância, foi de aproximadamente 12 minutos e 20 segundos por cenário (diurno ou noturno).

5.2.1 Resultados para Imagens Diurnas

Amostras Classificadas Incorretamente

A Tabela 5.3 apresenta os valores médios da Distância de Wasserstein, calculados separadamente para os canais RGB, considerando apenas pixels estatisticamente significativos. Observa-se que, para todas as classes, os valores de WD são elevados, indicando forte divergência estatística em relação ao conjunto de treinamento.

Tabela 5.3: Distância de Wasserstein média por canal (RGB) para amostras incorretas — SVM (diurno)

Classe	Canal R	Canal G	Canal B
go	0.334	0.319	0.334
stop	0.277	0.319	0.507
warning	0.459	0.478	0.573

Destaca-se a classe *warning*, que apresenta os maiores valores de WD em todos os canais, sugerindo que erros nessa classe estão associados a mudanças estatísticas mais intensas no padrão visual das imagens.

Amostras Classificadas Corretamente

A Tabela 5.4 mostra os valores correspondentes às amostras corretamente classificadas. Em comparação com os erros, os valores de WD são significativamente menores, indicando maior proximidade estatística entre as imagens de teste e o conjunto de treinamento quando o classificador acerta.

Tabela 5.4: Distância de Wasserstein média por canal (RGB) para amostras corretas — SVM (diurno)

Classe	Canal R	Canal G	Canal B
go	0.115	0.196	0.207
stop	0.125	0.086	0.078
warning	0.190	0.200	0.265

A separação clara entre os valores de WD obtidos para acertos e erros indica que a métrica é sensível a mudanças relevantes no domínio das imagens diurnas, mesmo quando a acurácia do classificador é elevada.

Definição do Limiar

Para cada classe, foi calculado um valor global de WD como a média aritmética dos três canais RGB. A Tabela 5.5 apresenta esses valores globais para amostras corretamente e incorretamente classificadas no cenário diurno. O limiar de aceitação foi definido como a média entre os dois valores globais (acertos e erros) de cada classe.

Tabela 5.5: Média global (RGB) da WD para acertos e erros no cenário diurno (SVM)

Classe	WD global (acertos)	WD global (erros)	Limiar
go	0.173	0.329	0.251
stop	0.097	0.368	0.233
warning	0.219	0.504	0.362

5.2.2 Resultados para Imagens Noturnas

Amostras Classificadas Incorretamente

A Tabela 5.6 apresenta os valores médios da WD significativa para as amostras incorretamente classificadas no cenário noturno. Os valores correspondem à média por canal RGB, calculada a partir das amostras de erro coletadas para cada classe.

Tabela 5.6: Distância de Wasserstein média por canal (RGB) para amostras incorretas — SVM (noturno)

Classe	Canal R	Canal G	Canal B
go	0.240	0.222	0.220
stop	0.342	0.224	0.204
warning	0.335	0.223	0.215

Observa-se que, para todas as classes, os valores de WD associados às amostras incorretas são relativamente elevados, indicando uma divergência estatística significativa entre essas imagens noturnas e a distribuição aprendida durante o treinamento. Esse efeito é mais pronunciado nas classes *stop* e *warning*, refletindo a maior dificuldade do modelo em lidar com variações de iluminação, reflexos e ruídos presentes no período noturno.

Amostras Classificadas Corretamente

A Tabela 5.7 apresenta os valores médios da Distância de Wasserstein significativa para as amostras corretamente classificadas no cenário noturno.

Tabela 5.7: Distância de Wasserstein média por canal (RGB) para amostras corretas — SVM (noturno)

Classe	Canal R	Canal G	Canal B
go	0.152	0.197	0.219
stop	0.259	0.165	0.167
warning	0.174	0.161	0.106

Em comparação com as amostras incorretas, as amostras corretamente classificadas apresentam valores de WD consistentemente menores, indicando maior similaridade estatística com os dados de treinamento. Ainda assim, nota-se que os valores globais de WD no cenário noturno são, em média, superiores aos observados no cenário diurno, evidenciando o impacto das condições de baixa iluminação mesmo quando a classificação é correta.

Definição do Limiar

Para a definição do limiar de decisão no cenário noturno, foi calculada a média global da Distância de Wasserstein para cada classe, obtida a partir da média dos três canais RGB. Em seguida, o limiar foi definido como a média aritmética entre o valor global das amostras corretamente classificadas e o valor global das amostras incorretamente classificadas.

A Tabela 5.8 resume os valores utilizados no cálculo do limiar para cada classe.

Tabela 5.8: Média global RGB da Distância de Wasserstein e limiar definido — SVM (noturno)

Classe	**WD global (acertos)**	**WD global (erros)**	**Limiar**
go	0.189	0.227	0.208
stop	0.197	0.257	0.227
warning	0.147	0.258	0.203

Os resultados confirmam que, também no cenário noturno, existe uma separação estatística clara entre amostras corretas e incorretas. A definição de limiares específicos para esse contexto permite ao SafeML II sinalizar situações de maior incerteza de forma mais precisa, reforçando a importância de considerar o período do dia como um fator relevante na análise de confiabilidade de sistemas de visão computacional embarcados.

5.3 Resultados da Coleta SafeML II para a CNN

Esta seção apresenta os resultados da análise de confiabilidade do classificador baseado em CNN utilizando o SafeML II. A análise foi realizada separadamente para os cenários diurno e noturno, considerando as três classes de interesse (*go*, *stop* e *warning*) e amostras corretamente e incorretamente classificadas.

Assim como na análise do SVM, para cada classe foram selecionadas até 15 amostras representativas de acertos e erros, permitindo a comparação das distâncias estatísticas associadas a cada situação. Os valores apresentados correspondem à Distância de Wasserstein significativa (com $p < 0{,}05$), calculada a partir da média dos canais RGB.

O tempo médio necessário para a execução completa dessa etapa de coleta, que envolve a seleção das amostras, o cálculo da Distância de Wasserstein pixel a pixel e a aplicação do critério estatístico de significância, foi de aproximadamente 12 minutos e 40 segundos por cenário (diurno ou noturno) para o classificador CNN.

5.3.1 Resultados no Cenário Diurno

Amostras Classificadas Incorretamente

A Tabela 5.9 apresenta os valores médios da Distância de Wasserstein por canal RGB para as amostras incorretamente classificadas no cenário diurno.

Tabela 5.9: Distância de Wasserstein média por canal (RGB) para amostras incorretas — CNN (diurno)

Classe	Canal R	Canal G	Canal B
go	0.534	0.492	0.490
stop	0.211	0.230	0.348
warning	–	–	–

Observa-se que, no cenário diurno, a classe *go* apresenta valores elevados de WD nas amostras incorretas, indicando divergência significativa entre essas imagens e a distribuição aprendida durante o treinamento. Para a classe *warning*, não foram observadas amostras incorretamente classificadas nesse cenário, o que impossibilita a definição direta de um limiar diurno para essa classe.

Amostras Classificadas Corretamente

A Tabela 5.10 apresenta os valores médios da Distância de Wasserstein para as amostras corretamente classificadas no cenário diurno.

Tabela 5.10: Distância de Wasserstein média por canal (RGB) para amostras corretas — CNN (diurno)

Classe	Canal R	Canal G	Canal B
go	0.123	0.178	0.185
stop	0.140	0.111	0.120
warning	0.215	0.224	0.250

Em todas as classes, os valores associados às amostras corretamente classificadas são substancialmente inferiores aos observados nas amostras incorretas, evidenciando a capacidade do SafeML II em diferenciar estatisticamente decisões confiáveis de situações de erro.

Definição do Limiar

A definição do limiar diurno foi realizada a partir da média global da Distância de Wasserstein, obtida pela média dos três canais RGB. O limiar é definido como a média aritmética entre os valores globais das amostras corretas e incorretas.

A Tabela 5.11 resume os valores utilizados nesse processo. Para a classe *warning*, como não houve erros no cenário diurno, adota-se o limiar definido a partir do cenário noturno.

Tabela 5.11: Média global RGB da Distância de Wasserstein e limiar definido — CNN (diurno)

Classe	WD global (acertos)	WD global (erros)	Limiar
go	0.162	0.505	0.334
stop	0.124	0.263	0.194
warning	–	–	0.292 (limiar noturno)

5.3.2 Resultados no Cenário Noturno

Amostras Classificadas Incorretamente

A Tabela 5.12 apresenta os valores médios da Distância de Wasserstein para as amostras incorretamente classificadas no cenário noturno.

Tabela 5.12: Distância de Wasserstein média por canal (RGB) para amostras incorretas — CNN (noturno)

Classe	Canal R	Canal G	Canal B
go	0.301	0.260	0.237
stop	0.464	0.359	0.279
warning	0.607	0.445	0.281

No cenário noturno, observa-se um aumento expressivo das distâncias estatísticas, especialmente para a classe *warning*, indicando maior sensibilidade da CNN às variações de iluminação e ruído presentes nesse contexto.

Análise das Amostras Classificadas Corretamente

A Tabela 5.13 apresenta os valores médios da Distância de Wasserstein para as amostras corretamente classificadas no cenário noturno.

Tabela 5.13: Distância de Wasserstein média por canal (RGB) para amostras corretas — CNN (noturno)

Classe	Canal R	Canal G	Canal B
go	0.164	0.203	0.224
stop	0.261	0.163	0.165
warning	0.159	0.140	0.119

Apesar das condições adversas, as amostras corretamente classificadas mantêm valores de WD inferiores aos observados nas amostras incorretas, reforçando a consistência do monitoramento estatístico.

Definição do Limiar para o Cenário Noturno

A Tabela 5.14 apresenta os valores globais de WD e os limiares definidos para o cenário noturno.

Tabela 5.14: Média global RGB da Distância de Wasserstein e limiar definido — CNN (noturno)

Classe	WD global (acertos)	WD global (erros)	Limiar
go	0.197	0.266	0.232
stop	0.196	0.367	0.281
warning	0.139	0.444	0.292

Os resultados indicam que a definição de limiares específicos para o cenário noturno é essencial, especialmente para classes mais sensíveis à iluminação, como *warning*. A adoção desses limiares permite que o SafeML II sinalize de forma mais precisa situações de maior incerteza, reforçando a importância da separação entre cenários diurnos e noturnos na análise de confiabilidade.

5.4 Análise Visual via *Heatmaps*

Uma das vantagens da abordagem SafeML II é permitir uma análise visual das regiões da imagem que mais contribuem para a divergência estatística entre os dados de treino e as amostras classificadas incorretamente. Essa análise é realizada por meio de *heatmaps* baseados na Distância de Wasserstein, calculada pixel a pixel, permitindo interpretar quais partes do semáforo e do contexto visual influenciam a perda de confiabilidade do classificador. Ressalta-se que a escala de cores utilizada nos mapas é normalizada para

fins de visualização: valores próximos de 1 indicam regiões de maior divergência *relativa* dentro da imagem, e não valores absolutos da Distância de Wasserstein.

Em cada visualização, são apresentados dois mapas para cada classe: (i) um mapa considerando todas as distâncias calculadas e (ii) um mapa filtrado pelo critério estatístico de significância ($p < 0{,}05$), no qual apenas os pixels cuja divergência é estatisticamente relevante são mantidos. Dessa forma, o segundo mapa evidencia exclusivamente as regiões que representam mudanças estruturais relevantes em relação ao conjunto de treino.

5.4.1 Heatmaps para o SVM (Cenário Diurno)

A Figura 5.1 apresenta os *heatmaps* gerados para a classe *stop* no cenário diurno. Observa-se que as maiores divergências estatísticas concentram-se na região da lâmpada vermelha e em áreas adjacentes. Após a aplicação do filtro de significância ($p < 0{,}05$), essas regiões permanecem bem definidas, indicando que a divergência observada não se deve a ruído aleatório, mas a variações estruturais relevantes em relação ao padrão aprendido no treinamento.

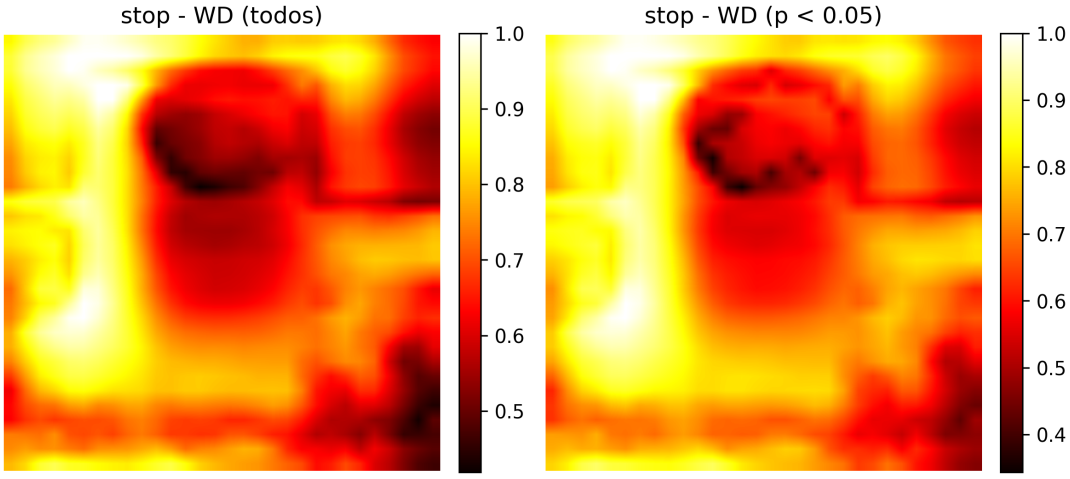

Figura 5.1: Heatmaps SafeML II para a classe *stop* no cenário diurno. À esquerda, mapa considerando todas as distâncias de Wasserstein; à direita, mapa filtrado por significância estatística ($p < 0{,}05$).

A Figura 5.2 apresenta os *heatmaps* correspondentes à classe *warning*. Nota-se uma concentração ainda mais intensa da divergência estatística na região central da imagem, associada à lâmpada amarela. Esse comportamento está alinhado com os resultados quantitativos apresentados anteriormente, nos quais a classe *warning* demonstrou maior

sensibilidade a variações de iluminação no cenário diurno. O filtro por p-valor elimina regiões periféricas irrelevantes, reforçando que a principal fonte de incerteza está associada à própria área luminosa do semáforo.

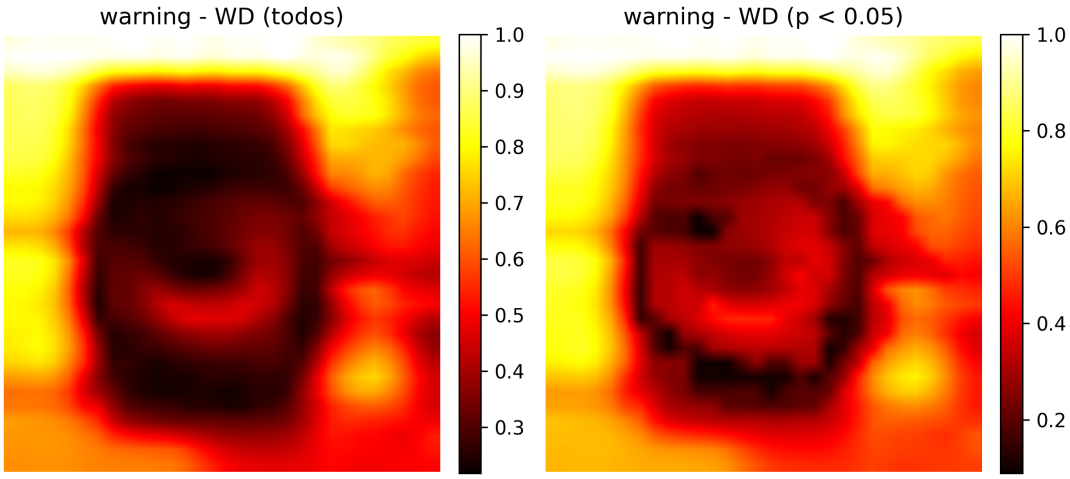

Figura 5.2: Heatmaps SafeML II para a classe *warning* no cenário diurno. À esquerda, WD considerando todos os pixels; à direita, apenas pixels estatisticamente significativos ($p < 0{,}05$).

Por fim, a Figura 5.3 apresenta os heatmaps da classe *go*. Diferentemente das classes anteriores, observa-se uma distribuição mais espalhada das divergências, envolvendo tanto a região da lâmpada verde quanto partes do fundo da imagem. Esse padrão indica que, embora o classificador apresente bom desempenho geral para essa classe, variações no contexto visual — como reflexos, céu ou estruturas metálicas — também influenciam a confiabilidade estatística das predições incorretas.

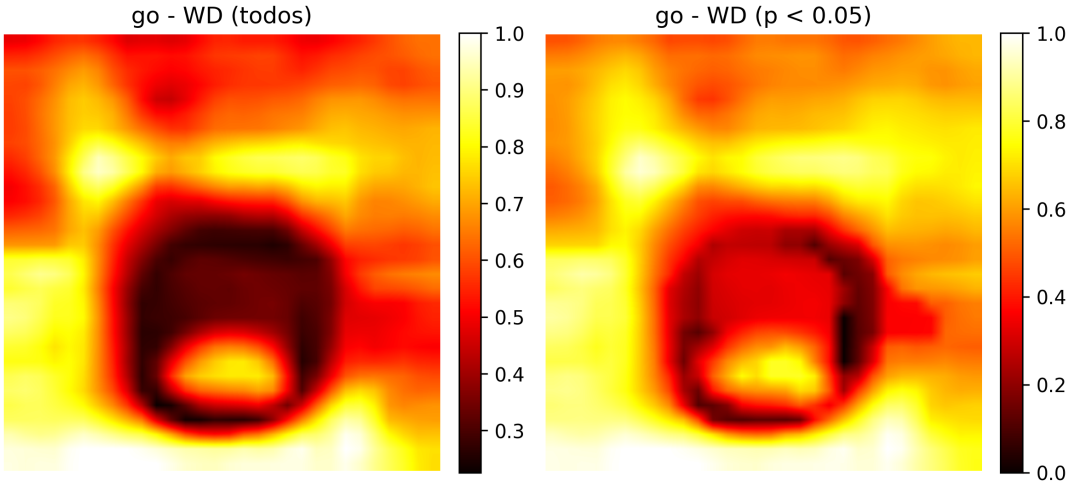

Figura 5.3: Heatmaps SafeML II para a classe *go* no cenário diurno. À esquerda, mapa completo de WD; à direita, mapa filtrado por significância estatística ($p < 0{,}05$).

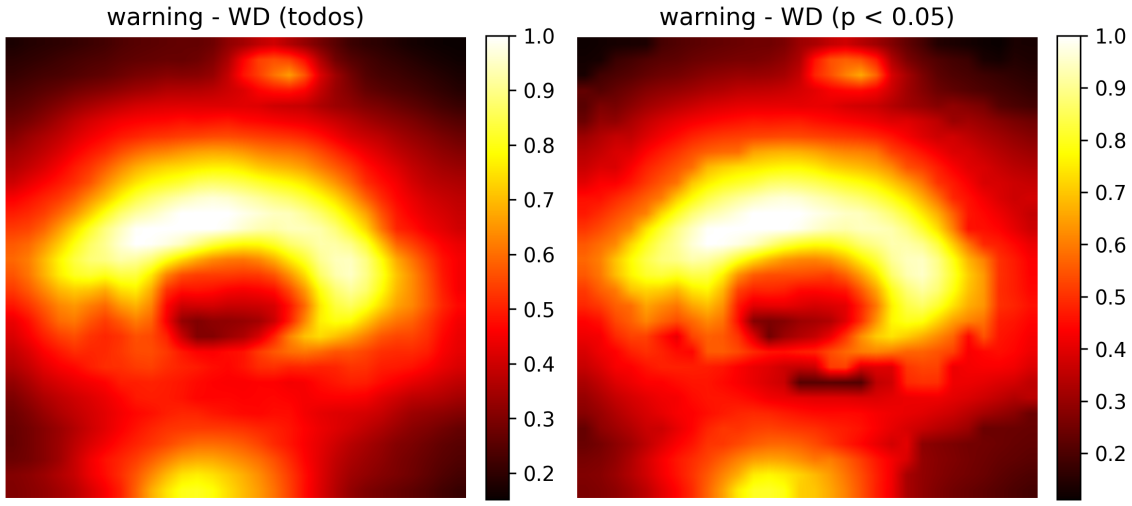

Figura 5.4: Heatmaps SafeML II para a classe *warning* no cenário noturno (SVM). À esquerda: WD normalizada considerando todos os pixels. À direita: WD considerando apenas pixels estatisticamente significativos ($p < 0{,}05$).

5.4.2 Heatmaps SafeML II — Classificador SVM (Cenário Noturno)

A Figura 5.4 apresenta os mapas de calor obtidos para a classe *warning* no cenário noturno. Observa-se que as regiões de maior intensidade de WD concentram-se principalmente na área correspondente à luz amarela do semáforo, bem como em partes do entorno imediato. No mapa filtrado por significância estatística ($p < 0{,}05$), essas regiões permanecem evidentes, indicando que a divergência observada está associada a alterações estruturais relevantes nos padrões visuais noturnos.

A Figura 5.5 apresenta os resultados para a classe *go*. Nota-se que as maiores intensidades de WD estão concentradas na região inferior do semáforo, associada à luz verde, além de áreas adjacentes. Em comparação com o cenário diurno, observa-se uma maior influência do fundo e de reflexos luminosos, característica comum em imagens noturnas, o que contribui para o aumento da divergência estatística observada.

Por fim, a Figura 5.6 apresenta os mapas correspondentes à classe *stop*. As regiões de maior intensidade de WD concentram-se predominantemente na área da luz vermelha, com destaque para o mapa filtrado por significância estatística, no qual essas regiões permanecem bem definidas. Esse comportamento indica que, no período noturno, variações na intensidade luminosa da luz vermelha exercem impacto significativo na divergência em

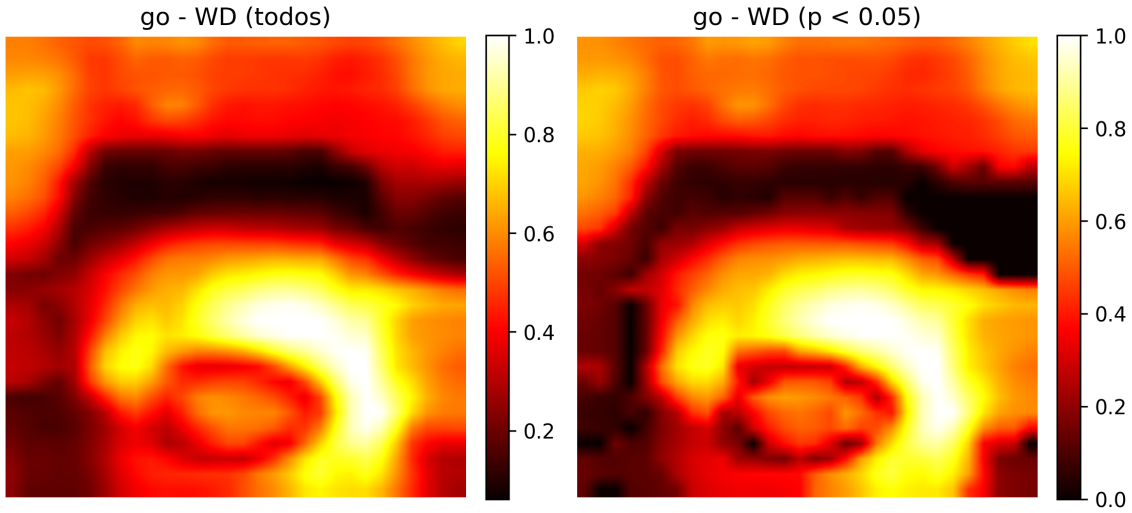

Figura 5.5: Heatmaps SafeML II para a classe *go* no cenário noturno (SVM). À esquerda: WD normalizada considerando todos os pixels. À direita: WD considerando apenas pixels estatisticamente significativos ($p < 0{,}05$).

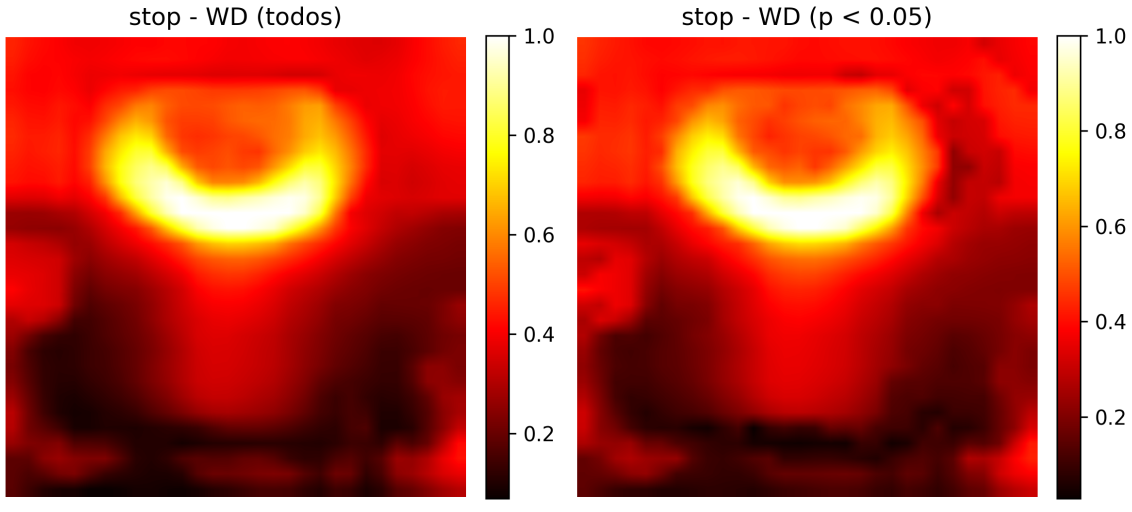

Figura 5.6: Heatmaps SafeML II para a classe *stop* no cenário noturno (SVM). À esquerda: WD normalizada considerando todos os pixels. À direita: WD considerando apenas pixels estatisticamente significativos ($p < 0{,}05$).

relação aos dados de treinamento.

5.4.3 Análise Visual dos Heatmaps – CNN (Cenário Diurno)

A Figura 5.7 e a Figura 5.8 apresentam os mapas de calor da Distância de Wasserstein (WD) obtidos para o classificador CNN no cenário diurno, considerando respectivamente as classes *go* e *stop*.

De forma geral, observa-se que as regiões de maior intensidade de WD concentram-se principalmente nas áreas centrais do recorte, correspondentes às regiões luminosas do semáforo. Para a classe *go*, essas regiões estão associadas à luz verde e ao seu entorno imediato, enquanto para a classe *stop* a maior divergência estatística aparece concentrada na área superior do semáforo, onde se localiza a luz vermelha.

Nos mapas filtrados por significância estatística, essas mesmas regiões permanecem em evidência, indicando que a divergência observada não é resultado de ruído aleatório, mas sim de diferenças estruturais relevantes entre as amostras de treino e os casos classificados incorretamente. Esse comportamento sugere que, mesmo em condições diurnas favoráveis, a CNN apresenta sensibilidade a variações de intensidade luminosa, contraste e saturação nas regiões críticas do semáforo.

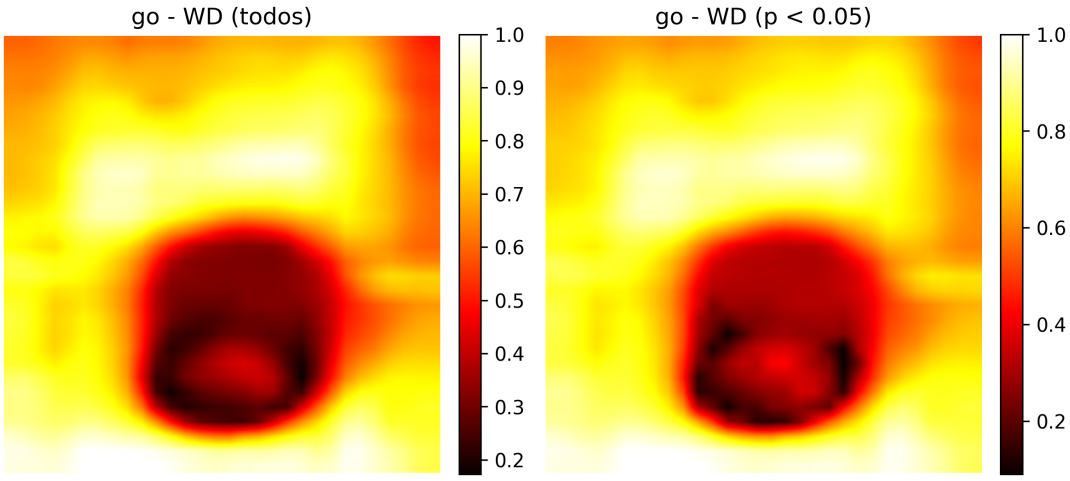

Figura 5.7: Heatmaps de WD para a classe *go* no cenário diurno (CNN): mapa completo (esquerda) e mapa filtrado por significância estatística $p < 0{,}05$ (direita).

Para a classe *warning*, o classificador CNN não apresentou erros de classificação no conjunto de teste diurno. Como consequência, não foi possível calcular os mapas de WD associados a amostras classificadas incorretamente para essa classe nesse cenário.

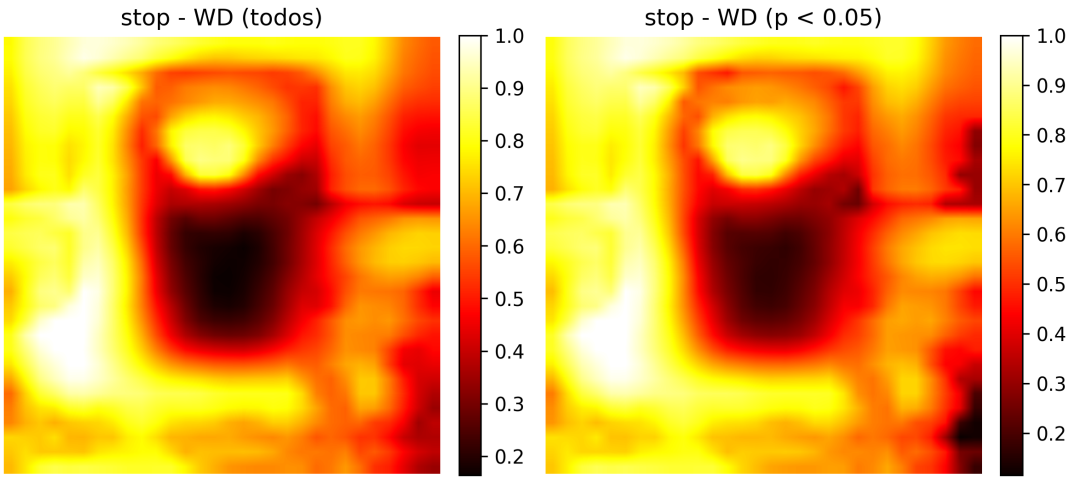

Figura 5.8: Heatmaps de WD para a classe *stop* no cenário diurno (CNN): mapa completo (esquerda) e mapa filtrado por significância estatística $p < 0{,}05$ (direita).

Dessa forma, a análise SafeML para a classe *warning* na CNN, bem como a definição de limiar de decisão, é realizada exclusivamente com base nos resultados obtidos no cenário noturno, apresentados na subseção correspondente.

5.4.4 Cenário Noturno

A análise visual dos *heatmaps* no cenário noturno evidencia diferenças importantes no comportamento do classificador CNN quando comparado ao cenário diurno. De modo geral, observa-se uma distribuição mais espalhada das regiões com alta intensidade de WD, refletindo a maior variabilidade visual causada pela iluminação artificial, reflexos e ruídos presentes nas imagens noturnas.

Para a classe *go*, ilustrada na Figura 5.9, as regiões de maior WD concentram-se predominantemente na área correspondente à lâmpada ativa do semáforo, mas também se estendem para regiões adjacentes. Esse comportamento indica que, à noite, pequenas variações de brilho e contraste no entorno do semáforo contribuem de forma mais significativa para a divergência estatística em relação ao conjunto de treinamento.

No caso da classe *stop*, apresentada na Figura 5.10, observa-se um padrão semelhante, porém com regiões de alta WD mais extensas e menos concentradas exclusivamente na lâmpada do semáforo. Isso sugere que, durante a noite, elementos do contexto visual — como iluminação pública e reflexos — passam a influenciar de forma mais acentuada a distribuição dos pixels, aumentando a divergência estatística mesmo em áreas fora da

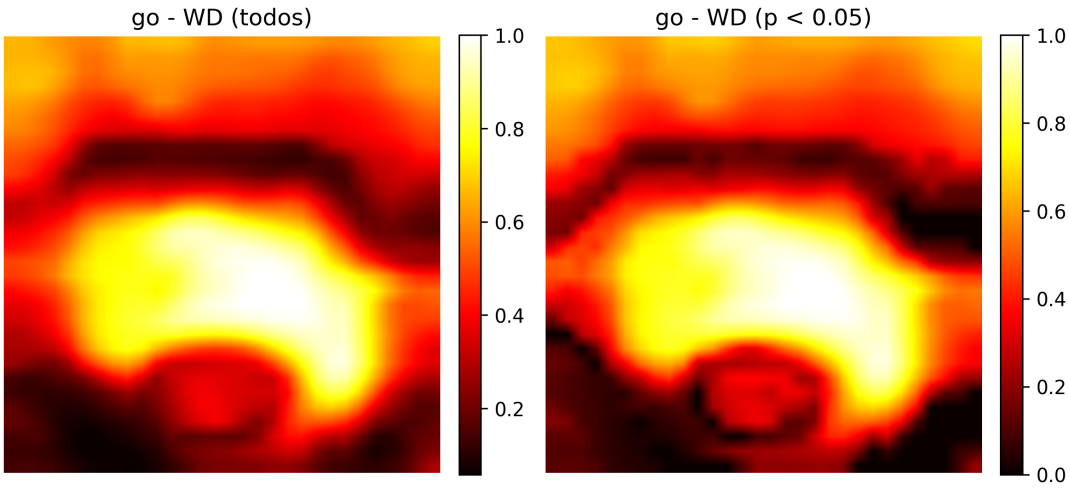

Figura 5.9: Heatmaps SafeML II para a classe *go* no cenário noturno (CNN). À esquerda, o mapa considerando todos os valores de WD; à direita, apenas os pixels estatisticamente significativos ($p < 0{,}05$).

região central do semáforo.

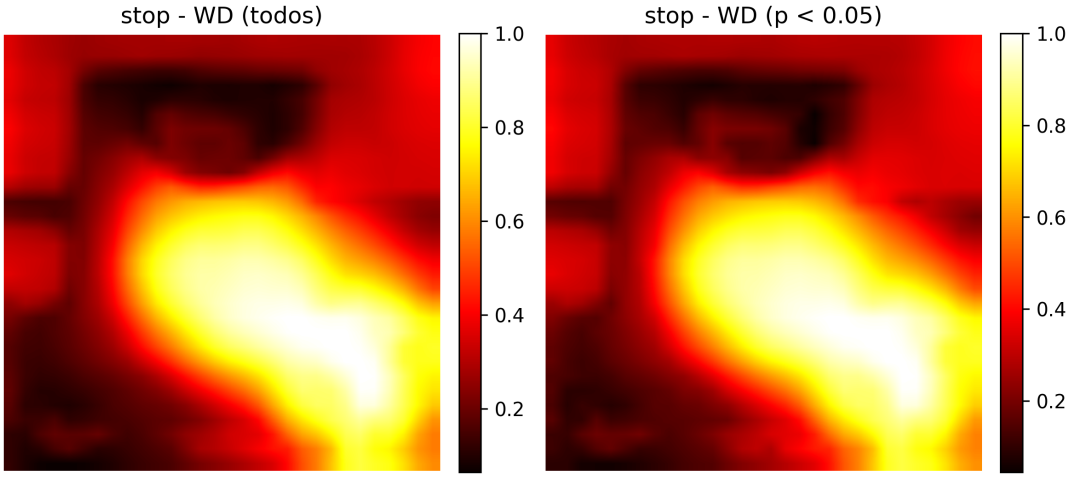

Figura 5.10: Heatmaps SafeML II para a classe *stop* no cenário noturno (CNN). À esquerda, o mapa considerando todos os valores de WD; à direita, apenas os pixels estatisticamente significativos ($p < 0{,}05$).

Por fim, a classe *warning*, ilustrada na Figura 5.11, apresenta as regiões de alta WD mais intensas e espacialmente mais amplas entre as três classes. Esse padrão visual é consistente com os resultados quantitativos obtidos anteriormente e indica que a identificação da luz amarela à noite é particularmente sensível a variações de iluminação e ruído visual. Tal comportamento justifica a adoção de um limiar específico para o cenário noturno nessa classe.

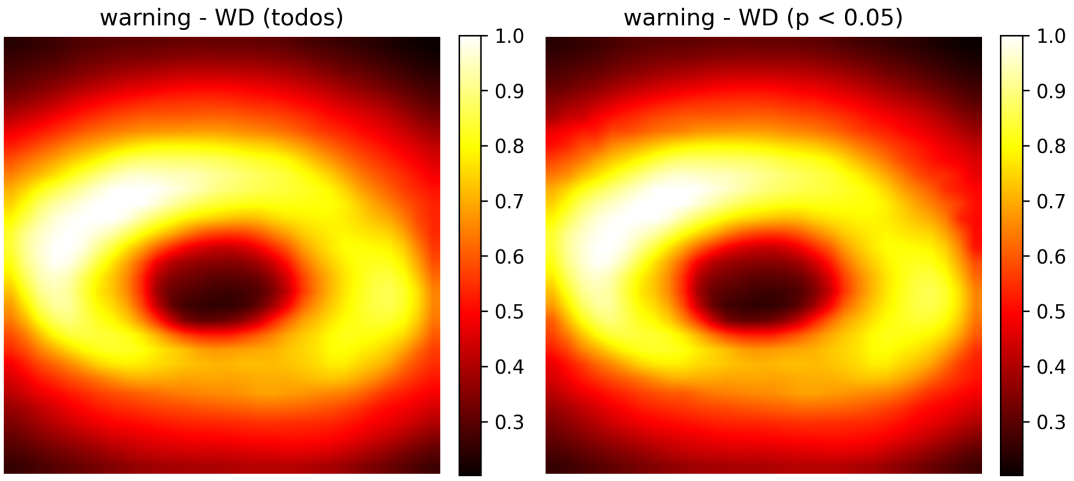

Figura 5.11: Heatmaps SafeML II para a classe *warning* no cenário noturno (CNN). À esquerda, o mapa considerando todos os valores de WD; à direita, apenas os pixels estatisticamente significativos ($p < 0{,}05$).

5.5 Avaliação do Limiar

Nesta seção é realizada a avaliação experimental dos limiares definidos a partir da coleta de dados do SafeML II. O objetivo é verificar se os valores de limiar estimados são capazes de distinguir, de forma consistente, amostras potencialmente confiáveis de amostras que apresentam maior divergência estatística em relação aos dados de treinamento.

A avaliação é conduzida a partir de um *buffer* composto por 30 imagens do conjunto de teste, sendo 10 imagens sequenciais para cada classe considerada (*go*, *stop* e *warning*). As imagens sequenciais representam o mesmo semáforo capturado em instantes consecutivos, permitindo analisar a estabilidade da medida de divergência ao longo de uma curta sequência temporal. Esse procedimento é aplicado de forma idêntica para os cenários diurno e noturno, bem como para ambos os classificadores avaliados neste trabalho.

5.5.1 Avaliação do Limiar para o SVM

Para cada classe, é analisado o valor médio da Distância de Wasserstein, calculada apenas sobre os pixels estatisticamente significativos, e sua comparação com o limiar previamente estabelecido.

O tempo médio necessário para a execução completa da avaliação em um cenário (diurno ou noturno) foi de aproximadamente 11 minutos e 30 segundos. Considerando que o buffer é composto por três classes, esse tempo corresponde a cerca de 3 minutos e

50 segundos por classe, ou aproximadamente 23 segundos por imagem dentro do *buffer* de 10 imagens sequenciais.

Cenário Diurno

A Tabela 5.15 apresenta os resultados da avaliação do limiar no cenário diurno. Observa-se que, para as três classes consideradas (*go*, *stop* e *warning*), os valores de WD média permaneceram abaixo dos respectivos limiares, resultando na aceitação de todas as amostras avaliadas.

Tabela 5.15: Avaliação do limiar SafeML II para o classificador SVM no cenário diurno.

Classe	**WD média**	**Limiar**	**Aceita**
go	0.2295	0.2510	Sim
stop	0.1763	0.2330	Sim
warning	0.2851	0.3620	Sim

Esses resultados indicam que, no cenário diurno, o limiar definido é suficientemente conservador para acomodar variações naturais das imagens sem comprometer a aceitação de amostras corretamente classificadas, preservando a confiabilidade do sistema.

Cenário Noturno

A Tabela 5.16 apresenta os resultados obtidos para o cenário noturno. Diferentemente do cenário diurno, observa-se que nem todas as amostras foram aceitas pelo critério de limiar. As classes *go* e *warning* apresentaram valores de WD média superiores aos respectivos limiares, resultando na rejeição das amostras, enquanto a classe *stop* permaneceu dentro do limite estabelecido.

Tabela 5.16: Avaliação do limiar SafeML II para o classificador SVM no cenário noturno.

Classe	**WD média**	**Limiar**	**Aceita**
go	0.2330	0.2080	Não
stop	0.1849	0.2270	Sim
warning	0.2543	0.2030	Não

Os resultados evidenciam que, no cenário noturno, as variações visuais associadas à iluminação artificial e ao aumento de ruído impactam diretamente a distribuição dos valores de WD. Nesse contexto, o mecanismo de limiar atua de forma mais restri-

tiva, identificando situações nas quais a divergência estatística em relação ao conjunto de treinamento ultrapassa o limite considerado confiável.

5.5.2 Avaliação do Limiar para o Classificador CNN

Para a CNN, o tempo médio necessário para a execução completa da avaliação em um cenário (diurno ou noturno) foi de aproximadamente 10 minutos e 18 segundos. Considerando que o *buffer* é composto por três classes, esse tempo corresponde a cerca de 3 minutos e 26 segundos por classe, ou aproximadamente 21 segundos por imagem dentro do buffer de 10 imagens sequenciais.

Cenário Diurno

A Tabela 5.17 apresenta os resultados da avaliação do limiar no cenário diurno para o classificador CNN. Observa-se que, assim como no caso do SVM, todas as amostras avaliadas apresentaram valores de WD média inferiores aos respectivos limiares, resultando na aceitação de todas as predições.

Tabela 5.17: Avaliação do limiar SafeML II para o classificador CNN no cenário diurno.

Classe	**WD média**	**Limiar**	**Aceita**
go	0.2327	0.3340	Sim
stop	0.1791	0.1940	Sim
warning	0.2839	0.2920	Sim

Os resultados indicam que, no cenário diurno, o classificador CNN apresenta uma separação clara entre as distribuições associadas aos acertos e os limiares definidos, com margens de segurança consistentes, especialmente para a classe *go*. Esse comportamento reflete a maior estabilidade visual das imagens diurnas, favorecendo a confiabilidade das decisões do modelo.

Cenário Noturno

A Tabela 5.18 apresenta os resultados obtidos para o cenário noturno. Diferentemente do observado para o SVM, todas as amostras avaliadas pela CNN foram aceitas pelo critério de limiar, mesmo em um contexto visual mais desafiador.

Tabela 5.18: Avaliação do limiar SafeML II para o classificador CNN no cenário noturno.

Classe	**WD média**	**Limiar**	**Aceita**
go	0.2319	0.2320	Sim
stop	0.1845	0.2810	Sim
warning	0.2556	0.2920	Sim

Nota-se que, para a classe *go*, o valor da WD média ficou muito próximo do limiar definido, indicando um caso limítrofe de aceitação. Ainda assim, o critério foi capaz de preservar a decisão do classificador, sugerindo que a CNN apresenta maior robustez às variações visuais noturnas quando comparada ao SVM, especialmente em termos da distribuição espacial das diferenças capturadas pela distância estatística.

5.5.3 Síntese e Análise dos Resultados Obtidos

Os resultados apresentados ao longo deste capítulo permitem uma análise integrada do desempenho e da confiabilidade dos classificadores avaliados, considerando tanto métricas tradicionais quanto a abordagem estatística proposta pelo SafeML II. De forma geral, os experimentos evidenciam que elevados índices de acurácia não são, por si só, suficientes para caracterizar o comportamento seguro de modelos de aprendizado de máquina em cenários críticos.

A avaliação do treinamento demonstrou que tanto o classificador SVM quanto a CNN alcançam desempenho global elevado no reconhecimento de semáforos, com taxas de acerto superiores a 98%. Entretanto, a análise por classe revelou diferenças relevantes, especialmente para a classe *warning*, que apresentou maior sensibilidade a variações no conjunto de dados. Esse comportamento reforça a limitação de métricas agregadas e justifica a necessidade de mecanismos adicionais de avaliação de confiabilidade.

A coleta de dados com o SafeML II mostrou de forma consistente que amostras incorretamente classificadas apresentam valores médios de Distância de Wasserstein significativamente superiores aos observados em amostras corretamente classificadas. Esse padrão foi identificado em ambos os classificadores e nos dois cenários avaliados, corroborando a hipótese de que a divergência estatística entre os dados de treino e de aplicação está diretamente associada à perda de confiabilidade das previsões.

A separação entre cenários diurnos e noturnos revelou-se fundamental para a cor-

reta interpretação dos resultados. Os experimentos indicaram que os valores de WD e os limiares de aceitação variam de maneira significativa conforme as condições de iluminação. No cenário noturno, observou-se maior variabilidade estatística, refletindo o impacto de fatores como ruído visual, reflexos e menor contraste, o que reforça a inadequação de um limiar único para todos os contextos de operação.

A análise visual por meio dos mapas de calor complementou os resultados quantitativos ao evidenciar regiões da imagem mais associadas à divergência estatística. Os heatmaps indicaram que, embora os maiores desvios estejam concentrados nas regiões das luzes do semáforo, áreas do entorno também contribuem para a perda de confiabilidade, demonstrando que o contexto visual exerce influência relevante sobre o comportamento dos classificadores.

Por fim, a avaliação dos limiares definidos a partir da coleta SafeML II mostrou que o mecanismo de aceitação e rejeição de previsões responde de forma coerente aos níveis de divergência observados. Nos testes realizados com buffers de imagens sequenciais, os limiares permitiram distinguir situações de operação dentro do domínio de treinamento daquelas associadas a maior incerteza estatística, tanto para o SVM quanto para a CNN. Esses resultados indicam que a abordagem adotada fornece subsídios relevantes para o monitoramento da confiabilidade de classificadores em tempo de execução.

6 Conclusão

O avanço do *Machine Learning* tem viabilizado a aplicação de sistemas inteligentes em domínios cada vez mais críticos, como veículos autônomos e sistemas avançados de assistência ao condutor. Entretanto, conforme discutido ao longo deste trabalho, altos valores de acurácia não são suficientes para garantir comportamento seguro em ambientes reais e dinâmicos. Nesse contexto, torna-se fundamental complementar os modelos de classificação com mecanismos capazes de estimar a confiabilidade de suas decisões em tempo de execução.

Este Trabalho de Conclusão de Curso teve como objetivo principal analisar a confiabilidade de classificadores aplicados ao reconhecimento automático de semáforos, avaliando a adequação do framework SafeML II como um mecanismo de monitoramento estatístico para esse domínio. Para isso, foram implementados e avaliados dois modelos distintos — um classificador SVM e uma CNN — utilizando o conjunto de dados LISA Traffic Light Dataset.

Os resultados obtidos demonstram que, embora ambos os modelos apresentem desempenho elevado segundo métricas tradicionais de classificação, seus comportamentos diferem significativamente quando analisados sob a ótica da confiabilidade. A aplicação do SafeML II evidenciou que amostras classificadas incorretamente tendem a apresentar maiores divergências estatísticas em relação ao conjunto de treino, mensuradas pela Distância de Wasserstein, quando comparadas às amostras corretamente classificadas. Esse comportamento foi consistente tanto para o SVM quanto para a CNN, validando a capacidade do SafeML II de atuar como um indicador de risco independente do modelo utilizado.

Outro achado importante foi a diferença no comportamento entre os classificadores na etapa de avaliação dos limiares. Enquanto a CNN apresentou um desempenho mais robusto, com todas as imagens erroneamente classificadas sendo corretamente sinalizadas como fora do domínio pelo SafeML II, o classificador SVM falhou em alguns casos, atribuindo baixa divergência estatística a amostras incorretas. Esse resultado sugere que, além

da CNN possuir maior capacidade discriminativa nos dados, suas falhas tendem a ocorrer em regiões do espaço de entrada mais afastadas do padrão de treinamento, o que facilita sua detecção por métodos de monitoramento como o SafeML. Assim, pode-se inferir que a CNN, neste contexto, não apenas obteve melhor desempenho tradicional, mas também maior previsibilidade do ponto de vista de segurança e confiabilidade.

Um resultado relevante deste trabalho foi a constatação de que a separação entre cenários diurnos e noturnos é essencial para a definição de limiares estatísticos mais adequados. As diferenças de iluminação impactam diretamente a distribuição dos pixels das imagens e, consequentemente, os valores da Distância de Wasserstein. A adoção de limiares específicos por classe e por cenário mostrou-se mais coerente do que a utilização de um único limiar global, contribuindo para decisões mais consistentes no monitoramento de confiabilidade.

A análise visual por meio de mapas de calor complementou os resultados quantitativos, permitindo identificar regiões da imagem que mais contribuem para a divergência estatística observada nos casos de erro. Esses mapas reforçam o papel explicativo do SafeML II, auxiliando na interpretação do comportamento dos classificadores e evidenciando padrões associados a variações de iluminação, reflexos e baixa definição visual, especialmente no cenário noturno.

Como limitações, destaca-se o custo computacional associado às etapas de coleta e avaliação dos limiares, que, na configuração atual, inviabiliza a aplicação direta do SafeML II em sistemas embarcados com restrições severas de tempo. Além disso, os experimentos foram conduzidos em um único conjunto de dados e com imagens previamente recortadas, não contemplando etapas como a detecção automática do semáforo em imagens de cena completa.

Como perspectivas de trabalhos futuros, além das estratégias de otimização para redução do tempo de execução do SafeML II e sua integração com pipelines de decisão em tempo real, destaca-se a possibilidade de investigar o uso das próprias medidas estatísticas empregadas pelo SafeML como mecanismo principal de discriminação entre classes. Os resultados obtidos indicam que tais medidas são capazes de capturar diferenças relevantes entre distribuições associadas às classes, sugerindo que uma abordagem baseada exclu-

sivamente em critérios estatísticos — inspirada ou derivada do SafeML — poderia ser explorada como um classificador em si, dispensando o uso de modelos tradicionais de *Machine Learning*, como SVMs ou redes neurais convolucionais. Essa linha de investigação pode ser promissora para o desenvolvimento de arquiteturas mais simples, interpretáveis e computacionalmente eficientes, com potencial para otimizar o desempenho e reduzir a latência em aplicações de tempo real.

Por fim, este trabalho contribui para a discussão sobre segurança e confiabilidade em sistemas inteligentes, demonstrando que a análise estatística de desvios de distribuição não apenas complementa métricas tradicionais de desempenho, mas também se apresenta como um caminho promissor para o desenvolvimento de soluções mais seguras, eficientes e responsáveis no contexto de veículos autônomos.

Bibliografia

ALVES, G. *Entendendo Redes Convolucionais (CNNs)*. 2018. Medium. Acesso em: 20 mai. 2024. Publicado em Neuronio BR. Disponível em: ⟨https://medium.com/neuronio-br/entendendo-redes-convolucionais-cnns-d10359f21184⟩.

ASLANSEFAT, K. et al. Toward improving confidence in autonomous vehicle software: A study on traffic sign recognition systems. *Computer*, IEEE, v. 54, n. 8, p. 66–76, 2021.

Aslansefat, K. et al. SafeML: Safety Monitoring of Machine Learning Classifiers through Statistical Difference Measure. *arXiv e-prints*, 2020. Disponível em: ⟨https://arxiv.org/abs/2005.13166⟩.

ASLANSEFAT, K. et al. Safeml: Safety monitoring of machine learning classifiers through statistical difference measures. In: SPRINGER. *International Symposium on Model-Based Safety and Assessment*. [S.l.], 2020. p. 197–211. Referência do SafeML 1 (Versão publicada).

BARBOSA, G. et al. Segurança em redes 5g: Oportunidades e desafios em detecção de anomalias e predição de tráfego baseadas em aprendizado de máquina. In: _____. *Minicursos do SBSeg*. Porto Alegre: SBC, 2021. p. 145–189. ISBN 9786587003658.

COUTINHO, B. *Modelos de Predição — SVM: Aprenda a criar seu primeiro algoritmo de classificação com SVM*. 2019. Medium. Acesso em: 20 mai. 2024. Publicado em Turing Talks. Disponível em: ⟨https://medium.com/turing-talks/turing-talks-12-classificacao-por-svm-f4598094a3f1⟩.

DUSTIN. *CNNs Explained Simply: A Friendly Guide to How Vision Models Really Work*. 2025. Medium. Acesso em: 20 mai. 2024. Publicado em Data And Beyond. Disponível em: ⟨https://medium.com/data-and-beyond/cnns-explained-simply-a-friendly-guide-to-how-vision-models-really-work-273eb61be005⟩.

HURWITZ, J.; KIRSCH, D. *Machine Learning For Dummies*. Hoboken, NJ: John Wiley & Sons, 2018. (IBM Limited Edition). IBM Limited Edition.

Instituto de Pesquisa Econômica Aplicada. *Taxa de mortes no trânsito está associada ao desenvolvimento econômico*. 2021. Acesso em: 2025. Disponível em: ⟨https://www.ipea.gov.br/portal/categorias/45-todas-as-noticias/noticias/15601-taxa-de-mortes-no-transito-esta-associada-ao-desenvolvimento-economico⟩.

JENSEN, M. B. et al. Vision for looking at traffic lights: Issues, survey, and perspectives. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 17, n. 7, p. 1800–1815, 2016.

JIA, Y. et al. The role of explainability in assuring safety of machine learning in healthcare. *IEEE Transactions on Artificial Intelligence (Early Access)*, IEEE, 2022.

JORGECARDETE. *Convolutional Neural Networks: A Comprehensive Guide*. 2024. Medium. Acesso em: 20 mai. 2024. Publicado em The Deep Hub. Disponível em: ⟨https://medium.com/thedeephub/convolutional-neural-networks-a-comprehensive-guide-5cc0b5eae175⟩.

KOOPMAN, P.; WAGNER, M. Challenges in autonomous vehicle testing and validation. In: SAE INTERNATIONAL. *SAE World Congress Experience*. [S.l.], 2016.

LIMA, C. V. A. *XAI LIME Tool: Uma ferramenta que avalia explicabilidade em Inteligência Artificial*. Monografia (Trabalho de Conclusão de Curso (Graduação em Ciência da Computação)) — Universidade Federal da Paraíba, 2024.

LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2017. p. 4765–4774.

NAZAT, S.; ABDALLAH, M. Xai-based feature ensemble for enhanced anomaly detection in autonomous driving systems. *arXiv preprint arXiv:2405.xxxxx*, 2024. Disponível em repositórios de pré-publicação.

Office of Educational Technology. *Artificial Intelligence and the Future of Teaching and Learning: Insights and Recommendations*. Washington, DC, 2023. Acesso em: 20 mai. 2024. Disponível em: ⟨https://www2.ed.gov/documents/ai-report/ai-report.pdf⟩.

PATERSON, C. et al. Safety assurance of machine learning for autonomous systems. *Reliability Engineering and System Safety*, Elsevier, v. 254, p. 111311, 2025.

PHILIPSEN, M. P. et al. Traffic light detection: A learning algorithm and evaluations on challenging dataset. In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems (ITSC)*. [S.l.]: IEEE, 2015. p. 2341–2345.

RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. "why should i trust you?": Explaining the predictions of any classifier. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. [S.l.: s.n.], 2016. p. 1135–1144.

SCULLEY, D. et al. Hidden technical debt in machine learning systems. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2015. p. 2503–2511.

Senna Martins Advogados. *Responsabilidade Civil em Acidentes com Veículos Autônomos: Desafios e Soluções*. 2025. Análise jurídica e dados de mercado brasileiro. Disponível em: ⟨https://sennamartins.com.br/responsabilidade-civil-veiculos-autonomos⟩.

SZELISKI, R. *Computer Vision: Algorithms and Applications*. 2. ed. Cham: Springer, 2021. Draft edition.

The Guardian. *Tesla Autopilot feature was involved in 13 fatal crashes, US regulator says*. 2024. Acessado em: Fev. 2025. Disponível em: ⟨https://www.theguardian.com/technology/2024/apr/26/tesla-autopilot-fatal-crash⟩.

Waymo LLC. *Waymo Safety Impact: Making roads safer with autonomous driving*. [S.l.], 2025. Dados acumulados até Setembro de 2025. Disponível em: ⟨https://waymo.com/safety/impact/⟩.

World Health Organization. *Road traffic injuries*. 2018. Acesso em: 2025. Disponível em: ⟨https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries⟩.