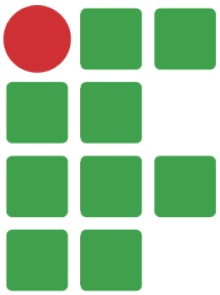


Instituto Federal de Educação, Ciência e Teconologia da Paraíba - IFPB
Unidade Acadêmica de Informática - UAI
Bacharelado em Engenharia da Computação

Liedson Augusto Maciel Costa

CONTROLE SEMAFÓRICO ADAPTATIVO BASEADO EM REDES NEURAIS CONVOLUCIONAIS PARA PREDIÇÃO DE TEMPO DE ESVAZIAMENTO DE FILAS VEICULARES EM AMBIENTE SIMULADO

Campina Grande
2026

Liedson Augusto Maciel Costa

CONTROLE SEMAFÓRICO ADAPTATIVO BASEADO EM REDES NEURAIS CONVOLUCIONAIS PARA PREDIÇÃO DE TEMPO DE ESVAZIAMENTO DE FILAS VEICULARES EM AMBIENTE SIMULADO

Monografia apresentada à Coordenação do curso de Bacharelado em Engenharia da Computação da Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, como parte dos requisitos necessários à obtenção do título de Bacharel em Engenharia da Computação.

Orientadores: Fagner de Araujo Pereira
Elmano Ramalho Cavalcanti

Campina Grande
Julho de 2026

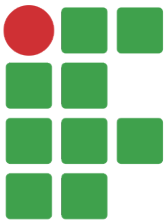
Catalogação na fonte:
Ficha catalográfica elaborada por Thiago Ferreira Cabral de Oliveira – CRB 15/628

M152c MACIEL COSTA, Liedson Augusto.
Controle Semafórico Adaptativo Baseado em Redes Neurais Convolucionais para Predição de Tempo de Esvaziamento de Filas Veiculares em Ambiente Simulado / Liedson Augusto Maciel Costa. - Campina Grande, 2026.
67 f.

Monografia (Graduação em Bacharelado em Engenharia de Computação) - Instituto Federal da Paraíba, 2026.
Orientadores: Prof. Fagner de Araujo Pereira,
Prof. Elmano Ramalho Cavalcanti.

1. 1. Semáforo Inteligente. 2. Rede Neural. 3. Otimização de Tráfego. I. Pereira, Fagner de Araujo. II. Cavalcanti, Elmano Ramalho. III. Instituto Federal de Educação, Ciência e Tecnologia da Paraíba. IV. Título.

CDU 004.8



CONTROLE SEMAFÓRICO ADAPTATIVO BASEADO EM REDES NEURAIS CONVOLUCIONAIS PARA PREDIÇÃO DE TEMPO DE ESVAZIAMENTO DE FILAS VEICULARES EM AMBIENTE SIMULADO

Liedson Augusto Maciel Costa

Orientadores: Fagner de Araujo Pereira
Elmano Ramalho Cavalcanti

MONOGRAFIA SUBMETIDA À COORDENAÇÃO DO CURSO DE ENGENHARIA DA COMPUTAÇÃO DO INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM ENGENHARIA DA COMPUTAÇÃO.

Aprovada por:

Prof. Fagner de Araujo Pereira, D.Sc.

Prof. Elmano Ramalho Cavalcanti, D.Sc.

Prof. Igor Barbosa da Costa, D.Sc.

Prof. Danyllo Wagner Albuquerque, D.Sc.

CAMPINA GRANDE, PB – BRASIL
JULHO DE 2026

Maciel Costa, Liedson Augusto

Controle Semafórico Adaptativo Baseado em Redes Neurais Convolucionais para Predição de Tempo de Esvaziamento de Filas Veiculares em Ambiente Simulado/Liedson Augusto Maciel Costa. – Campina Grande: IFPB/EngeComp, 2026.

XI, 55 p.: il.; 29, 7cm.

Orientadores: Fagner de Araujo Pereira
Elmano Ramalho Cavalcanti

Monografia (bacharelado) – IFPB/POLI/Programa de Engenharia da Computação, 2026.

Referências Bibliográficas: p. 53 – 55.

1. Semaforo Inteligente. 2. Rede Neural. 3. Otimização de Tráfego. 4. Simulação de Tráfego. I. de Araujo Pereira, Fagner *et al.* II. Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, POLI, Programa de Engenharia da Computação. III. Título.

Agradecimentos

Agradeço primeiramente a Deus, por me dar força e perseverança ao longo de toda a jornada acadêmica.

Aos meus orientadores, Prof. Fagner de Araujo Pereira e Prof. Elmano Ramalho Cavalcanti, pela dedicação, paciência e direcionamento ao longo do desenvolvimento deste trabalho.

À minha família, pelo apoio incondicional em todos os momentos desta caminhada, principalmente meus pais Mônica Rogéria de Souza Maciel e meu pai Gilmar Santos Costa. Também agradeço ao meu irmão Rielisson Joseff Sousa Costa, responsável por me ensinar diversos conceitos sobre Rede Neural que até então eu nao tinha conhecimento.

Ao meu amor Alice de Oliveira Barros, a mulher com que desejo viver ate os últimos dias da minha vida.

Aos amigos e colegas que, de alguma forma, contribuíram para a conclusão desta etapa.

À Superintendência de Trânsito e Transportes Públicos de Campina Grande (STTP), pelo fornecimento dos dados reais de temporização semafórica utilizados neste trabalho, em especial ao Sr. Helder de Barros Carlos e ao Sr. Victor Arruda Câmara Virgolino, que me acolheram e me ensinaram muito sobre vida profissional e pessoal.

Ao Instituto Federal de Educação, Ciência e Tecnologia da Paraíba (IFPB), pela estrutura e pelo ensino de qualidade que tornaram possível este trabalho.

Por fim, um agradecimento especial a minha querida Vó Maria Albino de Sousa, que nesse momento se encontra descansando nos braços de Deus, jamais esquecerei todo apoio e carinho que tive.

RESUMO

MACIEL COSTA, Liedson Augusto. **Controle Semafórico Adaptativo Baseado em Redes Neurais Convolucionais para Predição de Tempo de Esvaziamento de Filas Veiculares em Ambiente Simulado**. Campina Grande, 2026. Monografia (Bacharelado). Coordenação de Engenharia da Computação, Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, 2026.

O crescimento da frota veicular e a rigidez dos sistemas semafóricos de tempo fixo têm contribuído para o aumento de congestionamentos e a ineficiência do tráfego urbano. Semáforos convencionais são calibrados para condições médias, desperdiçando tempo de verde quando a fila é pequena e sendo insuficientes em horários de pico. Para mitigar esse problema, este trabalho propõe um sistema de controle semafórico adaptativo baseado em uma rede neural convolucional 2D (CNN), capaz de predizer o tempo necessário para o esvaziamento da fila de veículos em cada fase do semáforo, ajustando a duração do verde de forma dinâmica conforme a composição real da fila.

Todo o desenvolvimento e validação ocorrem em ambiente simulado no software SUMO (Simulation of Urban MObility), utilizando como cenário a reprodução da malha viária do cruzamento entre a Avenida Prefeito Severino Cabral e a Rua Deputado Ascendino Moura, em Campina Grande. Na simulação, detectores de área virtuais capturam, no momento da transição semafórica, os tipos de veículos presentes na fila (motocicleta, carro de passeio e ônibus) e suas posições relativas no detector. A partir dessa composição, a rede neural classifica o tempo de esvaziamento, utilizando uma função de perda com tolerância que aceita predições dentro de uma margem de erro de poucos segundos. A base de dados foi construída a partir de milhares de simulações com diferentes densidades de tráfego. A avaliação compara, no mesmo cenário simulado, o desempenho do sistema proposto com o do semáforo de tempo fixo. Os resultados indicam que a adaptação do tempo de verde à composição da fila melhora significativamente o aproveitamento do ciclo semafórico e reduz o desequilíbrio de atraso entre vias em cenários de alta demanda, embora apresente desempenho inferior ao controle de tempo fixo quanto ao atraso global em cenários de baixa e média demanda.

ABSTRACT

MACIEL COSTA, Liedson Augusto. **Adaptive Traffic Signal Control Based on Convolutional Neural Networks for Vehicle Queue Discharge Time Prediction in Simulated Environment**. Campina Grande, 2026. Undergraduate Thesis (B.Sc. in Computer Engineering). Computer Engineering Program, Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, 2026.

The growth of the vehicle fleet and the rigidity of fixed-time traffic light systems have contributed to increasing congestion and inefficiency in urban traffic. Conventional traffic lights are calibrated for average conditions, wasting green time when the queue is short and proving insufficient during peak hours. To mitigate this problem, this work proposes an adaptive traffic light control system based on a 2D Convolutional Neural Network (CNN), capable of predicting the time required to clear the vehicle queue in each signal phase, dynamically adjusting the green light duration according to the actual queue composition.

All development and validation take place in a simulated environment using the SUMO (Simulation of Urban MObility) software, reproducing the road network of the intersection between Avenida Prefeito Severino Cabral and Rua Deputado Ascendino Moura, in Campina Grande, Brazil. In the simulation, virtual area detectors capture, at the moment of signal transition, the types of vehicles in the queue (motorcycle, passenger car, and bus) and their relative positions within the detector. Based on this composition, the neural network classifies the queue clearance time, using a tolerance-aware loss function that accepts predictions within a margin of error of a few seconds. The dataset was built from thousands of simulations with varying traffic densities. The evaluation compares, within the same simulated scenario, the performance of the proposed system against a fixed-time traffic light. The results indicate that adapting the green light duration to the queue composition significantly improves the utilization of the traffic signal cycle and reduces the delay imbalance between approaches under high-demand scenarios, although it underperforms fixed-time control in terms of overall delay under low- and medium-demand scenarios.

Sumário

Lista de Siglas e Abreviaturas

API	Interface de Programação de Aplicações
CNN	Rede Neural Convolucional
DLR	Deutsches Zentrum für Luft- und Raumfahrt (Centro Aeroespacial Alemão)
IA	Inteligência Artificial
IDM	Intelligent Driver Model
RNA	Rede Neural Artificial
RNR	Rede Neural Recorrente
SCATS	Sydney Co-ordinated Adaptive Traffic System
SCOOT	Split Cycle Offset Optimisation Technique
STTP	Superintendência de Trânsito e Transportes Públicos
SUMO	Simulation of Urban MObility
TCP	Transmission Control Protocol
TraCI	Traffic Control Interface
GAP	Global Average Pooling

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Contextualização

O acelerado processo de urbanização e o contínuo crescimento da frota de veículos automotores impõem desafios complexos à mobilidade urbana, especialmente em cidades de médio e grande porte no Brasil. Problemas crônicos como congestionamentos frequentes, perdas significativas de tempo em deslocamentos, aumento da poluição atmosférica e maior incidência de acidentes de trânsito comprometem a qualidade de vida da população e acarretam elevados custos socioeconômicos e ambientais [1].

Os sistemas semafóricos desempenham papel central na gestão do tráfego urbano, organizando o fluxo de veículos e pedestres de forma a aumentar a segurança e reduzir congestionamentos. No entanto, a maioria das cidades brasileiras ainda opera com semáforos de ciclo fixo, cujos intervalos de verde são calibrados com base em médias históricas de fluxo. Essa abordagem é estruturalmente incapaz de responder às flutuações dinâmicas do tráfego: em períodos de baixa demanda, o tempo de verde é desperdiçado; em horários de pico, torna-se insuficiente para escoar a fila formada [2].

A cidade de Campina Grande (PB) ilustra bem esse cenário. De acordo com dados da STTP, em períodos sazonais como Natal e Réveillon o volume de veículos circulantes pode triplicar, com acréscimo de aproximadamente um milhão de deslocamentos [3]. Esse crescimento abrupto evidencia a rigidez dos semáforos convencionais diante de variações repentinas de demanda, gerando congestionamentos, aumento do tempo de espera e desperdício de combustível.

Campina Grande é também sede do Maior São João do Mundo, festa junina que atrai anualmente milhões de visitantes e representa um dos maiores eventos culturais do Brasil. O impacto no tráfego urbano é expressivo: somente nos primeiros 19 dias do São João de 2024, os equipamentos de monitoramento viário da STTP registraram

8.985.350 passagens de veículos, cerca de 1,6 milhão a mais do que no mesmo período do ano anterior [4]. Esses dados reforçam como a demanda por mobilidade na cidade está sujeita a variações abruptas e recorrentes, para as quais os semáforos de ciclo fixo não oferecem resposta adequada.

A aplicação de técnicas de IA ao controle semafórico tem demonstrado resultados expressivos em experiências nacionais e internacionais. Em Curitiba, a adoção de um sistema adaptativo resultou em redução significativa do tempo de deslocamento e em ganhos de eficiência para o transporte coletivo, com aumento substancial no número de ciclos diários completados pelos ônibus [5]. Essas experiências evidenciam o potencial das abordagens baseadas em aprendizado de máquina para superar as limitações dos sistemas estáticos.

A simulação microscópica de tráfego representa uma alternativa metodológica consolidada para o desenvolvimento e a avaliação de estratégias de controle semafórico, especialmente quando a implantação real não é viável ou segura. Nesse contexto, este trabalho utiliza o SUMO (*Simulation of Urban MObility*), simulador de código aberto desenvolvido pelo Centro Aeroespacial Alemão (DLR), em conjunto com sua interface de controle externo TraCI, que permite integrar o modelo de rede neural diretamente ao ciclo de simulação [6, 7].

Diante desse cenário, este trabalho propõe o desenvolvimento e a avaliação, em ambiente de simulação, de um controlador semafórico adaptativo baseado em redes neurais convolucionais. O sistema determina o tempo de verde de cada fase a partir da composição da fila de veículos detectada por sensores de faixa no momento da transição de fase, ou seja, quais tipos de veículos (motocicleta, carro de passeio e ônibus) estão parados e em quais posições dentro do detector. A hipótese que orienta este trabalho é apresentada e formalizada na Seção 1.2. A avaliação é conduzida integralmente em ambiente simulado, utilizando o SUMO, sobre um modelo do cruzamento entre a Av. Pref. Severino Cabral e a Rua Dep. Ascendino Moura, em Campina Grande. O desempenho do controlador neural é comparado ao do semáforo de tempo fixo em cenários de baixa, média e alta densidade de tráfego.

Esta pesquisa se justifica em quatro dimensões: pela relevância social, ao contribuir para a melhoria da mobilidade e da qualidade de vida urbana; pela relevância ambiental, ao reduzir o tempo de marcha lenta e, consequentemente, as emissões de poluentes; pela relevância tecnológica, ao integrar aprendizado de máquina e simulação de tráfego em uma solução validável e replicável; e pela relevância computacional, ao propor uma representação tensorial estruturada da composição de filas veiculares, adequada ao processamento por redes neurais convolucionais, e um pipeline completo e automatizado de geração de dados, treinamento e integração do modelo a um ambiente de simulação em tempo real via API TraCI, contribuições que podem ser reaproveitadas em outros problemas de predição sobre dados sequenciais

e posicionais.

1.2 Hipótese de Pesquisa

Este trabalho parte da seguinte questão de pesquisa: a adaptação dinâmica do tempo de verde à composição da fila de veículos, predita por uma rede neural convolucional, melhora a eficiência do controle semafórico em relação ao controle de tempo fixo?

A partir dessa questão, formula-se a hipótese central deste trabalho: o tempo de verde ajustado à composição real da fila, predito por uma CNN a partir do tipo e da posição dos veículos detectados, tem potencial para resultar em maior eficiência operacional do ciclo semafórico do que planos de tempo fixo, especialmente em cenários de alta densidade de tráfego.

Essa hipótese é avaliada por meio da comparação entre o controlador neural e seis planos de temporização fixa reais, sob três níveis de densidade de tráfego (baixo, médio e alto), utilizando como métricas o throughput médio, o tempo de verde desperdiçado, a taxa de esvaziamento e o atraso (*delay*) global e por via.

1.3 Objetivos

1.3.1 Objetivo Geral

Desenvolver e avaliar, em ambiente de simulação, um sistema de controle semafórico adaptativo baseado em rede neural convolucional, capaz de ajustar dinamicamente o tempo de verde de cada fase a partir da composição da fila de veículos detectada no início do ciclo, comparando seu desempenho ao de um semáforo de tempo fixo em um cruzamento urbano modelado no simulador SUMO.

1.3.2 Objetivos Específicos

- Avaliar se a predição do tempo de esvaziamento de filas veiculares, a partir da composição detectada por sensores de faixa, constitui uma estratégia viável de controle semafórico adaptativo, tomando como estudo de caso o cruzamento real entre a Av. Pref. Severino Cabral e a Rua Dep. Ascendino Moura, em Campina Grande.

- Investigar a capacidade de uma rede neural convolucional de generalizar a predição do tempo de esvaziamento para diferentes composições e densidades de fila, a partir de um conjunto de dados gerado por simulações automatizadas no SUMO.

- Comparar o desempenho operacional do controlador neural proposto com o de seis planos de temporização fixa reais, sob diferentes níveis de densidade de tráfego, utilizando como métricas o tempo médio de espera, o throughput, o desperdício de verde e a taxa de esvaziamento das filas.

- Analisar em que condições de demanda a estratégia adaptativa proposta apresenta vantagens ou desvantagens em relação ao controle de tempo fixo, identificando os fatores que explicam esse comportamento.

1.4 Organização do Trabalho

Os capítulos seguintes estão organizados da seguinte forma:

O **Capítulo 2 — Fundamentação Teórica** apresenta os conceitos que sustentam o trabalho: controle semafórico adaptativo, fundamentos de redes neurais convolucionais, e a ferramenta de simulação SUMO com sua interface de controle TraCI.

O **Capítulo 3 — Metodologia e Desenvolvimento** descreve as etapas de modelagem do cruzamento, geração e tratamento do conjunto de dados, projeto e treinamento da rede neural, e integração do modelo ao simulador como controlador adaptativo.

O **Capítulo 4 — Resultados e Discussão** apresenta a comparação de desempenho entre o controlador neural e o semáforo de tempo fixo nos diferentes cenários de densidade de tráfego, analisando o tempo médio de espera e o comportamento do sistema em cada condição.

O **Capítulo 5 — Considerações Finais** sintetiza as contribuições do trabalho, discute suas limitações e aponta direções para pesquisas futuras.

Capítulo 2

Fundamentação Teórica

A construção de um sistema de semáforo inteligente demanda sólida fundamentação teórica e o emprego de ferramentas tecnológicas adequadas. Este capítulo apresenta os conceitos que sustentam o desenvolvimento deste trabalho, organizados em três eixos: os fundamentos de engenharia de tráfego e controle semafórico, que contextualizam o problema abordado; os conceitos de aprendizado de máquina e redes neurais, que formam a base da solução proposta; e as ferramentas tecnológicas empregadas no desenvolvimento e na validação, incluindo o simulador de tráfego SUMO, a biblioteca PyTorch e a linguagem Python.

2.1 Engenharia de Tráfego e Controle Semafórico

O controle semafórico é um dos instrumentos centrais da engenharia de tráfego para a gestão de interseções urbanas. Seu objetivo fundamental é organizar a passagem alternada de fluxos conflitantes, reduzindo o risco de colisões e distribuindo o tempo disponível entre as vias que convergem em um cruzamento. Para compreender o funcionamento e as limitações dos sistemas semafóricos, é necessário apresentar alguns conceitos fundamentais da área.

2.1.1 Conceitos Fundamentais

O funcionamento de um semáforo é descrito por um conjunto de termos técnicos padronizados. O **ciclo semafórico** corresponde à sequência completa de todas as indicações luminosas (verde, amarelo e vermelho) oferecidas a todos os movimentos da interseção, retornando ao estado inicial. A duração total dessa sequência é denominada **tempo de ciclo**. Cada intervalo em que um determinado grupo de movimentos compatíveis recebe o direito de passagem é chamado de **fase**. O período de transição entre fases, que inclui o amarelo e, quando aplicável, o vermelho geral, é denominado **tempo de entreverdes** e tem a função de garantir a segurança na

troca de direitos de passagem [8].

A **fila de veículos** que se forma durante o vermelho é um elemento central para a análise de desempenho de uma interseção semaforizada. Quando o verde é ativado, essa fila inicia seu processo de **esvaziamento**, e o tempo necessário para que todos os veículos parados cruzem a linha de retenção depende de fatores como a quantidade de veículos, seus tipos (veículos leves, motocicletas, ônibus) e a capacidade da via. A **taxa de fluxo de saturação** representa a taxa máxima de veículos que podem atravessar a interseção por unidade de tempo durante o verde efetivo, sendo uma grandeza fundamental para o dimensionamento semafórico [8].

2.1.2 Tipos de Controle Semafórico

Os sistemas de controle semafórico podem ser classificados em três categorias principais, conforme o grau de resposta à demanda real de tráfego:

Controle de tempo fixo. Nessa modalidade, os tempos de verde, amarelo e vermelho de cada fase são previamente calculados e permanecem constantes ao longo do dia, independentemente das condições reais de tráfego. A calibração é tipicamente realizada com base em contagens volumétricas históricas, buscando atender à demanda média da interseção. Embora seja a estratégia de controle mais simples e de menor custo de implantação, sua rigidez implica desperdício de tempo de verde nos períodos de baixa demanda e insuficiência nos horários de pico. A grande maioria das interseções semaforizadas em cidades brasileiras de médio porte, incluindo Campina Grande, opera sob essa modalidade [2, 8].

Controle atuado. Os semáforos atuados utilizam detectores de presença veicular (laços indutivos, sensores infravermelhos, câmeras ou radares) para ajustar a duração do verde em função da demanda detectada em tempo real. Em sua forma mais comum, a atuação permite a extensão do verde enquanto houver veículos sendo detectados, até um limite máximo pré-configurado. Essa abordagem responde melhor às flutuações do tráfego, porém ainda opera com regras fixas e limiares determinísticos, sem capacidade de predição ou aprendizado [8].

Controle adaptativo. Os sistemas adaptativos representam o nível mais sofisticado de controle semafórico. Diferentemente dos atuados, que apenas reagem à presença de veículos, os sistemas adaptativos empregam algoritmos capazes de analisar padrões de tráfego e ajustar dinamicamente os parâmetros do ciclo semafórico. Exemplos clássicos incluem os sistemas SCOOT e SCATS, que coordenam múltiplas interseções em rede. Mais recentemente, abordagens baseadas em inteligência artificial, como aprendizado por reforço e aprendizado supervisionado, têm sido exploradas para o controle adaptativo de interseções individuais [2, 9].

O sistema proposto neste trabalho situa-se na categoria de controle adap-

tativo, porém com uma abordagem específica: em vez de otimizar globalmente o cruzamento, utiliza uma rede neural para predizer o tempo necessário para o esvaziamento da fila presente no início de cada fase, empregando essa predição diretamente como duração do verde.

2.2 Inteligência Artificial em Sistemas Inteligentes de Transporte

Sistemas Inteligentes de Transporte (*Intelligent Transportation Systems* – ITS) englobam o uso de tecnologias de sensoriamento, comunicação e processamento de dados para melhorar a eficiência, a segurança e a sustentabilidade da mobilidade urbana. Dentro desse campo, o controle semafórico é uma das aplicações mais estudadas para a incorporação de técnicas de inteligência artificial, dada a disponibilidade crescente de sensores de tráfego e o custo computacional relativamente baixo de modelos aplicados a interseções individuais [9].

Conforme discutido por Wei et al. (2019), as abordagens de IA aplicadas ao controle semafórico podem ser agrupadas em três grandes linhas: o aprendizado por reforço, no qual um agente aprende uma política de controle por meio de interação direta com o ambiente de tráfego; a lógica *fuzzy*, que traduz regras de controle especificadas por especialistas em ações contínuas; e o aprendizado supervisionado, no qual um modelo é treinado a partir de dados rotulados para predizer grandezas relevantes ao controle, como tempos de ciclo ou de esvaziamento de fila [9].

O aprendizado por reforço é a linha mais explorada na literatura recente, por permitir a otimização direta de uma política de controle sem a necessidade de rótulos pré-definidos. Entretanto, essa abordagem apresenta desafios relacionados à estabilidade do treinamento, ao tempo de convergência e à interpretabilidade das decisões tomadas pelo agente. Em contrapartida, o aprendizado supervisionado, embora dependente da qualidade e representatividade do conjunto de dados rotulado, oferece maior transparência sobre o que o modelo aprendeu e facilita a validação de suas predições de forma isolada, antes mesmo de sua integração ao ciclo de controle [9]. É nessa linha que o presente trabalho se insere: a rede neural convolucional não decide diretamente a política de controle do cruzamento, mas resolve um subproblema bem definido – a predição do tempo de esvaziamento da fila – cuja saída é então utilizada de forma direta como parâmetro de controle.

2.3 Simulação de Tráfego: SUMO e TraCI

A simulação de tráfego desempenha papel fundamental no estudo e no desenvolvimento de soluções voltadas para a mobilidade urbana. Ela possibilita a modelagem e a análise de cenários complexos sem a necessidade de realizar experimentos diretamente em vias reais, o que reduz custos e riscos operacionais. Além disso, permite a geração sistemática de dados em condições controladas, sendo especialmente útil para o treinamento e a validação de modelos de aprendizado de máquina.

O software Simulation of Urban MObility (SUMO) é um simulador microscópico e multimodal de código aberto, desenvolvido pelo Deutsches Zentrum für Luft- und Raumfahrt (DLR), o Centro Aeroespacial Alemão. Conforme sua documentação oficial, o SUMO permite representar em detalhe o comportamento individual de veículos, pedestres e transportes coletivos em uma malha viária digitalizada. Sua arquitetura flexível possibilita a definição de rotas, sinais de tráfego, fluxos veiculares e diferentes tipos de veículos com características distintas de aceleração, comprimento e velocidade máxima, tornando-o adequado tanto para pesquisas acadêmicas quanto para aplicações práticas em planejamento urbano e sistemas inteligentes de transporte [10].

O SUMO se torna particularmente poderoso quando combinado com a Traffic Control Interface (TraCI), uma API que permite o controle em tempo real das simulações por meio de conexões externas. Conforme a documentação do DLR, o TraCI opera sobre o protocolo TCP, possibilitando que aplicações escritas em linguagens como Python interajam dinamicamente com a execução do simulador [11]. Essa interface oferece comandos para manipular elementos da simulação enquanto ela ocorre, incluindo o controle de estados semafóricos, a inserção ou remoção dinâmica de veículos, a alteração de rotas em tempo real e a coleta de informações detalhadas sobre filas, tempos de espera e posições de veículos.

2.4 Aprendizado de Máquina Supervisionado

O aprendizado de máquina (machine learning) é uma subárea da inteligência artificial que se dedica ao desenvolvimento de algoritmos capazes de aprender padrões a partir de dados, sem que esses padrões sejam explicitamente programados. Conforme descrito por Bishop (2006), os métodos de aprendizado de máquina podem ser agrupados em três paradigmas principais: supervisionado, não supervisionado e por reforço [12].

No **aprendizado supervisionado**, que é o paradigma empregado neste trabalho, o modelo é treinado a partir de um conjunto de exemplos rotulados, ou

seja, pares de entrada e saída desejada. O objetivo é que o modelo aprenda uma função que mapeie as entradas às saídas correspondentes, de modo a generalizar para dados não vistos durante o treinamento. As tarefas de aprendizado supervisionado dividem-se em dois tipos: **regressão**, quando a saída é um valor contínuo (por exemplo, prever a temperatura), e **classificação**, quando a saída corresponde a uma categoria discreta (por exemplo, classificar uma imagem como "gato" ou "cachorro") [13].

2.4.1 Função de Perda e Treinamento

O treinamento de um modelo supervisionado consiste em ajustar seus parâmetros internos (pesos) de modo a minimizar uma **função de perda** (loss function), que quantifica a discrepância entre as predições do modelo e os rótulos reais do dataset. Para tarefas de classificação, a função de perda mais comum é a **entropia cruzada categórica** (cross-entropy loss), que penaliza predições que atribuem baixa probabilidade à classe correta [13].

A minimização da função de perda é realizada por meio de algoritmos de otimização baseados em **gradiente descendente**, nos quais os pesos do modelo são atualizados iterativamente na direção que reduz o erro. Variantes como o Adam (Adaptive Moment Estimation), proposto por Kingma e Ba (2015), combinam estimativas adaptativas do primeiro e segundo momentos do gradiente, proporcionando convergência mais estável e rápida em problemas com grande volume de dados [14].

2.4.2 Divisão de Dados e Generalização

Para avaliar a capacidade de generalização de um modelo, o dataset é tipicamente dividido em três conjuntos: **treinamento**, utilizado para o ajuste dos pesos; **validação**, utilizado para monitorar o desempenho durante o treinamento e ajustar hiperparâmetros; e **teste**, utilizado para a avaliação final do modelo em dados não vistos. Essa divisão é essencial para detectar problemas como o **sobreajuste** (overfitting), situação em que o modelo memoriza os dados de treinamento em vez de aprender padrões generalizáveis [13].

2.5 Redes Neurais Artificiais

As redes neurais artificiais (RNAs) são modelos computacionais inspirados na arquitetura do cérebro humano, projetados para identificar padrões complexos e extrair informações significativas de grandes volumes de dados. A estrutura básica de uma RNA é organizada em camadas de neurônios interconectados: a camada de entrada recebe os dados brutos, que são processados por uma ou mais camadas

ocultas, onde ocorrem as operações de maior complexidade, e a camada de saída fornece a resposta do modelo [15].

Cada conexão entre neurônios possui um **peso** associado e um **viés**, que são ajustados durante o treinamento. O processo de aprendizado é, essencialmente, a otimização desses parâmetros para que a saída do modelo se aproxime do resultado desejado. Esse ajuste ocorre em duas fases: a **propagação direta** (forward propagation), na qual os dados percorrem a rede da entrada à saída, e a **retropropagação** (backpropagation), na qual o gradiente do erro é calculado e propagado de volta pela rede para atualizar os pesos de cada conexão [15].

2.5.1 Redes Neurais Convolucionais (CNNs)

As Redes Neurais Convolucionais (CNNs) são um tipo especializado de rede neural que utiliza camadas convolucionais para detectar automaticamente padrões locais nos dados de entrada. Embora tenham sido originalmente desenvolvidas para processamento de imagens, onde os filtros (kernels) capturam padrões espaciais como bordas e texturas, as CNNs são aplicáveis a qualquer dado que possua estrutura posicional, ou seja, em que a posição relativa dos elementos carrega informação relevante [16].

A arquitetura de uma CNN é composta, em geral, por três tipos de camadas. As **camadas convolucionais** aplicam filtros aos dados de entrada, produzindo mapas de características que destacam padrões relevantes. As **camadas de pooling** realizam a redução da dimensionalidade dos mapas de características, preservando informações essenciais e diminuindo o custo computacional. E as **camadas totalmente conectadas** (densas) consolidam as características extraídas e geram a saída final do modelo [16].

2.5.2 CNNs em Comparação com Arquiteturas Alternativas

A escolha de uma arquitetura de rede neural para um problema de predição depende fundamentalmente da estrutura dos dados de entrada. Um Perceptron Multicamadas (MLP), por exemplo, trata a entrada como um vetor não estruturado, sem qualquer noção de proximidade ou ordem entre seus elementos: cada neurônio da primeira camada se conecta a todas as entradas com pesos independentes, de modo que a rede precisa aprender do zero, para cada posição, os padrões que uma CNN já assume por construção. Para o problema de predição do tempo de esvaziamento, em que a posição relativa de cada veículo na fila carrega informação relevante – um ônibus na cabeça da fila tem impacto diferente de um ônibus no fundo –, essa perda de estrutura espacial representa uma desvantagem em relação a arquiteturas que preservam explicitamente a ordem dos dados.

Redes Neurais Recorrentes (RNRs) e suas variantes, como LSTM (*Long Short-Term Memory*), são adequadas para dados sequenciais em que a ordem representa uma dependência temporal, como séries temporais ou texto. Embora a sequência de veículos na fila também possua uma ordem relevante, essa ordem é espacial e estática no momento da predição, e não evolui ao longo do tempo dentro de uma mesma amostra. Além disso, RNRs processam a sequência de forma estritamente sequencial, o que aumenta o custo computacional de treinamento e dificulta a captura de padrões locais de curto alcance, como a presença de um ônibus entre duas motocicletas, sem a necessidade de propagar informação por toda a sequência.

Arquiteturas baseadas em *Transformers*, por sua vez, utilizam mecanismos de atenção para capturar dependências entre elementos de uma sequência, independentemente da distância entre eles, e têm apresentado resultados expressivos em tarefas de processamento de linguagem natural e, mais recentemente, em visão computacional. Entretanto, essas arquiteturas costumam demandar volumes de dados de treinamento significativamente maiores para generalizar adequadamente, além de maior custo computacional, o que as torna menos adequadas a um problema com sequências curtas (tipicamente entre 17 e 23 veículos por faixa) e um conjunto de dados de porte moderado, como o utilizado neste trabalho.

Modelos de regressão tradicionais, como regressão linear ou métodos baseados em árvores (por exemplo, *Random Forest* e *Gradient Boosting*), exigem tipicamente a extração manual de *features* agregadas – como o número total de veículos por tipo ou a posição média da fila – perdendo a granularidade da posição individual de cada veículo. Embora sejam computacionalmente mais simples e frequentemente competitivos em problemas tabulares, esses modelos não incorporam nativamente a noção de vizinhança espacial entre elementos de entrada, que é justamente a propriedade que motivou a escolha de uma CNN neste trabalho.

Diante desse panorama, a CNN foi escolhida por representar o melhor compromisso entre a estrutura dos dados de entrada – uma sequência curta e espacialmente ordenada de veículos – e o volume de dados disponível para treinamento, permitindo capturar padrões locais de composição da fila sem exigir a extração manual de *features* nem demandar os grandes volumes de dados tipicamente necessários para o treinamento de arquiteturas mais complexas, como *Transformers*.

2.6 A Biblioteca PyTorch

PyTorch é uma biblioteca de código aberto para aprendizado de máquina e redes neurais artificiais, amplamente utilizada tanto em pesquisa quanto em aplicações práticas de inteligência artificial [17, 18]. Sua característica mais relevante para este trabalho é o modelo de *eager execution*, que executa as operações de forma dinâmica,

em contraste com arquiteturas de grafos estáticos adotadas por outros frameworks [13, 18]. Esse recurso facilita a depuração e o desenvolvimento iterativo, características especialmente úteis na etapa de prototipagem de arquiteturas descrita no Capítulo 3.

2.7 Trabalhos relacionados

A aplicação de inteligência artificial ao controle semafórico tem sido objeto de intensa pesquisa nas últimas décadas, podendo-se identificar três grandes linhas de abordagem: aprendizado por reforço, lógica *fuzzy* e aprendizado supervisionado, conforme discutido na Seção 2.2. Esta seção revisa criticamente quatro trabalhos representativos dessas linhas, identificando suas limitações metodológicas e as lacunas que motivam as escolhas de projeto adotadas neste trabalho.

No contexto brasileiro, Campos (2006) propôs um sistema de regulagem em tempo real para duas interseções integradas utilizando Redes Neurais Recorrentes em conjunto com controle adaptativo [19]. O objetivo era determinar automaticamente o tempo de ciclo a partir do fluxo e da velocidade dos veículos detectados nas vias. A principal limitação desse trabalho está na avaliação: a validação foi conduzida em simulador próprio, sem comparação com nenhum *baseline* de controle fixo, e de forma exclusivamente qualitativa, verificando apenas se a rede neural determinava corretamente o estado de bloqueio ou liberação de cada fase, sem métricas quantitativas de desempenho como tempo de espera ou comprimento de fila. Essa ausência de métricas operacionais e de comparação sistemática com o controle convencional é uma lacuna que o presente trabalho busca preencher.

No cenário internacional, Genders e Razavi (2016) desenvolveram um agente de controle semafórico baseado em Aprendizado por Reforço Profundo, utilizando o SUMO como ambiente de simulação [20]. Os autores propuseram uma codificação discreta do estado do tráfego como entrada de uma CNN, representando a presença e velocidade dos veículos em células ao longo das faixas. O agente foi comparado a um controlador neural de camada única, obtendo redução de 82% no atraso médio acumulado, 66% no comprimento médio de fila e 20% no tempo médio de viagem. Apesar de também empregar CNN e SUMO, a abordagem trata o problema como otimização sequencial via aprendizado por reforço, o que exige um processo de treinamento mais custoso computacionalmente e menos interpretável do que a classificação supervisionada adotada neste trabalho, na qual a contribuição de cada padrão de entrada para a predição pode ser analisada de forma mais direta.

Wei et al. (2019) apresentaram uma revisão abrangente dos métodos de controle semafórico baseados em IA, consolidando as três linhas de abordagem mencionadas e apontando que o aprendizado supervisionado, embora menos explorado,

oferece maior interpretabilidade e facilidade de validação em relação ao aprendizado por reforço [9]. Os próprios autores destacam, contudo, que essa linha carece de estudos que explorem representações de entrada mais ricas do que variáveis agregadas de fluxo, uma lacuna que este trabalho aborda por meio da representação detalhada da composição da fila.

Mais recentemente, Siqueira Filho (2024) aplicou redes neurais densas para o controle adaptativo de semáforos em simulador baseado no Intelligent Driver Model (IDM), comparando o sistema adaptativo ao controle de ciclo fixo [21]. A avaliação utilizou três métricas: média de veículos esperando, veículos passados por unidade de tempo e velocidade média dos veículos. Os resultados mostraram melhora moderada na fluidez do tráfego em cenários de alta demanda. A principal limitação identificada nesse trabalho é que a entrada do modelo era composta por variáveis agregadas do ciclo, sem distinção do tipo ou posição dos veículos na fila, o que impede o modelo de capturar assimetrias de composição relevantes – por exemplo, a diferença de impacto entre um ônibus e uma motocicleta na cabeça da fila. Além disso, a saída era um valor contínuo de duração do ciclo obtido por regressão, sem avaliação de sua aderência a planos reais de temporização de um cruzamento existente.

A síntese a seguir compara os quatro trabalhos revisados e a proposta deste trabalho, quanto à abordagem, entrada, saída, ambiente de validação e métrica principal empregada:

Campos (2006) Abordagem: RNR + controle adaptativo. Entrada: fluxo e velocidade agregados. Saída: estado de bloqueio/liberação da fase. Ambiente: simulador próprio, em linguagem C. Métrica principal: avaliação qualitativa, sem métricas quantitativas de desempenho.

Genders & Razavi (2016) Abordagem: aprendizado por reforço profundo (CNN). Entrada: grade discreta de presença e velocidade dos veículos. Saída: ação de controle semafórico. Ambiente: SUMO. Métrica principal: atraso médio, comprimento de fila e tempo de viagem.

Siqueira Filho (2024) Abordagem: rede neural densa (regressão). Entrada: variáveis agregadas do ciclo semafórico. Saída: duração do ciclo (valor contínuo). Ambiente: simulador próprio baseado no *Intelligent Driver Model* (IDM). Métrica principal: veículos esperando e throughput.

Este trabalho (2026) Abordagem: aprendizado supervisionado com CNN 2D. Entrada: tipo e posição individual de cada veículo na fila. Saída: classe discreta correspondente ao tempo de esvaziamento. Ambiente: SUMO, sobre um cruzamento real de Campina Grande. Métrica principal: throughput, delay global e por via, desperdício de verde e taxa de esvaziamento.

Essa comparação evidencia as lacunas discutidas ao longo desta seção: ausência de métricas quantitativas em Campos (2006), maior custo computacional e menor interpretabilidade do aprendizado por reforço em Genders e Razavi (2016), e agregação excessiva da entrada e da saída do modelo em Siqueira Filho (2024) – limitações que a representação detalhada da fila e a formulação como classificação discreta, adotadas neste trabalho, buscam superar.

O presente trabalho diferencia-se dos estudos anteriores ao adotar aprendizado supervisionado com uma CNN 2D cuja entrada é a composição detalhada da fila – tipos de veículos e suas posições nas faixas – e cuja saída é a classificação discreta do tempo de esvaziamento dessa fila. A definição das métricas de avaliação foi parcialmente inspirada nos trabalhos revisados: o throughput médio e o delay global seguem a linha adotada por Siqueira Filho (2024) e Genders e Razavi (2016), respectivamente. No entanto, duas métricas adicionais foram propostas – o tempo de verde desperdiçado após o esvaziamento da fila e o delay discriminado por via – com o objetivo de capturar aspectos da ineficiência do ciclo fixo que os trabalhos anteriores não contemplam. A validação é conduzida sobre um modelo real do cruzamento entre a Av. Pref. Severino Cabral e a Rua Dep. Ascendino Moura, em Campina Grande, utilizando o SUMO como ambiente de simulação.

Capítulo 3

Metodologia e Desenvolvimento

3.1 Visão Geral do Sistema

O sistema proposto tem como objetivo desenvolver um controlador semafórico adaptativo baseado em aprendizado de máquina e avaliar seu desempenho em comparação ao controle de tempo fixo, em ambiente simulado. A hipótese investigada é que adaptar o tempo de verde à composição real da fila de veículos pode resultar em melhor aproveitamento do ciclo semafórico do que manter tempos pré-definidos, independentemente da demanda presente. Todo o desenvolvimento e a validação são conduzidos no simulador SUMO, sem qualquer implantação em via real.

O desenvolvimento foi organizado em quatro etapas sequenciais. A primeira consiste na modelagem do cruzamento real entre a Av. Pref. Severino Cabral e a Rua Dep. Ascendino Moura no simulador SUMO, reproduzindo a geometria das faixas, os tipos de veículos presentes na via e as configurações semafóricas reais utilizadas no local. A segunda etapa compreende a geração do dataset de treinamento, realizada por meio de simulações automatizadas via TraCI, nas quais a composição da fila de veículos é registrada no início de cada fase verde e o tempo de esvaziamento real dessa fila é medido e utilizado como rótulo. A terceira etapa abrange o projeto e o treinamento da rede neural convolucional, que recebe como entrada a representação estruturada da fila e produz como saída a classe discreta correspondente ao tempo de esvaziamento previsto. A quarta e última etapa consiste na validação comparativa, na qual o modelo treinado é integrado ao SUMO como controlador semafórico ativo e seu desempenho é comparado ao de seis configurações de tempo fixo derivadas dos planos de temporização reais do cruzamento, em cenários de baixa, média e alta densidade de tráfego.

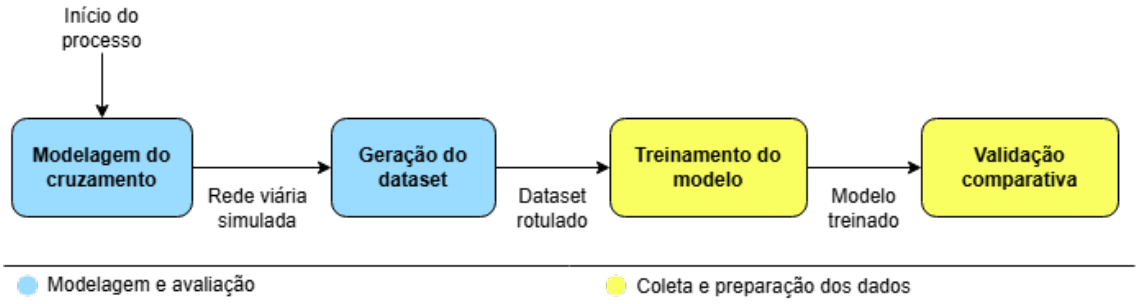


Figura 3.1: Visão geral das etapas de desenvolvimento do sistema

Uma decisão central do projeto foi tratar cada via do cruzamento de forma independente. O modelo não recebe o estado global da interseção, mas sim a composição da fila de uma única via no momento em que sua fase verde se inicia. Essa escolha reflete a premissa de projeto de que o tempo necessário para esvaziar uma fila depende fundamentalmente da quantidade, do tipo e da distribuição dos veículos presentes naquela via, tornando desnecessária a observação simultânea das demais aproximações para a tomada de decisão fase a fase.

3.2 Modelagem do Cruzamento no SUMO

O cenário de simulação utilizado neste trabalho é baseado no cruzamento real entre a Avenida Prefeito Severino Cabral e a Rua Deputado Ascendino Moura, localizado em Campina Grande, Paraíba. A escolha desse cruzamento se justifica pelo acesso às configurações reais de temporização semafórica praticadas no local, o que permite tanto a construção de um cenário de simulação fiel quanto a definição de baselines de comparação representativos da operação atual.

A malha viária foi obtida diretamente do OpenStreetMap por meio da ferramenta de importação nativa do SUMO, que converte dados geográficos reais em redes viárias simuláveis. Esse processo preserva a geometria das vias, o número de faixas e as características físicas do cruzamento, dispensando a necessidade de modelagem manual da infraestrutura.

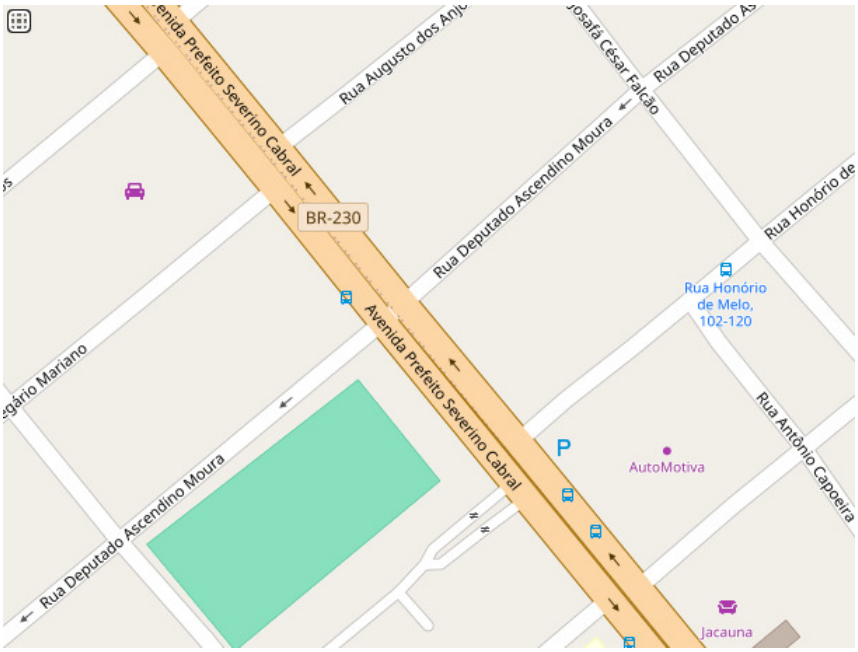


Figura 3.2: Recorte do cruzamento obtido do OpenStreetMap

O recorte importado do OpenStreetMap, ilustrado na Figura 3.2, contém elementos que extrapolam o escopo do estudo. Para focar exclusivamente no cruzamento de interesse, foram removidas a Rua Honório de Melo e a Rua Augusto dos Anjos, que não fazem parte do cenário analisado. Além disso, os sentidos de circulação dos veículos foram corrigidos para refletir a operação real das vias, e o conjunto semafórico foi inserido manualmente com os planos de temporização reais do cruzamento. O resultado é uma rede viária simplificada, centrada exclusivamente na interseção entre a Av. Pref. Severino Cabral e a Rua Dep. Ascendino Moura, conforme ilustrado na Figura 3.3.

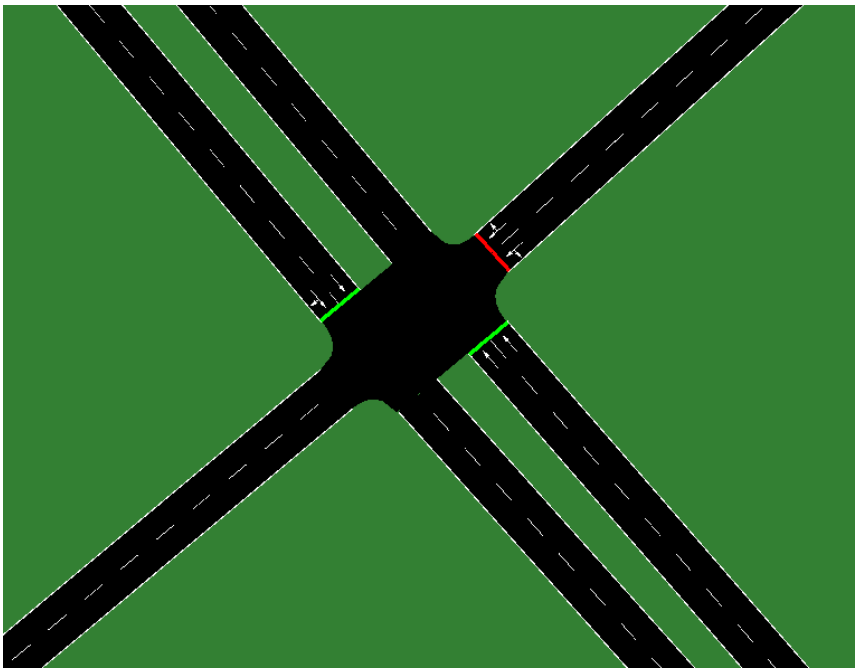


Figura 3.3: Rede viária refinada no SUMO

Foram definidos três tipos de veículos para compor o fluxo de tráfego nas simulações: motocicleta, carro de passeio e ônibus. Esses tipos correspondem às categorias nativas do SUMO, cujos atributos internos, como comprimento, aceleração máxima e distância mínima de segurança, foram mantidos conforme os valores padrão da ferramenta. O único parâmetro ajustado foi a velocidade máxima permitida, adaptada às condições da via modelada.

A lógica semafórica foi configurada com base nos planos de temporização reais do cruzamento. Para a etapa de geração do dataset, foi adotado o plano de maior duração disponível, com 48 segundos de verde para a Av. Pref. Severino Cabral e 18 segundos para a Rua Dep. Ascendino Moura, além de 4 segundos de amarelo e 3 segundos de vermelho geral entre as fases, conforme detalhado na Tabela 3.1.

Tabela 3.1: Plano semafórico utilizado na geração do dataset

Fase	**Duração (s)**
Verde — Av. Severino Cabral	48
Amarelo — Av. Severino Cabral	4
Vermelho geral	3
Verde — Rua Ascendino Moura	18
Amarelo — Rua Ascendino Moura	4
Vermelho geral	3

Os detectores de área foram posicionados nas faixas de cada via, com alcance de 90 metros para a Av. Pref. Severino Cabral e 100 metros para a Rua Dep.

Ascendino Moura. Esses detectores, acessados via interface TraCI, são responsáveis por capturar a composição da fila de veículos no início de cada fase verde, fornecendo as informações que alimentam tanto o processo de geração do dataset quanto o controlador neural durante a etapa de validação.

3.3 Geração do Dataset

O dataset de treinamento foi construído a partir de simulações automatizadas no SUMO, conduzidas inteiramente via interface TraCI. Para cada simulação, um script Python gerava dinamicamente um arquivo de rotas no formato `.rou`, definindo a quantidade e o tipo de veículos que circulariam pela rede viária. Esse processo dispensou qualquer intervenção manual entre uma simulação e outra, permitindo a execução de um grande volume de experimentos de forma sistemática.

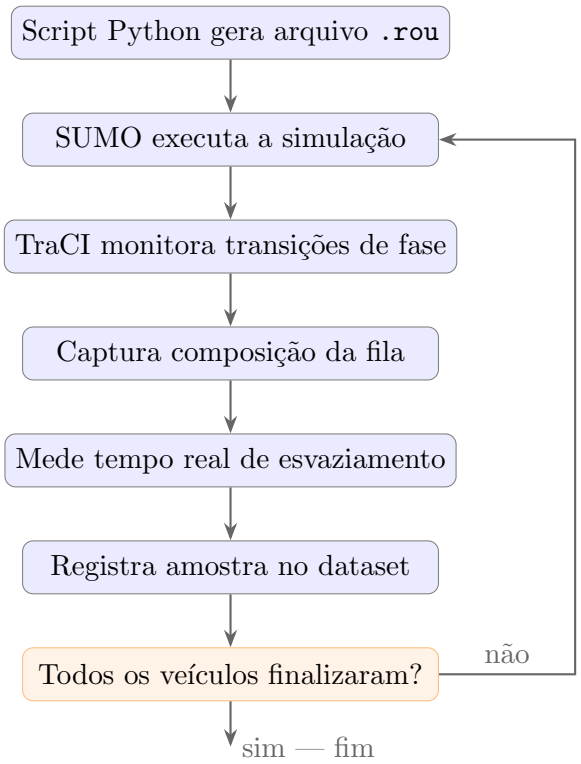


Figura 3.4: Fluxo de coleta automática do dataset via TraCI

Cada simulação era executada até que todos os veículos gerados tivessem completado seus trajetos, abrangendo múltiplos ciclos semafóricos. A cada transição de fase verde, o script registrava automaticamente três elementos: o tipo de cada veículo presente nas faixas da via que recebia o verde, sua posição relativa no detector, e o tempo real necessário para o esvaziamento completo dessa fila, utilizado como rótulo da amostra.

O tempo de esvaziamento é medido a partir do momento em que o verde é ativado até o instante em que o último veículo da fila cruza a linha de parada. Nos casos em que a fila não se esvazia dentro do ciclo verde disponível, o tempo é acumulado para o ciclo seguinte: se, por exemplo, restaram veículos após 20 segundos de verde e o ciclo subsequente precisou de 5 segundos adicionais para esvaziar o remanescente, o rótulo registrado é de 25 segundos. Esse tratamento garante que o dataset reflita o custo real de esvaziamento, independentemente de quantos ciclos foram necessários.

Com o objetivo de ampliar o volume do dataset sem introduzir distorções nos rótulos, cada amostra coletada foi duplicada por espelhamento de faixas: além do snapshot original, uma segunda versão foi gerada com as posições das duas faixas trocadas. Essa operação é válida sob a premissa de que as duas faixas de uma mesma via são fisicamente equivalentes neste cenário: ambas possuem a mesma largura, o mesmo limite de velocidade e recebem o verde simultaneamente, sem qualquer restrição de movimento (como faixas exclusivas de conversão) que diferencie o comportamento dos veículos entre elas. Sob essa premissa, o tempo de esvaziamento da fila depende da composição conjunta dos veículos nas duas faixas, e não de qual faixa específica cada veículo ocupa – de modo que trocar os veículos entre as faixas produz uma amostra igualmente válida, com o mesmo rótulo de tempo de esvaziamento da amostra original.

Essa duplicação não configura vazamento de informação entre os conjuntos de treinamento e teste, pois o espelhamento é aplicado antes da divisão entre treino e teste, e o embaralhamento aleatório das amostras (Seção 3.5) não garante que uma amostra e seu par espelhado permaneçam no mesmo conjunto. Essa possibilidade de uma amostra e sua versão espelhada serem separadas entre treino e teste, é reconhecida como uma limitação metodológica e discutida na Seção 5.2 deste trabalho, já que, na prática, isso pode inflar marginalmente a similaridade entre os conjuntos, dado que ambas as amostras carregam a mesma informação de composição.

Para garantir diversidade de composições de fila no dataset, as simulações foram organizadas em três grupos de volume de veículos. Em todas as simulações, o tempo total foi fixado em 3600 segundos, equivalente a uma hora de operação. Dentro de cada grupo, a quantidade de veículos por tipo foi sorteada aleatoriamente dentro de faixas predefinidas, distintas para cada via, conforme apresentado na Tabela 3.2. Essa variação de volume tem como objetivo expor o modelo a composições de fila diversas durante o treinamento, desde filas curtas com poucos veículos até filas longas com maior presença de ônibus e motocicletas misturados.

Tabela 3.2: Faixas de volume de veículos por grupo e via (simulações de 3600s)

Grupo	**Via**	**Mín. (veíc.)**	**Máx. (veíc.)**
A	Av. Severino Cabral	200	400
	Rua Ascendino Moura	100	250
B	Av. Severino Cabral	500	900
	Rua Ascendino Moura	300	600
C	Av. Severino Cabral	800	1300
	Rua Ascendino Moura	600	1000

A representação de cada amostra segue uma estrutura tensorial tridimensional com shape $(2, 2, N_{MAX})$, onde a primeira dimensão corresponde às duas faixas da via, a segunda às duas *features* por veículo, e a terceira à sequência de veículos detectados na faixa. Os veículos são ordenados do mais próximo ao mais distante da linha de parada, de forma que o primeiro elemento da sequência representa sempre o veículo na cabeça da fila. Essa ordenação é aplicada explicitamente pelo script de coleta e é mantida de forma consistente em todas as etapas do pipeline, que são: coleta, treinamento e validação, garantindo que a posição de cada veículo no tensor carregue significado espacial. O valor de N_{MAX} é calculado dinamicamente como o maior número de veículos observado em qualquer faixa ao longo de todo o dataset, situando-se tipicamente entre 17 e 23 veículos por faixa. Faixas com menos veículos que N_{MAX} recebem preenchimento com zeros (*zero-padding*) nas posições restantes, e o valor de N_{MAX} é salvo junto ao modelo para garantir consistência entre treinamento e inferência.

Cada veículo é representado por exatamente duas *features*. A primeira é o tipo do veículo, codificado como um valor escalar proporcional ao seu impacto estimado no tempo de esvaziamento da fila: motocicleta recebe 0,20, refletindo um tempo médio de passagem de 1 a 2 segundos; carro de passeio recebe 0,50, correspondendo a 3 a 4 segundos; e ônibus recebe 1,00, referente a 8 a 10 segundos. Essa codificação respeita a ordem física entre os tipos, mantém os valores no intervalo $[0, 1]$ para facilitar a convergência do modelo, e é semanticamente coerente com a saída da rede, que também prevê uma grandeza temporal. A segunda *feature* é a posição normalizada do veículo no detector, onde 0,0 indica um veículo encostado na linha de parada e 1,0 indica um veículo no limite do alcance do detector. Posições sem veículo recebem valor 0,0 em ambas as *features*, sinalizando ausência.

3.3.1 Preparação dos Dados

Os dados brutos capturados pelo TraCI passam por uma sequência de transformações antes de compor o tensor de entrada do modelo. Para cada veículo pre-

sente no detector no momento da transição de fase, o script consulta individualmente três atributos: o identificador do tipo de veículo, a posição em metros ao longo da via e a velocidade instantânea. A captura ocorre no instante em que o sistema detecta o início da fase amarela, representando um snapshot do estado da fila imediatamente antes da troca de direito de passagem.

A posição bruta, fornecida pelo SUMO como distância em metros desde o início da via, é convertida para um valor normalizado em três etapas. Primeiro, subtrai-se o offset do detector em relação ao início da via, obtendo a posição relativa dentro do alcance do sensor. Em seguida, o valor é invertido em relação ao comprimento do detector, de modo que posições próximas à linha de parada resultem em valores próximos de 0,0 e posições distantes resultem em valores próximos de 1,0. Por fim, o resultado é limitado ao intervalo $[0,0;\ 1,0]$ para garantir que eventuais ruídos de posição não gerem valores fora da escala esperada pelo modelo.

O tipo do veículo, retornado pelo SUMO como uma string identificadora do *vType* definido no arquivo de rotas, é convertido para um valor escalar por meio de um dicionário de mapeamento fixo: `motorcycle` recebe 0,20, `passenger` recebe 0,50 e `bus` recebe 1,00. Após a codificação, os veículos de cada faixa são ordenados por posição bruta de forma decrescente, posicionando o veículo mais próximo da linha de parada no índice 0 do tensor. Em seguida, os vetores de tipo e posição são preenchidos com zeros até o comprimento N_{MAX}, com o padding aplicado ao final da sequência, representando conceitualmente as posições vazias no fundo do detector.

O tensor final de cada amostra é montado empilhando as duas faixas da via em uma única operação, resultando no shape $(2,\ 2,\ N_{MAX})$. Os parâmetros necessários para reproduzir a estrutura do modelo em inferência, incluindo N_{MAX}, o número de classes de saída e os hiperparâmetros de arquitetura, que são persistidos em um arquivo de metadados gerado ao final do pipeline de preparação, garantindo consistência entre as etapas de treinamento, avaliação e validação comparativa.

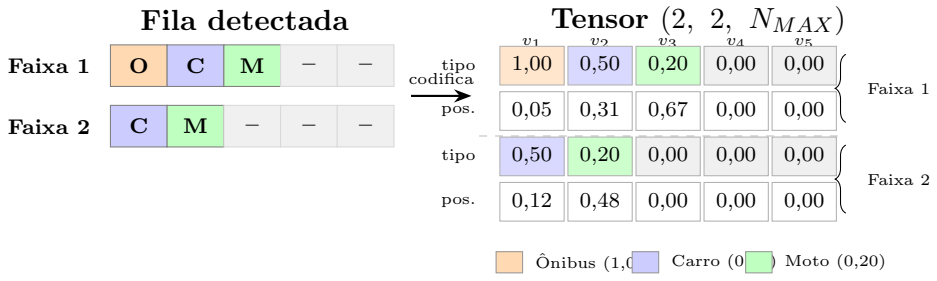


Figura 3.5: Representação tensorial da composição da fila de veículos

3.4 Arquitetura da Rede Neural

Antes de detalhar a arquitetura, é importante justificar a formulação do problema como classificação, e não como regressão. Embora o tempo de esvaziamento seja, por natureza, uma grandeza contínua, três fatores motivaram a adoção de uma abordagem por classificação discreta neste trabalho. Primeiro, o tempo de verde aplicado ao cruzamento é, na prática, um valor discreto em segundos, de modo que a saída do modelo já corresponde diretamente à unidade de controle utilizada pelo controlador semafórico, sem necessidade de arredondamento posterior. Segundo, a classificação permite o uso da função de perda tolerante descrita na Seção 3.5.1, que distribui probabilidade entre classes vizinhas ao alvo e penaliza de forma diferenciada erros pequenos e grandes – uma flexibilidade mais direta de implementar em um problema de classificação do que em uma regressão tradicional baseada em erro quadrático ou absoluto. Terceiro, a classificação fornece, para cada predição, uma distribuição de probabilidade sobre as classes vizinhas, o que permite estimar a confiança do modelo em cada predição, informação não obtida diretamente em uma regressão pontual.

Essa escolha tem como contrapartida uma limitação: a discretização impõe um teto de resolução igual a um segundo, de modo que variações mais finas no tempo de esvaziamento – por exemplo, frações de segundo – não são distinguidas pelo modelo. Essa perda de granularidade é considerada aceitável no contexto deste trabalho, uma vez que o tempo de verde aplicado pelo controlador semafórico já opera em granularidade de segundos inteiros, e a diferença de poucas frações de segundo entre predições não teria efeito prático sobre o controle da fase.

A rede neural desenvolvida para este trabalho é uma Rede Neural Convolucional 2D projetada para receber o tensor de composição da fila e produzir como saída a classe discreta correspondente ao tempo de esvaziamento previsto. A escolha por uma CNN se justifica pela natureza posicional dos dados de entrada: a ordem dos veículos no tensor carrega informação relevante pois um ônibus na cabeça da fila tem impacto diferente de um ônibus no fundo, e as camadas convolucionais são capazes de capturar esses padrões locais de forma eficiente, sem a necessidade de engenharia manual de features.

A arquitetura é composta por três blocos principais: um bloco convolucional compartilhado, um branch duplo de agregação e um bloco de classificação denso. O fluxo completo dos dados é ilustrado na Figura 3.6.

3.4.1 Bloco Convolucional

O bloco convolucional é formado por três camadas Conv2D sequenciais, cada uma seguida de normalização em lote (BatchNorm2d) e função de ativação Leaky-

ReLU com inclinação negativa de 0,01. A primeira camada recebe os 2 canais de entrada do tensor, correspondentes às duas faixas da via, e aplica 128 filtros com kernel de tamanho (2×5). Esse kernel colapsa a dimensão de altura do tensor já na primeira camada, unificando as duas *features* (tipo e posição) de cada veículo, enquanto a janela de 5 posições consecutivas permite capturar padrões locais na sequência da fila. A segunda e a terceira camadas aplicam, respectivamente, 64 e 32 filtros com kernels (1×3), refinando progressivamente as representações extraídas. Ao final do bloco convolucional, a dimensão espacial de veículos é reduzida em 8 posições em relação a N_{MAX}, resultado da soma dos raios dos kernels aplicados.

3.4.2 Branch Duplo

Após o bloco convolucional, os mapas de características seguem dois caminhos paralelos antes de serem combinados. O primeiro, denominado **branch local**, aplica uma operação de *Flatten*, preservando a posição espacial de cada padrão detectado pelos filtros. Esse branch permite que o modelo distinga, por exemplo, a presença de um ônibus na cabeça da fila de um ônibus no fundo, informação que é perdida em operações de agregação global.

O segundo, denominado **branch global**, aplica *Global Average Pooling* (GAP), calculando a média de cada filtro ao longo de todas as posições da sequência. Como cada filtro aprende a detectar padrões específicos de composição, como a presença de ônibus ou a concentração de motocicletas, o valor médio resultante do GAP funciona como uma contagem implícita aprendida via retropropagação, fornecendo ao modelo um resumo da composição global da fila independente da posição.

Essa combinação de branches surgiu da observação de que, sem o branch global, o modelo apresentava dificuldade em prever corretamente tempos de esvaziamento para filas grandes, na faixa de 27 a 40 segundos. O branch local sozinho capturava padrões posicionais, mas não traduzia adequadamente o volume total da fila em tempo de esvaziamento. A adição do GAP resolveu esse gargalo sem necessidade de criar features manuais de contagem. Os vetores produzidos pelos dois branches são concatenados em um único vetor antes de entrar no bloco de classificação.

3.4.3 Bloco de Classificação e Saída

O bloco de classificação é composto por três camadas lineares sequenciais. A primeira expande o vetor concatenado para uma representação de alta dimensionalidade, seguida de normalização em lote, LeakyReLU e Dropout com taxa de 0,50 para regularização. A segunda camada reduz essa representação para 512 neurônios,

seguida de LeakyReLU e Dropout com taxa de 0,30. A terceira e última camada projeta os 512 neurônios para o número de classes de saída, que corresponde ao número de segundos distintos observados no dataset, tipicamente 49 classes representando tempos de esvaziamento de 0 a 48 segundos.

A camada de saída não aplica *softmax* explicitamente, pois essa operação é incorporada internamente pela função de perda durante o treinamento, seguindo a prática recomendada pelo PyTorch para evitar instabilidade numérica. Durante a inferência, o tempo previsto é obtido diretamente pelo índice da classe com maior logit, o que é equivalente ao argmax da distribuição softmax, e é aplicado diretamente como duração do verde da fase correspondente.

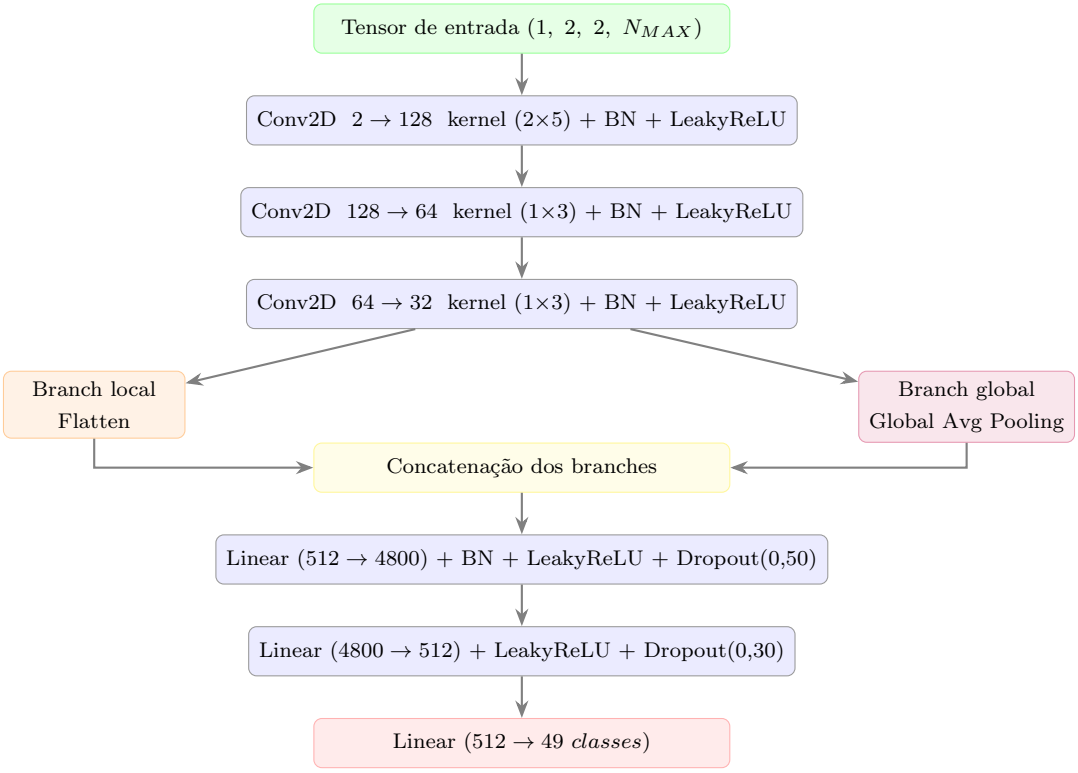


Figura 3.6: Arquitetura da rede neural convolucional com branch duplo

3.5 Treinamento do Modelo

O treinamento foi conduzido sobre o dataset gerado nas simulações, previamente preparado por um pipeline de balanceamento e divisão. O processo de coleta produziu um total de 637.692 configurações distintas de fila observadas ao longo das simulações. Antes da divisão, as amostras foram embaralhadas aleatoriamente para evitar viés de ordenação, e classes com menos de 2 amostras foram duplicadas para garantir representatividade mínima, enquanto as 26 classes com mais de 5.000 amostras foram truncadas nesse limite para evitar desequilíbrio excessivo entre as classes

de tempo. Esse balanceamento reduziu o conjunto de 637.692 para 162.570 amostras – uma redução deliberada, motivada pela necessidade de evitar que classes de tempo muito frequentes (tipicamente os tempos de esvaziamento intermediários) dominassem o treinamento em detrimento de classes raras, como tempos muito curtos ou muito longos.

O conjunto balanceado foi então dividido em três partes independentes, na proporção 70% para treinamento, 15% para validação e 15% para teste, com seed fixa para reprodutibilidade, resultando em 113.779 amostras de treinamento, 24.394 de validação e 24.397 de teste, distribuídas em 49 classes (0 a 48 segundos). O conjunto de validação é utilizado para monitorar o desempenho ao longo das épocas – orientando os critérios de *early stopping*, a redução da taxa de aprendizado e a seleção do melhor *checkpoint* do modelo –, enquanto o conjunto de teste é reservado como um *holdout* avaliado uma única vez, após o encerramento completo do treinamento, garantindo que nenhuma decisão tomada durante o treinamento seja influenciada pelos dados de teste.

3.5.1 Função de Perda

A função de perda utilizada é uma variante da entropia cruzada categórica adaptada para problemas de classificação ordinal, denominada *TolerantCrossEntropyLoss*. Em vez de atribuir probabilidade 1 exclusivamente à classe correta, a função distribui essa probabilidade de forma uniforme entre todas as classes dentro de uma janela de tolerância $\pm T$ em torno do alvo. Para um alvo de 20 segundos com tolerância $T = 3$, por exemplo, as classes 17, 18, 19, 20, 21, 22 e 23 recebem probabilidade $\frac{1}{7}$ cada, enquanto as demais recebem probabilidade zero.

Formalmente, seja y a classe alvo (o tempo real de esvaziamento, em segundos) e T a tolerância adotada. Define-se a distribuição-alvo suavizada $q(c)$ sobre cada classe $c \in \{0, 1, \ldots, C-1\}$, onde C é o número total de classes, como:

$$q(c) = \{\ 12T + 1, se|c - y| \leq T0, casocontrário \tag{3.1}$$

A função de perda é então calculada como a entropia cruzada entre essa distribuição-alvo suavizada $q(c)$ e a distribuição de probabilidade predita pelo modelo $p(c)$, obtida a partir dos logits de saída por meio de uma função *softmax*:

$$\mathcal{L} = -\sum_{c=0}^{C-1} q(c) \cdot \log p(c) \tag{3.2}$$

Para um alvo de 20 segundos com tolerância $T = 3$, por exemplo, as classes 17, 18, 19, 20, 21, 22 e 23 recebem probabilidade $q(c) = \frac{1}{7}$ cada, enquanto as demais recebem probabilidade zero. Quando $T = 0$, a Equação 3.1 se reduz à distribuição

one-hot tradicional, e a função de perda equivale à entropia cruzada categórica padrão – daí a escolha de $T = 0$ como caso de referência entre as variantes avaliadas na Seção 3.5.3.

Essa formulação relaciona-se com a classificação ordinal porque reconhece explicitamente que as classes possuem uma relação de ordem e proximidade entre si – diferentemente da entropia cruzada categórica padrão, que trata todas as classes incorretas como igualmente erradas, independentemente de sua distância até a classe correta. Ao distribuir probabilidade entre classes vizinhas proporcionalmente à tolerância operacional aceitável (alguns segundos, no contexto do controle semafórico), a função penaliza predições distantes do alvo mais severamente do que predições próximas, sem exigir a alteração da arquitetura de saída da rede para uma regressão ordinal explícita.

Essa abordagem reflete a natureza do problema: prever 18 segundos quando o tempo real de esvaziamento é 19 segundos é operacionalmente aceitável e não deve ser penalizado da mesma forma que um erro de 10 segundos. A ideia geral de distribuir probabilidade entre classes vizinhas está relacionada a técnicas como *label smoothing* e regressão ordinal com classes vizinhas, presentes na literatura de aprendizado de máquina [13].

Foram treinados quatro modelos com valores distintos de tolerância: $T = 0$ (equivalente à entropia cruzada padrão), $T = 1$, $T = 2$ e $T = 3$. O modelo com $T = 3$ foi selecionado como o modelo principal para a etapa de validação comparativa, por apresentar o melhor balanço entre acurácia e generalização nas avaliações realizadas durante o desenvolvimento.

3.5.2 Configuração do Treinamento

O otimizador utilizado foi o AdamW, com taxa de aprendizado inicial de 0,0001 e decaimento de pesos (*weight decay*) de 0,001. O tamanho do lote (*batch size*) foi calculado dinamicamente como o total de amostras de treinamento dividido por 1.000, de modo que o conjunto de treino fosse percorrido em aproximadamente 1.000 batches por época. O gradiente foi limitado a norma máxima de 1,0 por clipping para estabilizar o treinamento.

O número máximo de épocas foi definido em 1.000. Dois critérios de parada antecipada foram adotados: *early stopping* com paciência de 20 épocas, interrompendo o treinamento caso a loss de validação não apresentasse melhora em relação ao melhor valor histórico observado, e parada por convergência, acionada quando a loss de validação atingia valor inferior a 0,5. A taxa de aprendizado foi reduzida pela metade sempre que a loss de validação não melhorasse por 15 épocas consecutivas, com valor mínimo de 10^{-5}, seguindo a estratégia *ReduceLROnPlateau*.

O melhor modelo de cada treinamento foi salvo com base em dois critérios simultâneos: menor loss de validação observada e diferença entre loss de validação e loss de treinamento inferior a 0,08, garantindo que o checkpoint salvo não apresentasse sobreajuste significativo. Checkpoints periódicos foram salvos a cada 10 épocas como salvaguarda adicional.

3.5.3 Variantes Avaliadas

Antes de chegar à arquitetura final, foram desenvolvidas e avaliadas variantes intermediárias da rede neural, todas dentro do paradigma de classificação. As variantes incluíram modelos de Perceptron Multicamadas com diferentes configurações de profundidade, e versões da CNN com diferentes configurações de kernels e profundidade convolucional. A variante selecionada para o trabalho foi a que obteve maior acurácia nas avaliações, incorporando o bloco convolucional de três camadas com branch duplo descrito na seção anterior.

3.5.4 Comparação com Modelos Supervisionados Alternativos

Para avaliar se os ganhos observados decorrem especificamente da arquitetura CNN proposta, ou se seriam obtidos por qualquer modelo preditivo minimamente competente treinado sobre os mesmos dados, foram treinados quatro modelos supervisionados alternativos, utilizando exatamente os mesmos conjuntos de treinamento e teste da CNN: Regressão Linear, *Random Forest* em sua variante de classificação, *Random Forest* em sua variante de regressão (com predição arredondada para o inteiro mais próximo), e um Perceptron Multicamadas (MLP) com duas camadas ocultas de 128 e 64 neurônios.

Para cada amostra, o tensor de entrada $(2, 2, N_{MAX})$ foi achatado em um vetor unidimensional antes de ser fornecido a esses modelos, uma vez que nenhum deles é capaz de processar nativamente a estrutura espacial bidimensional utilizada pela CNN. Todos os modelos foram avaliados sobre o mesmo conjunto de teste, utilizando as mesmas métricas de tolerância definidas na Seção 3.5.1 (acurácia com tolerância de ± 0, ± 1, ± 2 e ± 3 segundos), além do erro médio absoluto (MAE), da raiz do erro quadrático médio (RMSE) e do viés médio das predições, garantindo uma comparação justa e nos mesmos termos utilizados para avaliar a CNN.

3.6 Protocolo de Validação

A validação comparativa foi conduzida por meio de 3.150 simulações no SUMO, organizadas em 150 cenários de tráfego distintos, cada um executado sob 3 sementes aleatórias (*seeds*) diferentes e, para cada semente, testado com os 7 controladores – uma execução com o controlador neural e seis com os planos de temporização fixa derivados dos dados reais do cruzamento. Essa estrutura de pareamento garante que, para uma mesma combinação de cenário e semente, todos os sete controladores enfrentem exatamente o mesmo padrão de chegada de veículos, isolando o efeito do controlador como única variável experimental. A repetição de cada cenário sob múltiplas sementes permite estimar a variabilidade inerente à estocasticidade interna do simulador – geração e comportamento dos veículos –, possibilitando o cálculo de desvio padrão, intervalo de confiança e testes de significância estatística sobre as diferenças observadas entre controladores, conforme detalhado na Seção 3.6.4.

Para cada cenário, o volume total de veículos por via foi sorteado uma única vez e mantido fixo (*congelado*) entre as três sementes testadas, de modo que a variação entre execuções de um mesmo cenário decorra exclusivamente da semente do simulador, e não de uma nova amostragem de volume. As simulações foram executadas em paralelo, com até 8 processos SUMO simultâneos, cada um conectado a uma instância independente via TraCI, e os resultados de cada execução individual – cenário, semente e controlador – foram persistidos em disco de forma incremental, a cada 50 execuções concluídas, permitindo a retomada da rodada em caso de interrupção sem perda do progresso já realizado. Cada execução individual simula 3.600 segundos de tráfego antes de encerrar.

3.6.1 Cenários de Teste

Os 150 cenários foram distribuídos igualmente entre os três grupos de volume definidos na etapa de geração do dataset: 50 cenários por grupo. Em cada cenário, o volume de veículos por via foi sorteado aleatoriamente dentro dos mesmos intervalos utilizados no treinamento, conforme a Tabela 3.2. A variação entre cenários do mesmo grupo é resultado do ruído aleatório dentro desses intervalos, com sementes distintas para cada simulação.

A composição por tipo de veículo nos cenários de teste seguiu a distribuição de 50% carros de passeio, 40% motocicletas e 10% ônibus, aplicada igualmente para as duas vias. Segundo dados do Departamento Estadual de Trânsito da Paraíba (DETRAN-PB) relativos à frota de Campina Grande em fevereiro de 2026, a proporção real entre essas três categorias de veículo é de aproximadamente 51% de automóveis, 48% de motocicletas e 0,7% de ônibus e micro-ônibus [22]. A distribuição adotada nos cenários de teste aproxima-se dessa proporção real para carros

e motocicletas, mas eleva deliberadamente a participação de ônibus para 10%, de modo a garantir um número mínimo de ocorrências dessa categoria nos 150 cenários de teste – o que não ocorreria de forma consistente caso a proporção real de 0,7% fosse adotada diretamente, dada a baixa frequência absoluta de ônibus na frota local. Essa escolha é relevante porque o ônibus é o tipo de veículo com maior impacto individual estimado no tempo de esvaziamento da fila, tornando importante avaliar o comportamento do modelo diante de sua presença, ainda que de forma proporcionalmente superior à observada na frota real.

Durante a geração do dataset de treinamento, por outro lado, os veículos foram distribuídos de forma aproximadamente uniforme entre os três tipos, resultando em cerca de 33% de cada categoria. Essa escolha teve como objetivo garantir que o modelo fosse exposto a um volume suficiente de exemplos de todas as combinações possíveis de composição durante o treinamento, incluindo composições raras no tráfego real, como filas com alta concentração de ônibus, que seriam subrepresentadas caso a distribuição de treinamento seguisse a mesma proporção observada no tráfego real.

Essa diferença entre as distribuições de treinamento e teste representa, portanto, um compromisso metodológico: o treinamento prioriza a diversidade de composições para favorecer a generalização do modelo, enquanto o teste prioriza o realismo da avaliação. Ainda assim, essa diferença constitui uma fonte potencial de viés, já que o desempenho reportado neste trabalho reflete a distribuição específica adotada nos cenários de teste, e composições de tráfego significativamente diferentes – por exemplo, com maior concentração de ônibus do que a testada – poderiam resultar em desempenho distinto do modelo. Essa limitação e seus efeitos são retomados de forma mais ampla nas Considerações Finais deste trabalho.

3.6.2 Planos de Temporização Fixa

Os seis planos de temporização fixa utilizados como baseline foram fornecidos pela STTP e correspondem às configurações reais operadas no cruzamento entre a Av. Pref. Severino Cabral e a Rua Dep. Ascendino Moura. Cada plano define os tempos de verde, amarelo e vermelho geral para as duas fases do ciclo, com duração total de ciclo variando de 30 a 80 segundos, conforme apresentado na Tabela 3.3.

Tabela 3.3: Planos de temporização fixa fornecidos pela STTP

Fase	**T1 (80s)**	**T2 (72s)**	**T3 (60s)**	**T4 (50s)**	**T5 (40s)**	**T6 (30s)**
Verde F1 (Severino)	48	40	30	24	19	12
Amarelo F1	4	4	4	4	4	4
Vermelho geral	3	3	2	2	1	1
Verde F2 (Ascendino)	18	18	16	12	10	8
Amarelo F2	4	4	4	4	4	4
Vermelho geral	3	3	3	3	2	2

A aplicação de cada plano ao longo do dia segue uma programação semanal definida pela STTP, apresentada na Tabela 3.4. O plano T1, de maior duração de ciclo, é aplicado nos períodos de maior demanda, como o intervalo entre 7h e 19h nos dias úteis (segunda a sexta). Os planos de menor duração, como T5 e T6, são utilizados nos períodos de baixa demanda, como a madrugada e os horários noturnos. Aos domingos, o plano T2 vigora entre 7h e 20h, refletindo um fluxo de tráfego inferior ao dos dias úteis nesse horário.

Tabela 3.4: Programação semanal dos planos de temporização fornecida pela STTP

Dia	**Início**	**Fim**	**Plano**
	00:00	05:00	T6
	05:00	06:00	T5
	06:00	06:30	T4
	06:30	07:00	T2
Segunda a Sexta	07:00	19:00	T1
	19:00	20:00	T2
	20:00	21:00	T3
	21:00	22:00	T4
	22:00	23:00	T5
	23:00	00:00	T6
	00:00	05:00	T6
	05:00	06:00	T5
	06:00	06:30	T4
	06:30	07:00	T2
Sábado	07:00	13:00	T1
	13:00	20:00	T2
	20:00	21:00	T3
	21:00	22:00	T4
	22:00	23:00	T5
	23:00	00:00	T6
	00:00	05:00	T6
	05:00	06:00	T5
	06:00	07:00	T3
Domingo	07:00	20:00	T2
	20:00	21:00	T3
	21:00	22:00	T4
	22:00	23:00	T5
	23:00	00:00	T6

É importante destacar uma escolha metodológica desta validação: embora cada plano fixo seja aplicado, na operação real do cruzamento, apenas em horários específicos do dia – por exemplo, os planos T5 e T6, de menor duração de ciclo, são utilizados somente em períodos de baixa demanda, como a madrugada –, todos os seis planos foram testados nos três grupos de volume de tráfego (baixo, médio e alto), incluindo grupos de volume aos quais o plano não seria efetivamente submetido na operação real da STTP. Essa escolha foi deliberada: o objetivo não é reproduzir a operação real do cruzamento tal como programada atualmente, mas sim isolar

o efeito de cada estratégia de temporização – fixa ou adaptativa – sob as mesmas condições de demanda, permitindo avaliar, por exemplo, como um plano originalmente projetado para baixa demanda (como o T6) se comportaria caso fosse exposto a um cenário de alta demanda, e comparar esse comportamento ao do controlador neural sob a mesma condição. Essa análise é particularmente relevante para evidenciar a rigidez dos planos de tempo fixo diante de variações de demanda, tema central da motivação deste trabalho, discutida na Contextualização deste documento.

O tempo mínimo de verde adotado no controlador neural foi definido em 8 segundos, valor correspondente ao menor tempo de verde praticado pela STTP nesse cruzamento, referente à fase da Rua Dep. Ascendino Moura no plano T6. Esse valor garante tempo suficiente para a travessia segura de pedestres, sendo aplicado sempre que o modelo prevê um tempo de esvaziamento inferior a esse limite ou quando nenhum veículo é detectado na faixa no momento da transição de fase.

3.6.3 Métricas de Avaliação

O desempenho de cada controlador foi avaliado por meio de quatro métricas, calculadas separadamente para cada grupo de volume e agregadas como média sobre os 50 cenários do grupo.

O **throughput médio** mede a taxa de veículos que cruzam a linha de parada por segundo de verde concedido, calculado para cada ciclo de verde como o número de veículos que cruzaram dividido pela duração do verde. Essa métrica reflete a eficiência do controlador em aproveitar o tempo de verde disponível.

O **tempo de verde desperdiçado** mede, em segundos, o intervalo entre o último veículo cruzar a linha de parada e o encerramento da fase verde. Esse valor só é computado em ciclos nos quais a fila se esvaziou completamente, refletindo diretamente o excesso de verde concedido além do necessário.

O **delay global** corresponde ao tempo médio de espera por veículo no cruzamento, calculado como a diferença entre o tempo de espera acumulado de cada veículo ao sair do detector e o tempo acumulado ao entrar, isolando o delay específico da passagem pelo cruzamento. Os resultados são reportados também de forma desagregada por via, como **delay por via**, permitindo identificar assimetrias no desempenho do controlador entre a Av. Pref. Severino Cabral e a Rua Dep. Ascendino Moura.

Cada controlador é avaliado individualmente, sem agregação dos planos fixos em um valor único, de modo que o controlador neural seja comparado diretamente contra cada plano real do cruzamento.

3.6.4 Análise Estatística

Para cada métrica e cada grupo de volume (baixo, médio e alto), os resultados são reportados como média, intervalo de confiança de 95% (IC95) e desvio padrão sobre as $n = 150$ execuções pareadas (50 cenários $\times$ 3 sementes) de cada controlador. Além da agregação descritiva, a significância das diferenças observadas entre o controlador neural e cada um dos seis planos fixos é avaliada por meio de testes estatísticos pareados, uma vez que ambos enfrentam exatamente o mesmo tráfego dentro de cada combinação de cenário e semente.

Dois testes pareados são aplicados a cada comparação: o teste de Wilcoxon *signed-rank*, não paramétrico, e o teste t pareado, paramétrico. O teste de Wilcoxon é adotado como referência principal, por não assumir normalidade na distribuição das diferenças – premissa que não se sustenta de forma geral neste estudo, especialmente no grupo de volume alto, onde a variabilidade entre cenários é maior e as distribuições de algumas métricas apresentam assimetria e valores atípicos (*outliers*). O teste t pareado é reportado em conjunto, como verificação complementar; divergências entre os dois testes, quando ocorrem, são discutidas pontualmente no Capítulo 4. Diferenças com $p < 0{,}05$ em ambos os testes são consideradas estatisticamente significativas; comparações em que ao menos um dos testes resulta em $p \geq 0{,}05$ são reportadas como estatisticamente inconclusivas, e não como equivalência confirmada entre os controladores.

Capítulo 4

Resultados e Discussão

Este capítulo apresenta os resultados obtidos em três etapas distintas: primeiro, a avaliação do desempenho preditivo do modelo no conjunto de teste (*holdout*) do dataset, nunca utilizado durante o treinamento; em seguida, a comparação desse desempenho com o de quatro modelos supervisionados alternativos, sobre o mesmo conjunto de teste; por fim, a comparação do controlador neural com os seis planos de temporização fixa nas simulações de validação, conforme o protocolo descrito na Seção 3.6, incluindo a significância estatística das diferenças observadas.

4.1 Desempenho do Modelo no Dataset

O treinamento do modelo foi monitorado ao longo das épocas por meio das curvas de loss e acurácia sobre o conjunto de validação, apresentadas nas Figuras 4.1 e 4.2.

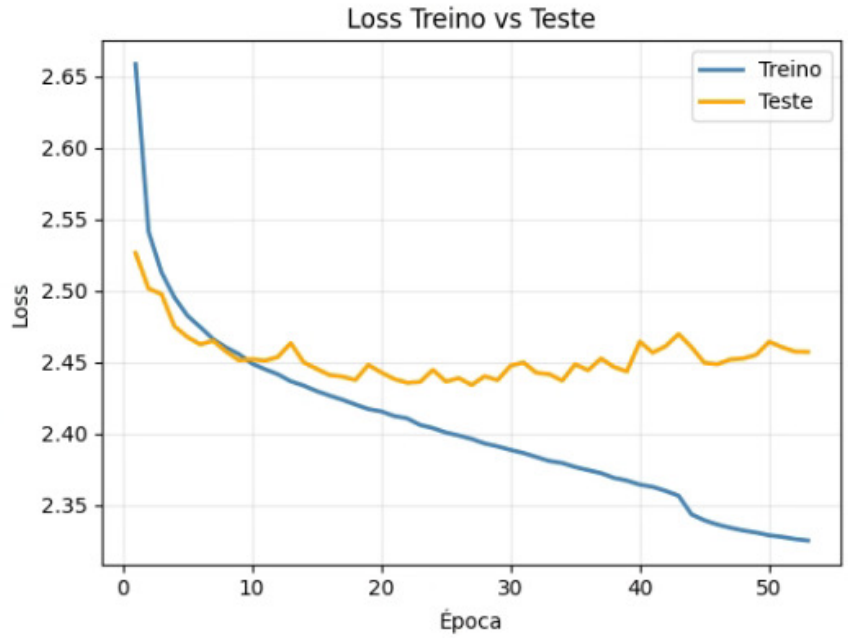


Figura 4.1: Evolução da loss de treino e validação ao longo das épocas

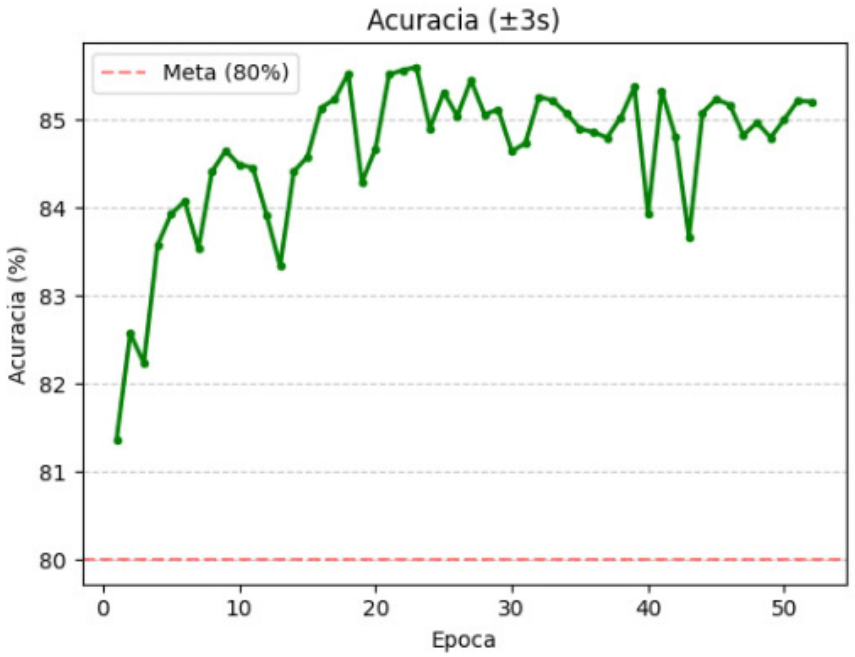


Figura 4.2: Evolução da acurácia com tolerância ±3s ao longo das épocas, sobre o conjunto de validação

As curvas de loss convergem de forma consistente, com a loss de validação atingindo seu melhor valor (2,4317) na época 26 e não sendo superada nas épocas seguintes, apesar de o treinamento ter continuado por mais de 40 épocas adicionais

antes da interrupção. O checkpoint correspondente à época 26 foi selecionado como modelo final, por satisfazer simultaneamente os critérios de menor loss de validação e diferença aceitável entre as losses de treino e validação (*gap* de 0,0304, abaixo do limite de 0,08 adotado).

O desempenho preditivo desse modelo foi então avaliado uma única vez sobre o conjunto de teste (*holdout*), reservado integralmente para essa finalidade e nunca utilizado durante o treinamento ou a seleção do modelo. Sobre as 24.397 amostras de teste, o modelo obteve acurácia de 84,92% com tolerância de ±3 segundos – a métrica oficial de avaliação, alinhada à tolerância adotada na função de perda –, com erro médio absoluto (MAE) de 2,09 segundos, raiz do erro quadrático médio (RMSE) de 2,88 segundos e viés médio de $+0,39$ segundos, indicando uma leve tendência do modelo a superestimar o tempo de esvaziamento. A Tabela 4.1 detalha a acurácia obtida sob diferentes níveis de tolerância.

Tabela 4.1: Acurácia da CNN no conjunto de teste, por nível de tolerância

Tolerância	**Acurácia (%)**
±0s (acerto exato)	16,38
±1s	45,29
±2s	71,84
±3s (oficial)	84,92

A queda acentuada de acurácia entre ±3s (84,9%) e o acerto exato (16,4%) é esperada e decorre diretamente do desenho da função de perda: a *TolerantCrossEntropyLoss* não penaliza diferenças pequenas dentro da margem de tolerância, de modo que o modelo é otimizado para acertar a janela operacionalmente relevante, e não o segundo exato. A proximidade entre a loss final de teste (2,4443) e a melhor loss de validação (2,4317) indica que o modelo generaliza de forma consistente para dados nunca vistos, sem sinais de sobreajuste ao conjunto de validação utilizado durante o treinamento.

4.1.1 Comparação com Modelos Supervisionados Alternativos

Para avaliar se o desempenho obtido decorre especificamente da arquitetura CNN proposta, a Tabela 4.2 compara seus resultados aos dos quatro modelos alternativos descritos na Seção 3.5.4 (Regressão Linear, *Random Forest* de classificação e de regressão, e MLP), todos avaliados sobre o mesmo conjunto de teste.

Tabela 4.2: Comparação entre a CNN e modelos supervisionados alternativos, no mesmo conjunto de teste ($n = 24.397$)

Modelo	**Acc. ±3s (%)**	**MAE (s)**	**RMSE (s)**	**Viés (s)**	**Tempo de treino**
CNN (proposta)	**84,92**	2,09	2,88	$+0,39$	— (*checkpoint* pré-treinado)
MLP	81,90	1,82	3,09	$+0,27$	47,34 s
Random Forest (regressão)	81,70	**1,72**	**2,79**	$+0,01$	11,97 s
Random Forest (classificação)	78,80	2,04	3,50	—	4,64 s
Regressão Linear	74,10	2,66	3,88	$0,00$	**0,12 s**

A CNN obteve a maior acurácia na métrica de tolerância operacional (±3s), superando o segundo colocado (MLP) por 3 pontos percentuais e o pior modelo (Regressão Linear) por quase 11 pontos percentuais. Entretanto, o resultado não é uniforme em todas as métricas: o *Random Forest* de regressão obteve o menor erro médio absoluto (1,72s) e o menor RMSE (2,79s), além do viés mais próximo de zero ($+0,01$s), superando a CNN nessas três métricas.

Essa aparente contradição se explica pela diferença entre os critérios de otimização de cada modelo. Os baselines de regressão são treinados para minimizar o erro médio ponto a ponto, o que naturalmente favorece métricas como MAE e RMSE. A CNN, por sua vez, é treinada com a *TolerantCrossEntropyLoss*, que não distingue entre acertar exatamente o alvo e acertar dentro de uma margem de poucos segundos, exatamente a régua de avaliação que importa para a aplicação de controle semafórico, na qual uma diferença de 1 ou 2 segundos na duração do verde não tem efeito prático observável. Em outras palavras, o *Random Forest* de regressão erra, em média, valores menores, mas a CNN erra com mais frequência *dentro da margem operacionalmente aceitável*, que é o critério relevante para a decisão de controle.

Do ponto de vista de custo computacional, todos os baselines treinam em segundos (de 0,12s para a Regressão Linear a 47,34s para o MLP), ordens de grandeza mais rápido do que o treinamento da CNN. Isso evidencia um compromisso relevante: a CNN oferece a maior acurácia na métrica operacional de interesse, à custa de um treinamento consideravelmente mais custoso do que alternativas mais simples, que permanecem competitivas e podem constituir opções viáveis caso o custo de treinamento seja um fator limitante em contextos de recursos computacionais restritos.

4.2 Throughput Médio

O throughput médio mede a taxa de veículos que cruzam a linha de parada por segundo de verde concedido. A Figura 4.3 apresenta os resultados para os três grupos de volume, com barras de erro representando o intervalo de confiança de 95% sobre as 150 execuções pareadas de cada controlador.

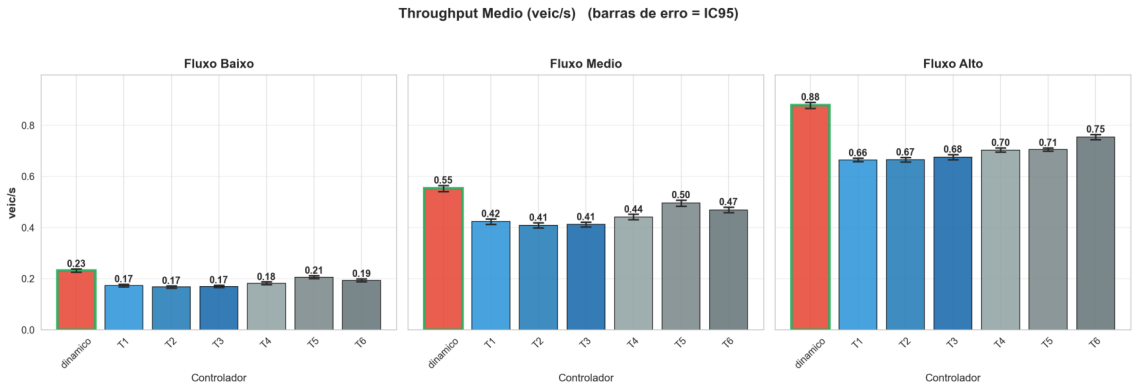


Figura 4.3: Throughput médio por controlador, para os fluxos Baixo, Médio e Alto (veíc/s). Barras de erro representam IC95.

O controlador neural (dinâmico) apresentou o maior throughput em todos os três grupos de volume: 0,232 veíc/s no fluxo baixo, contra 0,206 veíc/s do melhor plano fixo (T5); 0,553 veíc/s no fluxo médio, contra 0,496 veíc/s do melhor fixo (também T5); e 0,878 veíc/s no fluxo alto, contra 0,753 veíc/s do melhor fixo (T6). A vantagem relativa do controlador neural cresce com o volume de tráfego, chegando a aproximadamente 17% acima do melhor plano fixo no fluxo alto. Todas as diferenças entre o controlador neural e os seis planos fixos, em todos os grupos de volume, mostraram-se estatisticamente significativas ($p < 0{,}001$ em ambos os testes pareados, Wilcoxon e t), confirmando que a vantagem observada não decorre da variabilidade aleatória das simulações.

4.3 Tempo de Verde Desperdiçado

O tempo de verde desperdiçado mede o intervalo entre o esvaziamento completo da fila e o encerramento da fase verde. A Figura 4.4 apresenta os resultados para os três grupos de volume.

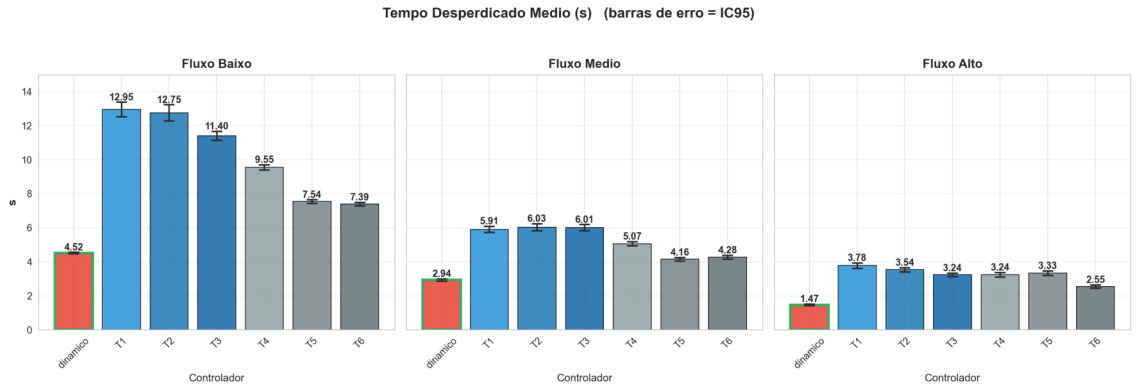


Figura 4.4: Tempo de verde desperdiçado médio por controlador, para os fluxos Baixo, Médio e Alto (s). Barras de erro representam IC95.

O controlador neural apresentou o menor desperdício de verde em todos os grupos de volume. No fluxo baixo, desperdiçou em média 4,52 segundos por ciclo, enquanto os planos fixos T1 e T2 desperdiçaram 12,95 e 12,75 segundos, respectivamente, quase o triplo. No fluxo médio, o desperdício do controlador neural foi de 2,94 segundos, contra 6,03 segundos do pior plano (T2). No fluxo alto, o controlador neural registrou 1,47 segundo de desperdício médio, valor inferior ao de todos os planos fixos, sendo o mais próximo o T6, com 2,55 segundos. Todas essas diferenças são estatisticamente significativas ($p < 0{,}001$ em ambos os testes, para todos os planos fixos e grupos de volume).

Esses resultados confirmam que o controlador neural utiliza o tempo de verde de forma substancialmente mais eficiente do que os planos fixos, com significância estatística consistente em todos os cenários avaliados. A redução do desperdício é uma consequência direta da predição adaptativa: ao ajustar a duração do verde à composição observada da fila, o sistema evita manter o verde aberto além do necessário, liberando tempo de ciclo para as demais fases.

4.4 Taxa de Esvaziamento

A taxa de esvaziamento indica a proporção de ciclos em que a fila foi completamente esvaziada durante o verde concedido. A Figura 4.5 apresenta os resultados para os três grupos de volume.

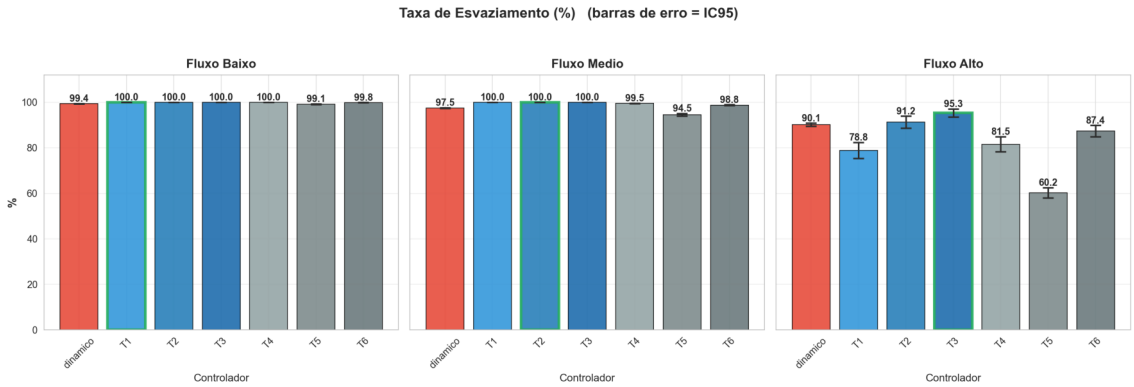


Figura 4.5: Taxa de esvaziamento por controlador, para os fluxos Baixo, Médio e Alto (%). Barras de erro representam IC95.

Nos fluxos baixo e médio, os planos fixos com maior duração de ciclo (T1, T2 e T3) atingiram taxa de esvaziamento próxima ou igual a 100%, enquanto o controlador neural registrou 99,36% e 97,47%, respectivamente, uma diferença pequena, porém estatisticamente significativa na maioria das comparações ($p < 0{,}001$). Essa diferença decorre do fato de que os planos fixos com tempos longos garantem verde suficiente para praticamente qualquer composição de fila nesses volumes, enquanto o controlador neural, ao ajustar o verde ao tempo previsto, assume o risco de subestimar ocasionalmente o tempo necessário.

No fluxo alto, o cenário se inverte parcialmente. O plano T5 registrou a menor taxa de esvaziamento entre todos os controladores (60,23%), evidenciando que, em quase 40% dos ciclos, o verde foi insuficiente para esvaziar a fila. O controlador neural manteve 90,14%, superando T1 (78,84%) e T4 (81,54%) com significância estatística ($p < 0{,}001$). Já em relação a T2 (91,25%) e T6 (87,36%), as diferenças no fluxo alto não atingiram significância estatística em ambos os testes pareados ($p \geq 0{,}05$ para pelo menos um dos testes), configurando um empate estatístico. O plano T3 apresentou a maior taxa de esvaziamento no fluxo alto (95,29%), com diferença estatisticamente significativa em relação ao controlador neural ($p < 0{,}001$), evidenciando que planos de duração intermediária tendem a se adaptar melhor a esse volume do que planos de verde muito longo (T1) ou muito curto (T5).

4.5 Delay Global

O delay global corresponde ao tempo médio de espera por veículo no cruzamento. A Figura 4.6 apresenta os resultados para os três grupos de volume, revelando um quadro mais complexo do que as métricas de eficiência operacional anteriores.

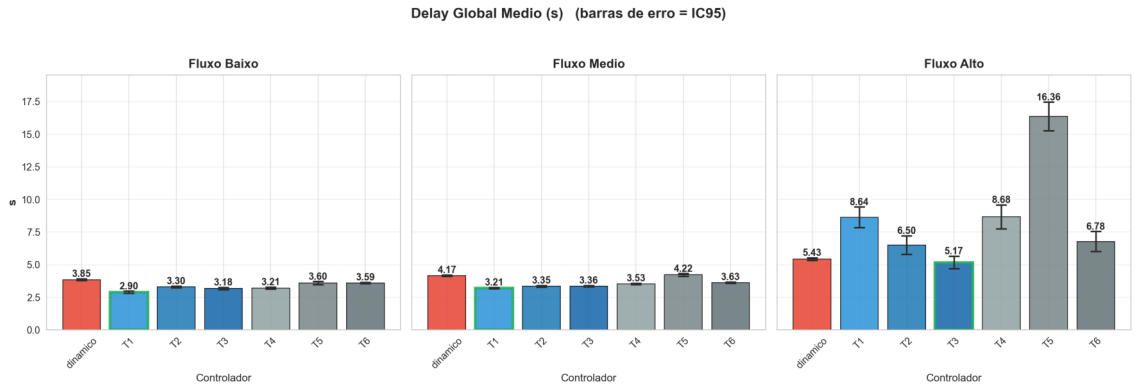


Figura 4.6: Delay global médio por controlador, para os fluxos Baixo, Médio e Alto (s). Barras de erro representam IC95.

Nos fluxos baixo e médio, os planos fixos com ciclo mais longo apresentaram delay global inferior ao do controlador neural, com diferenças estatisticamente significativas na quase totalidade das comparações. No fluxo baixo, o melhor plano fixo (T1) registrou 2,90 segundos, contra 3,85 segundos do controlador neural. No fluxo médio, T1 registrou 3,21 segundos, contra 4,17 segundos do controlador neural. Essa desvantagem do controlador neural nesses volumes está diretamente relacionada à sua estratégia de distribuir o verde de forma mais equilibrada entre as vias, como detalhado na Seção 4.6.

No fluxo alto, o quadro se inverte de forma acentuada. O controlador neural registrou delay global de 5,43 segundos, valor inferior ao de T1 (8,64s), T2 (6,50s), T4 (8,68s), T5 (16,36s) e T6 (6,78s), com significância estatística em todas essas comparações ($p < 0{,}05$). Apenas o plano T3 apresentou delay global levemente menor (5,17 segundos); no entanto, essa diferença específica não atingiu significância estatística em ambos os testes pareados (Wilcoxon $p = 4{,}1 \times 10^{-8}$, mas t pareado $p = 0{,}242$), configurando um resultado estatisticamente inconclusivo entre os dois testes. Dado que o teste de Wilcoxon é adotado como referência neste trabalho por não assumir normalidade – premissa pouco sustentável no fluxo alto, onde a variabilidade entre cenários é elevada, esse caso é reportado como uma comparação que exige cautela na interpretação, e não como uma vitória inequívoca de qualquer um dos dois controladores.

O resultado do plano T5 é o mais expressivo do experimento: com apenas 19 segundos de verde para a Severino e 8 para a Ascendino, o sistema praticamente colapsa no fluxo alto, atingindo delay médio de 16,36 segundos por veículo – quase o triplo do valor obtido pelo controlador neural na mesma condição.

4.6 Delay por Via

A desagregação do delay por via revela o mecanismo por trás dos resultados globais observados na seção anterior. A Figura 4.7 apresenta o delay médio da Av. Pref. Severino Cabral e da Rua Dep. Ascendino Moura separadamente para cada controlador, nos três grupos de volume.

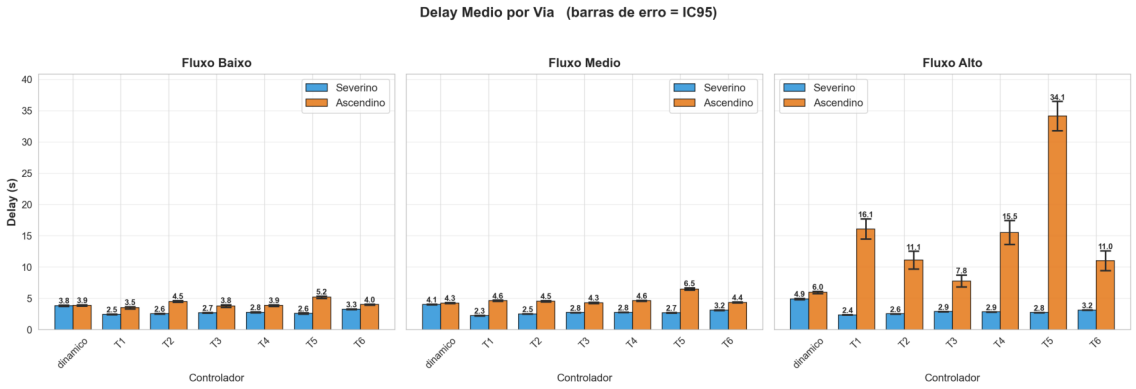


Figura 4.7: Delay médio por via (Severino e Ascendino) e controlador, para os fluxos Baixo, Médio e Alto (s).

O delay na Av. Pref. Severino Cabral foi consistentemente maior no controlador neural do que nos planos fixos, em todos os grupos de volume, com significância estatística em praticamente todas as comparações. No fluxo alto, por exemplo, o controlador neural registrou 4,91 segundos na Severino, enquanto os planos fixos T1 a T5 mantiveram valores entre 2,39 e 2,91 segundos. Essa diferença reflete diretamente o comportamento do controlador neural: ao adaptar o verde da Severino à composição real da fila, o sistema concede menos verde do que os planos fixos concedem por padrão, aumentando o tempo de espera individual dos veículos da via principal.

O efeito oposto, e mais expressivo, é observado na Rua Dep. Ascendino Moura no fluxo alto. O controlador neural registrou delay de apenas 5,98 segundos, enquanto os planos fixos apresentaram valores dramaticamente maiores: T1 com 16,12 segundos, T2 com 11,12 segundos, T4 com 15,54 segundos, T6 com 11,04 segundos e, no caso mais extremo, T5 com 34,14 segundos, quase seis vezes o valor do controlador neural. Todas essas diferenças são estatisticamente significativas ($p < 0{,}05$ em ambos os testes pareados). Esse resultado evidencia o principal problema dos semáforos de ciclo fixo em cenários de alta demanda: ao priorizar rigidamente a via principal com tempos de verde fixos e desproporcionais, esses planos penalizam severamente a via secundária, que acumula filas sem receber verde suficiente para esvaziá-las.

O controlador neural, ao dimensionar o verde de cada fase conforme a de-

manda observada em tempo real, evita esse desequilíbrio estrutural. No fluxo alto, a diferença entre o delay da Severino (4,91s) e o da Ascendino (5,98s) sob o controlador neural é de apenas 1,07 segundo, contra diferenças de até 31,23 segundos entre as vias sob o plano T5. Esse equilíbrio entre vias, mantido de forma consistente pelo controlador neural mesmo sob alta demanda, é um dos resultados centrais deste trabalho: a Ascendino recebe verde proporcional à sua fila, o que reduz substancialmente o delay dessa via sem comprometer desproporcionalmente a Severino.

4.7 Discussão Geral

Os resultados obtidos permitem caracterizar o controlador neural como uma solução com perfil de desempenho distinto dos planos de tempo fixo, com vantagens e limitações específicas a cada métrica e cenário, um perfil sustentado, agora, por significância estatística sobre 150 execuções pareadas por comparação, e não apenas por diferenças pontuais entre médias.

Em eficiência operacional, o controlador neural foi superior em todos os cenários avaliados, com significância estatística consistente: maior throughput e menor desperdício de verde em todos os grupos de volume, em todas as comparações contra os seis planos fixos. Esses resultados confirmam, com rigor estatístico, a hipótese central do trabalho de que adaptar o tempo de verde à composição da fila melhora o aproveitamento do ciclo semafórico.

Em relação ao delay global e ao delay por via, os resultados revelam um trade-off inerente à estratégia adaptativa, já identificado na literatura de sistemas de controle que buscam equilíbrio entre múltiplos objetivos concorrentes. Ao distribuir o verde de forma mais equilibrada entre as vias, o controlador neural aumenta o delay na via principal em volumes baixo e médio, situação em que os planos fixos com tempos longos garantem espera reduzida para a Severino, com significância estatística. No fluxo alto, contudo, essa mesma estratégia se mostra benéfica para o sistema como um todo: o controlador neural evita o colapso da via secundária observado em quase todos os planos fixos, mantendo o delay da Ascendino em níveis operacionais (5,98s) enquanto planos como T1, T4 e, em particular, T5 atingem valores de 16 a 34 segundos nessa via, diferenças estatisticamente significativas em todos os casos. A única exceção parcial é o delay global no fluxo alto frente ao plano T3, cuja diferença em relação ao controlador neural não atingiu significância estatística de forma consistente entre os dois testes aplicados, sendo reportada como estatisticamente inconclusiva.

Em termos de taxa de esvaziamento, o controlador neural apresentou desempenho ligeiramente inferior aos melhores planos fixos nos volumes baixo e médio, onde ciclos de verde longo garantem esvaziamento completo na quase totalidade dos

ciclos. No fluxo alto, o controlador neural superou a maioria dos planos fixos, com o plano T5 registrando taxa de apenas 60,23%, evidenciando que ciclos de verde curto e fixo são particularmente inadequados para cenários de alta demanda; a exceção é o plano T3, que apresentou a maior taxa de esvaziamento nesse grupo de volume (95,29%), com diferença estatisticamente significativa em relação ao controlador neural.

A comparação com modelos supervisionados alternativos, apresentada na Seção 4.1, complementa essa análise sob outra perspectiva: embora a CNN proposta obtenha a maior acurácia na métrica de tolerância operacional ($\pm$3s), o *Random Forest* de regressão obtém o menor erro médio absoluto, a um custo de treinamento ordens de grandeza menor. Isso reforça que a superioridade da CNN não é absoluta em todas as dimensões possíveis de avaliação, mas específica à métrica alinhada ao objetivo de controle, um ponto relevante tanto para a interpretação dos resultados operacionais quanto para decisões futuras de implementação, discutidas no Capítulo 5.

Em síntese, o controlador neural demonstra desempenho superior, estatisticamente significativo e consistente, em cenários de alta demanda, precisamente os cenários em que a rigidez dos semáforos de ciclo fixo gera os maiores prejuízos à mobilidade urbana, e nos quais escolher o plano fixo correto exigiria conhecimento prévio e preciso da demanda esperada. Em cenários de baixa e média demanda, os planos fixos mais longos apresentam menor delay global, porém à custa de desperdício de tempo de verde significativamente maior e de desequilíbrio crescente entre as vias conforme a demanda aumenta. A escolha entre as duas abordagens depende, portanto, do perfil de demanda predominante no cruzamento e dos objetivos de otimização priorizados pelo gestor de tráfego – sendo o controlador neural particularmente indicado para cruzamentos sujeitos a variações abruptas e imprevisíveis de demanda, como o cenário de Campina Grande descrito na Contextualização deste trabalho.

Capítulo 5

Considerações Finais

5.1 Conclusões

Este trabalho propôs o desenvolvimento e a avaliação, em ambiente simulado, de um controlador semafórico adaptativo baseado em aprendizado de máquina. O sistema utiliza uma rede neural convolucional 2D para prever o tempo de esvaziamento da fila de veículos no início de cada fase verde, ajustando dinamicamente a duração do verde conforme a composição real da fila detectada pelos sensores de faixa. A validação foi conduzida integralmente no simulador SUMO, sobre um modelo do cruzamento real entre a Av. Pref. Severino Cabral e a Rua Dep. Ascendino Moura, em Campina Grande, comparando o controlador neural a seis planos de temporização fixa derivados dos dados reais do cruzamento.

As principais contribuições deste trabalho podem ser sintetizadas da seguinte forma:

- Modelagem de um cruzamento real de Campina Grande no simulador SUMO, a partir de dados geográficos do OpenStreetMap e de planos de temporização semafórica reais fornecidos pela STTP.
- Desenvolvimento de uma estratégia de geração automatizada de dataset via interface TraCI, capaz de capturar a composição da fila de veículos e seu respectivo tempo de esvaziamento sem qualquer intervenção manual entre simulações.
- Proposta de uma representação tensorial estruturada da composição da fila, incorporando o tipo e a posição individual de cada veículo, capaz de preservar informação espacial relevante para a predição.
- Projeto de uma arquitetura de rede neural convolucional com branch duplo (local e global), capaz de capturar tanto padrões posicionais quanto a com-

posição agregada da fila, e de uma função de perda tolerante adaptada à natureza ordinal do problema de predição de tempo.

- Integração do modelo treinado ao simulador SUMO como controlador semafórico ativo, substituindo a lógica de ciclo fixo em tempo de simulação.
- Avaliação comparativa sistemática do controlador neural contra seis planos de temporização fixa reais, sob três níveis de densidade de tráfego, utilizando quatro métricas operacionais complementares (throughput, desperdício de verde, taxa de esvaziamento e delay).

Os parágrafos seguintes detalham os principais resultados obtidos a partir dessas contribuições.

É importante destacar que o sistema proposto não opera como um controlador de ciclo semafórico no sentido tradicional. Em vez de decidir quando trocar de fase ou qual fase ativar, o modelo resolve um problema mais específico: dado o estado atual da fila de veículos em uma via, quanto tempo é necessário para que essa fila se esvazie completamente. Essa predição é então utilizada como duração do verde, fazendo com que o semáforo se adapte à demanda real sem necessidade de otimização global do cruzamento.

O modelo demonstrou desempenho preditivo consistente sobre um conjunto de teste reservado integralmente para avaliação final, nunca utilizado durante o treinamento, atingindo acurácia de 84,92% com tolerância de ± 3 segundos, erro médio absoluto de 2,09 segundos e viés de $+0,39$ segundos. A comparação com quatro modelos supervisionados alternativos (Regressão Linear, *Random Forest* de classificação e de regressão, e um Perceptron Multicamadas) mostrou que a CNN obtém a maior acurácia na métrica de tolerância operacional, embora o *Random Forest* de regressão apresente menor erro médio absoluto a um custo de treinamento consideravelmente menor. Esse resultado indica que a vantagem da arquitetura proposta está associada especificamente ao alinhamento entre sua função de perda e a métrica relevante para o controle semafórico, e não a uma superioridade incondicional sobre qualquer critério de avaliação.

Na validação comparativa, conduzida com múltiplas repetições por cenário e testes estatísticos pareados, o controlador neural apresentou desempenho superior em eficiência operacional em todos os cenários avaliados, registrando o maior throughput médio e o menor tempo de verde desperdiçado nos três grupos de volume, com significância estatística em praticamente todas as comparações contra os seis planos fixos. No fluxo alto, o controlador neural também se destacou na proteção da via secundária: enquanto os planos fixos T1, T2, T4, T5 e T6 geraram delays de 11 a 34 segundos na Rua Dep. Ascendino Moura, o controlador neural manteve o

delay dessa via em 5,98 segundos, com diferenças estatisticamente significativas em todos os casos, evidenciando a limitação estrutural dos semáforos de ciclo fixo em cenários de alta demanda.

Por outro lado, os resultados também revelaram limitações do controlador neural em relação ao delay global nos volumes baixo e médio, onde os planos fixos com tempos longos garantiram menor espera média por veículo, com significância estatística. Esse comportamento reflete um trade-off inerente à estratégia adaptativa: ao distribuir o verde de forma mais equilibrada entre as vias, o sistema aumenta o delay na via principal em situações de baixa e média demanda, mas evita o colapso da via secundária em situações de alta demanda. No fluxo alto, apenas a comparação de delay global contra o plano T3 não atingiu significância estatística consistente entre os dois testes aplicados, sendo reportada como estatisticamente inconclusiva.

Diante desses resultados, sustentados por análise estatística sobre múltiplas repetições, considera-se que a hipótese central do trabalho foi confirmada de forma consistente para cenários de alta demanda, e parcialmente confirmada de forma geral: adaptar o tempo de verde à composição real da fila melhora, com significância estatística, a eficiência do uso do ciclo semafórico em todos os cenários e protege a via secundária em cenários de alta demanda, mas não minimiza o delay global nos cenários de baixa e média demanda. O controlador neural representa uma abordagem com perfil de desempenho distinto dos planos fixos, com vantagens estatisticamente robustas em cenários de maior variabilidade de demanda.

5.2 Limitações

Algumas limitações do trabalho devem ser consideradas na interpretação dos resultados.

A validação foi conduzida integralmente em ambiente simulado, sem implantação em via real. O simulador SUMO oferece um ambiente controlado e reprodutível, mas não captura todas as variáveis presentes no tráfego urbano real, como condições climáticas, comportamento irregular de motoristas, pedestres e veículos de emergência.

Ao longo do desenvolvimento deste trabalho, duas limitações metodológicas inicialmente identificadas foram corrigidas. O conjunto de dados passou a contar com um conjunto de validação dedicado, separado do conjunto de teste: a validação orienta os critérios de *early stopping*, a redução da taxa de aprendizado e a seleção do melhor *checkpoint* do modelo, enquanto o conjunto de teste é reservado como *holdout* avaliado uma única vez, ao final do treinamento, conforme detalhado na Seção 3.5. Da mesma forma, as simulações de validação comparativa passaram a utilizar sementes aleatórias controladas, com o volume de veículos por cenário

fixado e repetido sob múltiplas sementes, permitindo tanto a reprodutibilidade dos resultados quanto o cálculo de significância estatística das diferenças observadas, conforme descrito na Seção 3.6.4.

A distribuição de tipos de veículos no treinamento (aproximadamente 33% de cada tipo) diferiu da distribuição adotada nos testes (50% carros, 40% motos, 10% ônibus). Como discutido na Seção 3.6.1, essa diferença foi deliberada: o treinamento prioriza a diversidade de composições, incluindo situações raras no tráfego real, enquanto o teste prioriza o realismo, aproximando-se da proporção real observada na frota de Campina Grande. Ainda assim, essa diferença representa uma fonte potencial de viés, já que o desempenho reportado reflete especificamente a distribuição adotada nos cenários de teste.

O controlador neural não é capaz de pular fases nem alterar a ordem do ciclo semafórico. Vias sem demanda ainda recebem o verde mínimo de 8 segundos antes de ceder passagem, o que representa ineficiência em cenários de demanda muito assimétrica.

Os parâmetros dos detectores de área e a codificação dos tipos de veículos estão definidos de forma estática no código, sem persistência no arquivo de metadados do modelo. Alterações nesses parâmetros exigiriam novo treinamento e atualização manual dos scripts de inferência.

5.3 Ameaças à Validade

Além das limitações já discutidas, é importante categorizar de forma mais sistemática as principais ameaças à validade deste estudo experimental.

Ameaças à validade de conclusão dizem respeito à capacidade de sustentar estatisticamente as diferenças observadas entre o controlador neural e os planos fixos. Essa ameaça foi mitigada ao longo do desenvolvimento deste trabalho: cada cenário passou a ser executado sob três sementes aleatórias distintas, totalizando 150 execuções pareadas por comparação, e as diferenças entre o controlador neural e cada plano fixo foram avaliadas por meio de testes pareados de Wilcoxon e t, conforme descrito na Seção 3.6.4. A quase totalidade das comparações mostrou-se estatisticamente significativa, com poucas exceções pontuais, documentadas no Capítulo 4, em que os dois testes não convergiram quanto à significância. Ainda assim, o número de sementes por cenário (três) representa um ponto de atenção: um número maior de repetições reduziria ainda mais a incerteza associada à estocasticidade interna do simulador, embora a um custo computacional proporcionalmente maior.

Ameaças à validade interna referem-se a fatores que podem ter influenciado os resultados sem relação direta com a variável de interesse. As duas principais

ameaças identificadas inicialmente, o uso do mesmo conjunto de dados para orientar decisões de treinamento e para a avaliação final do modelo, e a ausência de sementes aleatórias controladas nas simulações de validação comparativa, foram mitigadas ao longo do desenvolvimento deste trabalho, conforme descrito na Seção 5.2. Uma ameaça residual permanece quanto ao número de sementes por cenário: com apenas três repetições, a estimativa de variabilidade de cada métrica, embora suficiente para a análise estatística realizada, é menos precisa do que a que seria obtida com um número maior de repetições.

Ameaças à validade de constructo dizem respeito a se as métricas e os dados utilizados realmente capturam o fenômeno de interesse. A diferença entre a distribuição de tipos de veículos utilizada no treinamento e a utilizada nos cenários de teste, discutida na Seção 3.6.1, é uma ameaça relevante: o desempenho reportado reflete uma composição de tráfego específica, e outras composições poderiam levar a resultados distintos. Da mesma forma, a duplicação de amostras por espelhamento de faixas, embora fundamentada na equivalência física das faixas do cruzamento estudado, pressupõe que essa equivalência se mantém em qualquer cenário de aplicação do modelo, o que pode não ser verdade em cruzamentos com faixas exclusivas ou com restrições de movimento diferentes entre faixas.

Ameaças à validade externa limitam a generalização dos resultados obtidos para além do escopo estudado. Todo o desenvolvimento e a validação foram conduzidos em ambiente exclusivamente simulado, no SUMO, sem qualquer implantação em via real, de modo que fatores não representados na simulação – como ruído de sensoriamento, comportamento irregular de motoristas, condições climáticas ou a presença de pedestres e ciclistas – não foram avaliados. Além disso, a validação restringiu-se a um único cruzamento, com geometria, volume de tráfego e composição de veículos específicos de Campina Grande, o que impede concluir se o modelo aprendeu padrões de esvaziamento de fila generalizáveis a outros cruzamentos ou se aprendeu características específicas do cenário estudado. Quanto à escolha de modelo, a comparação com quatro modelos supervisionados alternativos (Regressão Linear, *Random Forest* de classificação e de regressão, e um Perceptron Multicamadas), apresentada na Seção 4.1, permite afirmar que a CNN proposta obtém a maior acurácia especificamente na métrica de tolerância operacional relevante para o controle semafórico, embora não domine em todas as métricas de erro possíveis. Uma ameaça residual permanece: a comparação foi realizada apenas com o cruzamento estudado, e não é possível afirmar que a mesma hierarquia de desempenho entre modelos se manteria em cruzamentos com características geométricas ou de tráfego distintas.

5.4 Trabalhos Futuros

Com base nos resultados obtidos e nas limitações identificadas, algumas direções para trabalhos futuros são sugeridas.

A implantação do controlador em um ambiente de simulação mais complexo, com múltiplas interseções coordenadas, representaria um avanço natural em relação ao escopo deste trabalho, que se restringiu a uma interseção isolada. A extensão para redes de semáforos exigiria mecanismos de coordenação entre os controladores de cada cruzamento.

A unificação das distribuições de veículos entre treinamento e teste eliminaria a fonte de viés residual discutida na Seção 5.2, permitindo avaliar o desempenho do modelo em condições ainda mais consistentes. A adoção de um conjunto de validação dedicado e a comparação com modelos supervisionados alternativos, inicialmente apontadas como limitações deste trabalho, foram incorporadas ao longo do desenvolvimento e encontram-se detalhadas nos Capítulos 3 e 4; trabalhos futuros podem aprofundar essa comparação incluindo técnicas adicionais, como *Gradient Boosting* e redes recorrentes, e avaliando sua sensibilidade a diferentes composições de tráfego.

A implementação de um mecanismo de controle mais flexível, capaz de pular fases ou ajustar a ordem do ciclo conforme a demanda, poderia melhorar o desempenho em cenários de demanda muito assimétrica, superando a limitação do uso do `setPhaseDuration` sobre um `tlLogic` estático.

A incorporação de variáveis adicionais ao tensor de entrada, como a velocidade dos veículos ou o histórico de ciclos anteriores, poderia melhorar a capacidade preditiva do modelo, especialmente no fluxo alto, onde o erro médio absoluto foi mais elevado.

Por fim, a validação do sistema com dados reais de contagem veicular do cruzamento estudado, em substituição às simulações com distribuições estimadas, permitiria avaliar a aderência do modelo a condições de tráfego efetivamente observadas em Campina Grande.

Além das direções de pesquisa apontadas, a migração da solução proposta de um ambiente simulado para uma implantação real em Campina Grande enfrentaria desafios de naturezas distintas. Do ponto de vista de sensoriamento, o sistema depende da identificação do tipo de veículo e de sua posição relativa na fila, informações obtidas nas simulações por meio de detectores de área ideais do SUMO, sem ruído ou falhas de leitura. Em uma implantação real, essa entrada dependeria de sensores como laços indutivos, radares ou câmeras com processamento de visão computacional, sujeitos a ruído, oclusão entre veículos e falhas intermitentes de detecção – fatores que podem degradar a qualidade da composição de fila fornecida ao

modelo e, consequentemente, a precisão de suas predições.

Do ponto de vista computacional, o modelo treinado realiza uma única inferência por transição de fase, o que representa custo computacional baixo o suficiente para execução em tempo real mesmo em hardware modesto, como controladores semafóricos embarcados ou unidades de processamento instaladas no próprio cruzamento. O principal custo computacional do trabalho está concentrado na etapa de treinamento do modelo, que não precisaria ser repetida com frequência em uma implantação operacional, salvo em cenários de recalibração periódica do sistema.

Do ponto de vista de integração, os controladores semafóricos reais operados pela STTP utilizam controladores programáveis próprios, com lógicas e protocolos de comunicação distintos da interface TraCI utilizada neste trabalho para comunicação com o SUMO. A integração real exigiria o desenvolvimento de uma camada intermediária de software capaz de traduzir a predição do modelo em comandos compatíveis com o controlador físico instalado no cruzamento, além de mecanismos de segurança e supervisão – como limites rígidos de tempo mínimo e máximo de verde, e um modo de operação de contingência baseado em tempo fixo – para lidar com eventuais falhas do modelo ou dos sensores sem comprometer a segurança da interseção.

Referências Bibliográficas

[1] INSTITUTO DE PESQUISA ECONÔMICA APLICADA (IPEA). *Mobilidade Urbana no Brasil: desafios e perspectivas*. Comunicado do Ipea 94, Ipea, Brasília, 2011. Disponível em: <https://repositorio.ipea.gov.br>. Acessado em: 07 set. 2025.

[2] ROESS, R. P., PRASSAS, E. S., MCSHANE, W. R. *Traffic Engineering*. 4th ed. Upper Saddle River, NJ, Prentice Hall / Pearson, 2011. ISBN: 978-0-13-613573-9.

[3] STTP CAMPINA GRANDE. "Fluxo de veículos triplica durante o período do Natal em Campina Grande e tem aumento de cerca de 1 milhão no Ano Novo". https://sttp.campinagrande.pb.gov.br/fluxo-de-veiculos-triplica-durante-o-periodo-do-natal-em-campina-grande-e- 2024. Acessado em: 07 set. 2025.

[4] STTP CAMPINA GRANDE. "Sucesso: STTP registra quase 130 mil passagens gratuitas nas catracas dos ônibus durante os primeiros dias d'O Maior São João do Mundo". https://sttp.campinagrande.pb.gov.br/sucesso-sttp-registra-quase-130-mil-passagens-gratuitas-nas-catracas-dos-c 2024. Acessado em: 14 mar. 2026.

[5] ESTADÃO MOBILIDADE. "Gasta muito tempo no trânsito? Conheça tecnologia que economiza 83 horas por ano". https://mobilidade.estadao.com.br/mobilidade-para-que/gasta-muito-tempo-no-transito-conheca-tecnologia-que-economiza-83-horas-po 2023. Acessado em: 07 set. 2025.

[6] LOPEZ, P. A., BEHRISCH, M., BIEKER-WALZ, L., et al. "Microscopic Traffic Simulation using SUMO". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2575–2582. IEEE, 2018. doi: 10.1109/ITSC.2018.8569938.

[7] WEGENER, A., PIÓRKOWSKI, M., RAYA, M., et al. "TraCI: An Interface for Coupling Road Traffic and Network Simulators". In: *Proceedings of*

the 11th Communications and Networking Simulation Symposium (CNS), pp. 155–163. ACM, 2008. doi: 10.1145/1400713.1400740.

[8] DENATRAN, CONTRAN. *Manual Brasileiro de Sinalização de Trânsito — Volume V: Sinalização Semafórica*. Resolução CONTRAN 483, Departamento Nacional de Trânsito / Conselho Nacional de Trânsito, Brasília, 2014. Disponível em: <https://www.gov.br/infraestrutura/pt-br/assuntos/transito/arquivos-senatran/educacao/publicacoes/manual_vol_v_-2.pdf>. Acessado em: 07 set. 2025.

[9] WEI, H., ZHENG, G., GAYAH, V. V., et al. "A Survey on Traffic Signal Control Methods", *arXiv preprint*, v. abs/1904.08117, 2019. Disponível em: <https://arxiv.org/abs/1904.08117>.

[10] DLR – GERMAN AEROSPACE CENTER. "SUMO Documentation". https://sumo.dlr.de/docs/index.html, 2025. Acessado em: 09 set. 2025.

[11] DLR – GERMAN AEROSPACE CENTER. "TraCI – SUMO Traffic Control Interface". https://sumo.dlr.de/docs/TraCI.html, 2025. Acessado em: 09 set. 2025.

[12] BISHOP, C. M. *Pattern Recognition and Machine Learning*. New York, Springer, 2006. ISBN: 978-0-387-31073-2.

[13] GOODFELLOW, I., BENGIO, Y., COURVILLE, A. *Deep Learning*. MIT Press, 2016. Disponível em: <https://www.deeplearningbook.org/>. Acessado em: 07 set. 2025.

[14] KINGMA, D. P., BA, J. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations (ICLR)*, 2015. Disponível em: <https://arxiv.org/abs/1412.6980>.

[15] IBM. "O que é uma rede neural?" https://www.ibm.com/br-pt/think/topics/neural-networks, 2024. Acessado em: 09 set. 2025.

[16] IBM. "Redes Neurais Convolucionais (CNNs)". https://www.ibm.com/br-pt/think/topics/convolutional-neural-networks, 2024. Acessado em: 09 set. 2025.

[17] PYTORCH FOUNDATION. "PyTorch: An open source machine learning framework". https://pytorch.org/, 2025. Acessado em: 07 set. 2025.

[18] PASZKE, A., GROSS, S., MASSA, F., et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, 2019. Disponível em: <https://arxiv.org/abs/1912.01703>.

[19] CAMPOS, R. L. B. L. *Regulagem em Tempo-Real de um Sistema Integrado de Semáforos: Uma Aplicação de Redes Neurais Recorrentes e Controle Adaptativo*. Dissertação (mestrado em gestão do conhecimento e tecnologia da informação), Universidade Católica de Brasília, Brasília, 2006. Disponível em: <https://bdtd.ucb.br:8443/jspui/bitstream/123456789/1484/1/Texto%20Completo.pdf>. Acessado em: 21 mar. 2026.

[20] GENDERS, W., RAZAVI, S. "Using a Deep Reinforcement Learning Agent for Traffic Signal Control", *arXiv preprint*, v. abs/1611.01142, 2016. Disponível em: <https://arxiv.org/abs/1611.01142>.

[21] SIQUEIRA FILHO, L. A. D. "Redes Neurais Aplicadas a Semáforos de Trânsito". Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) — Pontifícia Universidade Católica de Goiás, Goiânia, 2024. Disponível em: <https://repositorio.pucgoias.edu.br/jspui/bitstream/123456789/8632/1/REDES%20NEURAIS%20APLICADAS%20A%20SEM%C3%81FOROS%20DE%20TR%C3%82NSITO.pdf>. Acessado em: 21 mar. 2026.

[22] DETRAN-PB – DEPARTAMENTO ESTADUAL DE TRÂNSITO DA PARAÍBA. "Frota: Município / Tipo de Veículos – Dados até Fevereiro 2026". https://detran.pb.gov.br/institucional-1/estatisticas/veiculos/frota/4-frota-tipo-municipio-totalacumulado.pdf, 2026. Acessado em: 07 jul. 2026.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
Campus Campina Grande - Código INEP: 25137409
R. Tranqüilino Coelho Lemos, 671, Dinamérica, CEP 58432-300, Campina Grande (PB)
CNPJ: 10.783.898/0003-37 - Telefone: (83) 2102.6200

Documento Digitalizado Restrito

Versão Final do TCC

Assunto:	Versão Final do TCC
Assinado por:	Liedson Augusto
Tipo do Documento:	Relatório
Situação:	Finalizado
Nível de Acesso:	Restrito
Hipótese Legal:	Direito Autoral (Art. 24, III, da Lei no 9.610/1998)
Tipo do Conferência:	Cópia Simples

Documento assinado eletronicamente por:

- **Liedson Augusto Maciel Costa, DISCENTE (202111250018) DE BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO - CAMPINA GRANDE**, em 04/08/2026 15:13:43.

Este documento foi armazenado no SUAP em 04/08/2026. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 1937094
Código de Autenticação: 38976c85ba