GUSTAVO HENRIQUE LARSEN

PREDIÇÃO DE TEMPOS DE VIAGEM EM LINHAS DE ÔNIBUS BASEADO EM DADOS DE TRÁFEGO E REDES NEURAIS ARTIFICIAIS

São Paulo

2020

GUSTAVO HENRIQUE LARSEN

Versão Corrigida

PREDIÇÃO DE TEMPOS DE VIAGEM EM LINHAS DE ÔNIBUS BASEADO EM DADOS DE TRÁFEGO E REDES NEURAIS ARTIFICIAIS

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para a obtenção do Título de Mestre em Ciências.

Orientador: Prof. Dr. Leopoldo Rideki Yoshioka

São Paulo

2020

GUSTAVO HENRIQUE LARSEN

PREDIÇÃO DE TEMPOS DE VIAGEM EM LINHAS DE ÔNIBUS BASEADO EM DADOS DE TRÁFEGO E REDES NEURAIS ARTIFICIAIS

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para a obtenção do Título de Mestre em Ciências.

Área de concentração: Sistemas Eletrônicos

Orientador: Prof. Dr. Leopoldo Rideki Yoshioka

São Paulo

2020

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, _____ de _____ de _____

Assinatura do autor: _____

Assinatura do orientador: _____

Catalogação-na-publicação

Larsen, Gustavo

Predição de Tempos de Viagem em Linhas de Ônibus Baseado em Dados de Tráfego e Redes Neurais Artificiais / G. Larsen -- versão corr. -- São Paulo, 2020.

127 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Sistemas Eletrônicos.

1.ENGENHARIA ELÉTRICA 2.APRENDIZADO COMPUTACIONAL 3.SISTEMAS INTELIGENTES DE TRANSPORTES 4.REDES E COMUNICAÇÃO DE DADOS 5.MINERAÇÃO DE DADOS I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos II.t.

Nome: LARSEN, Gustavo Henrique

Título: Predição de Tempos de Viagem em Linhas de Ônibus Baseado em Dados de Tráfego e Redes Neurais Artificiais

Dissertação apresentada à Escola Politécnica, Universidade de São Paulo para obtenção do título de Mestre em Ciências.

Aprovado em:

Banca Examinadora

Prof. Dr. _____

Instituição: _____

Julgamento: _____

Prof. Dr. _____

Instituição: _____

Julgamento: _____

Prof. Dr. _____

Instituição: _____

Julgamento: _____

Dedico este trabalho a Deus: em cumprimento a vontade Dele para a minha vida; e ao propósito que Ele tem ao me fazer terminá-lo. Porque Dele, por Ele e para Ele, são todas as coisas; A Ele seja toda a Honra, o Louvor e a Glória, Eternamente. Amém.

Agradecimentos

Primeiramente, sou grato a Deus por me capacitar a finalizar este trabalho, colocar pessoas e recursos em meu caminho, e a me moldar com todas as dificuldades que passei, sem jamais me deixar desamparado.

À minha esposa Karla por me acompanhar, auxiliar e incentivar durante todo o período do mestrado e por todos os outros períodos difíceis em nossas vidas. Sempre potencializou aquilo que há em mim, e juntos pudemos ir mais longe.

Aos meus pais Orli e Denise, meu irmão Gunnar e toda a minha família por toda a paciência e compreensão pelo período que estive ausente deles enquanto estive estudando, e por toda a educação que me proporcionaram.

Ao Prof. Dr. Leopoldo Yoshioka pela confiança, incentivo e pelas sábias recomendações e indicações, que me guiaram no desenvolvimento deste trabalho, e pelo Prof. Dr. Claudio Marte por compartilhar comigo seus conhecimentos e experiências sobre os temas da área de Transporte.

Ao Dr. Kleber Hodel, por me apresentar ao programa de mestrado da Poli, todos os conselhos e por me instigar o interesse do assunto dessa pesquisa. Ao Prof. Dr. Armando Laganá e Prof. Dr. João Francisco Justo Filho por me direcionar no início do mestrado e por expandir meus horizontes acadêmicos.

À empresa Scipopulis me atender e conceder acesso à plataforma do painel do ônibus. E às empresas SPTrans e Google por disponibilizarem as APIs de forma aberta, e possuírem uma boa documentação na forma de utilizá-las, o qual foi imprescindível para a execução e resultados desse trabalho.

À Universidade de São Paulo e a Escola Politécnica pelo excelente curso que tive o privilégio de cursar, por todo o corpo docente e pelos recursos oferecidos que foram muito valiosos. À FAPESP, CNPq e CAPES pelos recursos que viabilizam o Programa de Pós-Graduação da USP. À Mercedes-Benz do Brasil que proporcionou um ambiente criativo e flexível, onde pude compartilhar minhas ideias, e me trouxe condições para realização do Mestrado na Poli USP.

“Pois será como a árvore plantada junto a ribeiros de águas, a qual dá o seu fruto no seu tempo; as suas folhas não cairão, e tudo quanto fizer prosperará.”

Salmos 1:3 (ACF)

Resumo

O conceito de cidades inteligentes é uma tendência nas grandes cidades. Sistemas Inteligentes de Transporte desempenham um papel essencial no fornecimento de informações que possibilitam a previsão de tempos de viagem de ônibus. Informações precisas sobre tempos de viagem ajuda no planejamento dos passageiros e da agência responsável pelo transporte público. O objetivo deste trabalho é propor uma nova metodologia de previsão de tempos de viagem dos ônibus com base em dados abertos coletados em tempo real. A metodologia apresenta um processo para realizar predições precisas de tempos de viagem de ônibus, combinando um método de previsão estatística, um método de aprendizagem de máquina, e em conjunto com dados coletados em tempo real. Será apresentado todas as etapas do processo, incluindo a coleta de vários tipos de dados, armazenamento, análise do banco de dados, desenvolvimento e implementação das técnicas de aprendizado de máquina. Um banco de dados (*dataset*) foi construído a partir da coleta dos dados de geolocalização da frota de ônibus da cidade de São Paulo, dados de tráfego em tempo real, previsão de tráfego do Google Maps, dados meteorológicos e outros dados históricos. A seguir, treinamos uma Rede Neural Artificial (RNA). No processo de treinamento da RNA, alternamos o conjunto de dados e seus hiperparâmetros para descobrir a combinação que forneceu o menor erro de previsão. O erro médio percentual absoluto obtido foi de 9,10%, refletindo em uma raiz do erro quadrático médio de 297 segundos em uma linha que possui um tempo médio de viagem de 35 minutos. Esta pesquisa demonstrou que o método proposto forneceu uma previsão mais precisa do tempo de viagem de ônibus do que os métodos anteriores, a partir de dados da coletados em tempo real pela *web*.

Palavras-chave: Transporte Inteligente, AVL, Mineração de Dados, Aprendizado de Máquina.

Abstract

The concept of smart cities is a trend in big cities. Intelligent Transport Systems plays an essential role in providing information that enables bus travel times prediction. Accurate travel times information improves the planning of the passengers and the agency responsible for public transport. The objective of this work is to propose a new methodology for buses travel times prediction based on open data collected in real time. The methodology presents a process for predicting accurate bus travel times, combining a statistical forecasting method, a machine learning method, along with real time data collected. All steps of the process will be presented, including the collect process for many different types of data, storage, database analysis, development and implementation of machine learning techniques. A dataset was built by collecting the geolocation of the bus fleet in the city of São Paulo, real-time traffic data, traffic forecast from Google Maps, meteorological data and other historical data. Finally, we train an Artificial Neural Network (ANN). In the ANN training process, we alternate the dataset and its hyperparameters to find the combination that provided the most accurate prediction. The mean absolute percentage error obtained was 9.10%, reflecting a root mean square error of 297 seconds on a bus line that has an average travel time of 35 minutes. This research demonstrated that the proposed method provided a prediction of bus travel time more accurate than previous methods, based on data collected in real time over the web.

Keywords: Intelligent Transportation System, AVL, Data Mining, Machine Learning.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AFC	Automatic Fare Counting
ANTP	Associação Nacional de Transportes Públicos
APC	Automatic Passenger Counting
API	Application Programming Interface
APTS	Advanced Public Transport System
ARIMA	AutoRegressive Integrated Moving Average
ATIS	Advanced Traveller Information System
ATMS	Advanced Traffic Management System
AVL	Automatic Vehicle Location
AWS	Amazon Web services
CCP	Coeficiente de Correlação de Pearson
CSV	Comma Separated Values
DBN	Deep Belief Network
EMS	Emergency Management System
GCP	Google Cloud Platform
GPS	Global Positioning System
GPX	GPS eXchange Format
GPU	Graphics Processing Unit
GTFS	General Transit Feed Specification
HTTP	Hypertext Transfer Protocol
IA	Inteligência Artificial
INMET	Instituto Nacional de Meteorologia
IoT	Internet of Things
ITS	Intelligent Transportation System
JSON	JavaScript Object Notation
KML	Keyhole Markup Language
KNN	K-Nearest Neighbor
LSTM	Long Short-Term Memory
MaaS	Mobility as a Service
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error

MSE	Mean Squared Error
OSO	Ordem de Serviço de Operação
PaaS	Platform as a Service
POI	Point of Interest
RDP	Remote Desktop Protocol
REST	Representation State Transfer
RFID	Radio-Frequency Identification
RMSE	Root Mean Square Error
RNA	Rede Neural Artificial
RVM	Relevance Vector Machine
SGD	Stochastic Gradient Descent
SIG	Sistema de Informação Geográfica
SOAP	Simple Object Access Protocol
SPTrans	São Paulo Transporte
SVM	Support Vector Machine
TSP	Transit Signal Priority
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTC	Universal Time Coordinated
VBA	Visual Basic for Applications
VM	Virtual Machine
WSDL	Web Services Description Language
XML	eXtensible Markup Language
XML-RPC	XML-Remote Procedure Call

Sumário

1 INTRODUÇÃO

1.1 Contextualização

Está em curso uma revolução das sociedades conectadas decorrente do avanço das tecnologias digitais. As cidades inteligentes (conhecido também como *smart cities*) são uma tendência mundial, que promove a integração de diferentes tecnologias para melhorar a qualidade de vida da população e a eficiência das cidades de forma sustentável. Pecar e Papa definiram cidade inteligente como sendo uma cidade conectada cuja infraestrutura se relaciona entre si (PECAR; PAPA, 2017). A integração da infraestrutura física, redes de energia e comunicação, atividades econômicas e sociais, e a mobilidade urbana buscam proporcionar um impacto positivo à vida do cidadão. Esses avanços tecnológicos começam a impactar também os meios que as pessoas utilizam para se locomover, através de sistemas de transporte inteligente (MOHANTY; CHOPPALI; KOUGIANOS, 2016).

Uma das novas formas preconizadas para oferecer um transporte inteligente, é por meio do conceito de mobilidade como um serviço - *Mobility as a Service (MaaS*). O MaaS consiste num novo modelo de negócio, onde o usuário pode solicitar o serviço de mobilidade sob demanda (quando precisar) por meio de aplicativos. Trata-se de um conceito bastante amplo, englobando desde opções de rotas, diferentes modais, facilidade de pagamentos de tarifas de transporte, solicitações de serviços de taxi, aluguel de veículos, compartilhamento de veículos (incluindo bicicletas e patinetes elétricos), *car-pooling*, comercialização de dados de mobilidade, ou até um sistema de mobilidade que integra todas as opções com as diferentes rotas, tempos de viagem e tarifas, (LI; VOEGE, 2017). Muitos desses serviços de mobilidade utilizam técnicas emergentes como *web data mining*, *crowdsourcing*, *Big Data e cloud computing*, para tornar os serviços mais eficazes e escaláveis.

Web data mining é um processo utilizado para extrair dados de um servidor que agrega os dados brutos. O servidor que agrega os dados pode ser tanto um banco de dados, que armazena dados históricos, ou pode apenas retornar os valores atuais. Portanto, o processo de *web data mining* pode ocorrer apenas uma única vez para coletar todo um período de dados, ou pode ser cíclico, conforme periodicidade de atualização do servidor que agrega os dados. Sendo que para as coletas cíclicas, geralmente é necessário criar um servidor execute um programa para realizar a tarefa de forma automática (MUGHAL, 2018).

O *crowdsourcing* é uma forma de obtenção de dados por meio de um processo colaborativo, onde os usuários de dispositivos móveis aceitam compartilhar os dados de localização e outras informações (LIU et al., 2013a). Trata-se de uma ferramenta que tem se mostrado eficaz e aplicado cada vez mais para fornecer *insights* para resolução de problemas de mobilidade. Cria-se assim, a possibilidade de que dezenas de milhões de dispositivos móveis se tornem sensores mineradores de dados, contribuindo para alimentar o banco de dados do provedor do aplicativo. Reunindo as diferentes informações geradas por cada um dos usuários, pode-se construir um grande conjunto de dados, que pode ser considerado *Big Data*.

O *Big Data* é um conjunto de dados com uma dimensão grande demais para serem tratados por sistemas tradicionais. O *Big Data* pode integrar diferentes bancos em si (estruturados e não estruturados), e na sua forma inicial pode possuir alguns dados com valores nulos ou errados, por conta do grande volume de dados, velocidade, variedade, veracidade e valor (5Vs). Ignorando ou tratando os dados errôneos é possível traçar uma linha de tendência de uma determinada variável estudada em relação ao tempo ou a outros fatores dentro do *Big Data*. Com um grande conjunto de dados de mobilidade é possível compreender o comportamento das viagens dos usuários, que servirá como base para predizer, por exemplo, o tempo de viagem dos próximos usuários que percorrerão um trajeto similar. A acurácia das estimativas dependerá da qualidade dos dados utilizados e da capacidade de previsão dos modelos matemáticos utilizados, (HE; CAO; LIU, 2015).

Um exemplo de aplicativo que utiliza técnicas de *crowdsourcing* e *Big Data* é o aplicativo Waze, que é bastante disseminado entre os usuários motoristas no Brasil. O Waze coleta informações de localização e velocidade dos veículos por meio dos dados fornecidos pelo GPS do dispositivo móvel, também solicita aos usuários informações sobre o trânsito, estado da via, e ocorrências de acidentes ou veículos com problemas mecânicos. Com o uso desse tipo de ferramenta torna-se possível saber a situação do trânsito em tempo real, incluindo ocorrências de eventos na via. A ideia desta pesquisa é fazer uso dos dados que podem ser obtidos por meio de buscas na *web,* e realizar a previsão do tempo de viagem de um ônibus a partir dos dados de posicionamento em tempo real, dados históricos de viagens, previsão de tráfego, dados históricos de trânsito, dados meteorológicos, pesquisa origem destino entre outras.

Não somente aplicativos de trânsito, como mencionado anteriormente, a demanda por aplicativos de mobilidade urbana tem crescido muito ultimamente em todos os modais. Estas incluem aplicativos para a micromobilidade (bicicletas e patinetes elétricos), transporte

individual (motos e automóveis) e coletivos (ônibus, metrô e trem). Os aplicativos de maior preferência dos usuários são aqueles mais confiáveis, com informações mais precisas, sugestão de melhores rotas para evitar o trânsito, ou opções de trajetos e modais para os usuários, além de fornecer informações de tarifa e horário de partida e estimativa de tempo de viagem. Por exemplo, os aplicativos Moovit e Citymapper coletam informações de diversos servidores e recebe dados dos usuários para entregar dados mais precisos e completos sobre as opções de rotas, preço, tempo e acomodação em vagões ou ocupação dos ônibus para os próximos usuários.

Uma das transformações que se observa é a crescente integração entre os diferentes modais de transporte conforme apontado por um estudo recente sobre a mobilidade publicado pelo Metrô de São Paulo, na pesquisa Origem Destino de 2017 (METRÔ SÃO PAULO, 2019). Um exemplo foi o salto de mais de 400% nas viagens de táxi, em São Paulo, nos últimos dez anos, por conta de surgimento de novos serviços de mobilidade por aplicativos, enquanto o transporte individual aumentou em 9,2% e o transporte público por ônibus caiu 8,1%. Dentre os principais motivos apontados dessa queda, identifica-se a baixa qualidade do serviço do ônibus (lotação, segurança, tempo de espera no ponto e informação ao usuário sobre horário de embarque e desembarque).

Com relação ao transporte público por ônibus, um dos principais problemas enfrentados por usuários é a imprevisibilidade do tempo de chegada do próximo ônibus. O tempo de chegada é uma informação que afeta ao passageiro que está esperando no ponto de ônibus (por consequência na vulnerabilidade e segurança do passageiro), o planejamento do serviço e dos usuários, e a regularidade da linha (afeta o carregamento do veículo), (MORI et al., 2015). A imprevisibilidade pode ser agravada ainda mais quando o ônibus disputa espaço na via com ônibus de outras linhas e com automóveis (para linhas que não utilizam faixa preferenciais ou corredor de ônibus), tornando susceptível aos efeitos de congestionamentos em horários de picos. Esses são alguns dos motivos que dificultam a aplicação das tecnologias nos serviços de mobilidade para o transporte público por ônibus, (HENSHER, 2017).

1.2 Motivação

A disponibilização das informações dos ônibus aos usuários aumenta o conforto e a segurança dos passageiros que vão embarcar no ônibus. O passageiro consegue planejar o horário para se deslocar até o ponto de ônibus, minimizando o tempo de espera no ponto de

ônibus, reduzindo a vulnerabilidade às ameaças e assaltos (LU et al., 2018). Os horários de partida e chegada mais precisos fazem com que o usuário passe a confiar mais no sistema de transporte coletivo, pois não precisará chegar no ponto de ônibus com uma margem muito grande de antecedência, nem precisará ficar esperando um tempo desnecessário, e o passageiro poderá saber a hora que ele chegará ao destino dele com mais certeza. O impacto da informação sobre a previsibilidade do ônibus para os passageiros será aprofundado no item 2.2.3 desta dissertação.

Os tempos de viagem de um ônibus podem variar consideravelmente de um ônibus para outro, mesmo operando na mesma linha. Se a operação do ônibus não possuir algum tipo de controle, pode gerar uma irregularidade na frequência em que os ônibus passam em um determinado ponto de ônibus. Isso faz com que os passageiros passem a esperar mais no ponto de ônibus, e quando o ônibus chega ao ponto, em breve um outro ônibus chega praticamente vazio. Esse fenômeno é chamado de comboio (*bunching*). As previsões de tempos de viagem precisas podem contribuir com os métodos de controle da operação do ônibus, prevenindo a ocorrência desse tipo de irregularidade.

O aumento da previsibilidade do ônibus pode trazer grandes benefícios à sociedade. Além dos benefícios diretos (ao usuário e ao operador do transporte público), existem os benefícios indiretos. Como por exemplo, com o aumento da qualidade do serviço de transporte público, o ônibus tende a atrair mais usuários do transporte individual, reduzindo o volume de tráfego, contribuindo para a diminuição de congestionamentos e emissões de poluentes. E isso faz com que os tempos de viagem sejam menores, mais uniforme, afetando positivamente na saúde da população, por conta da redução de poluição, (CHEN et al., 2017), sendo que, os usuários que migrarem do transporte individual para o coletivo, terão um benefício na saúde em função do deslocamento à pé até o ponto de ônibus (FERRIS; WATKINS; BORNING, 2010).

Portanto, a disponibilização de informações de tempos de viagem precisas é algo de grande relevância que contribui para a melhoria da mobilidade. Possui o potencial de causar impactos socioeconômicos positivos para a sociedade, dentro do contexto de MaaS e cidades inteligentes.

1.3 Objetivos

O objetivo dessa pesquisa é desenvolver uma nova metodologia de previsão dos tempos de chegada dos ônibus nos pontos de parada. A metodologia proposta baseia-se na combinação de métodos estatísticos e de aprendizado de máquina, e busca melhorar a capacidade de previsão a partir dados de geolocalização dos ônibus, estimativa das condições de trânsito em tempo real, e outros dados históricos.

Os objetivos específicos desta pesquisa são os seguintes:

- Implementar uma máquina virtual para computação na nuvem, de forma a garantir que o processo computacional ocorra sem interrupção durante 24 horas por dia, por 30 dias consecutivos;
- Coletar e tratar dados de geolocalização dos ônibus, dados de tráfego em tempo real e previsão de tráfego;
- Desenvolver um modelo de previsão de tempos de chegada dos ônibus baseado em redes neurais artificiais e previsão ingênua (Naïve);
- Realizar experimentos computacionais a fim de escolher a melhor configuração de hiperparâmetros das redes neurais artificiais e determinar o conjunto de dados de entrada mais relevantes para o treinamento da rede neural artificial; e
- Comparar resultados de previsão obtidos pelos métodos propostos nesta pesquisa com os métodos propostos por outros pesquisadores.

1.4 Estrutura do Trabalho

Os demais capítulos desta dissertação estão organizados da seguinte forma:

- Capitulo 2: apresenta o estado da arte com relação aos principais temas desse trabalho, ou seja, a coleta e monitoramento das geolocalizações dos ônibus e dados de tráfego, os principais temas sobre sistemas transporte inteligente, os métodos de aprendizado de máquina e estatísticos para previsão, e por fim, os trabalhos que foram usados como referência para essa pesquisa.
- Capítulo 3: descreve a metodologia proposta nesta pesquisa.
- Capítulo 4: apresenta a aplicação da metodologia proposta em uma linha de ônibus de São Paulo, trata da interpretação dos resultados obtidos e uma

exploração dos hiperparâmetros da RNA, e das correlações entre os dados de entrada.

- Capítulo 5: apresenta as conclusões desta pesquisa e apresenta propostas de trabalhos futuros.
- Por fim, apresentamos as referências bibliográficas utilizadas nesta pesquisa.
- Os APÊNDICES A, B, C e D: apresentam os códigos elaborados nesta pesquisa para a coleta de dados e para a realização da parte experimental desta pesquisa.

2 Embasamento Teórico e Estado da Arte

2.1 Apresentação do Capítulo

Este capítulo abrange os principais temas abordados neste trabalho, como sistemas transporte inteligente, métodos de aprendizado de máquina e estatísticos para previsão. Esta seção inclui a forma de coleta e monitoramento das geolocalizações dos ônibus e dados de tráfego e suas documentações. E por fim, os principais trabalhos que nortearam esta pesquisa.

2.2 Sistema de Transporte Inteligente

2.2.1 Arquitetura

Nas últimas décadas os Sistemas de Transporte Inteligente (ITS) vêm desempenhando um papel importante para melhorar de forma eficiente o desempenho dos sistemas de transporte, aumentar a segurança e proporcionar mais alternativas de locomoção para os viajantes. Trata-se de uma solução que integra a tecnologia eletrônica embarcada nos veículos, sistemas computacionais, redes de comunicação e algoritmos baseados em associação a modelos matemáticos, (WILLIAMS, 2008).

Um avanço significativo do ITS nos anos recentes tem sido o fato de uma quantidade muito grande de dados poderem ser processados de diferentes formas, por diferentes atores envolvidos, incluindo não somente as agências de trânsito e de transporte, mas também os operadores de transporte, as empresas de ofertas de novos serviços de transporte e plataformas de mobilidade (ANTP, 2015). A disponibilidade de uma quantidade cada vez maior de dados está ampliando as possibilidades de desenvolvimento de ITS, passando de um sistema orientado às tecnologias para um sistema orientado aos dados de fontes múltiplas. Diante desse novo cenário, o emprego de técnicas de *data analytics* e *Big Data* estão se popularizando. Para encontrar os padrões de comportamento ou para realizar previsões a partir de grande massa de dados os algoritmos de aprendizagem de máquinas estão se tornando cada vez mais relevantes.

O ITS é subdividido em quatro principais segmentos que incluem: Sistema Avançado de Informação ao Usuário (ATIS - *Advanced Traveller Information System*), Sistema Avançado de Gerenciamento de Tráfego (ATMS - *Advanced Traffic Management System*), Sistema

Avançado de Transporte Público (APTS - *Advanced Public Transport System*), e Sistema de Gerenciamento de Emergência (EMS - *Emergency Management System*), (MANDHARE; KHARAT; PATIL, 2018).

- ATIS é utilizado para fornecer informações aos viajantes, antes e durante o deslocamento, como por exemplo, informações sobre o tráfego, rodízio, obras na rodovia, preço das tarifas, período de operação do transporte público, horários de partida e chegada, ocupação, desvios de rota e alterações nas linhas;
- ATMS é utilizado pelo departamento de tráfego para gerenciar, monitorar o fluxo e regular o tráfego de todos os veículos, usando informações em tempo real para interferir e ajustar controladores viários;
- APTS é utilizado para aumentar a eficiência operacional de todo o transporte público e melhorar as condições, tornando o sistema de transporte mais confiável e de melhor qualidade;
- EMS é utilizado para planejar e fornecer ajuda em condições de emergência.

Dentre os quatro sistemas citados acima, apenas os sistemas ATIS e APTS serão explorados nesta pesquisa, pois o foco do estudo está voltado aos segmentos do ITS voltados ao transporte público e aos passageiros que o utilizam.

Para possibilitar o emprego do ATIS e o APTS no transporte público, é necessário planejar e documentar as linhas de ônibus, descrevendo os itinerários, e os horários de partida, os dias e horários de operação das linhas de ônibus. Para servir de base para o monitoramento dos ônibus, e confrontar o planejado com a situação em tempo real, esse planejamento é documentado através da Ordem de Serviço de Operação (OSO) (SMT E SPTRANS, 2018).

A OSO está disponível apenas para a agência e a empresa contratada para prestar o serviço público. Como uma das propostas do ATIS é disponibilizar as informações ao usuário, é necessário disponibilizar as informações da OSO de forma aberta e pública. A forma mais comum e padronizada, adotado por pelo menos 2500 agências de transporte em mais de 50 países (TRANSITLAND, 2019), é por meio da Especificação Geral sobre *Feeds* de Transporte Público (GTFS), que descreveremos nos próximos itens.

2.2.2 Especificação Geral sobre *Feeds* de Transporte Público

A Especificação Geral sobre *Feeds* de Transporte Público (GTFS) é um padrão aberto que define um formato comum para especificar a programação de horários do transporte público, além de informações geográficas associadas ao itinerário. Trata-se de um conjunto de arquivos no formato de texto (extensão “csv” ou “txt”), contendo o itinerário, rotas (sequência de coordenadas geográficas), datas e horários de funcionamento, frequência, paradas e tarifas associadas do transporte público de uma determinada agência de transporte, (FAYYAZ S.; LIU; ZHANG, 2017).

O GTFS descreve de maneira precisa as informações sobre as linhas e a programação horária, bem como os atributos geográficos e econômicos associados. Por ser documentado em forma de texto, as informações contidas nele não são visuais (mapa com todas as informações), nem podem ser analisadas separadamente, uma vez que os arquivos-texto estruturados estão interligados entre si.

Como o GTFS é aberto ao público, permite que pesquisadores e desenvolvedores o utilizem como fonte de dados para analisar o transporte público ou para construir aplicativos de mobilidade urbana. Os aplicativos podem possuir funcionalidades que simplifiquem a visualização das informações pelos usuários por meio de mapas e ícones. Outra possibilidade no uso do GTFS é automatizar a atualização, sempre que houver modificações em alguma linha, os usuários podem ser notificados quando consultarem o aplicativo, pois as informações de seu itinerário estarão atualizados (ANTRIM; BARBEAU, 2013).

Por conta que o GTFS foi criado pela Google em parceria com a agência de trânsito TriMet, (GOLDSTEIN; DYSON, 2013), a documentação completa do GTFS pode ser consultada no (GOOGLE, 2019a). O GTFS é composto pelos seguintes arquivos:

Tabela 1 – Descrição do Conteúdo de Arquivos que Compõe o GTFS

Nome do Arquivo	**Conteúdo**
Agency	Contém informações sobre as agências prestadoras do transporte público, URL da agência, versão do GTFS, fuso horário e idioma do arquivo.
Calendar	Possui códigos e significados para os dias da semana, dias úteis e fim de semana, em que o serviço é prestado, bem como o período de vigência (início e término).
Fare_atributes	Composto das informações sobre os possíveis custos para cada modal de transporte público oferecido pela agência, bem como, o custo para a utilização de mais de um modal em uma viagem e tempos limites para baldeações e transferências de modais.
Fare_rules	Contém informações sobre o modal de transporte público para cada identificador de rota e regras de tarifas a serem aplicadas.
Frequencies	Contém a informação do headway para cada horário de cada linha.
Routes	Para cada identificador de linha possui as informações de qual agência presta o serviço, qual o código e nome da linha, cor da linha e cor do texto da linha.
Shapes	É a descrição da forma da linha representada pelos pontos de coordenadas geográficas, informando o identificador do *shape*, identificador da linha e a sequência dos pontos com a respectiva latitude e longitude.
Stops_times	Para cada um dos horários de partida de cada linha, descreve o horário de chegada em cada identificador de ponto de ônibus.
Stops	Para cada identificador de ponto de ônibus, descreve o nome do ponto, referência, e latitude e longitude do ponto.
Trips	Para cada identificador de linha, descreve os códigos de dia da semana de operação, o letreiro escrito no ônibus, direção de ida ou volta e identificador da forma da linha.

Fonte: Traduzido e adaptado de (GOOGLE, 2019a)

Apesar da proposta do GTFS ser dinâmica quanto ao planejamento do transporte público, outras informações são relevantes para os usuários, como por exemplo: a posição, a hora prevista de embarque e desembarque e a ocupação do ônibus que o passageiro está

aguardando, pois essas informações podem diminuir o tempo de espera e aumentar a segurança dos passageiros, conforme (BARBEAU, 2018). Portanto um sistema de informação mais detalhado é necessário para aumentar a qualidade do transporte público por ônibus.

2.2.3 Sistema Avançado de Informação ao Viajante

O sistema avançado de informação ao usuário (ATIS) faz parte de uma das áreas do ITS, e segundo a ANTP (2012), pode ser desdobrado em três classes: 1) informações antes do deslocamento; 2) informações durante o deslocamento; e 3) serviços pessoais de informação.

As informações antes do deslocamento podem ser de vários tipos: programação diária das linhas de uma determinada agência de transporte (GTFS), previsão de despacho dos ônibus, localização dos veículos de transporte público (AVL), condições da rodovia e regras de trânsito predominante (rodízio de veículos). As informações durante o deslocamento incluem as estimativas de horário de chegada em cada ponto de ônibus ou terminal final e a localização atual do veículo.

O sistema automático de localização (AVL) fornece as posições dos veículos da frota para uma central de monitoramento remoto. O AVL funciona a partir de um receptor de posicionamento global (GPS), que recebe os sinais de uma constelação de satélites e calcula a localização por meio de um processo de triangulação. O módulo eletrônico embarcado no veículo, denominado de plataforma telemática veicular, recebe os dados de posicionamento e outras informações do veículo, tais como velocidade, rotação do motor, nível de ocupação, estado das portas entre outras. Os dados coletados são enviados para uma central de monitoramento via rede de telefonia celular 3G ou 4G, (HICKMAN, 2004). Após a coleta dos dados de cada AVL, o provedor do transporte público disponibiliza esses dados centralizados em um servidor, o qual torna possível a consulta da localização dos ônibus nos smartphones dos passageiros.

Porém a simples disponibilização das informações de localização dos AVL aos passageiros pode não ser úteis o suficiente, pois o passageiro terá apenas uma ideia do horário que o ônibus irá passar no ponto e ele terá que realizar várias consultas para verificar o andamento do ônibus. Portanto é necessário fornecer previsões de tempo de chegada, pois conforme indicado por (GOOZE; WATKINS; BORNING, 2013), a percepção do usuário sobre a qualidade do serviço aumenta conforme a precisão da informação recebida (localização correta do ônibus, previsão de chegada correta, frequência de atualização da informação).

WATKINS et al.(2011) verificou que a disponibilidade de informações precisas sobre o tempo de viagem dos ônibus contribui para a segurança dos usuários, aumenta a confiabilidade do transporte público e ainda ajuda na tomada de decisão dos usuários ao optar pelo serviço de mobilidade escolhido e o melhor horário.

Por conta que o ATIS tem a possibilidade de interferir na decisão dos passageiros, isto pode interferir positivamente na operação dos ônibus, conforme relatado por WANG et al.(2018), em que o autor propõe disponibilizar a informação de carregamento dos ônibus para os usuários. Essa informação aliada à informação de tempo de chegada dos ônibus auxiliaria na tomada de decisão dos passageiros sobre embarcar no ônibus que está chegando lotado ou no próximo ônibus com mais lugares disponíveis. Conforme os resultados apresentados pelos autores, a operação de ônibus conseguiu se autorregular, evitando a formação de comboio sem precisar intervir na operação do ônibus. O que ocorreu nesse caso foi que a maioria das pessoas passaram a optar por embarcar no próximo ônibus que estava mais vazio e menos atrasado. Permitindo assim, que o ônibus mais cheio e atrasado perdesse menos tempo nos pontos de ônibus.

As informações em tempo real dos AVL podem beneficiar as próprias agências de transporte público. Pois com essas informações é possível monitorar as linhas e tomar medidas de regulação e fazer com que o intervalo entre as passagens nos pontos de ônibus seja mais constante (*headway*), melhorando assim o serviço prestado. Para que isso seja executado, é necessário utilizar os conceitos dos Sistemas de Transporte Público Avançado (APTS), que será visto no item a seguir.

2.2.4 Sistemas de Transporte Público Avançado

O sistema avançado de transporte público (APTS) trata sobre gestão de frota, bilhetagem eletrônica, priorização semafórica, alteração de rotas, transferência intermodal e métodos de controle da linha. O principal objetivo do APTS é melhorar a situação atual do transporte público, reagendando e gerenciando adequadamente as frotas disponíveis, para incentivar os passageiros a usar mais o transporte público em comparação com os veículos particulares, (SINGH; GUPTA, 2015).

Uma das necessidades de se utilizar o APTS no transporte público por ônibus é para regularizar a distância temporal os ônibus na mesma linha (*headway*). Trata-se de um fator que

afeta diretamente o tempo de espera do passageiro no ponto de parada, e consequentemente a percepção de qualidade do serviço.

A irregularidade do *headway* é um efeito que normalmente ocorre nas linhas de ônibus. Mesmo que a operação se inicie com um *headway* uniforme, com o passar do tempo o espaçamento entre os ônibus tornam-se desbalanceadas, com veículos muito próximos ou muito afastados uns dos outros. A irregularidade do *headway* pode ser iniciada de diversas formas dentre as quais é destacado a seguir:

- Motoristas com comportamentos diferentes;
- Características automobilísticas diferentes nos ônibus;
- Situação do trânsito alterada de um veículo para outro;
- Quantidade pontos a parar para embarque ou desembarque;
- Quantidade de tempo a ficar parado em cada ponto;
- Situação semafórica ao ônibus se aproximar do semáforo; e
- Situação climática.

HICKMAN (2001) afirma que se a irregularidade do *headway* não for controlada ou corrigida, a tendência é que se intensifique cada vez mais. Pois, o ônibus com *headway* maior começa a transportar mais passageiros que se acumulam nos pontos, enquanto que o ônibus com menor *headway* tende a se aproximar cada vez do veículo anterior, pois haverá menor quantidade de passageiros. Como resultado, pode-se ocasionar o efeito comboio (*bunching*).

Existe diversas formas de se controlar o headway a fim de impedir o efeito comboio, aumentar a regularidade dos ônibus e para diminuir o tempo de espera dos passageiros nos pontos de ônibus. A seguir será apresentado a principais manobras de regulação:

- **Controle de velocidade (*Speed Control*):** Alerta o motorista sobre o atraso/adiantamento para dirigir mais rápido/devagar, podendo ser utilizado em corredores ou faixas exclusivas, onde não há tanta interferência de outros veículos ou outras linhas de ônibus.(ZHANG; LO, 2018)
- ***Short Turning*:** Realiza o desembarque total antes do ponto final para disponibilizar mais cedo o ônibus, normalmente utilizado quando há uma alta demanda no sentido contrário nos horários de pico. (ULUSOY; CHIEN; WEI, 2010), (CHEN et al., 2015)
- **Despacho (*Dispatching*):** Antecipa/adianta horário de despacho do ônibus no ponto de partida. (BERREBI; WATKINS; LAVAL, 2015)

- **Embarque ou desembarque limitado (*Limited Board and Alight*):** Restringe quantidade de embarque/desembarque ou não permite um deles. (DELGADO; MUNOZ; GIESEN, 2012)
- **Inserir veículo (*Extra Bus*):** Insere um ônibus reserva a partir de qualquer ponto na linha, tenta-se evitar essa opção pelo alto custo operacional, uma vez que outras medidas podem ser utilizadas se adotadas previamente. (YU et al., 2015)
- **Meia viagem (*Deadheading*):** o ônibus atende apenas um sentido da linha, sendo utilizado quando há uma grande diferença na demanda de passageiros entre o sentido transportado e o oposto. (YU; YANG; LI, 2012)
- **Priorização semafórica (TSP – *Transit Signal Priority*):** Antecipa ou atrasa o tempo de troca do semáforo, utilizado quando há infraestrutura necessária e quando não há outro veículo com prioridade maior, como ambulâncias e carro de bombeiro. (ESTRADA et al., 2016), (HU; PARK; LEE, 2015)
- **Pular parada (*Stop Skiping*):** O veículo para em apenas alguns pontos selecionados, utilizado quando há informação eficiente aos usuários e for programado com antecedência. (LIU et al., 2013b), (SUN; HICKMAN, 2005)
- **Retenção (*Holding)*:** Veículo permanece parado por algum período em algum ponto depois de finalizar o embarque/desembarque, é geralmente praticado quando a via possui um acostamento (*shoulder lane*) de ônibus para não prejudicar outras linhas que passam no mesmo ponto e quando há uma folga (slack time) para essa operação. (HICKMAN, 2001), (XUAN; ARGOTE; DAGANZO, 2011)
- **Ultrapassagem (*Overtaking*):** O veículo menos carregado ultrapassa veículo mais carregado, quando na situação de comboio. (WU; LIU; JIN, 2017)

Conforme (BERREBI et al., 2018), os métodos de controle do *headway* mais eficientes levam em consideração a previsão dos tempos de viagem dos ônibus. Pois dessa maneira, antes mesmo que o *headway* seja distorcido, ele já será corrigido baseado nas previsões de chegada de cada ônibus.

Para possibilitar a elaboração de previsões precisas de tempos de viagem de ônibus, é importante levar em consideração o tempo de permanência em cada ponto de ônibus (*dwell time*), uma vez que o tempo de permanência pode representar um dos principais componentes do tempo total de trajeto, (MENG; QU, 2013). O tempo de permanência consiste no tempo em

que o ônibus ficou parado no ponto (somatória dos tempos de abertura e fechamento de portas com os embarques e desembarques de passageiros).

O tempo de permanência no ponto de ônibus pode ser medido através de sistemas de câmeras fazendo reconhecimento de imagem ou através de RFID instalados nos pontos de ônibus, (ASSAF; WILLIAMS, 2011). Uma maneira de estimar o tempo de permanência no ponto de ônibus com precisão é através do uso dos Contadores de Passageiros Automático (APC), que são sensores instalados nas portas dos ônibus que coletam a quantidade de passageiros que embarcaram e desembarcaram em cada ponto de ônibus, (RAJBHANDARI; CHIEN; DANIEL, 2003). Porém o custo de instalação e manutenção desses sensores são elevados.

Outro método para estimar o tempo de permanência é fazendo uso dos dados de bilhetagem eletrônica (AFC - *Automatic Fare Counting*), que estima a quantidade de passageiros embarcando em cada ponto. Porém este método não é tão preciso quanto os citados anteriormente, (MILKOVITS, 2008). Uma vez que nem todos os passageiros utilizam o pagamento eletrônico, os passageiros podem realizar o pagamento eletrônico a qualquer momento durante o deslocamento do ônibus, e não é possível estimar a quantidade de passageiros que desembarcam, (DOU et al., 2015).

Apesar da importância do tempo de permanência para estimar o tempo total de trajeto, essa variável não será analisada. Dentre os três métodos para obter o tempo de permanência, seria possível utilizar o AFC, que é utilizado na agência de transporte público selecionada. Porém, pela falta de disponibilização desses dados por parte da agência, será desconsiderado da análise.

Existem alguns estudos sobre métodos de controle da operação da linha de ônibus que dependem de previsões precisas de tempos de viagem de ônibus. Para que se obtenha métodos de controle operacionais de linha de ônibus mais eficientes, maior de ser a precisão da previsão. Caso isso não seja satisfeito, o método de controle pode piorar em relação a um método que não utilize previsões (seja apenas corretivo). Nos próximos parágrafos, serão apontados alguns estudos que demandam informações precisas das previsões dos tempos de viagem dos ônibus:

He (2015) propôs uma estratégia “anticomboio” para melhorar a tabela horária dos ônibus e a confiabilidade do *headway,* utilizando informações precisas de tempos de chegada do ônibus atual e do próximo. A estratégia consiste em aplicar manobras de controle de velocidade dos ônibus e retenção nos pontos de parada para regular o *headway*.

Berrebi, Watkins e Laval (2015) desenvolveram uma política de despacho de ônibus otimizada baseada na informação perfeita sobre os horários de chegada dos ônibus. Dessa forma, os autores conseguiram diminuir o *headway* nos pontos de controle e diminuiu o tempo de espera dos passageiros.

Berrebi et al. (2018) realizaram uma comparação dos métodos de retenção de ônibus de outros autores e ainda compararam os desempenhos dos métodos com e sem previsões de viagem. Em todos os métodos que os autores fizeram os testes, a precisão da previsão influenciava na performance dos mecanismos de retenção dos ônibus, fazendo com que os ônibus perdessem menos tempo parado nos pontos de ônibus desnecessariamente.

2.3 Métodos de Previsão

Conforme visto na seção anterior, a previsão do tempo de viagem é algo muito importante para os usuários do sistema e para a agência provedora do transporte público, e quanto maior a precisão dessa previsão, melhores os resultados para o cliente e para a qualidade do serviço prestado. Portanto, a seguir será abordado alguns métodos de previsão, começando dos mais simples até os mais complexos. E por fim, será revisado as formas de medir e diagnosticar o modelo e a precisão da previsão.

2.3.1 Métodos Estatísticos

No campo da Estatística, quando uma determinada variável é observada ao longo do tempo, pode-se gerar uma série temporal da variável observada. Através da análise das séries temporais é possível inferir a frequência de determinados eventos, como picos e vales, ou a projeção futura da variável analisada, (MONTGOMERY; JENNINGS; KULAHCI, 2015).

Existem vários estudos dedicados para realizar previsão a partir de séries temporais, tais como os métodos auto regressivos integrados de médias móveis (ARIMA) e suas combinações (AR, MA, ARMA, SARIMA), (BOX et al., 2015). A forma ideal para se trabalhar com séries temporais é coletar dados de uma certa variável, ordenadas sequencialmente no tempo, preferencialmente em intervalos equidistantes. Caso isso não seja satisfeito, se torna uma amostra aleatória, (RASHIDI; RANJITKAR, 2013).

Existem duas abordagens diferentes que podem ser utilizadas na análise de séries temporais. A primeira é a análise no domínio do tempo. A segunda é a análise no domínio da frequência. Na análise no domínio do tempo avalia-se a evolução temporal do objeto de estudo, utilizando as funções de auto covariância e auto correlação. Na análise no domínio da frequência avalia-se a frequência com que determinados eventos ocorrem, ignorando os componentes harmônicos da série, e utiliza-se da densidade espectral, (MONTGOMERY; RUNGER, 2013).

Como base inicial para previsão, pode-se utilizar o método *naïve* (ingênuo), no qual o valor previsto será igual ao último valor observado, independentemente da quantidade de passos à frente. Este método, inclusive, é utilizado como *benchmark* para comparar os resultados de uma previsão proposta, (ZANINI, 2000).

Outro método relativamente simples é a média móvel, onde cada previsão é calculada a partir da média dos valores da janela anterior, sendo que a suavização da previsão é proporcional ao tamanho da janela, ou seja, os ruídos são ignorados e reage mais lentamente às mudanças. Existe ainda, outras variações da média móvel, como exemplo, a média móvel exponencial dupla e a tripla. A média móvel segue a seguinte expressão 1

$$X_t = \frac{1}{N}\sum_{i=1}^{N} X_{t-i} \tag{1}$$

sendo N o número total de dados históricos considerados, (acrescentar t), X_t é o valor estimado e X_{t-i} ($i = 1 \sim N$) os últimos dados históricos. Tanto o método ingênuo como a média móvel, seguem a mesma expressão, o valor de N para o método ingênuo é sempre 1, pois considera apenas o último dado histórico.

Dentre os métodos de previsão mencionados anteriormente, será utilizado como parte desta pesquisa apenas os métodos de média móvel e ingênuo, por sua simplicidade, por outros pesquisadores utilizarem destes métodos e por entregar resultados satisfatórios para o problema tratado, (KUMAR; VANAJAKSHI; SUBRAMANIAN, 2017), (MORI et al., 2015).

2.3.2 Aprendizado de Máquina

A inteligência artificial (IA) tornou-se uma tendência recentemente. Pessoas de diferentes áreas estão aplicando a IA para tornar suas tarefas mais fáceis. Por exemplo, economistas estão usando IA para prever preços futuros de mercado para obter lucro, médicos usam IA para classificar se um tumor é maligno ou benigno, meteorologistas usam a IA para prever o clima, recrutadores de recursos humanos usam IA para verificar se o currículo dos candidatos atende aos critérios mínimos para o trabalho (ERTEL, 2017). O que está por trás da ferramenta de IA são os algoritmos de aprendizado de máquina (*machine learning*) que a tornam inteligente e tão útil.

O aprendizado de máquina é um método computadorizado para treinar um modelo a partir de uma amostragem de dados. Depois de treinado, o modelo é capaz de classificar, ou prever valores, ou imagens. O aprendizado de máquina tem tomado grande importância para as pesquisas científicas, (HE et al., 2020), (LAI et al., 2020) e (ADEWALE; HADACHI, 2020). Dentre as técnicas de aprendizado de máquina podemos destacar a seguintes.

- **SVM** (*Support Vector Machine*), é um modelo de aprendizado de máquina aplicado a pequenas amostras, não linear e com reconhecimento de padrões de alta dimensão, seus modelos de aprendizado são supervisionados e são capazes de realizar classificação e regressão;
- **KNN** (*K-Nearest Neighbor*) é um algoritmo de aprendizado não paramétrico e iterativo. Seu objetivo é usar um banco de dados no qual os pontos de dados são separados em várias classes para prever a classificação de um novo ponto;
- **Regressão Linear** modela um valor de previsão de destino com base em variáveis independentes. É usado principalmente para descobrir a relação entre variáveis e previsão;
- **Florestas Aleatórias** (*Random forest*) são um método de aprendizado para classificação, regressão e outras tarefas que operam construindo uma grande variedade de árvores de decisão no momento do treinamento. Geram a classe que é o modo das classes (classificação) ou previsão média (regressão) das árvores individuais;

- **Algoritmo Genético** é um algoritmo de busca estocástico que atua em uma população de possíveis soluções, e é baseado na mecânica da genética e seleção de populações;
- **Rede Neural Artificial** (RNA) é um sistema de computação inspirados nas redes neurais biológicas. Sendo capaz de aprender e executar tarefas considerando amostras como exemplo.

Dentre os diferentes tipos de aprendizado de máquina, o modelo A RNA é um conjunto multicamada de *Perceptrons*, que é um modelo computacional baseado no neurônio biológico. Onde os dendritos do neurônio representam as entradas do *Perceptron* e o terminal do axônio representam a saída do *Perceptron*. Assim como no neurônio biológico, os sinais elétricos são modulados em várias quantidades, para o *Perceptron*, isso representa os pesos. Uma célula de neurônio ativa o terminal axônio apenas quando a soma dos sinais recebidos pelo núcleo atinge um certo limite, fazendo uma analogia à função de ativação do *Perceptron* (GRAUPE, 2007), conforme representado na Figura 1.

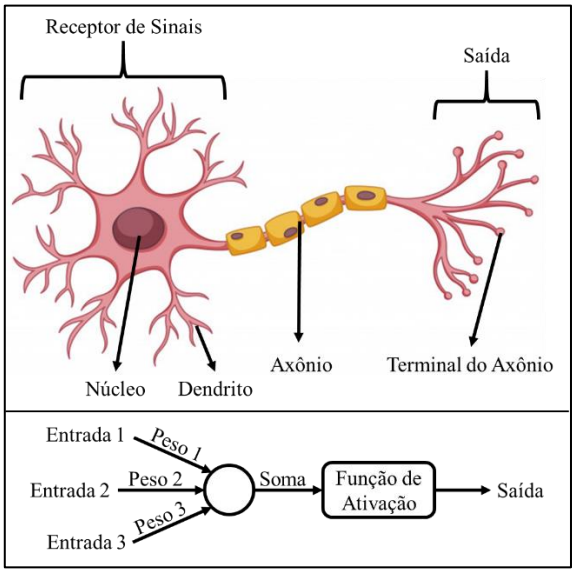


Figura 1 – Comparação entre um Neurônio Biológico e o Modelo *Perceptron*

Fonte: (FREEPIK, 2020) e Autor, elaborado através do PowerPoint

A função de ativação nada mais é do que uma equação configurável dentro do *Perceptron*. Ao receber a soma das suas entradas realiza o cálculo da equação e envia para a

saída o resultado. Existem vários tipos de funções de ativações, as mais comuns são: Linear, Sigmoide, Tangente Hiperbólica, Linear Retificada, Arco Tangente, Gaussiana e exponencial.

Um conjunto de *Perceptrons* organizados e conectados em camadas forma uma a Rede Neural Artificial (RNA), exemplificado na Figura 2. As RNAs podem fazer regressões, aproximações, previsões ou classificações.

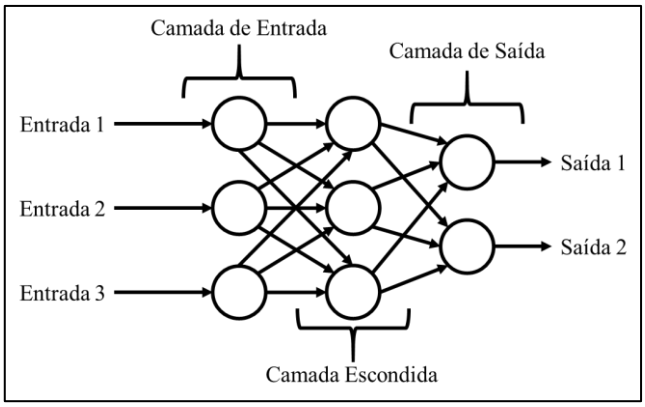


Figura 2 – Exemplo de Rede Neural Artificial *Feedfoward* com suas Camadas
Fonte: Autor

Para poder utilizar uma RNA, é necessário realizar o treinamento dela. Para isso, é preciso disponibilizar exemplos de valores nas entradas e nas saídas da RNA em um banco de dados. A partir disso, a RNA irá calcular as saídas conforme os dados de entrada e comparar com os valores de saída desejados para obter o erro da rede. Em seguida, a RNA alterará os pesos de cada *Perceptron* e recalculará o erro da rede, e seguirá dessa forma até que alguma condição estabelecida pelo criador da RNA seja atendida, (YEGNANARAYANA, 2012). Estas são as formas de colocar um limite para o treinamento de uma RNA:

- **Quantidade iterações (Épocas)**: estipula um número máximo de iterações dos recálculos e ajustes de pesos da RNA;
- **Erro mínimo desejado**: a RNA finaliza o treinamento quando alcança este valor determinado de erro e tipo de erro (os tipos de erro estão listados no item 2.3.3);
- **Parada Antecipada (*Early-Stop*)**: um conjunto de dados são separados do conjunto de treino e é informado um valor de paciência, então, é calculado o erro do modelo para esses dados separados a cada iteração, e o treinamento é finalizado quando a quantidade de vezes seguidas que o erro não diminuir no final de cada iteração for igual ao valor definido de paciência.

A RNA pode ser concebida em diferentes arquiteturas, dependendo do tipo de aplicação, a RNA pode ser do tipo *FeedForward*, retroalimentada ou reticulada. A RNA pode ser treinada de diferentes maneiras, dependendo da resposta que se deseja obter da RNA, (HAYKIN, 2009), podendo ser dos seguintes tipos:

- **Supervisionado**: usa um conjunto de entradas e saídas desejadas. A tarefa do aprendizado é produzir a saída desejada para cada entrada. Nesse caso, a função do peso está relacionada à eliminação de deduções incorretas.
- **Não Supervisionado**: funciona para algoritmos mais complicados em comparação com o aprendizado supervisionado, pois as informações sobre os dados são raras ou inexistentes. Onde é capaz de encontrar entidades como grupos, *clusters* e realizar estimativas de densidade e reduções de dimensões.
- **Com Reforço**: toma as medidas adequadas para maximizar a recompensa em uma situação específica. O aprendizado com reforço difere do aprendizado supervisionado de uma maneira que no aprendizado supervisionado, os dados de treinamento possuem uma resposta e o modelo é treinado com a resposta correta, enquanto no aprendizado por reforço não há resposta, mas o agente de reforço decide o que fazer para executar a tarefa especificada. Na ausência de um conjunto de dados de treinamento, ele deve aprender com sua experiência executando tentativas e aprendendo com erros e acertos.

Os hiperparâmetros de uma RNA são um conjunto de variáveis que determinam a estrutura da rede e como ela será treinada. Cada variável pertencente ao hiperparâmetro pode influenciar na rede positivamente ou negativamente. Por exemplo, a quantidade de camadas escondidas e largura de cada camada (quantidade de *Perceptrons*), e o tipo de otimizador (responsável pela atualização dos pesos), influenciam na velocidade de aprendizado do modelo, e por consequência na quantidade de épocas até finalizar o treino. No item 4.15 será abordada a análise e influência dos hiperparâmetros na RNA. Os outros hiperparâmetros que podem ser inseridos na RNA são:

- **Peso inicial**: Pode afetar o tempo de aprendizado e o número de iterações, mas não tem influência sobre o resultado do treino, pois os pesos são computados novamente ao final de cada iteração;
- **Momento**: ajuda a evitar oscilações e a conhecer a direção do próximo passo com o conhecimento dos passos anteriores;

- **Taxa de aprendizagem**: controla o quanto alterar nos pesos do modelo em resposta ao erro estimado cada vez que os pesos do modelo são atualizados. Um valor muito baixo de taxa de aprendizagem pode resultar em um longo processo de treinamento e pode até ficar parado, enquanto um valor muito alto pode resultar no aprendizado de um conjunto sub ótimo de pesos rápido demais ou em um processo de treinamento instável;
- **Porcentagem de abandono**: é uma técnica em que neurônios selecionados aleatoriamente são ignorados durante o treinamento. Eles são "abandonados" aleatoriamente. Isso significa que sua contribuição para a ativação dos neurônios a jusante é removida temporariamente e quaisquer atualizações de peso não são aplicadas ao neurônio na passagem para trás. A aplicação dessa técnica pode evitar o sobre ajuste (*overfitting*), bem como atrasar o processo de treinamento;
- **Tamanho do lote**: define o número de amostras que serão propagadas para a rede. Ao invés de treinar a rede com todo o conjunto de dados de treino, faz com que ela seja treinada em pequenos lotes de dados do conjunto. Este hiperparâmetro faz com que o processo de treinamento seja mais rápido e requeira menos memória para processamento, porém pode deixar o treinamento instável.

Como a determinação otimizada desses hiperparâmetros pode ser algo muito trabalhoso, uma vez que envolve métodos de Busca em Grade, Busca Aleatória e otimização Bayesiana, uma prática comum de ser aplicada é não inserir esses hiper-parâmetros na RNA e alterar apenas aqueles que tem uma influência maior na rede: a função de ativação, o otimizador, a largura e a camadas da rede, (JÚNIOR, 2018).

Dependendo do tamanho do banco de dados gerados, se caso houver poucas amostras, pode-se adotar medidas de validação cruzada (*cross validation*), (TSAMARDINOS; RAKHSHANI; LAGANI, 2015). Onde, para um mesmo banco de dados, separa-se diferentes partes da amostra de dados para treino e teste, exemplificado na Figura 3. E o resultado é a média de cada um dos treinos. A vantagem da utilização desse método é a possibilidade de reduzir o viés do modelo. Quando se possui um grande banco de dados, a utilização desse método se torna desnecessário.

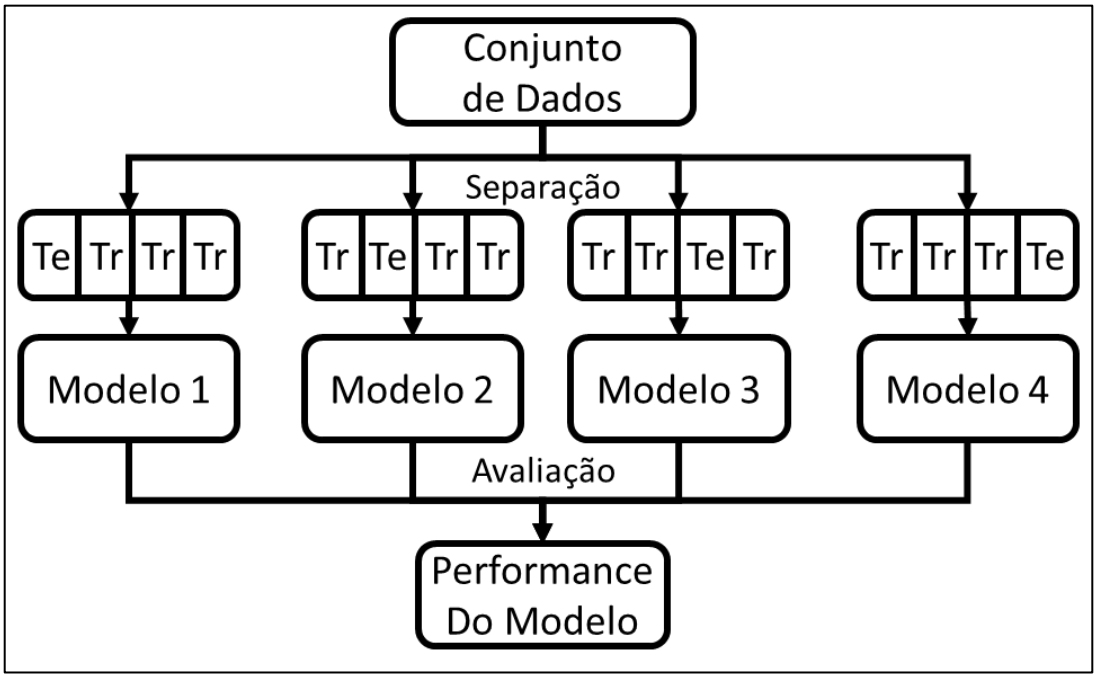


Figura 3 – Exemplificação de Validação Cruzada

Fonte: Autor.

Para séries temporais, onde deve-se utilizar os primeiros dados para treino e os últimos dados para teste e validação, pode-se empregar variações da validação cruzada, como por exemplo a divisão de séries temporais. Neste caso, pode-se dividir o banco de dados e realizar os treinos e testes para cada divisão, obedecendo a mesma regra de separar as últimas amostras para teste, e com isso chegar a um resultado mais estável de performance do modelo da RNA.

2.3.3 Diagnóstico do Modelo

Dados os métodos de previsão citados anteriormente, existem várias métricas de precisão para validação e comparação entre modelos, a seguir, serão apresentadas algumas delas. Seja y_i, os valores previstos, e x_i, os valores esperados.

1) **RMSE**: A Raiz do Erro Quadrático Médio é a raiz quadrada das médias das diferenças quadráticas entre os valores previstos e os valores esperados, sendo muito utilizada com a finalidade de minimizar grandes erros, conforme definido pela expressão 2. Quanto menor for o RMSE, melhor será o modelo de previsão.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(y_i - x_i)^2}{n}} \quad (2)$$

2) **MAE:** O Erro Médio Absoluto mede a magnitude média dos erros em um conjunto de previsões, sem considerar sua direção. É a média na amostra de teste das diferenças absolutas entre previsão e observação real em que todas as diferenças individuais têm pesos iguais, descrito na expressão 3. Quanto menor for o erro médio absoluto, menor é o erro do modelo;

$$MAE = \frac{\sum_{i=1}^{N}(y_i - x_i)}{N} \quad (3)$$

3) **MAPE:** O Erro Médio Absoluto Percentual é o MAE expresso em porcentagem relativo ao esperado, e por estar em porcentagem permite a comparação com erros de outros modelos e métodos de previsão, fórmula 4. Quanto menor a porcentagem, menor será o erro.

$$MAPE = \frac{100}{N} \sum_{t=1}^{N} \frac{|y_i - x_i|}{x_i} \quad (4)$$

4) **CCP**: Coeficiente de Correlação de Pearson mede o sentido e grau da correlação linear e entre duas variáveis, porém a Correlação de Pearson não é afetada por diferenças escalares que mantenham a mesma correlação linear, seguindo a fórmula matemática definida pela expressão (5), sendo ($\bar{x}$) e ($\bar{y}$) as médias dos valores esperados e previstos respectivamente. Valores mais próximos de 1 ou -1, significa que mais correlacionadas as variáveis estão;

$$\rho = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{[\sum_{i=1}^{n}(x_i - \bar{x})^2][\sum_{i=1}^{n}(y_i - \bar{y})^2]}} \quad (5)$$

5) **R^2:** O Coeficiente de Determinação é uma medida estatística que reflete o quanto as previsões de regressão se aproximam dos dados reais, expressão 6. Valores mais próximos de 1 significam que a variável pode ser determinada pelos regressores presentes no modelo.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (6)$$

Dentre as expressões apresentadas, as que serão utilizadas nesse trabalho serão o R^2 e o CCP para a análise do conjunto de dados de entrada com a saída desejada, e o MAPE como o principal valor de erro a ser otimizado, pois expressa um resultado percentual relativo à variável a ser prevista.

Através da utilização do MAPE para o cálculo do erro, possibilita a comparação com outros métodos de diferentes autores em linhas de ônibus com extensões diversas, pois como exemplo, um erro quadrático médio de 3 minutos pode ser pouco para uma viagem de duas horas, enquanto os mesmos 3 minutos podem ser um erro alto para uma vigem de apenas 10 minutos. A aplicação dessas expressões e resultados estão no item 4.16 dessa pesquisa.

2.4 Computação na Nuvem

A seguir será apresentado a proposta da computação na nuvem, qual o impacto que ela gera na sociedade e na forma em que vivemos, bem como os provedores desse serviço que são referência global. Por fim, será abordado esse tema dentro da mobilidade urbana e o seu papel para essa pesquisa.

2.4.1 Plataformas para Computação na Nuvem

Platform as a Service, também conhecido como PaaS (Plataforma como Serviço) é um serviço oferecido por algumas empresas para realizar atividades na nuvem, dentre essas atividades destaca-se o IoT (Internet das Coisas), computação na nuvem (*Cloud Computing*), criptografias e autenticações na nuvem, armazenamento de dados na nuvem, e provisão de serviços de tradução e reconhecimento de fala e imagens na nuvem, entre outros. Esses serviços permitem que os clientes possam desenvolver, gerenciar, conectar com rapidez e manter as

aplicações funcionando na nuvem sem a complexidade de se montar e manter uma infraestrutura local para manter o serviço sempre ativo, (PAHL, 2015).

As atividades cotidianas, como bancos, e-mail, streaming de mídia e comércio eletrônico, usam serviços na nuvem. No lado comercial, aplicativos, infraestrutura, armazenamento e vendas estão presentes na nuvem. A computação em nuvem é essencialmente a oferta de um aplicativo ou serviço oferecido em vários dispositivos ou locais. Esta oferta pode ser fornecida em três tipos diferentes de computação na nuvem: Privado, Híbrido e Nuvem Pública. O privado seria baseado apenas nas instalações do fornecedor, enquanto o híbrido aumentaria esse local com alguns estabelecimentos públicos. A nuvem pública seria fornecida totalmente do lado de fora por um fornecedor terceiro.

A computação na nuvem será um importante recurso que será utilizado nesse trabalho, principalmente pela necessidade de coletar dados (*data mining*) de maneira remota, automática e contínua (24 horas) por um período de pelo menos 1 mês ininterrupto. Por esse motivo a estabilidade de um servidor na nuvem é muito maior do que um servidor local. Já que um servidor local necessitaria resistir a possíveis quedas de energia elétrica, flutuações de conexão com a internet e interrupções do sistema operacional, (HOSSEINI-MOTLAGH; AHADPOUR; HAERI, 2015).

Conforme PARAISO et al. (2012), os três maiores provedores de PaaS são as seguintes plataformas: *Amazon Web Services* (AWS), *Google Cloud Platform* (GCP) e a Azure (da Microsoft). Apesar dos serviços serem similares, cada plataforma possui vantagens e desvantagens nos diferentes aspectos do serviço prestado.

Como o presente trabalho necessita de apenas um dos vários serviços disponíveis dentro das plataformas, será apresentada uma comparação, na Tabela 2, entre o que cada plataforma oferece dentro do nível gratuito sobre máquinas virtuais na nuvem (VM - *Virtual Machine*). No capítulo 4.4 abordaremos a respeito da escolha do provedor da plataforma.

Tabela 2 – Lista de Configurações das Máquinas Virtuais para Computação na Nuvem

Configuração	**AWS**	**Azure**	**GCP**
Virtual CPU	1	1	1
RAM	1 GB	4 GB	4 GB
HD	30 GB	50 GB	50 GB
Tempo no Nível Gratuito	1 ano	3 meses	1 ano

Fonte: Páginas *web* de cada provedor, (AMAZON, 2019), (GOOGLE, 2019b), (AZURE, 2019)

2.4.2 API

API (*Application Programming Interface*) é um conjunto de processamentos padronizados de um software que permitem que usuários externos utilizem as funcionalidades do software (incluindo o acesso aos dados) sem a necessidade de conhecer os detalhes de implementação do software. Trata-se de uma técnica que facilita muito o desenvolvimento de aplicações através do processamento de dados externos ou na nuvem, (MAHIDDINI, 2017).

Através do emprego de APIs possibilita que empresas criem seu próprio *web service*, que são formas padronizadas de acessar uma API. O *web service* facilita a interação entre duas máquinas conectadas à rede, pois foi projetado para ter uma interface que é representada em um formato processável por máquina especificado em WSDL (*Web Service Description Language*).

A comunicação de um *web service* deve ser realizada a um servidor HTTP (*Hypertext Transfer Protocol*), que basicamente é o endereço URI (*Uniform Resource Identifier*) de um servidor ou *site*. E a forma de comunicação pode ser SOAP (*Simple Object Access Protocol*), REST (*Representation State Transfer*) ou XML-RPC (*XML Remote Procedure Call*), que pode-se entender como o protocolo ou linguagem da requisição HTTP. E o método da requisição HTTP pode ser “GET”, “POST”, “PUT”, “PATCH” e “DELETE”, o funcionamento desses métodos é similar ao acessar um site: o “GET” é como abrir e carregar a página de um *site*, o “POST” é como preencher e submeter o formulário desse site, o “PUT” é como resubmeter o formulário de cadastro alterando todas as informações, o “PATCH” é como alterar apenas uma única informação do cadastro, e o “DELETE” é para excluir o seu cadastro do site.

Com o emprego de APIs e *web services*, também possibilita a criação aplicativos de mobilidade por usuários interessados em acessar dados de localização de ônibus em tempo real e gerar, por exemplo, informações tempo de chegada a um determinado ponto. Na próxima seção será apresentado algumas APIs de mobilidade presentes na cidade de São Paulo e disponibilizadas abertamente a qualquer desenvolvedor ou pesquisador.

Na área da mobilidade, as APIs são muito utilizadas para tornar aplicativos mais rápidos, diminuir o processamento interno dos programas, e diminuir o volume de dados transmitidos entre o provedor do aplicativo e os usuários. Um exemplo disso é, quando um provedor de rotas recebe uma solicitação sobre rotas mais rápidas ou trajeto mais curto entre dois pontos, ao invés dele enviar todas informações das vias (velocidades médias e comprimentos) para o celular do usuário processar e calcular o trajeto mais rápido ou mais curto, o provedor de rotas processa todas essas informações externas ao aplicativo (na nuvem) e envia para o cliente apenas as informações mais relevantes sobre as possíveis rotas, (YOON et al., 2010).

2.4.3 API OlhoVivo SPTrans

A SPTrans é o órgão responsável pela administração da operação de uma frota de aproximadamente 14 mil ônibus do município de São Paulo, que circulam em 17 mil km de vias, 1400 linhas, cerca de 19 mil pontos de paradas, atendendo mensalmente aproximadamente 220 milhões de passageiros, (SPTRANS, 2019a).

OlhoVivo é um serviço gratuito de informação de mobilidade da cidade de São Paulo disponibilizado pela SPTrans. Os usuários podem acessar esse sistema via *web*, por meio de computador ou dispositivo móvel, podendo consultar a localização do ônibus, o horário do próximo ônibus, velocidade e tempo de viagem nos principais corredores da cidade. A SPTrans disponibiliza também, no seu *site*, uma área para os desenvolvedores, onde podem ser obtidos dados de transporte públicos da cidade de São Paulo (SPTRANS, 2019b).

A API OlhoVivo provê informações em tempo real do monitoramento da frota de ônibus de São Paulo. Por meio dessa API, a SPTrans torna público os dados dos equipamentos embarcados (AVL) instalados nos veículos. Para algumas linhas específicas (corredores de ônibus) o sistema gera uma estimativa de tempo de chegada e velocidade média. Para cada tipo de informação desejada, uma solicitação específica deve ser realizada. No caso de um monitoramento contínuo pode-se realizar múltiplas solicitações ao longo do tempo, obtendo-se

assim atualizações frequentes de dados dos AVLs e gerar banco de dados próprio. A Figura 4 ilustra a interface digital da plataforma OlhoVivo.

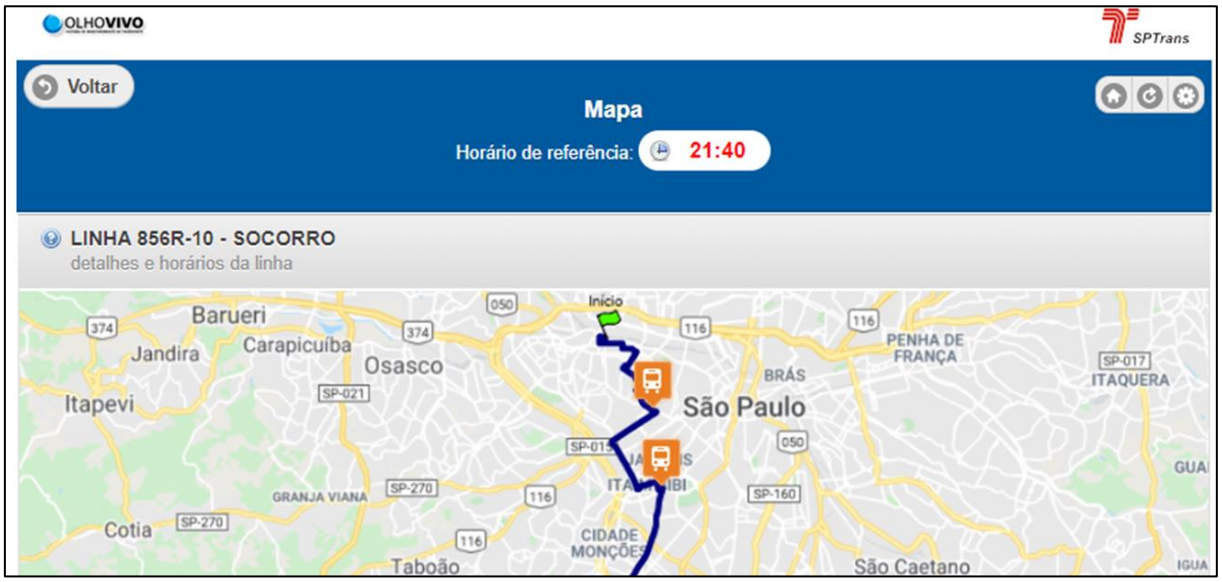


Figura 4 – Ilustração da Plataforma *web* do OlhoVivo

Fonte: (SPTRANS, 2020)

Para ter acesso a essa API é necessário criar uma chave de acesso, que é utilizado pela SPTrans gerenciar o tráfego de dados. A preocupação é limitar o acesso de forma a não sobrecarregar o sistema. Essa chave é utilizada no início de cada seção, quando é feita a autenticação na API, antes de começar a coleta de dados.

2.4.4 Painel Scipopulis

A Scipopulis é uma *startup* de inovação, focada em cidades inteligentes e dedicada à mobilidade urbana, (SCIPOPULIS, 2019a). Ela teve grande contribuição para a mobilidade urbana de São Paulo ao elaborar o volume 20 do caderno técnico da ANTP, (ANTP, 2015), que retrata sobre o Big Data gerado sobre os dados da SPTrans. Uma das atividades da Scipopulis é coletar o máximo de informações de mobilidade disponíveis na cidade de São Paulo para gerar e disponibilizar painéis e relatórios simplificados em uma plataforma para os usuários.

A empresa Scipopulis coleta diariamente o GTFS gerado pela SPTrans e utiliza a API do OlhoVivo para obter a geolocalização de todos os ônibus da SPTrans, e realiza uma estimativa de velocidade através da geolocalização de cada ônibus, (ANTP, 2015). E, com base

nos dados dos trechos compreendidos entre dois pontos de parada faz o cálculo de velocidade média para cada trecho e por faixa horária, (YAI, 2015).

A Scipopulis disponibiliza uma plataforma *online*, (SCIPOPULIS, 2019b), que permite ao usuário monitorar as velocidades médias dos ônibus, visualizar históricos graficamente por dias da semana, e exportar os dados de velocidade média de cada trecho para cada faixa horária, conforme observado na Figura 5.

Figura 5 – Painel de Monitoramento Scipopulis

Fonte: (SCIPOPULIS, 2019b)

Os dados sobre velocidade média informados no painel da Scipopulis podem ser calculados através da coleta de dados brutos da API OlhoVivo da SPTrans em conjunto com a trajetória da linha através do GTFS. Esses dados devem ser limpos e analisados, inclusive com ônibus de outras linhas que passam na mesma via, a fim de se chegar em uma velocidade média concisa.

Ao utilizar esse painel, pode-se obter dados históricos sobre qualquer período e qualquer linha de ônibus da SPTrans, que já passaram por uma limpeza e tratamento prévio. Dessa maneira, pode-se economizar tempo e recursos computacionais em obter dados históricos que possam contribuir com o modelo de previsão.

2.4.5 API Google Maps

Google Maps é um serviço *web* que provê informações geográficas e de mobilidade graficamente. Ele oferece serviços de mobilidade como planejador de rotas e navegação dinâmica para motoristas, disponibiliza informações sobre pontos de interesse (POI - *Points of Interest*), imagens 360° de ruas e lugares (*Street View*). Exibe informações geográficas como relevo, topografia, imagens de satélite, além de contar com uma ferramenta SIG (Sistemas de Informações Geográficas) própria denominada Google My Maps, (GOOGLE MAPS, 2019a). O Google Maps também é utilizado no ramo acadêmico como fonte de dados de mobilidade para pesquisadores, (ROTHFELD et al., 2019), (RAHMANI; KOUTSOPOULOS; JENELIUS, 2017).

Outro exemplo são as APIs de mobilidade do Google Maps, que também são gratuitas, porém há um limite diário de 100.000 solicitações por dia de cada API. O acesso ao serviço, bem como a documentação está dentro do GCP (GOOGLE, 2019c). O Google Maps proporciona os seguintes serviços *web* na Tabela 3:

Tabela 3 – Descrição dos Serviços *web* do Google Maps

Serviço	Descrição
Geocoding API	Converte entre endereços e coordenadas geográficas
Places Web Service	Implementa o preenchimento automático e adiciona informações atualizadas sobre milhões de locais
Elevation API	Fornece dados de elevação de qualquer ponto no mundo
Directions API	Calcula rotas entre vários locais
Roads API	Ativa a funcionalidade de aderência à rua para acompanhar com precisão rastros do GPS
Time Zone API	Fornece dados de fuso horário para qualquer lugar do mundo
Geolocation API	Encontra um local com base em informações de torres de celulares e nós de rede WiFi
Distance Matrix API	Estima o tempo e a distância de percurso para vários destinos

Fonte: Autor, baseado nas informações em (GOOGLE MAPS, 2019b)

Uma parte significativa das informações do Google Maps são provenientes de dados compartilhados pelos próprios usuários, em tempo real. Esse processo é conhecido como *crowdsourcing*. O usuário precisa ter uma conexão de dados ativa no seu dispositivo móvel. Através dessa conexão o Google Maps enriquece o seu serviço com informações de trânsito em tempo real, calcula a velocidade média por trecho, ao mesmo que mantém o mapa atualizado.

Para o propósito dessa pesquisa poderão ser utilizados as seguintes APIs: Google Maps *Directions* e Google Maps *Distance Matrix*. Essas APIs fornecerem previsões de tempo de deslocamento para determinados trechos e faixa de horários. Há três variações entre os modelos que podem ser solicitados a essas APIs:

- **Melhor Estimativa:** É o modelo padrão das APIs do Google Maps, onde o tempo de viagem previsto deve se aproximar melhor do tempo de viagem a ser desempenhado, uma vez que o provedor possui informações sobre as condições históricas do tráfego e o tráfego em tempo real.
- **Pessimista:** Indica que o tempo de viagem previsto deve ser maior que o tempo de viagem na maioria dos dias, porém dias com condições de tráfego excessivamente ruins podem exceder o tempo dessa previsão.
- **Otimista:** Indica que o tempo de viagem previsto deve ser menor que o tempo de viagem na maioria dos dias, porém dias com condições de tráfego excessivamente livres podem ser menor que o tempo informado nessa previsão.

O valor padrão "Melhor Estimativa" fornecerá as previsões mais corretas para a maioria dos casos, conforme (GOOGLE, 2020). Porém é possível que a previsão do tempo de viagem "Melhor Estimativa" possa ser menor do que o modelo "Otimista" ou, alternativamente, maior que o modelo "Pessimista", por conta da maneira em que o modelo de previsão de "Melhor Estimativa" integra informações de tráfego em tempo real.

Outra opção para essas APIs é a possibilidade de coletar uma previsão de trânsito futuro, é como se essas APIs do Google Maps fossem capazes de informar qual o tempo de viagem se fosse iniciar a viagem daqui 5 minutos (ou qualquer outro valor de tempo). Dessa maneira é possível saber se a previsão do tempo de viagem será maior ou menor no futuro e sua magnitude, em qualquer um dos 3 modelos (Melhor Estimativa, Pessimista ou Otimista).

Quanto mais próximo for a previsão do trânsito futuro do momento da coleta, mais próximo esse modelo de previsão se baseia nos dados de trânsito em tempo real, e quanto mais distante esses horários forem, a previsão tomará como base o trânsito típico para aquele determinado horario. A escolha da API, modelos de previsão, bem como sua utilização está descrita no item 4.5 dessa pesquisa.

2.5 Estado da Arte

A seguir será apresentado a evolução das pesquisas científicas relacionadas à predição de tempos de viagem de ônibus até o presente. Os artigos que utilizam técnicas de aprendizado de máquina ou algum tipo de dado de trânsito em tempo real terão um foco maior. E por fim os resultados de cada autor será analisado para uma discussão de resultados posterior.

2.5.1 Pesquisas na Área de Predição de Tempos de Viagens

Um dos primeiros artigos científicos sobre o tema, foi apresentado por LIN e ZENG (1999), onde já se discutia sobre predição em tempo real de ônibus a partir de dados de AVL para servir de informação ao usuário. Sendo que a partir do ano 2002 pode-se encontrar artigos utilizando RNA para prever os tempos de chegada dos ônibus, (CHIEN; DING; WEI, 2002).

Em 2003, CATHEY e DAILEY (2003) realizaram uma prescrição de partidas e chegadas de ônibus usando o filtro de Kalman a partir dos dados de AVL. Enquanto em 2004, SHALABY e FARHAN (2004) elaboraram um modelo de predição utilizando dados de AVL, APC e o filtro de Kalman. JEONG e RILETT(2005) desenvolveram um modelo de predição em tempo real baseado em redes neurais.

Em 2009, CHEN PENG; YAN XIN-PING e LI XU-HONG (2009) iniciaram a pesquisa de predição de tempos de viagem de ônibus utilizando RVM (*Relevance Vector Machine*), que também utilizava regressões. Enquanto que PADMANABAN, VANAJAKSHI e SUBRAMANIAN (2009) estudaram o impacto do tempo de permanência nos pontos de ônibus no tempo total de trajeto.

No ano de 2011, ZHU et al. (2011) realizaram pesquisas sobre a influência do tráfego na via sobre a predição de chegada dos ônibus. Em 2012, BAPTISTA, BOUILLET e POMPEY (2012) utilizaram o algoritmo KNN (*K-Nearest Neighbors*) para realizar suas predições. E em 2013, LIN et al.(2013) realizaram predições baseadas em dados de AVL, AFC e utilizaram os modelos de Filtro de Kalman e RNA, e em suas conclusões apontaram que a RNA superou os resultados do Filtro de Kalman.

Algumas revisões da literatura foram realizadas, existindo uma revisão elaborada por ALTINKAYA, MEHMET e ZONTUL (2013), onde é explicado os principais modelos computacionais utilizados para a previsão do tempo de chegada dos ônibus. Os métodos

destacados pelo autor foram o uso da velocidade média; modelos estatísticos (como série temporal e regressão), filtro de Kalman; aprendizado de máquina (como SVM, RNA); e modelos híbridos. Em sua conclusão, os autores mencionaram que para cada caso específico o método ideal pode variar, dependendo da necessidade e das características do banco de dados. Em 2016, CHOUDHARY; KHAMPARIA e GAHIER (2016) realizaram outra pesquisa e incluíram a lógica *Fuzzy* e o KNN (*K- nearest Neighbour Three*) em sua revisão.

As pesquisas mais recentes nessa área é o artigo dos autores de (PETERSEN; RODRIGUES; PEREIRA, 2019), que desenvolveram uma rede neural convolucional LSTM (*Long Short-Term Memory*) para realizar as predições. (KUMAR et al., 2019) apresentaram o modelo KNN para classificação e estimativa do tempo de viagem de ônibus recursivo. (CHEN et al., 2019) usaram o DBN (*Deep Belief Network*) com o algoritmo de retro-propagação para fazer as previsões dos ônibus. Além disso, (ACHAR et al., 2020) aplicaram um filtro Kalman para otimizar a previsão do horário de chegada dos ônibus.

Para efeito de comparação os artigos tratados em detalhe a seguir são exemplos de outros pesquisadores do mesmo tema que reportaram sobre os resultados de precisão da previsão, utilizando como critério o MAPE. Esses resultados servirão como base para comparação futura do resultado dessa pesquisa.

2.5.2 Modelos de Previsão Baseado em Redes Neurais Artificiais

JEONG e RILETT(2004), coletaram dados de seis meses de uma única linha através de um dispositivo AVL que capturava dados a cada cinco segundos. Eles desenvolveram um modelo RNA que recebe como entrada os tempos de viagem nos trechos anteriores, tempo de permanência no ponto de ônibus e tempo de atraso em cada parada, e tem como saída os tempos de chegada em cada ponto de ônibus. No caso, os autores desenvolveram uma RNA para cada trecho da linha. E realizaram diversos treinos e testes das RNA alterando a quantidade de camadas escondidas, funções de ativações e otimizadores.

Na apresentação de resultados, os autores mostram apenas os erros percentuais das melhores RNA, para cada função de ativação e otimizador, excluindo os resultados para as quantidades de camadas escondidas e para cada trecho em que a RNA foi treinada separadamente. Eles compararam também os modelos de RNA com um modelo histórico e um modelo de regressão, conforme mostrado na Tabela 4:

Tabela 4 – MAPE dos Modelos de Previsão (%)

Modelos	Local	SA	FIM	PC	FP	NT
Modelos Baseados em Dados Históricos	Centro	14,63	13,03	10,77	14,34	12,82
	Periferia	7,58	7,27	7,02	8,74	6,43
Modelo de Regressão	Centro	24,85	22,81	14,88	23,48	19,97
	Periferia	15,25	14,54	11,57	17,56	13,92
Modelo RNA	Centro	5,07	4,13	5,2	8,36	7,17
	Periferia	2,88	1,96	4,58	4,46	4,87

Fonte: (JEONG; RILETT, 2004)

Legenda:

SA: Sem Agrupamento;

FIM: Fim de semana;

PC: Horário de Pico;

FP: Horário fora de Pico;

NT: Noite.

Por fim, os autores concluem que a RNA é capaz de identificar uma relação não linear entre tempos de trajeto e as variáveis independentes. A RNA resultou em valores de erros percentuais consideravelmente menores em relação aos modelos de regressão e histórico. Apesar dos valores baixos de erro para a RNA, os autores comentam que os dados fornecidos como entrada normalmente não estão presentes nas linhas de ônibus urbanas, mas que com novas técnicas de coleta de dados de ITS, isso não seria um impeditivo.

2.5.3 Modelos de Previsão Baseado em Dados de RFID e Taxi

XINGHAO et al (2013), coletaram dados de um mês dos AVLs referente a duas linhas de ônibus. Para simular as informações coletadas por meio do RFID, os autores obtiveram por meio de sistema de câmeras, as informações semafóricas (hora exata em que os semáforos alteraram para verde e para vermelho) e a velocidade média dos táxis dos trechos das duas linhas de ônibus de 10 de julho de 2012, em duas faixas horárias entre 8:00 e 10:00 e entre 15:00 e 17:00.

Na elaboração do modelo, os autores capturaram as velocidades dos táxis e fizeram uma regressão simples para as velocidades coletadas dos ônibus, que gerou seu primeiro resultado,

mostrando o quão correlacionado esses dados estão, chegando a um coeficiente de determinação de aproximadamente 80%, conforme mostrado na Figura 6 a seguir:

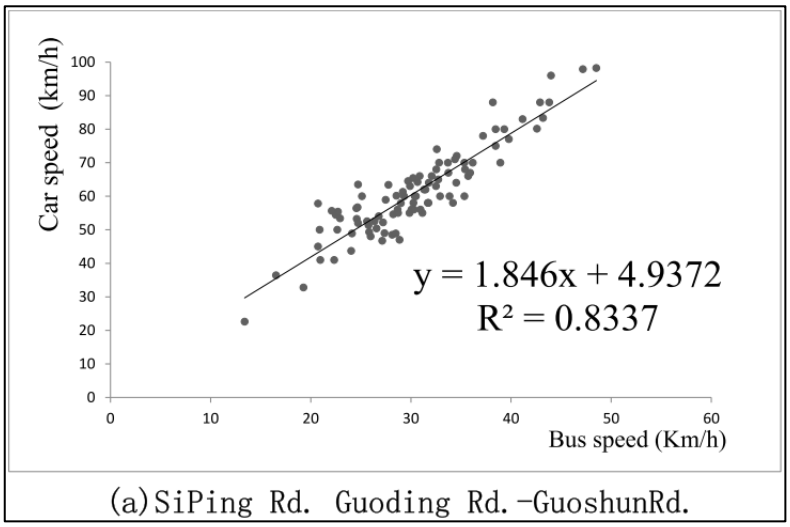


Figura 6 – Coeficiente de Determinação entre Táxi e Ônibus

Fonte: (XINGHAO et al., 2013)

Em seguida, os autores realizaram a previsão de tempo de chegada dos ônibus a partir do momento em que o ônibus passou por leitores de etiquetas eletrônicas de rádio frequência (RFID) e somaram com o resultado da regressão de velocidade entre ônibus e táxi. Também fizeram uma estimativa média da previsão baseada nos dados históricos do AVL, desconsiderando as informações da velocidade do táxi. Chegando nos gráficos da Figura 7 e resultados da Tabela 5 para comparar os dois métodos:

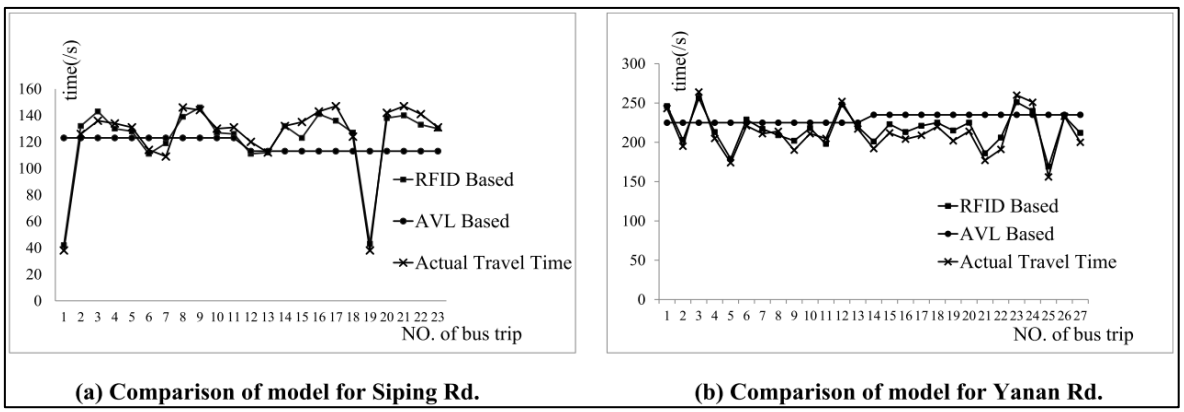


Figura 7 – Comparação entre Métodos de Previsão Média Histórica do AVL e RFID Simulado com Dados de Velocidade do Táxi em Tempo Real

Fonte: (XINGHAO et al., 2013)

Tabela 5 – Resultados da Precisão da Previsão da Média Histórica do AVL e da Previsão Através do RFID Simulado com Dados de Velocidade do Táxi em Tempo Real

Route	Algorithm	MAE(s)	MAPE (%)	$\left\|\varepsilon_i\right\|_{\max}$ (s)
BUS 55	AVL Based	22.17	18.99	85
	RFID Based	5.04	4.53	35
BUS 71	AVL Based	27.48	11.90	79
	RFID Based	8.26	3.89	32

Fonte: (XINGHAO et al., 2013)

2.5.4 Modelos Dinâmicos de Previsão Usando Dados de Geolocalização, Redes Neurais Artificiais e Filtro de Kalman

FAN e GURMU (2015) coletaram dados de novembro de AVL de 2008 a maio de 2009 de uma linha. Eles elaboraram modelos baseados em média histórica, filtro de Kalman e RNA para previsão de tempo de chegada. Para validação dos modelos, os autores criaram três seções da linha, a primeira seção cobrindo 9 pontos intermediários, a segunda seção cobrindo 19 pontos intermediários e a terceira seção cobrindo todos os 35 pontos de ônibus.

Para o treino da RNA, os autores normalizaram os dados, utilizaram da tangente hiperbólica como função de ativação e a retro propagação como algoritmo de aprendizado, porém não é comentado sobre a quantidade de camadas escondidas que foram utilizadas. Na apresentação de resultados, os autores compararam os resultados dos erros percentuais obtidos dos três modelos para as três seções, conforme mostrado na Figura 8:

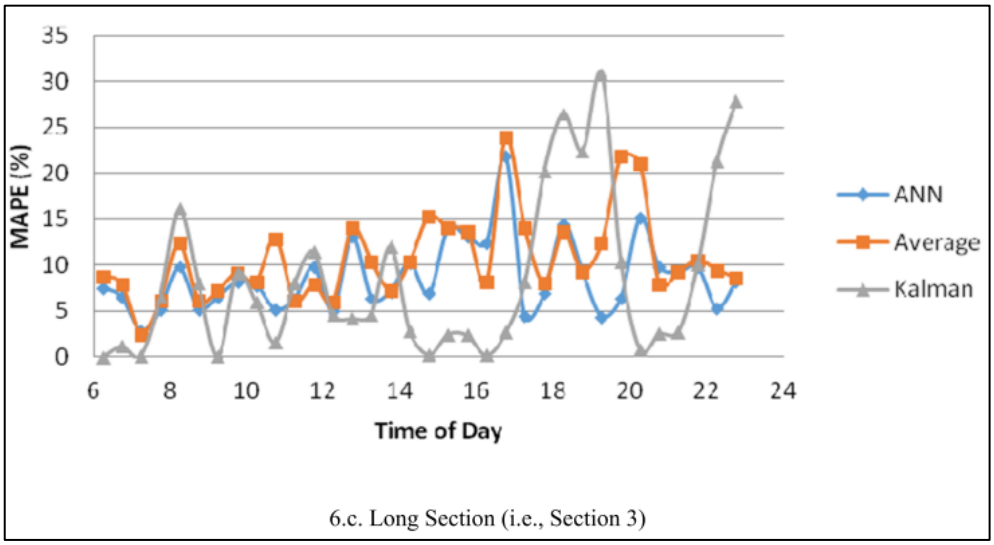


Figura 8 – MAPE x Hora do Dia da Seção 3

Fonte: (FAN; GURMU, 2015)

Por fim, os autores comentaram que obtiveram melhores resultados quando o tempo de viagem observado estiver mais próximo da média, ilustrado na Figura 9. E concluíram que a RNA obteve o melhor desempenho na previsão em relação aos outros dois métodos. O modelo baseado no filtro de Kalman conseguiu entregar uma boa previsão, porém mostrou-se vulnerável a grandes alterações de tempos de viagem para pouco tempo de reação.

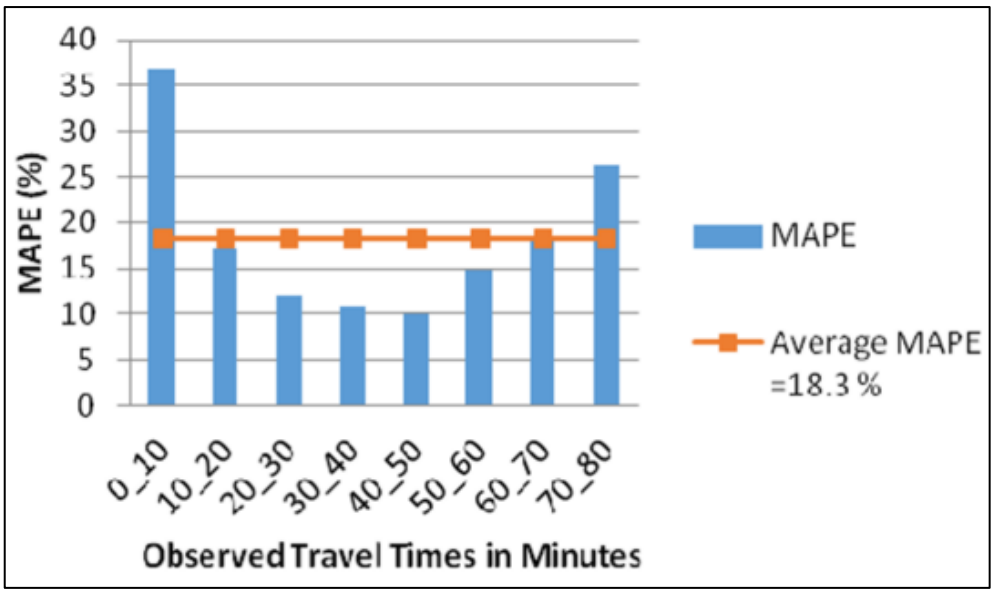


Figura 9 – MAPE para cada Agrupamento de Tempo de Viagem

Fonte: (FAN; GURMU, 2015)

2.5.5 Modelo de Previsão com Dados de Geolocalização de Múltiplas Linhas

No artigo “Um modelo de previsão do horário de chegada de ônibus nas paradas com múltiplas linhas” de YIN et al. (2017), os autores dedicaram-se na previsão de tempo de chegada dos ônibus apenas até o próximo ponto, tomando como base de dados apenas as seguintes informações de entrada:

1. Tempo de viagem dos três ônibus precedentes da mesma linha;
2. Velocidade do ônibus a ser previsto durante o último trecho percorrido;
3. Tempos de viagem de todos os ônibus que percorreram o trecho a ser previsto.

Com essas três informações, os autores elaboraram uma RNA e uma Máquina de Vetores de Suporte (SVM). Para simplificar o problema, os autores separaram entre horários de pico e fora de pico, e sentidos bairro-centro e centro-bairro, formando quatro grupos de análise. Além disso, os autores realizaram três diferentes combinações:

- S0: inclui todos os três valores de entrada como modelo;
- S1: altera a entrada 1, considerando apenas um ônibus predecessor da linha. Os outros dois valores de entrada são iguais aos da S0;
- S2: remove a entrada 3 enquanto permanece outros dois valores de entrada.

Nos resultados obtidos dos autores foi apontado que os modelos em RNA tiveram um desempenho melhor que em SVM. No estudo de caso, o MAPE do Cenário 0 foi menor, chegando a pouco menos de 10% na maioria dos grupos de análise, conforme a Figura 10:

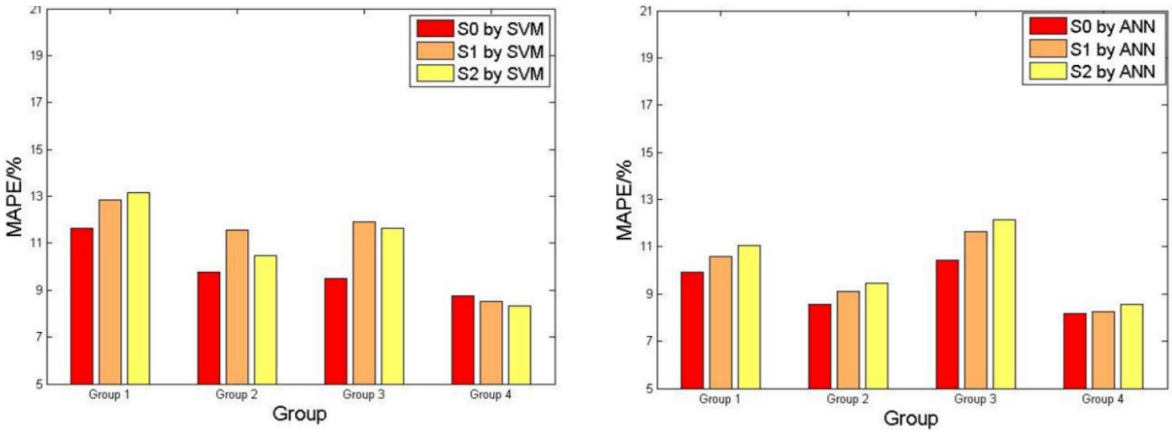


Figura 10 – MAPE para cada Grupo, Entrada dos Modelos Baseados em SVM e RNA

Fonte: (YIN et al., 2017)

2.6 Considerações Finais

Este capítulo apresentou os principais conceitos que serão tratados nesta pesquisa. Inicialmente, o ITS foi introduzido e foi apresentado o formato aberto da Especificação Geral sobre *Feeds* de Transporte Público (GTFS). Direcionando para duas áreas de influência do ITS, a primeira focada em regularizar a linha, e a outra em informar os passageiros. Também foi abordado alguns modelos de tempos de viagem e o impacto que a regularização da linha pode causar na qualidade do serviço. Em seguida foi discutido sobre alguns métodos de previsão estatísticos e métodos que utilizam RNA para gerar previsões, bem como, as métricas para diagnosticar a acurácia dos modelos.

Na seção seguinte abordamos sobre a computação na nuvem, onde mencionamos sobre algumas plataformas de serviço, e exemplificando formas de instanciar um servidor na nuvem para coleta cíclica de dados utilizando as API.

Por último, analisamos as pesquisas anteriores como referência e futura comparação de métodos e resultados. Apresentamos também os diferentes tipos de dados coletados e comparações realizadas por diferentes pesquisadores.

Tendo-se esses conhecimentos de diversas áreas como base, nesta pesquisa vamos propor uma metodologia que permita realizar a previsão de tempo de viagem de um ônibus com exatidão. A metodologia proposta descreverá desde o processo de coleta de dados até o treino da RNA, para futuras experimentações e análises combinatórias. Será explorada a capacidade da RNA de conseguir aprender com os dados coletados para poder entregar resultados precisos de previsão de tempos de viagem.

3 METODOLOGIA

3.1 Visão Geral e Arquitetura

No capítulo anterior foi visto que a previsão de tempo de viagem pode ser feita basicamente utilizando o método estatístico ou métodos de aprendizado de máquina. Esta pesquisa propõe uma metodologia baseada na combinação desses métodos, tomando como vantagem a possibilidade do emprego de múltiplos dados disponíveis atualmente. Por exemplo, dados de trânsito em tempo real, dados de previsão de trânsito, dados de rastreamento da frota de ônibus, dados de contagem de passageiros, dados meteorológicos entre outros.

Os métodos estatísticos são clássicos e consagrados por sua simplicidade e eficiência. O ponto fraco dos métodos estatísticos é que possuem uma fórmula definida e seu resultado probabilístico é sempre o mesmo, isso os torna engessados e inflexíveis com a nova realidade digital e com tantos dados disponíveis.

Por sua vez, os métodos de aprendizado de máquina estão em grande evolução e cada vez mais estão aumentando a sua capacidade de processar uma quantidade maior de dados e aprimorando o processo de aprendizado. Apesar desse método conseguir realizar previsões a partir de amostras de dados, quanto melhor for a qualidade dos dados informados, melhor será a eficiência desse método. Para atestar que dados possuem uma qualidade alta é necessário verificar a correlação entre o dado de entrada e o dado que se deseja prever, uma vez que, quanto mais forte for a correlação, maior a probabilidade de ser obter uma previsão mais precisa.

Uma das formas de se obter dados com essa característica é utilizando os métodos estatísticos para gerar as previsões de tempo de viagem. Como essa previsão estatística possui uma boa assertividade, existe uma grande possibilidade de ser fortemente correlacionada com os resultados desejados do aprendizado de máquina. Isso torna a previsão estatística um dado promissor a ser inserido no processo de aprendizagem. Dessa forma, será possível combinar os dois métodos em um processo mais robusto e que se beneficie dos pontos fortes de ambos os lados.

A metodologia proposta apresenta um processo para realizar predições precisas de tempos de viagem de ônibus, combinando um método de previsão estatística, um método de aprendizagem de máquina, e em conjunto com dados coletados em tempo real. Será apresentado todas as etapas do processo, incluindo a coleta de vários tipos de dados, armazenamento, análise do banco de dados, desenvolvimento e implementação das técnicas de aprendizado de máquina.

Este processo pode ser aplicado em qualquer via, ou linha de ônibus que possua dados de AVL e dados de trânsito em tempo real. Os dados de trânsito em tempo real podem ser provenientes de veículos de passeio ou de táxi, portanto, o processo pode ser aplicado a vias segregadas de ônibus, como faixas exclusivas ou corredores. Caso não haja dados de trânsito em tempo real, pode-se testar a metodologia utilizando algum outro tipo de dado que possa afetar o tempo de viagem dos ônibus, como por exemplo, os dados de bilhetagem (AFC) ou contagem de passageiros (APC).

Para utilizar a presente metodologia necessita-se basicamente de um computador conectado à internet para coletar, processar os dados de ônibus e de trânsito em tempo real e/ou outra base de dados, e utilizar as técnicas de aprendizado de máquina para realizar as previsões.

Para aplicar esta metodologia, primeiramente deve-se selecionar uma via, ou uma linha de ônibus, ou objeto de estudo, e coletar os seus dados geográficos (extensão, pontos de ônibus). A seguir, deve-se configurar um computador para coletar os dados de ônibus e de trânsito em tempo real. O computador deve ser capaz de limpar, formatar e interpretar os dados coletados corretamente conforme os dados geográficos.

Em seguida, um método de aprendizado de máquina deve ser selecionado. Entre as técnicas utilizadas dentro do aprendizado de máquina, citadas no item 2.3.2, a RNA será utilizada para o problema estudado, pois ela é capaz de tirar proveito de uma grande quantidade de dados de diferentes proveniências e ela pode facilmente articular os conjuntos de dados e hiperparâmetros a fim de encontrar a configuração ideal.

Um grande benefício que a RNA pode trazer é sua capacidade de aprender a partir dos dados informados. A RNA pode recalcular os pesos de cada dado de entrada e camadas escondidas a fim de conseguir encontrar uma função matemática que se aproxime da saída desejada. Caso algum dado de entrada não influencie em nada na saída (nem indiretamente), é bem provável que a RNA calcule os pesos para essa entrada bem próximo de zero (dependendo da função de ativação) para não afetar os resultados da função matemática obtida através do treinamento da RNA.

A RNA deve ser configurada e treinada a partir dos dados coletados. Deve ser verificado quais conjuntos de dados e quais hiperparâmetros retornam o melhor resultado. Após o treino, a RNA pode ser utilizada para entregar as previsões de tempos de viagem dos ônibus com alta precisão. Todo este processo está ilustrado na Figura 11:

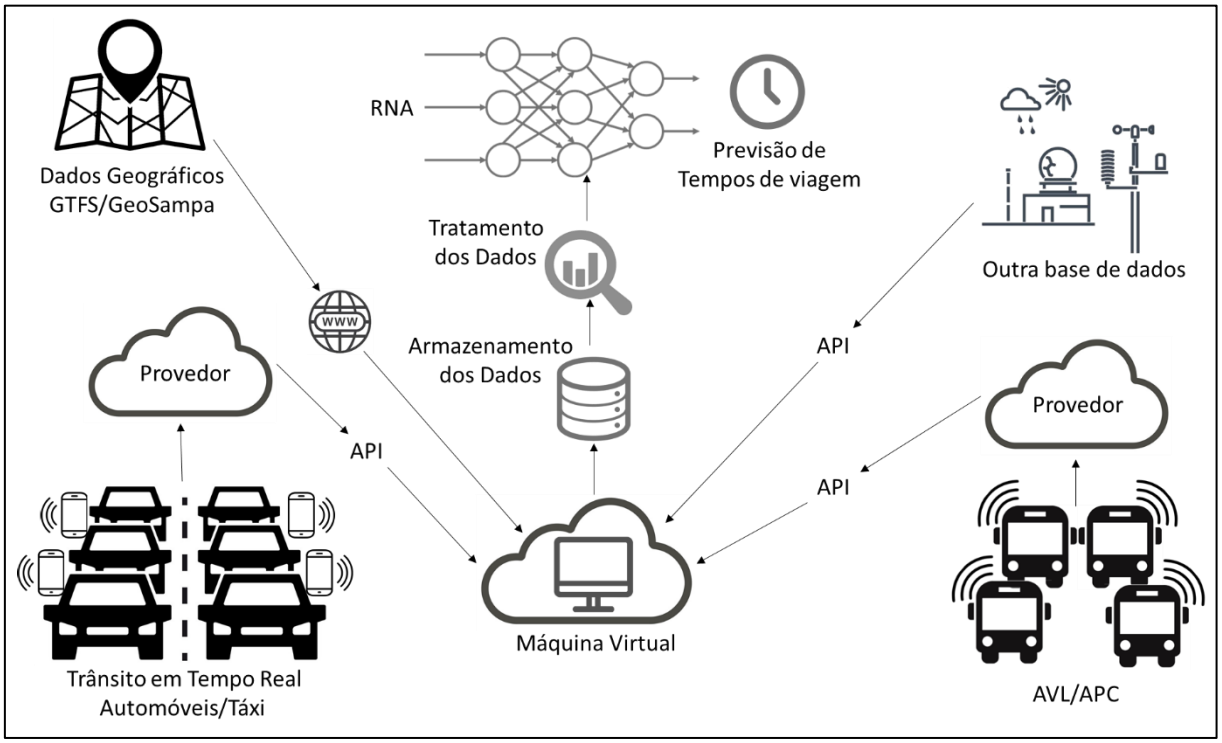


Figura 11 – Diagrama com a Visão Geral da Metodologia

Fonte: Autor.

Da perspectiva do usuário, uma máquina aprenderá a influência dos dados coletados no tempo de viagem dos ônibus através de uma RNA. Em seguida, ela gerará um algoritmo sofisticado que com base em cada dado coletado, a IA (Inteligência Artificial) da máquina estimará o tempo de viagem dos ônibus com alta precisão. Os dados em tempo real são necessários para a detecção de tráfego pesado na via, pois a RNA já será informada e considerará essa entrada no processo de previsão. Este é um dos pontos fracos de muitos métodos de previsão, conforme citado no capítulo anterior.

Devido à alta complexidade da RNA e as atualizações frequentes nas APIs *online* para coleta de dados, será necessário atualizar a RNA com os novos dados coletados para manter a entrega de previsões com alta precisão.

3.2 Análise dos Dados Geográficos

O primeiro passo deriva da limpeza e tratamento dos dados para prepará-los a serem inseridos na RNA. Apesar do aprendizado de máquina conseguir aprender a partir de exemplos, esses exemplos fornecidos podem possuir algum tipo de erro ou estarem levemente

corrompidos. O que pode afetar em muito no processo de aprendizagem, pois a máquina poderá aprender a errar através dos dados inseridos incorretamente. Dessa forma é muito importante a limpeza, tratamento e verificação dos dados antes de realizar o aprendizado de máquina.

É necessário coletar os dados geográficos da via ou linha de ônibus a ser analisada, como por exemplo, as latitudes e longitudes de cada vértice de toda a extensão do objeto de estudo são necessárias para fazer os cálculos de deslocamento dos ônibus corretamente. E devem ser coletadas as geolocalizações dos pontos de parada (posição na via onde o ônibus para) ou as geolocalizações dos pontos de ônibus (posição na calçada onde os passageiros esperam os ônibus).

Esses dados geográficos podem ser coletados manualmente, com um receptor de sinal GPS (os *smartphones* possuem essa antena), ou, podem ser coletados através de algum órgão público da cidade a ser estudada (exemplo: GeoSampa), ou através da agência provedora do transporte público da cidade (GTFS). Nos próximos parágrafos será apresentada uma metodologia para a análise de dados geográficos provenientes do GTFS, por ser mais comum para este tipo de pesquisa.

Para poder analisar o horário que os ônibus passam pelos pontos de parada, será necessário fazer alguns ajustes nos dados do GTFS. Por exemplo, os dados de posição dos pontos de parada devem ser inseridos no arquivo *shape* do GTFS, pois este contém apenas a trajetória geográfica da linha, sendo que a informação dos pontos de parada está em outros dois arquivos: *Stops* e *Stops_times*.

A informação geográfica dos pontos de paradas não interpola a trajetória do itinerário, ou seja, a trajetória é pela via e o ponto de ônibus fica na calçada. A simples inserção dos pontos de paradas no *shape* não pode ser realizado, pois os ônibus não saem de sua trajetória para embarques ou desembarques.

O traço azul da Figura 12 representa a trajetória do ônibus, enquanto o “balão” em vermelho indica a localização do ponto de ônibus. Nota-se que a localização do ponto de parada fica ligeiramente afastado da trajetória, ou seja, na calçada.

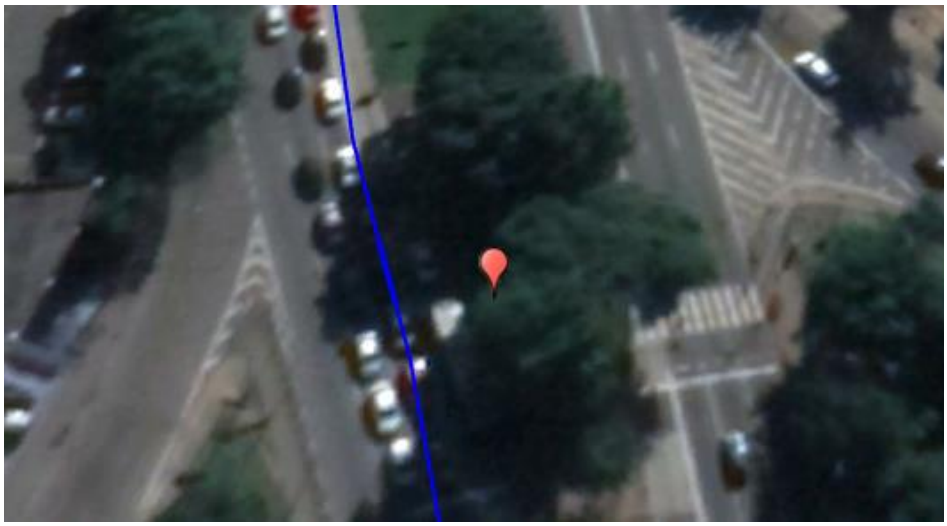


Figura 12 – Exemplo da Visualização da Rota de uma Linha de Ônibus (Linha em Azul) e a Localização de um Ponto de Parada (“Balão” em Vermelho)

Fonte: Autor, elaborado utilizando Google Earth

Por esse motivo, os pontos de parada devem ser ajustados no trajeto das linhas de ônibus para melhor adaptação de referência da localização do ônibus. Pontos de ônibus duplicados devem ser removidos.

A Figura 13 mostra um exemplo de alteração adotada. A localização do ponto de ônibus na calçada foi alterada para a localização de onde o ônibus ficaria parado durante o embarque de passageiros. Ao passo que a Figura 14 traz um exemplo de correção para o caso de um ponto de ônibus duplicado no arquivo GTFS.

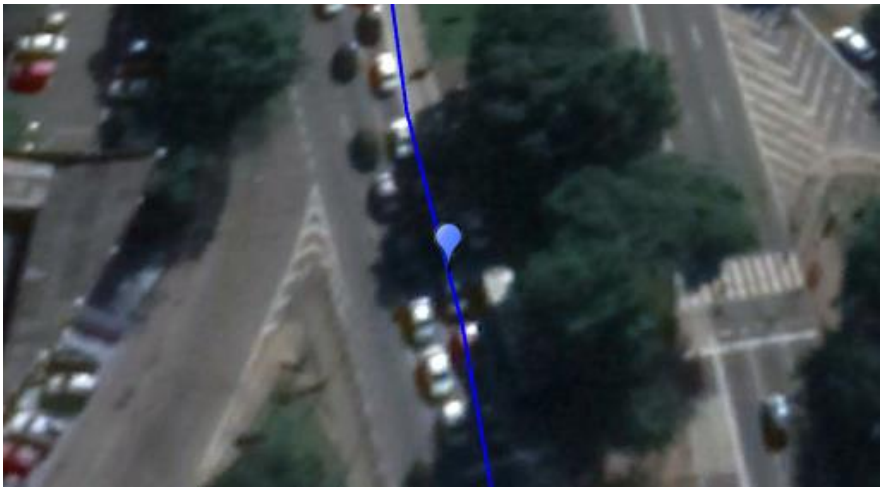


Figura 13 – Exemplo de Alteração do Ponto de Parada para Melhorar a Análise Posterior

Fonte: Autor, elaborado utilizando Google Earth

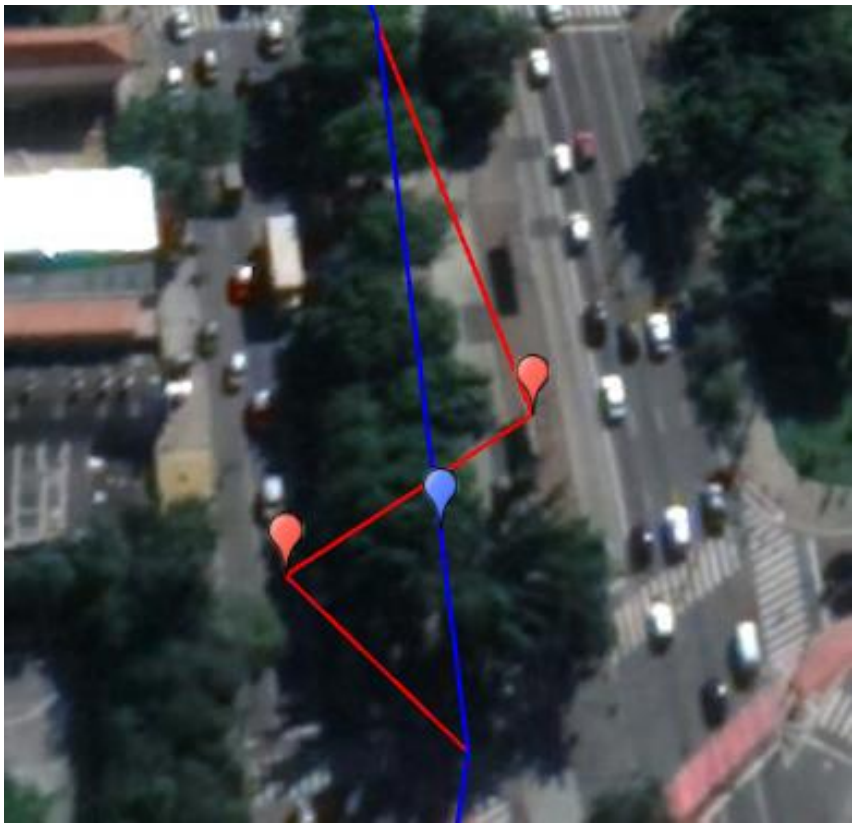


Figura 14 – Exemplo de Ponto de Parada Duplicado

Fonte: Autor, elaborado no Google Earth

Dessa forma impede que o ônibus que esteja na linha normal tenha interpretação de localização errada. Para correção da localização dos pontos de parada deve ser utilizado o seguinte procedimento:

1. Calcular a distância do ponto de ônibus para cada aresta do *shape* utilizando a fórmula (7) de Haversine (SINNOTT, 1984) :

$$d = 2R_{Terra} \times ArcSen\sqrt{Sen^2\left(\frac{\pi(\Delta Lat)}{90}\right) + Cos\left(\frac{\pi Lat_1}{90}\right)Cos\left(\frac{\pi Lat_2}{90}\right)Sen^2\left(\frac{\pi(\Delta Lon)}{90}\right)} \quad (7)$$

Onde d é a distância entre os dois pontos geodésicos, R_{Terra} é o raio da Terra, ΔLat é a diferença entre as duas latitudes, ΔLon é a diferença entre as longitudes, Lat_1 é a latitude do primeiro ponto e Lat_2 é a latitude do segundo ponto.

2. Verificar qual a aresta mais próxima utilizando a equação 8.

$$P = d_1 + d_2 - d_a \tag{8}$$

Onde P é a proximidade, d_1 é a distância do primeiro vértice do *shape*, d_2 é a distância do segundo vértice e d_a é a distância da aresta compreendida entre os dois vértices.

3. Calcular a razão entre as duas distancias com a equação 9, sendo r a razão, d_1 é a distância do primeiro vértice, d_2 é a distância do segundo vértice.

$$r = \frac{d_1}{d_2} \tag{9}$$

4. Converter as coordenadas geodésicas em cartesianas com as equações 10, 11 e 12, sendo X, Y e Z as coordenadas compreendidas nos respectivos eixos cartesianos:

$$X = \mathrm{Cos}(Lat) \times \mathrm{Cos}(Lon) \tag{10}$$

$$Y = \mathrm{Cos}(Lat) \times Sen(Lon) \tag{11}$$

$$Z = Sen(Lat) \tag{12}$$

5. Calcular a média proporcional no plano cartesiano com as expressões 13, 14 e 15:

$$X_p = (X_1 \times r) + \big(X_2 \times (1 - r)\big) \tag{13}$$

$$Y_p = (Y_1 \times r) + \big(Y_2 \times (1 - r)\big) \tag{14}$$

$$Z_p = (Z_1 \times r) + \big(Z_2 \times (1 - r)\big) \tag{15}$$

6. Realizar a conversão do plano cartesiano de volta para coordenadas geodésicas utilizando as expressões 16, 17 e 18:

$$Hyp = \sqrt{X_p^2 + Y_p^2} \tag{16}$$

$$Lat_p = ATan2\left(Hyp\ , Z_p\right) \times \frac{180}{\pi} \tag{17}$$

$$Lon_p = ATan2\left(X_p\ , Y_p\right) \times \frac{180}{\pi} \tag{18}$$

3.3 Coleta de Dados

É possível coletar dados de geolocalização de ônibus através de APIs, conforme explicado nos itens 2.4.2 e 2.4.3. Para realizar a coleta ciclicamente nas APIs é necessário um computador conectado à internet e elaborar um programa que colete de forma automatizada os resultados das APIs. O programa ainda deve formatar de uma forma que facilite a utilização dos dados e reúna-os em um único arquivo. Recomenda-se a utilização de uma máquina virtual, conforme explorado no item 2.4.1, por oferecer uma estabilidade na conexão à internet e a rede elétrica.

Os dados de trânsito em tempo real, podem ser obtidos de forma semelhante, conforme exemplificado no item 2.4.5. Pode-se utilizar o mesmo computador, porém outro código ou programa precisa estar sendo executado em paralelo.

Os mesmos dados de trânsito e AVL podem ser coletados de outros provedores também. Isto pode contribuir para verificar qual provedor possui dados mais confiáveis, ou se a combinação desses dados semelhantes pode gerar um resultado ainda melhor.

Diferentes tipos de dados também podem ser coletados se estiverem disponíveis, como por exemplo: dados meteorológicos, bilhetagem ou carregamento do ônibus e médias históricas, pois pode ser que a RNA encontre alguma correlação entre esses novos dados para aprimorar a precisão dos tempos de viagem. Os dados poderão ser removidos caso afetem negativamente a RNA.

3.4 Tratamento dos Dados

Após a coleta, é necessário que os dados sejam limpos antes de serem interpretados, uma vez que o sistema atual de GPS possui uma tolerância em seu posicionamento. Afinal, existe a possibilidade de um ônibus que estiver parado enviar dados de deslocamento dentro de um determinado raio dependendo do tempo que o veículo ficar parado.

Como os dados de AVL são transmitidos ciclicamente pelos veículos e por conta da tolerância do GPS, os dados do AVL quase nunca corresponderão exatamente com um dos vértices dos dados geográficos. Para que seja obtida a distância percorrida de um determinado AVL, é necessário interpolar a posição GPS do AVL com os vértices mais próximos, seguindo o mesmo procedimento do item 3.2.

É possível acontecer erros na coleta do sinal GPS por conta da recepção da antena nos veículos, zonas de sombra, sinal refletido do satélite GPS. Além disso, o motorista pode esquecer de desligar o AVL quando estiver retornando para o início da linha ou parado nos pontos iniciais e finais esperando uma próxima partida. Por isso, os dados devem ser preparados para a análise, os dados duplicados ou com erros devem ser excluídos e deve-se analisar se a trajetória de cada ônibus está dentro do conforme (velocidade instantânea, progresso da viagem, tempo parado nos semáforos ou pontos de parada).

3.5 Visualização dos Dados

Para verificar se a forma de tratamento está adequada e reanalisar os dados tratados, sugere-se realizar uma análise visual dos deslocamentos dos ônibus. Com base nos horários e nas posições GPS de cada ônibus, pode-se simular os dados em um programa SIG (Sistema de Informação Geográfica) ou em um mapa sinótico, para verificar que os dados estão condizentes com uma operação normal de um ônibus.

Ao visualizar esses os dados dos AVL, pode-se e identificar se algum veículo apresenta algum comportamento anormal (desaparecer no meio da viagem, deslocamento no sentido contrário a linha, grandes tempos de espera nos terminais, velocidades absurdas ou teletransporte). Caso alguma anomalia seja detectada, deve-se voltar ao item anterior e identificar o problema na limpeza e tratamento dos dados.

3.6 Predição Estatística

Após obter os dados limpos e tratados, pode-se aplicar as técnicas estatísticas de predição. Dentre os possíveis métodos estatísticos vistos no item 2.3.1, o método Naïve é um dos mais simples e amplamente utilizado por diversos pesquisadores para realizar previsões de tempo de viagem de ônibus.

O método Naïve utiliza apenas o último resultado de tempo de viagem para realizar a predição. Portanto é necessário estimar os tempos de viagem dos ônibus para poder utilizar esse método. Dependendo da frequência em que os dados de AVL são coletados ou atualizados na API, pode ser necessário interpolar os horários de passagem pelos pontos de parada.

3.7 Conversão de Dados

Com todo tratamento e limpeza dos dados prontos, pode-se analisar cada viagem registrada por cada AVL em particular. Dessa forma, pode-se obter informações precisas das partidas e tempos de viagem de cada AVL em cada ponto de parada. Consequentemente, pode-se descobrir os tempos total de viagem, e repetir o mesmo processo para todas as viagens de cada AVL para cada um dos dias a serem analisados.

Como esses dados serão utilizados para treinar uma RNA, os dados devem estar formatados de uma forma que o algoritmo de RNA possa interpretar de maneira correta e ser utilizado posteriormente para a previsão. Dessa forma, sugere-se que os dados sejam convertidos para a seguinte forma:

- A data da coleta deve ser substituída pelo dia da semana, conforme criticidade do tráfego 0=Domingo, 1=Sábado e 2 a 6 sendo Segunda-feira a Sexta-feira;
- Inserir uma nova variável *dummy* para indicar se naquele dia foi um domingo ou feriado (0), se não foi feriado (1), e para o Sábado terá valor (0,5);
- Converter os horários de partida em minutos do dia;
- Converter os tempos de viagem para cada trecho para segundos;
- Converter o tempo total de trajeto para segundos;
- Verificar a necessidade de adaptações nos outros dados coletados.

Após a conversão, os dados devem ser concatenados em um único arquivo (“.txt” ou “.csv”) para abastecer a RNA.

3.8 Desenvolvimento da Rede Neural Artificial

Com os dados tratados em mãos, deve-se escolher um programa para desenvolver a RNA. De preferência, um programa ou um ambiente que possa ler e normalizar o conjunto de dados, construir um modelo de RNA, gerar gráficos, calcular e salvar as métricas de precisão da previsão.

Os novos algoritmos de RNA estão adaptados a realizar o processamento com GPU (*Graphics Processing Unit*). Portanto, recomenda-se que seja verificado se o equipamento que realizará o treino da RNA possui este recurso e as devidas configurações para poder utilizá-lo.

3.9 Treino da Rede Neural Artificial

Diferentes conjuntos de dados podem ser inseridos na RNA, dessa forma, pode-se descobrir qual combinação de conjunto de dados é mais relevante para o treino da RNA e qual combinação gera a previsão otimizada. Pode-se alternar entre si diferentes hiperparâmetros, funções de ativações e otimizadores da RNA, com a finalidade de testar diferentes modelos de RNA, e comparar os diferentes resultados de cada uma delas.

Conforme explicado no item 2.3.2, deve-se configurar as funções de ativação na RNA. Por conta que a maioria dessas funções retornam valores entre 0 e 1 ou entre -1 e 1, elas são muito influenciadas pelo valor escalar que recebem de entrada. Isso pode gerar um problema para a RNA para o caso de alguma entrada possuir um valor muito discrepante das demais entrada, pois isso faz com que as entradas com valores baixos sejam ignoradas e dificulta o processo de aprendizagem e corre o risco da RNA não conseguir corrigir os pesos das entradas.

Para diminuir esse valor escalar e resolver o problema normalização existem algumas formas de se normalizar os dados, as mais utilizadas são a Padronização (Fórmula 19, sendo X o dado a ser normalizado, μ é a média e σ é o desvio padrão) e a Normalização (Fórmula 20). A Normalização redimensiona os valores em um intervalo entre 0 e 1. A Padronização redimensiona os dados para ter uma média igual a 0 e um desvio padrão igual a 1. Ambos são

utilizados para facilitar o treino da RNA, fazendo com que o erro convirja mais rápido, e para que os resultados das previsões sejam melhores.

$$z = \frac{X - \mu}{\sigma} \tag{19}$$

$$X_n = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{20}$$

Deve-se avaliar a forma de aprendizado e como o erro está evoluindo a cada iteração para evitar o sobre ajuste ou verificar a convergência do erro, caso seja necessário deve-se utilizar as funções de parada antecipada ou um número máximo de iterações. Dependendo do tamanho do banco de dados gerado e das características dos dados deve-se avaliar a necessidade de utilização da validação cruzada.

A recomendação para a fração entre treino e teste é de aproximadamente 80% para treino e 20% para teste. Esta razão de 80/20 é popularmente utilizada e se originou através do Princípio de Pareto. Essa separação deve ser realizada sendo as primeiras 80% das amostras para treino e as últimas 20% para teste, uma vez que o problema estudado é uma série temporal.

Se amostras aleatórias forem selecionadas para o teste do modelo, pode ocorrer o vazamento (*leaking*) de informações. É como se fosse realizar um teste de previsão de tempo de viagem em um modelo computacional que já possui informações do passado e do futuro da amostra que se quer prever.

Amostras aleatórias para teste poderiam ser utilizadas se o objetivo fosse realizar uma estimativa de tempo de viagem de um ônibus que teve dados de viagem corrompidos ou interrompidos. Nesse caso pode-se separar um determinado período e informar os dados de viagens de ônibus para realizar as estimativas de viagem dentro daquele período.

Como o objetivo desse estudo é realizar previsões de viagens que ainda não ocorreram, a RNA deve ser treinada com dados em forma sequencial e testada em um período posterior aos dados de treino.

3.10 Considerações Finais sobre a Metodologia

Neste capítulo foi documentado qual o conceito que esta metodologia carrega e todos os procedimentos necessários para poder realizá-la. Foi ilustrado como realizar a coleta automática de dados, limpeza, tratamento e visualização para compor um *dataset*. Foi visto como prever os tempos de viagem de ônibus através do método Naïve. E o que compreende o desenvolvimento e treino de uma RNA.

Em seguida, será apresentado um exemplo da aplicação desta metodologia proposta, com a finalidade de testar sua validade. Uma vez que, dependendo da diferença entre os conjuntos de dados ou formações diferentes, adaptações devem ser levadas em consideração.

4 EXPERIMENTOS E RESULTADOS

4.1 Metas e Resultados Desejados

Dentre as possíveis métricas de análise e alvos para este trabalho, as principais metas para a metodologia proposta consistem em:

- Propor exemplos de coleta e tratamento de dados para a metodologia proposta;
- Verificar a existência de alguma correlação entre os dados da SPTrans e Google Maps para a mesma via;
- Verificar se é possível treinar uma RNA a partir de dados de trânsito em tempo real, se o treino da RNA irá convergir com os dados informados;
- Verificar se há algum incremento na precisão de previsão de tempos de viagem quando dados de trânsito em tempo real estão disponíveis;
- Analisar quais entradas para RNA são mais relevantes;
- Analisar resultados para diferentes hiperparâmetros da RNA.

Para alcançar esses objetivos, será otimizado o Erro Absoluto Médio Percentual (MAPE), por ser a medida de precisão adotada pela maioria das bibliografias consultada. Dessa forma, será possível comparar e discutir sobre os resultados de outros pesquisadores.

4.2 Materiais e Ferramentas

Nesta seção será listado a relação de materiais e ferramentas necessários para a elaboração desse experimento. Para todos os itens listados a seguir existem recursos opcionais que realizam a mesma função ou semelhantes, dependendo da cidade escolhida. Os itens utilizados estão listados a seguir:

- Mapa do sistema viário da cidade de São Paulo incluindo corredores e faixas exclusivas de ônibus (GeoSampa);
- Informações do sistema público de transporte da cidade de São Paulo (GTFS SPTrans);
- Dados de geolocalização dos ônibus da linha 4708-10-0 do mês de março de 2019 (API OlhoVivo da SPTrans);

- Dados de velocidade média por hora dos trechos por onde a linha 4708-10-0 passa do mês de fevereiro de 2019 (Scipopulis);
- Dados em tempo real e de previsão de tempo de trajeto dos trechos por onde a linha 4708-10-0 passa do mês de março de 2019 (Google Maps API);
- Dados meteorológicos da cidade de São Paulo do mês de março de 2019 (INMET);
- Planilha eletrônica (Microsoft Excel e Google Fusion Table);
- Software de programação (VBA, R Studio e Notepad++);
- Sincronização de banco de dados (OneDrive);
- Máquina Virtual (GCP - Google Cloud Platform);
- Ferramentas SIG (QGIS, Google Earth e Google MyMaps);
- Ambiente de Programação Python (Anaconda, PyCharm);
- Bibliotecas Python para desenvolvimento de Redes Neurais Artificiais (Keras versão 2.3.1 e TensorFlow versão 2.1.0).

4.3 Escolha da Linha e Ajustes

Primeiramente uma linha de ônibus deve ser selecionada. O primeiro ponto importante a ser levado em consideração para a escolha da linha é não ter faixa exclusiva ou corredor exclusivo de ônibus em nenhuma parte da linha. Isto possibilitará que os dados estejam mais correlacionados, pois como a base da previsão de tráfego que a API do Google Maps fornece provém dos carros, é imprescindível que os ônibus a serem analisados circulem na mesma via que esses carros. Lembrando que caso tivesse sido coletado dados de trânsito de táxis que trafegam em vias com faixa exclusiva de ônibus, poderia se escolher linhas de ônibus que trafegam sobre elas.

Portanto, para a escolha da linha utilizou-se o GTFS da SPTrans conforme Figura 15, disponível na página de desenvolvedores da SPTrans, (SPTRANS, 2019b), com este conjunto de arquivos, é possível saber as trajetórias de cada linha. Com a finalidade de selecionar uma linha de comprimento médio, de 5 a 10 km de extensão, foi utilizado o arquivo *shapes* do GTFS, que contém a distância percorrida de cada identificador, para eliminar linhas com extensões maiores que 10 km.

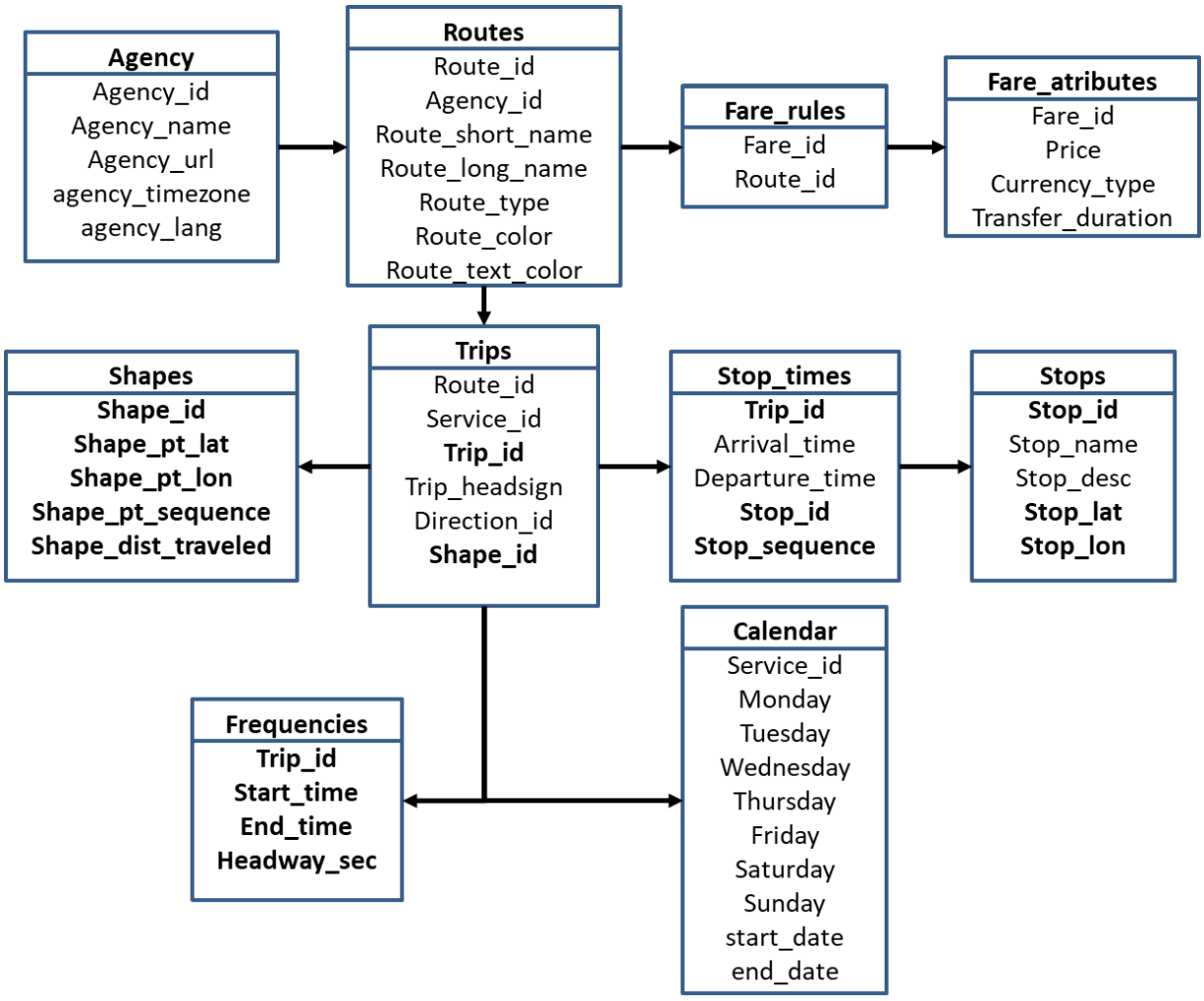


Figura 15 – Estrutura do GTFS da SPTrans, com Principais Informações em Negrito

Fonte: Autor, adaptado a partir de (SPTRANS, 2019b)

Em seguida, foi coletado os dados geográficos das vias com faixas ou corredores exclusivos de ônibus, provenientes do GeoSampa, (SÃO PAULO, 2020). Os arquivos *shapefile* dos corredores e faixas exclusivas foram baixados do GeoSampa e convertidos para KML utilizando o programa QGIS. O arquivo *shape* do GTFS também foi convertido em KML. Com os arquivos KML criados, foi possível acessá-los no Google Earth e no QGIS para uma visualização completa e possibilitando a seleção de linhas de forma visual e detalhada.

Conforme ilustrado na Figura 16, os traços em verde correspondem às linhas com menos de 10 km de extensão. Enquanto os traços em vermelho são as faixas exclusivas de ônibus e os traços preto são os corredores de ônibus.

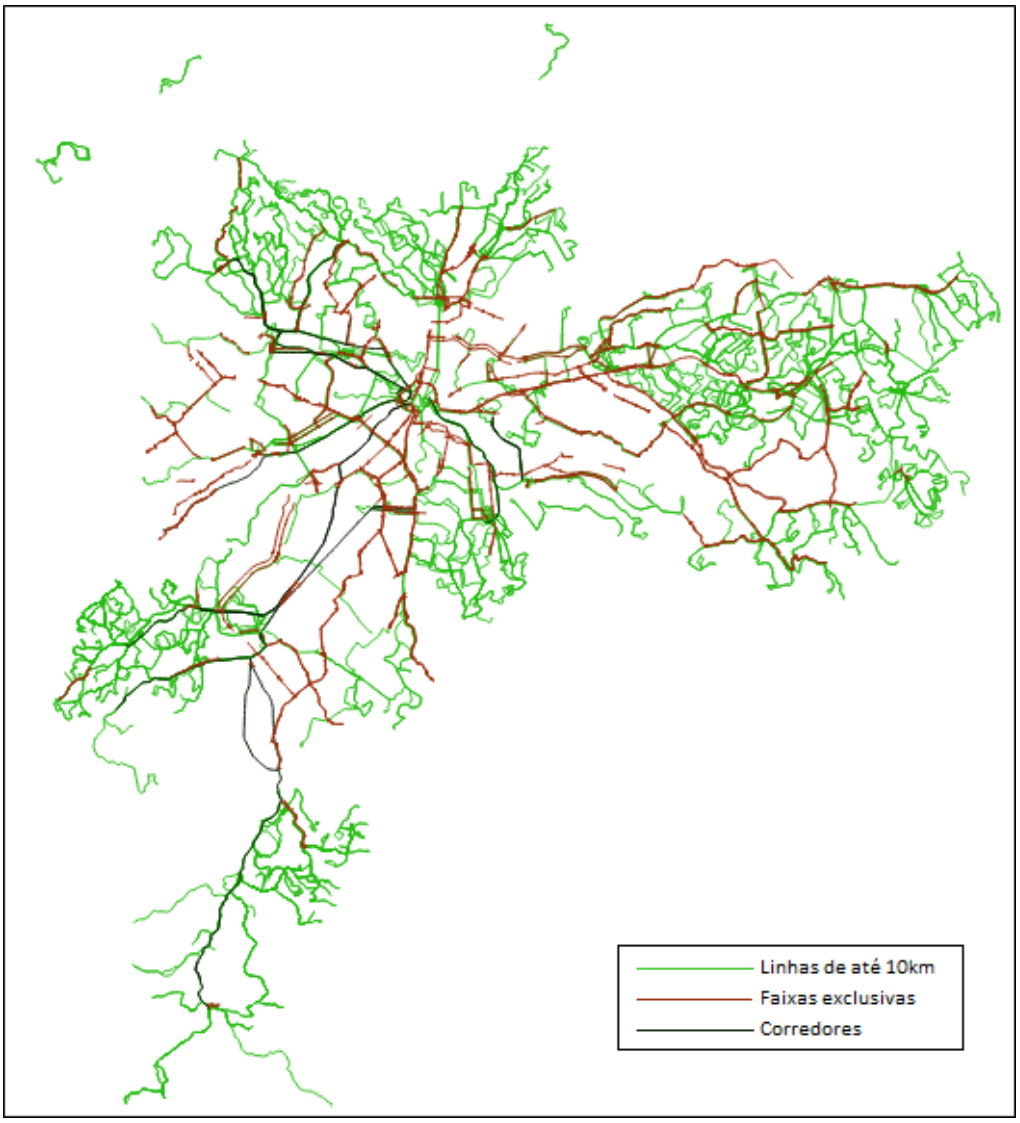


Figura 16 – Visualização dos KMLs no QGIS

Fonte: Autor, elaborado no QGIS

Em seguida, foi escolhida uma linha de ônibus com a ajuda do Google Maps na visão de trânsito típico em diferentes horários e dias da semana, para que os efeitos do trânsito possam ser mais bem observados. Essa ferramenta fornece uma visão gráfica do comportamento do trânsito para as vias exibidas no mapa digital. Ela exibe a frequência e intensidade do trânsito para cada faixa horária selecionada. Uma vez que os dados serão extraídos do próprio Google Maps, a visão de trânsito típico é bem representativa quanto a coleta de dados pela mesma plataforma.

Esta visualização de trânsito típico pode ser observada na Figura 17, onde pode-se selecionar o dia da semana e avançar ou retroagir o horário para verificar as cores de cada via.

Sendo a coloração verde para a via com trânsito livre e a coloração vermelho escuro para a via com trânsito mais carregado.

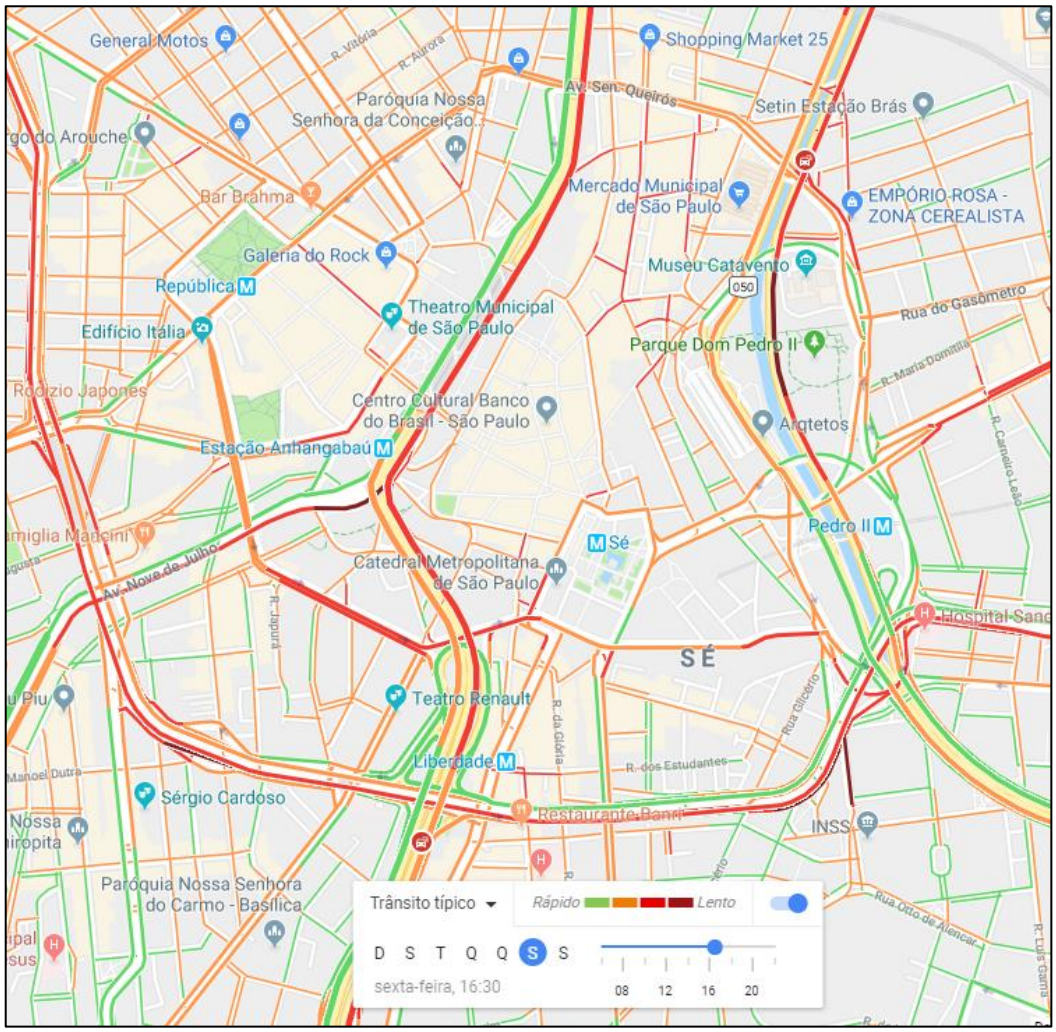


Figura 17 – Google Maps com Visualização de Trânsito Típico de Sexta-Feira as 17h

Fonte: Autor, elaborado utilizando Google Maps

Enfim, a linha 4708-10-0 (METRÔ VL. MARIANA) com saída do Jardim Clímax e destino no metrô da vila Mariana foi escolhida para a análise, ilustrada na Figura 18. Esta linha possui 8,6 km de extensão e 34 pontos de ônibus intermediários (excluindo o terminal inicial e final), e ao longo da trajetória da linha não há nenhum corredor ou faixa preferencial de ônibus na via. Em média o tempo total de viagem dessa linha é de 35 minutos.

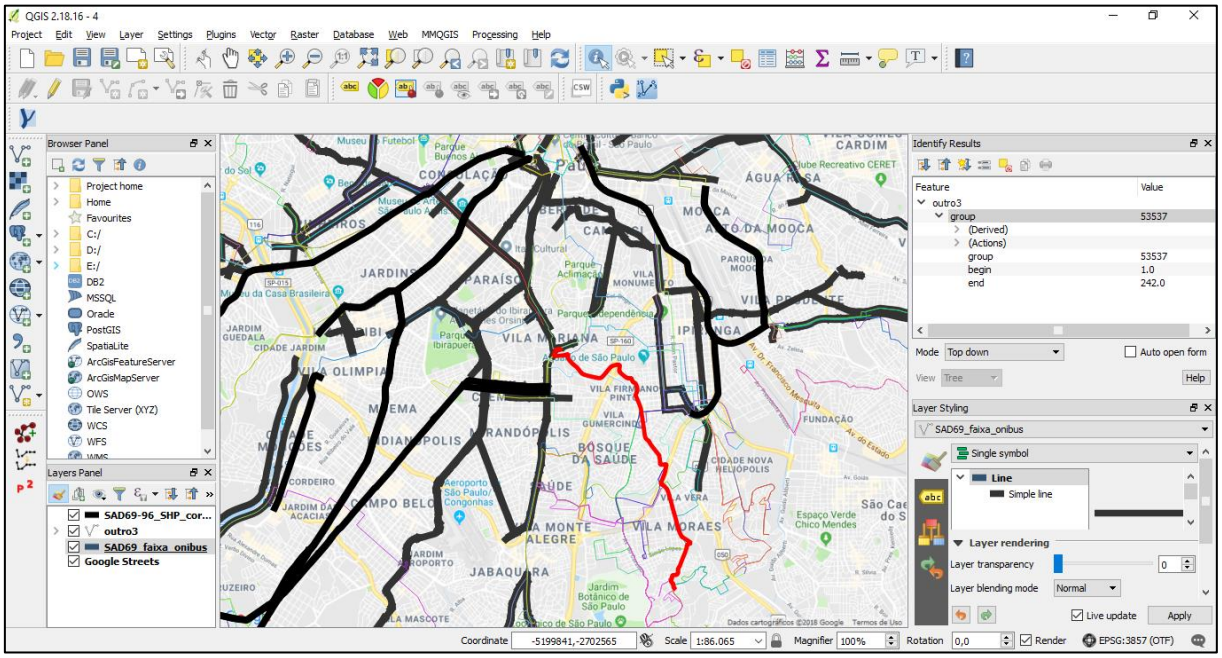


Figura 18 – Visualização das Linhas de Ônibus da SPTrans (Linhas coloridas) e dos Corredores ou Faixas Exclusivas de Ônibus (Linhas Pretas) e Linha Selecionada (Linha Vermelha) no QGIS e Google Maps

Fonte: Autor, elaborado utilizando QGIS com plugin do Google Maps

Após a escolha, o primeiro ajuste a realizar é fazer com que os pontos de ônibus estejam alinhados com os vértices do GTFS, conforme explicado na metodologia no item 3.2. A Figura 19 exemplifica esse processo.

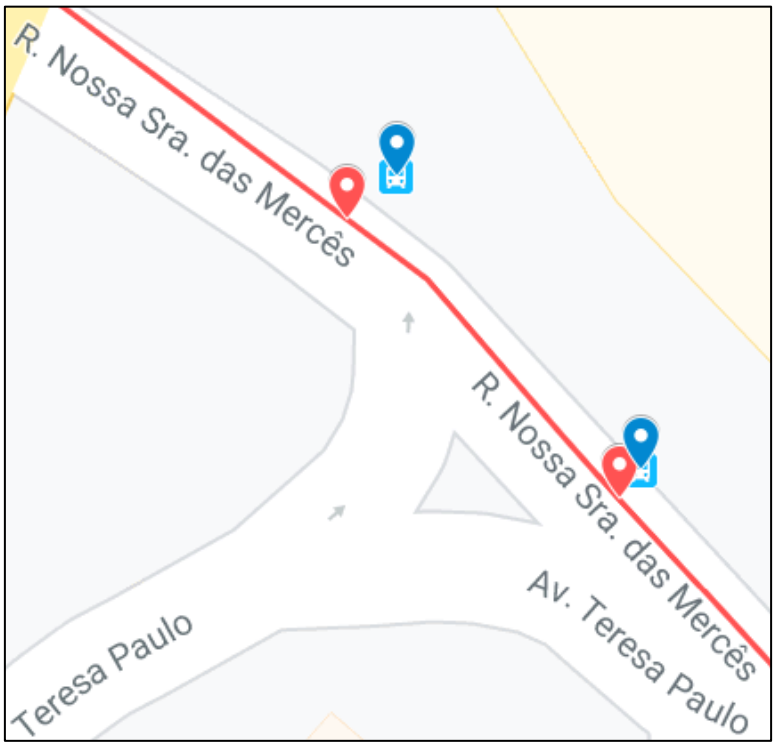


Figura 19 – Ajuste da Geolocalização dos Pontos de Ônibus (Balão Azul) para Posição na Via do Ponto de Parada do Ônibus (Balão Vermelho)

Fonte: Autor, elaborado utilizando Google MyMaps

Enfim, a Figura 20 sumariza a linha escolhida com todos os seus pontos de parada ajustados. Sendo a linha vermelha a trajetória do ônibus e os pontos vermelhos são os pontos de parada.

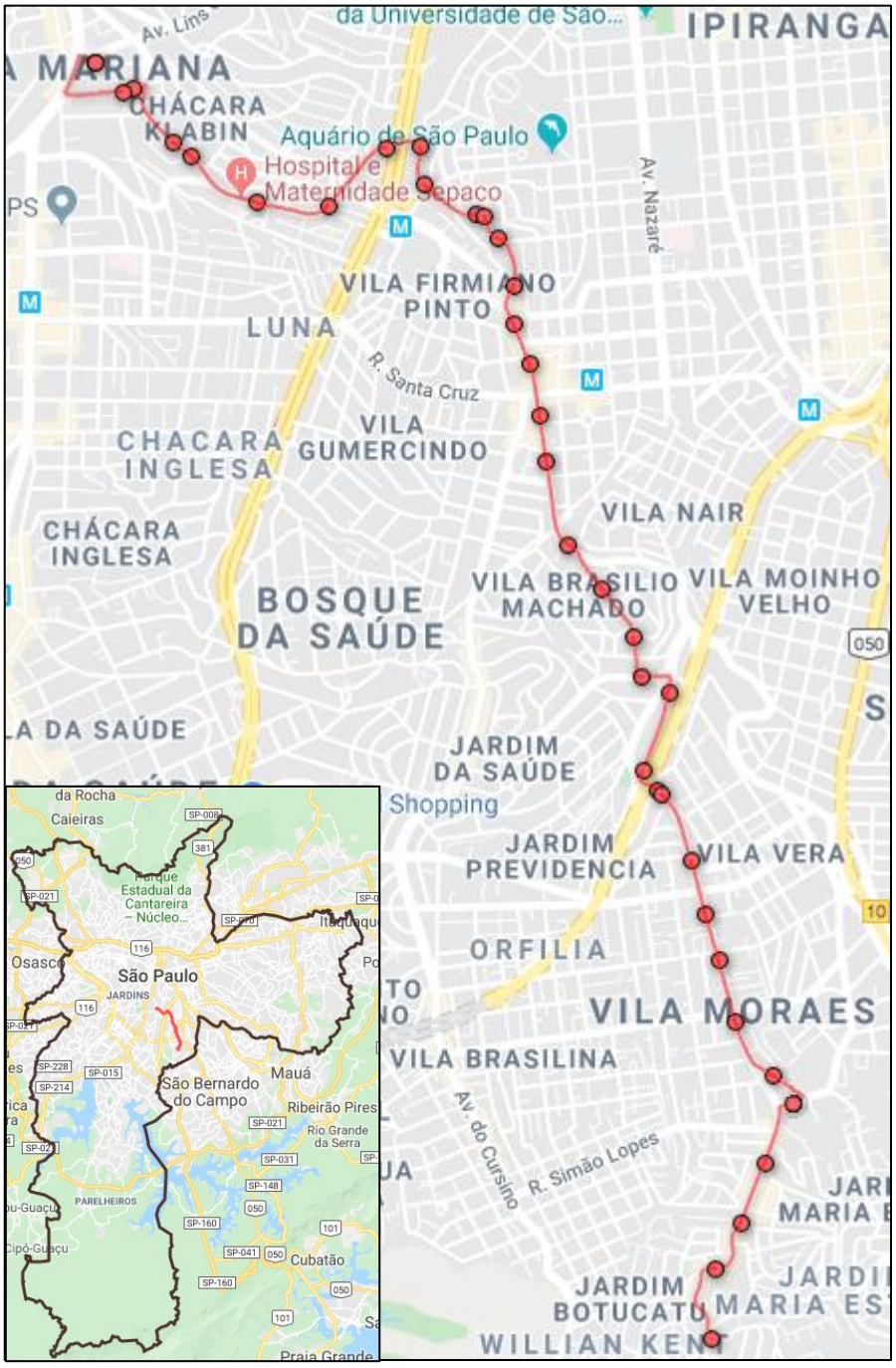


Figura 20 – Linha Escolhida, 4708-10-0 (METRÔ VL. MARIANA)

Fonte: Autor, elaborado utilizando Google Fusion Tables e Google MyMaps

Com o objeto de estudo definido, será iniciado a preparação para coleta dos dados dinâmicos de AVL e trânsito em tempo real para criação do banco de dados que será a base de aprendizado da RNA.

4.4 Instanciando e Configurando a Máquina Virtual

A computação na nuvem pode trazer grandes vantagens em estabilidade, disponibilidade do equipamento e seu baixo custo frente a popularização do *PaaS*, conforme explicado no item 2.4.1.

Uma máquina virtual foi instanciada usando a plataforma GCP, por conta da maior flexibilidade ao configurar o equipamento remoto, menor custo por hora (debitado do valor gratuito), maior tamanho de disco rígido para armazenamento dos dados e maior período de permanência no nível gratuito.

Ao instanciar a máquina virtual, ela ficou disponível para acesso remoto via RDP (*Remote Desktop Protocol*) em poucos minutos. Em seguida, foi necessário apenas instalar os programas que realizaram a coleta de dados via API (R Studio) e sincronização do banco de dados (OneDrive).

4.5 Coleta de Dados de AVL e Tráfego

Antes de iniciar a coleta foi necessário criar chaves das APIs, tanto para a API da SPTrans, como para a do Google Maps. Sendo que a forma de autenticação das chaves nas duas plataformas é diferente. A API da SPTrans requer uma postagem (POST) da chave de acesso antes de realizar as múltiplas solicitações (GET). Enquanto a API do Google Maps requer que a chave seja enviada como parâmetro de API em todas as solicitações (GET).

Um código no R Studio foi elaborado para fazer as solicitações nas APIs, coletar automaticamente os dados dos AVL da SPTrans ciclicamente a cada 30 segundos e realizar a tabulação dos dados recebidos (de JSON para CSV). Foi utilizada a API que retorna uma lista com todos os veículos de uma determinada linha com suas devidas posições geodésicas, pois apenas uma linha foi selecionada para ser estudada.

A API Google Maps *directions,* disponível dentro da plataforma GCP, foi utilizada para a coleta dos dados de trânsito. Sendo empregada a mesma máquina virtual, porém com mais dois códigos em R. O primeiro código foi utilizado para coleta do tempo de trajeto para cada trecho entre dois pontos de parada, sendo que a coleta foi realizada em tempo real. O outro código foi usado para a coleta do tempo total de trajeto com partidas futuras, com antecipação máxima de 20 minutos e mínima de 5 minutos, variando 5 minutos entre eles. O modelo de

previsão de tráfego do Google torna-se cada vez mais influenciado pelo modelo atual à medida que o horário de partida se aproxima do horário atual.

Tentou-se coletar dados históricos através da plataforma, porém a API sempre respondia com mensagens de erro, portanto é impossível a coleta de dados históricos através da API do Google Maps. Dentre os três possíveis modelos de tráfego que a API do Google Maps pode informar serão coletados nos modelos “Melhor Estimativa” e “Pessimista”, por compreender da melhor representação que o modelo padrão fornece quanto aos automóveis na via e o pessimista pode representar um modelo mais lento como é o caso dos ônibus.

No primeiro código para coleta dos dados do Google Maps, os parâmetros de origem e destino são variados conforme o trecho e informados na solicitação à API. No segundo código, a origem e destino são fixos, porém, possuem uma grande extensão, dando margem para a API escolher uma rota diferente do itinerário do ônibus. Na API é possível colocar parâmetros por onde o veículo deve passar para fixar a rota conforme os pontos de parada, independentemente da situação atual ou prevista. O único parâmetro alterado a cada solicitação é o horário de partida em segundos, em que é somado o valor de segundos desde as 00:00:00 de 1 de janeiro de 1970 UTC até o horário atual e então acrescentado o tempo da previsão futura.

Foi coletado dados de AVL, trânsito em tempo real e previsão de trânsito durante todo o mês de março de 2019. Estes dados foram sincronizados entre a máquina virtual e o equipamento que utilizará esses dados para a limpeza e o treino da RNA.

4.6 Coleta da Velocidade Média dos Ônibus no Trecho

Outros dados provenientes de outras fontes foram coletados. Foi concedido acesso a plataforma do Painel de Monitoramento de Ônibus da empresa Scipopulis, onde foi possível coletar a velocidade média comercial hora a hora dos ônibus em cada trecho da linha estudada.

Dentro da plataforma da Scipopulis, a linha estudada foi dividida entre os 35 trechos que compõem a linha completa. As velocidades médias para cada um desses trechos e a velocidade média da linha completa foram coletadas de todo o mês de fevereiro de 2019. Esses dados servirão como um histórico de velocidades médias, sendo que os dados são exportados no formato CSV.

Esses dados coletados da plataforma da Scipopulis poderiam ser obtidos através da coleta da geolocalização dos ônibus da API da SPTrans de fevereiro de 2019. Se esses dados de AVL fossem coletados direto da API, seria necessário aplicar alguns processos de limpeza

e tratamento dos dados para depois calcular a velocidade média. Ao coletar esses dados já tratados através da plataforma da Scipopulis traz uma vantagem em não precisar coletar os mesmos dados da API OlhoVivo e realizar a série de limpezas e cálculos para obter a velocidade média do mês anterior ao selecionado para estudo.

4.7 Coleta da Flutuação Horária das Viagens Diárias

Para representar uma ideia de volume de veículos motorizados circulando nas vias da cidade de São Paulo, e por consequência, gerar um reflexo no trânsito da cidade, foi coletado a flutuação das viagens ao longo do dia através da Pesquisa Origem Destino 2017 (METRÔ SÃO PAULO, 2019), conforme Figura 21. Os dados do gráfico foram tabulados em viagens iniciando por hora do dia.

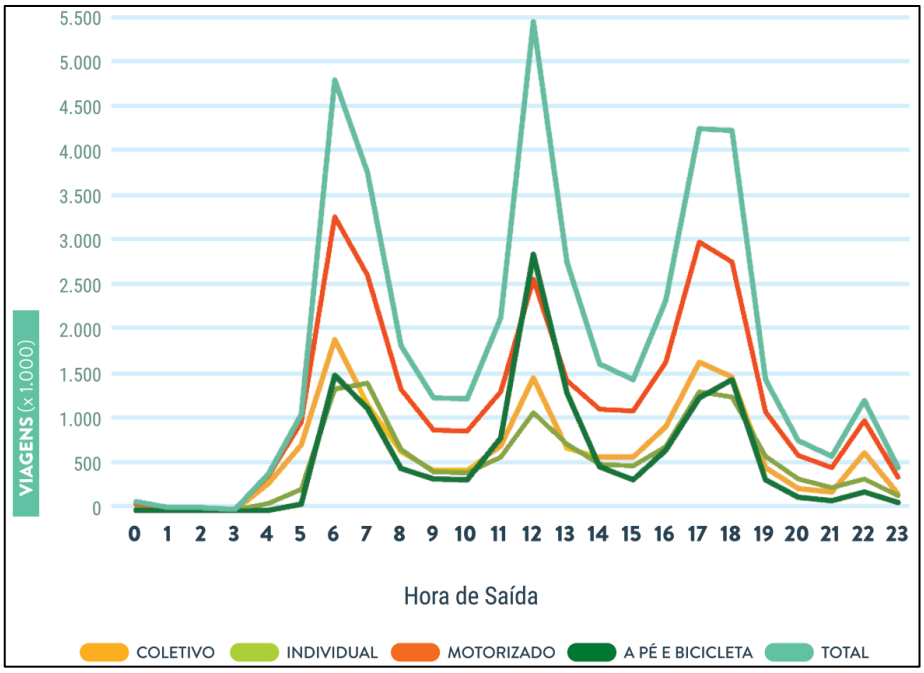


Figura 21 – Flutuação Horária das Viagens Diárias por Modo em 2017

Fonte: (METRÔ SÃO PAULO, 2019)

4.8 Coleta de Dados Meteorológicos

Conforme pesquisas anteriores, a precipitação altera parâmetros de operação do sistema urbano, e influencia no comportamento dos usuários (GONÇALVES, 2018). E pode ser que os outros dados meteorológicos estejam correlacionados com tempo total de trajeto. Uma vez que dependendo da temperatura ou velocidade do vento incentive as pessoas a utilizarem modelos de transporte privado, e por consequência, influencie no trânsito.

Portanto, dados meteorológicos foram coletados através do portal INMET (Instituto Nacional de Meteorologia), que foi utilizado em pesquisas cientificas anteriores (GONÇALVES, 2018). Os dados foram coletados de uma estação automática localizada no Mirante de Santana (Código A701), à 10 km de distância da linha selecionada. É possível coletar informações sobre temperatura, humidade do ar, velocidade do vento e volume de precipitação. O único ajuste necessário nessa base de dados foi converter os horários UTC para o horário local de São Paulo.

4.9 Classificação dos Dados de AVL

Os dados brutos obtidos da API da SPTrans foram: nome, número e letreiro da linha, número identificador do AVL, presença de acessibilidade do veículo, horário universal da última transmissão do ônibus, horário local da coleta na API, latitude e longitude do AVL.

Os dados foram exportados para o programa Excel, e o horário universal dos ônibus foram convertidas para o horário local. Os arquivos gerados em cada dia foram separados em uma pasta diferente para a correta interpretação e comparação com a especificação GTFS.

Para conseguir interpretar corretamente os dados coletados, uma planilha no Excel foi elaborada, onde foi possível verificar com precisão em qual aresta do *shape* cada dado da coleta está localizado, utilizando o mesmo procedimento descrito no item 3.2 deste capítulo, sendo criada uma formula no VBA do Excel para automatização desse cálculo.

E por fim, para facilitar o entendimento do deslocamento dos AVL, o número identificador do AVL foi ordenado de forma crescente. Então foi interpretada a posição do AVL relativo ao *shape*, bem como a distância do itinerário percorrido, a porcentagem percorrida em relação a distância final e a diferença de tempo entre da última atualização do AVL. E com isso, calculou-se as velocidades para cada atualização dos ônibus.

Para automatização do processo foi elaborado um algoritmo de limpeza e preparação dos dados. O procedimento de limpeza segue conforme diagrama lógico da Figura 22. Os outros dados coletados (trânsito em tempo real e previsão de trânsito, velocidades médias e dados meteorológicos) não necessitaram de nenhum tipo de limpeza.

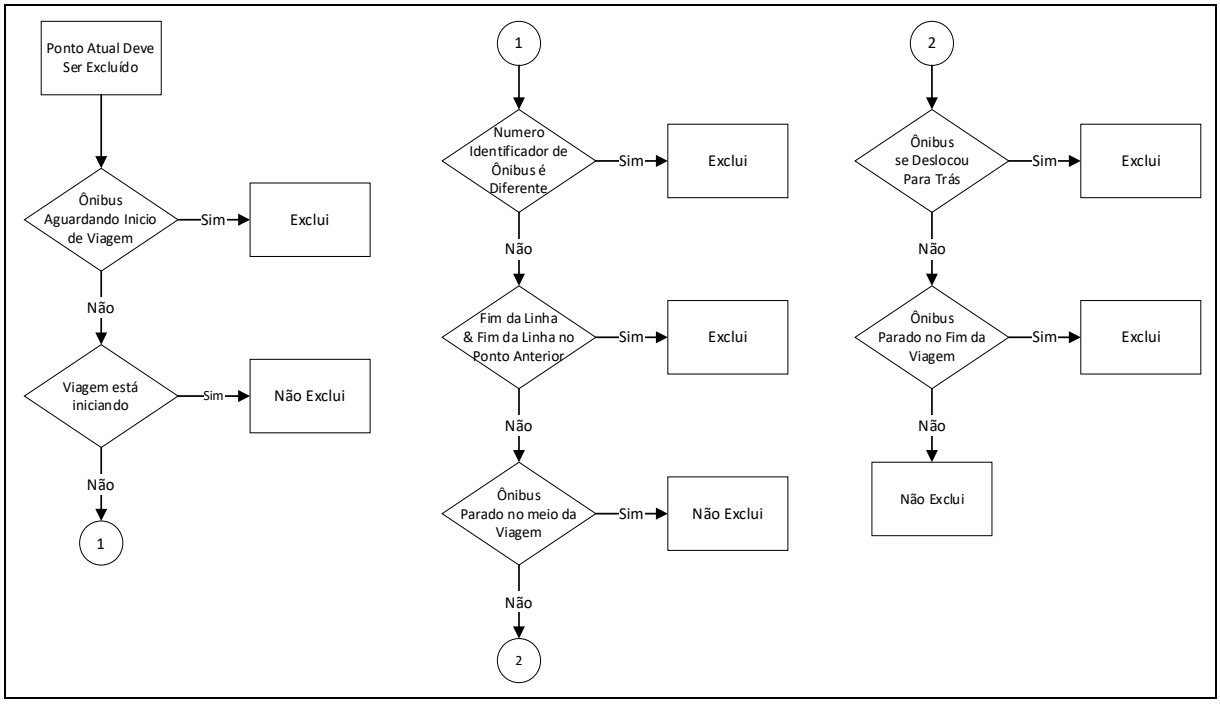


Figura 22 – Fluxograma de Limpeza dos Dados de AVL

Fonte: Autor, elaborado utilizando Visio

4.10 Visualização dos Dados de AVL

Após a limpeza dos dados de AVL, foi realizada uma visualização gráfica de todos os ônibus da linha específica para todo o período selecionado do estudo. Para isso, os dados de horário, latitude e longitude de cada ônibus foram inseridos em um único arquivo .GPX através de um código elaborado em VBA.

O programa Google Earth foi utilizado para poder observar o deslocamento ao longo do tempo de todos os ônibus da linha selecionada de todo o período de março de 2019, conforme Figura 23.

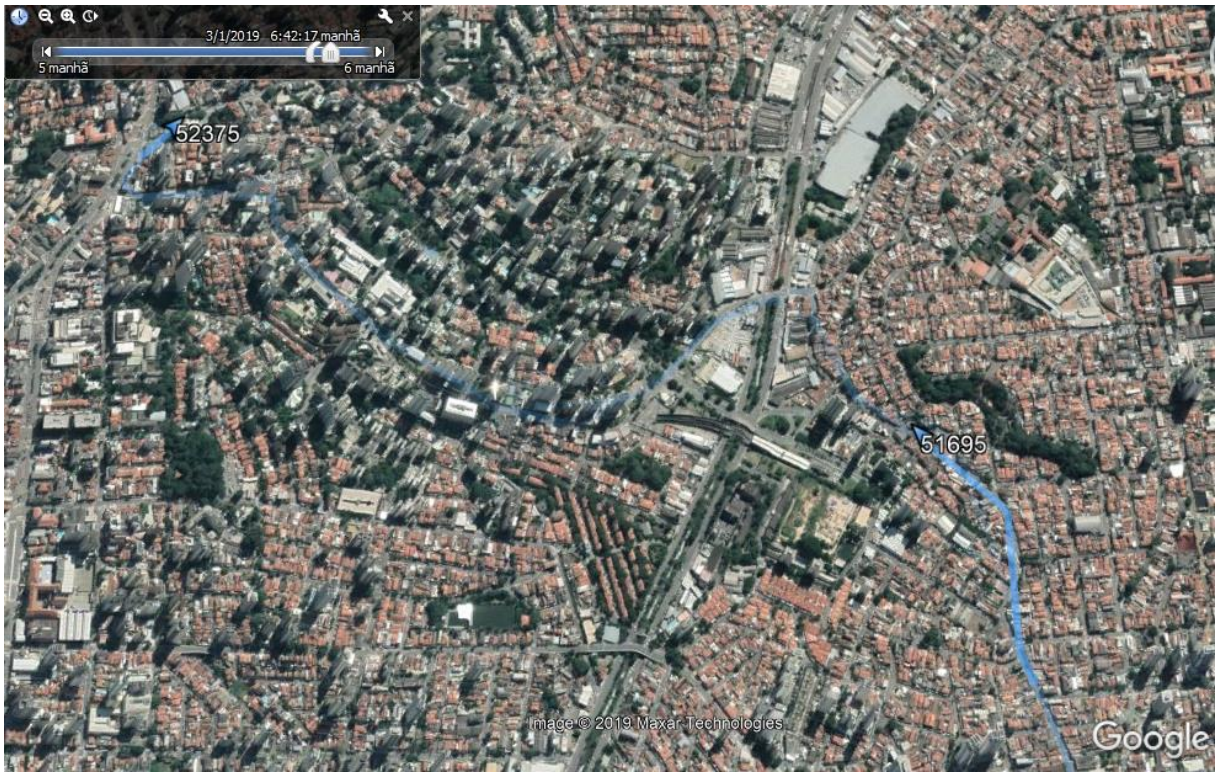


Figura 23 – Dados Tratados Sendo Reproduzido no Google Earth Pro, Triângulos como Ônibus e Linha Azul como Trajetória Recém Percorrida.
Fonte: Autor, elaborado utilizando Google Earth

Foi verificado se não houve nenhum comportamento estranho nos deslocamentos dos ônibus, como por exemplo: deslocamentos no sentido contrário a linha (mesmo por alguns segundos); tempo excessivo parado nos terminais (inicial ou final); velocidades muito alta ou alteração repentina no deslocamento; e ônibus operando fora de rota.

No total, foram verificadas 1570 viagens de ônibus, que é a somatória de viagens de ônibus operado pela linha estudada em todo o período de março de 2019. Dessa forma, validou-se o algoritmo de limpeza conforme explicado na metodologia do item 3.5.

4.11 Análise Estatística

Primeiramente foi analisado o comportamento da variável a ser prevista ao longo do tempo, conforme o gráfico da Figura 24. Onde é possível perceber a repetibilidade e frequência dos maiores tempos de trajeto (três picos diários).

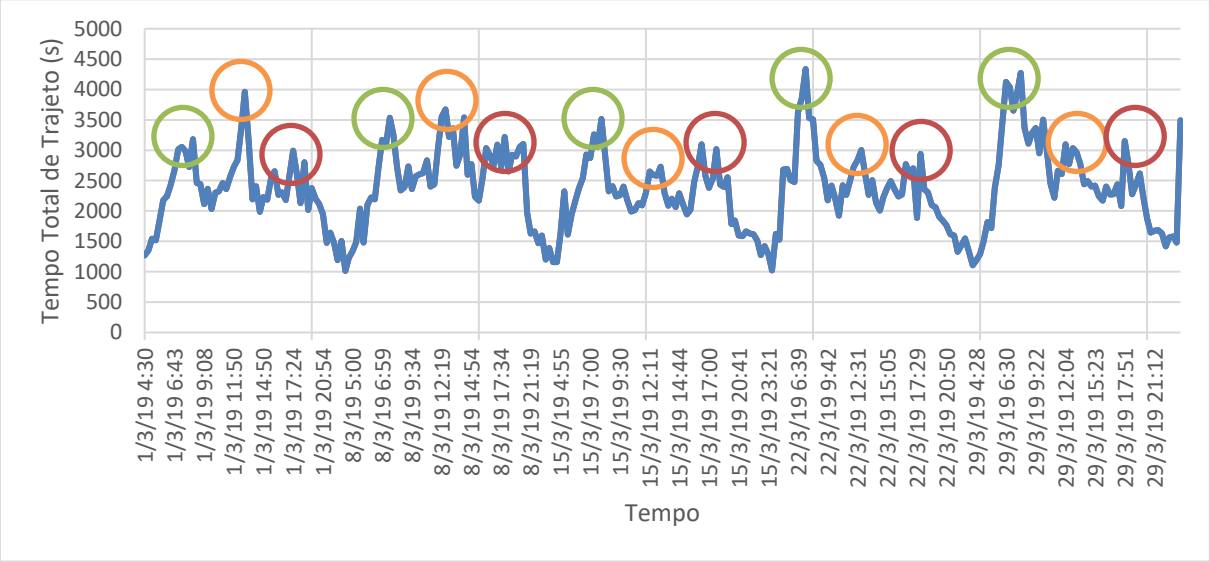


Figura 24 - Evolução do Tempo Total de Trajeto ao Longo do Tempo, Apenas Sextas-feiras, sendo os Círculos Verde o Pico da Manhã, os Círculos Laranja os Picos do Almoço e os Círculo Vermelho o Pico da Noite

Fonte: Autor, elaborado no Excel

Pode-se perceber que os picos e vales se repetindo na maioria dos dias no gráfico gerado possui um comportamento similar à flutuação horária das viagens apresentadas na Figura 21 (três picos ao longo de um dia). Porém, os maiores picos normalmente estão localizados na parte da manhã, e os picos do início da tarde e da noite não são tão altos. Isso faz sentido por se tratar de um trajeto apenas no sentido bairro-centro. Apesar de existir essa repetibilidade entre os picos de tempo total de trajeto, os picos possuem valores escalares diferentes. O tempo médio de viagem é de 35 minutos, e desvio padrão do tempo total de trajeto é de 612 segundos.

4.12 Predição Estatística

O ponto de partida para a melhora da precisão da predição do tempo total de trajeto será a comparação com a previsão ingênua, que entrega como previsão o último dado de tempo total de trajeto. Nesse caso, o erro MAPE foi calculado e o resultado foi de 12,89%, a raiz do erro quadrático médio obtido foi de 531 segundos.

Os erros para as médias móveis também foram calculados, considerando 2 e 3 dados anteriores, nesse caso, os erros MAPE foram de 12,20% e 12,60% respectivamente.

4.13 Concatenação do Dados

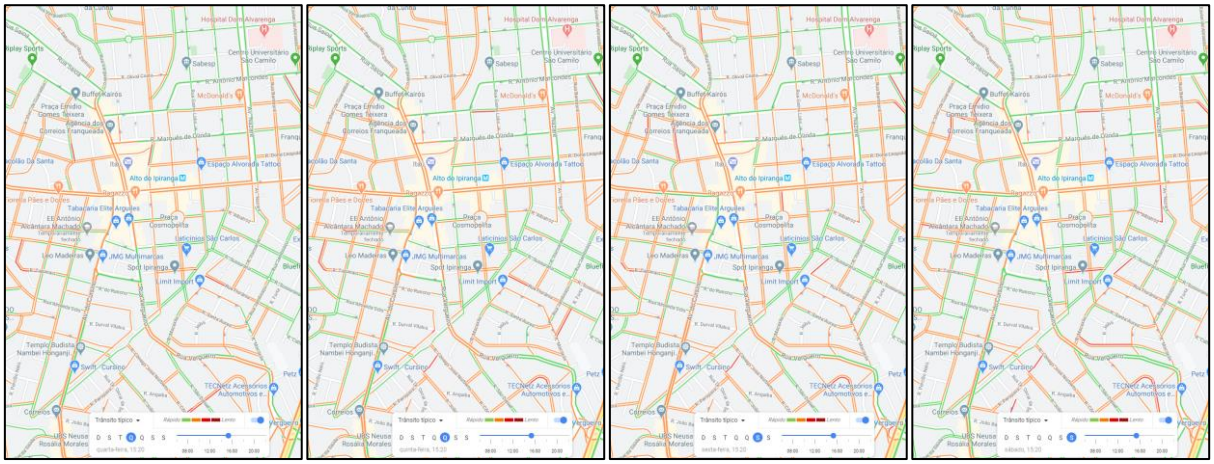
Em posse dos dados de AVL, trânsito do Google Maps, Velocidades Médias, Flutuação de Viagens Horária, Meteorológicos e da predição Naïve, é necessário concatená-los em um único arquivo e aplicar a devida conversão dos dados, conforme explicado no item 3.7.

Para a conversão da informação sobre os dias da semana para um formato numérico que possa representar de forma gradual a criticidade do tráfego/tempos de viagem dos ônibus, foi analisado a visão do trânsito típico do Google Maps em cada dia da semana em um mesmo horário (escolhido empiricamente). Conforme aumenta a quantidade de linhas laranja ou vermelhas no mapa do trânsito típico significa que mais crítico é aquele dia da semana em relação ao tráfego e por consequência ao tempo de viagem.

A visualização de apenas um trecho do que foi observado está ilustrado na Figura 25, onde o item (a) representa a trajetória da linha selecionada dentro do recorte do mapa e as figuras (b) a (h) ilustram o trânsito típico para dia da semana.

Conforme observado na Figura 25, pode-se perceber visualmente que o domingo é o dia com menos tráfego, por isso o valor *dummy* referente ao dia da semana deve ser 0, o próximo dia com menor tráfego é o sábado e seu valor *dummy* para dia da semana será 1. A sexta-feira apresentou mais itens laranja e vermelho dentro da trajetória da linha, por isso seu valor será 6. Entre os dias da semana compreendidos de segunda-feira à quinta-feira não foi possível observar alterações significativas nos mapas, e por esta razão serão os valores *dummy* de 2 á 5 respectivamente.

(a) Trajetória da linha (b) Trânsito Típico no Domingo (c) Trânsito Típico na Segunda-feira (d) Trânsito Típico na Terça-feira



(e) Trânsito Típico na Quarta-feira (f) Trânsito Típico na Quinta-feira (g) Trânsito Típico na Sexta-feira (h) Trânsito Típico no Sábado

Figura 25 – Visualização de apenas um trecho da trajetória da linha, e característica do trânsito típico do Google Maps para cada dia da semana

Portando os dados referente a cada viagem de ônibus foram salvos em uma planilha eletrônica para facilitar a exportação para “.txt”, que é um dos formatos de arquivo interpretados pela RNA. A Tabela 6 lista o conjunto de dados de entrada para se obter os tempos de viagem. Este conjunto de dados serão informados para a RNA realizar o treino e teste. Apesar de alguns dados serem semelhantes, como é o caso dos dados de trânsito provenientes do Google Maps e suas variações de modelos, o conjunto todo de dados serão inseridos e a própria RNA avaliará os dados inseridos, e se caso um determinado dado não for relevante para o modelo, o resultado da RNA não será influenciado quando o dado for removido do *dataset*.

Tabela 6 – Conjunto de Dados de Entrada para a RNA para Obter os Tempos de Trajeto

N°	**Entradas**	**Unidade**	**Intervalo do Conjunto de Dados**
1	Dia Semana (Domingo = 1; Sábado = 2; Segunda-feira à Sexta-feira = 3 a 7)	-	(1-7)
2	Feriado (Domingo e Feriado = 0; Sábado = 0,5; Sem Feriado = 1)	-	(0-1)
3	Horário do dia (Minuto do dia em que o ônibus é despachado)	Minutos	(0-1439)
4	Viagens de Veículos Motorizados	x1000 Viagens	(0-3250)
5	Viagens de Transporte Coletivo	x1000 Viagens	(0-1900)
6	Viagens de Transporte Individual	x1000 Viagens	(0-1450)
7	Temperatura	° C	(16-32)
8	Humidade	%	(28-91)
9	Vento	m/s	(0-6)
10	Precipitação	mm	(0-27)
11	Velocidade Mediana do Último Mês	km/h	(10-27)
12	Predição Naïve	Segundos	(959-4770)
13	Trânsito Atual Google Maps	Segundos	(1089-2494)
14	Trânsito Atual Google Maps Pessimista	Segundos	(1257-3451)
15	Trânsito Previsto Google Maps	Segundos	(1074-2707)
16	Trânsito Previsto Google Maps Pessimista	Segundos	(1211-3587)

Fonte: Autor

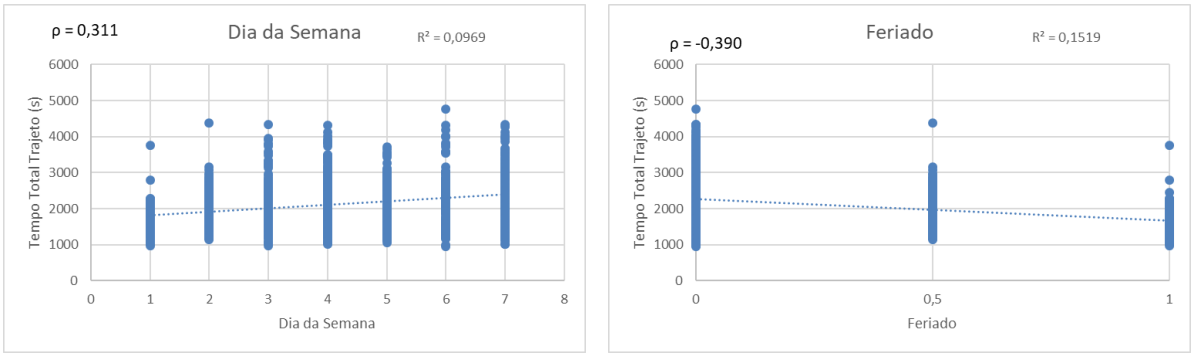
Dentre o conjunto de dados totais (1570 observações de tempos de viagem para cada um dos 35 trechos e o tempo total de trajeto) os primeiros 80% foram utilizados para treino (será informado os dados de entrada e saída para a RNA realizar o aprendizado) e os últimos 20% serão utilizados para teste (será informado apenas a entrada para a RNA calcular a previsão e comparar com o resultado esperado) e para verificar o erro do modelo desenvolvido.

A previsão será realizada a partir do momento em que o ônibus for despachado. Os tempos de chegada para cada um dos 34 pontos de ônibus intermediários até o terminal final serão previstos, será desconsiderado o tempo de permanência dos ônibus em cada ponto de ônibus. Os testes e resultados foram realizados para cada um dos 34 pontos intermediários e para o terminal final. A performance do modelo é proporcional para cada um dos pontos, ou seja, se o modelo corresponder com uma boa previsão de tempo total de trajeto, haverá uma boa previsão de tempos de viagem entre os pontos de ônibus e vice-versa.

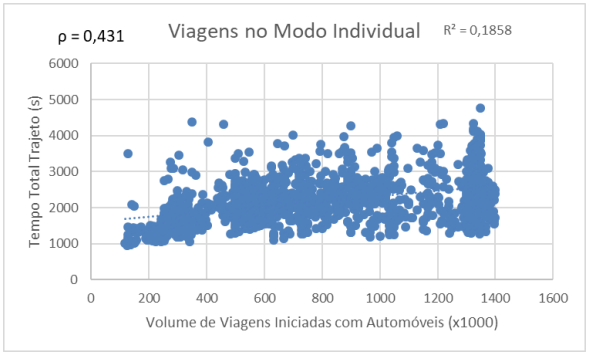
Com o objetivo de simplificar a visualização dos resultados alcançados, será apresentado apenas os resultados obtidos para a previsão do tempo total de trajeto (tempo entre o despacho do ônibus até chegar no terminal final). Esta opção foi selecionada para levar em consideração a extensão total da linha e todos os dados de trânsito das vias por onde a linha passa, e como o terminal final é o ponto mais longe do início da viagem, o tempo de viagem é maior, sua variabilidade é maior e, por consequência, o tempo de viagem total é o mais difícil para se realizar as previsões.

4.14 Correlação Dados de Entrada com a Saída RNA

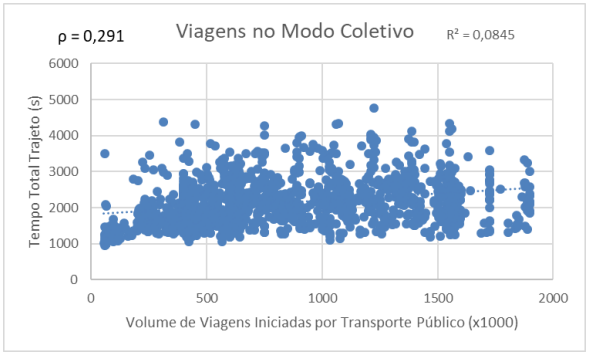
Como primeiro resultado, foi analisado a correlação e coeficiente de determinação dos dados coletados com o tempo total de viagem, ou seja, a correlação entre as entradas e saída da RNA, resultando no conjunto de gráficos da Figura 26. Dessa forma é possível analisar, antes do treino, quais as entradas que possivelmente serão mais relevantes para a RNA e o qual as variáveis de entrada se alteram com a saída.



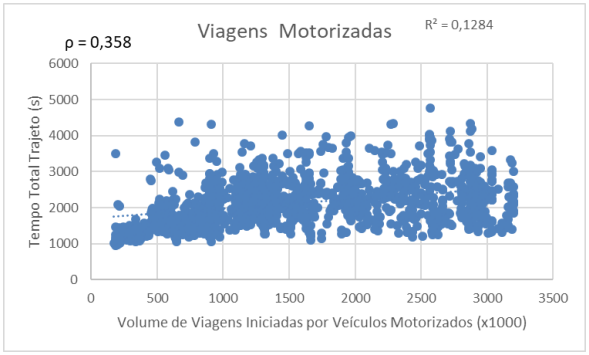
(a) (b)


ρ = 0,431
Viagens no Modo Individual
R² = 0,1858
Tempo Total Trajeto (s)
Volume de Viagens Iniciadas com Automóveis (x1000)

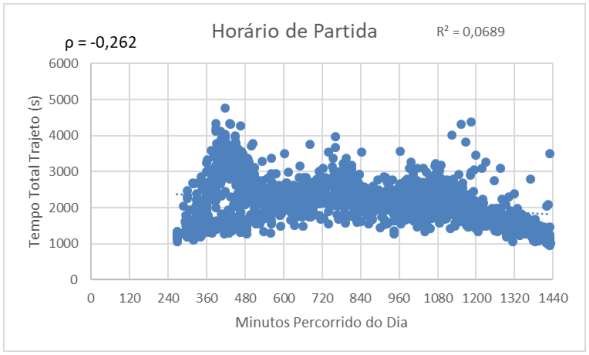
(c)


ρ = 0,291
Viagens no Modo Coletivo
R² = 0,0845
Tempo Total Trajeto (s)
Volume de Viagens Iniciadas por Transporte Público (x1000)

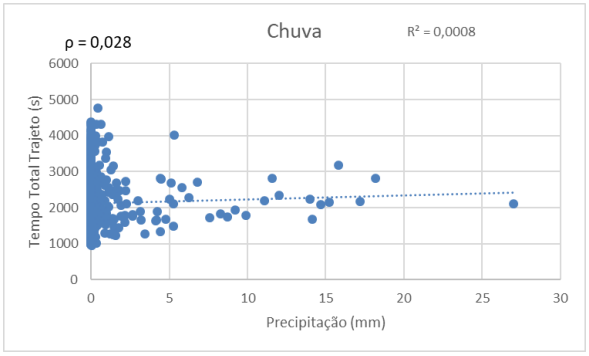
(d)


ρ = 0,358
Viagens Motorizadas
R² = 0,1284
Tempo Total Trajeto (s)
Volume de Viagens Iniciadas por Veículos Motorizados (x1000)

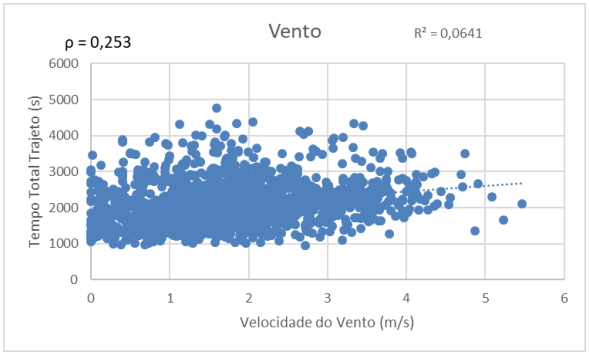
(e)


ρ = -0,262
Horário de Partida
R² = 0,0689
Tempo Total Trajeto (s)
Minutos Percorrido do Dia

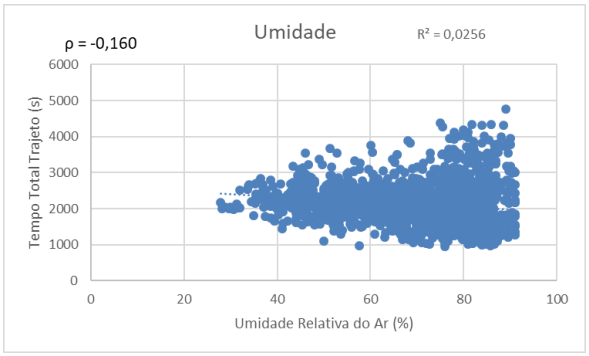
(f)


ρ = 0,028
Chuva
R² = 0,0008
Tempo Total Trajeto (s)
Precipitação (mm)

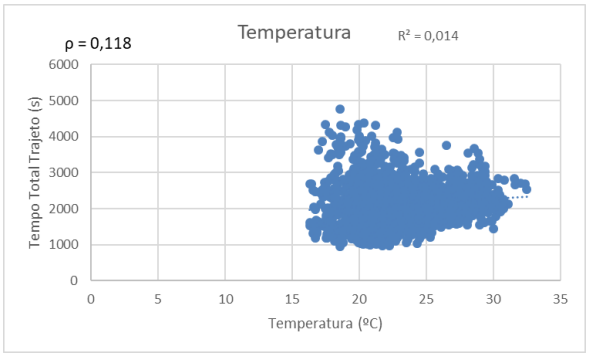
(g)


ρ = 0,253
Vento
R² = 0,0641
Tempo Total Trajeto (s)
Velocidade do Vento (m/s)

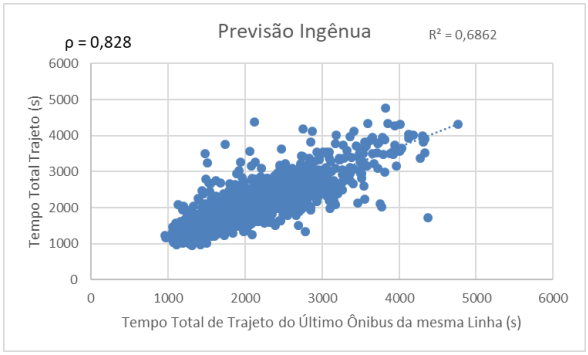
(h)


ρ = -0,160
Umidade
R² = 0,0256
Tempo Total Trajeto (s)
Umidade Relativa do Ar (%)

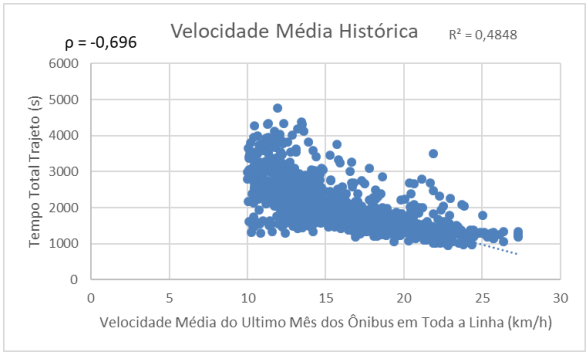
(i)


ρ = 0,118
Temperatura
R² = 0,014
Tempo Total Trajeto (s)
Temperatura (ºC)

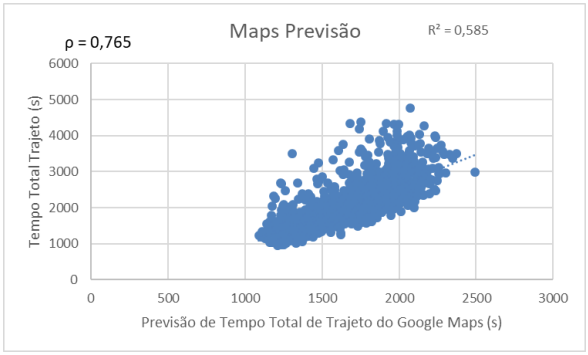
(j)



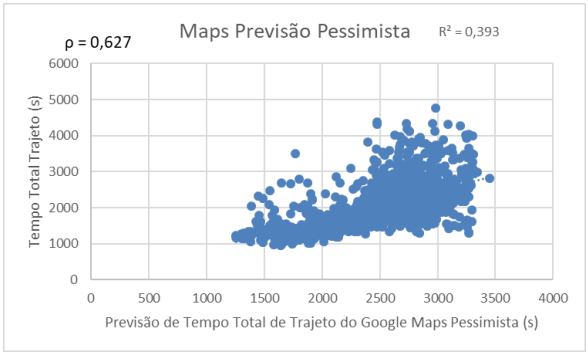
(k)



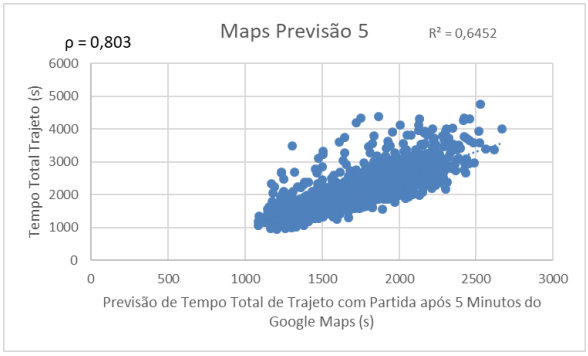
(l)



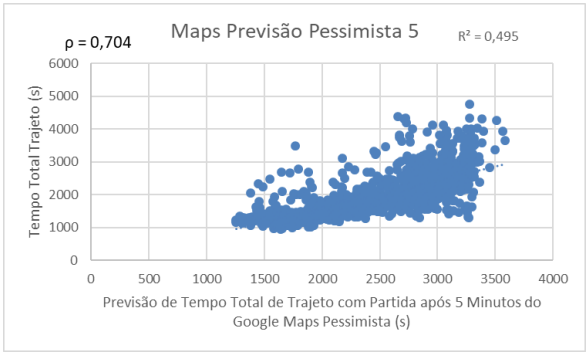
(m)



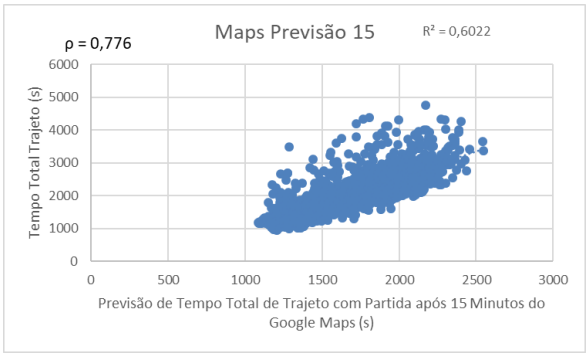
(n)



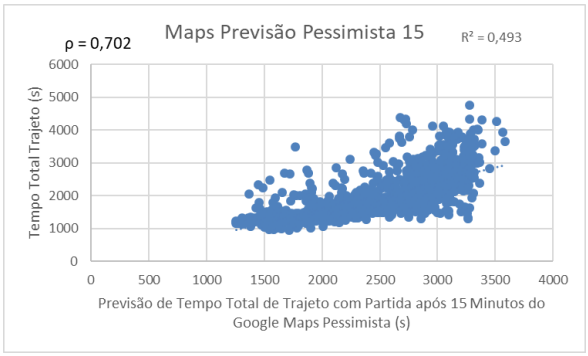
(o)



(p)



(q)



(r)

Figura 26 – Conjunto de Coeficientes de Determinação das Entradas e Saídas da RNA

Fonte: Autor, elaborado no Excel

A partir dos gráficos apresentados e pelos coeficientes de determinação e correlação, pode-se interpretar primeiramente que os dados coletados através das diferentes APIs (Google Maps e SPTrans) estão moderadamente correlacionados. Isso significa que pode ser um bom dado de entrada para a RNA. Ao passo que os dados meteorológicos estão fracamente correlacionados, porém pode ser que a RNA consiga descobrir algum padrão no comportamento desse dado em relação aos outros dados de entrada, ou pode ser que esses dados não contribuam durante o treino, ou nos próprios resultados da RNA.

Através das correlações é possível constar que quanto mais próximo da sexta-feira, os tempos de viagem vão aumentando. Nos feriados, os tempos de viagem são menores. As velocidades médias históricas tiveram uma boa correspondência quanto aos dados atuais, e conforme a velocidade média aumenta, os tempos de viagem vão diminuindo.

As correlações mais fortes, tanto no valor determinístico como na própria correlação, são os dados da previsão ingênua e previsão do Google Maps com uma previsão de trânsito 5 minutos no futuro. Provavelmente, isso seja por conta que o ônibus tem um modelo de viagem mais atrasado em comparação aos automóveis, e o trânsito de veículos que esse ônibus irá enfrentar será posterior aos dos automóveis.

Percebe-se também que dentre as bases de dados, não há nenhuma entrada que tenha correlações tão fortes como a correlação extraída por (XINGHAO et al., 2013), descrito no capítulo 2.5.3. Essa diferença ocorreu porque os autores coletaram manualmente os dados de táxi e os momentos em que os ônibus chegavam e saiam dos pontos de parada, através de gravações de vídeo. E os dados coletados neste trabalho foram através de APIs e dispositivos AVLs, no qual possuem uma latência maior e uma precisão menor. Sendo possível, apenas estimar o momento em que o ônibus passou nos pontos de parada, uma vez que, sem os dados de bilhetagem, não foi possível estimar o tempo de permanência em cada ponto.

Apesar dos problemas com a precisão dos dados coletados, a coleta remota e automatizada permite uma maior escalabilidade, tanto na própria coleta, como no tratamento e análise dos dados. O próprio autor de (XINGHAO et al., 2013) sugere a utilização de RFID instalados nos pontos de ônibus para coletar remotamente os mesmos dados e obter os mesmos resultados.

4.15 Análise dos Hiperparâmetros

4.15.1 Análise das Saídas

Os resultados da precisão da previsão do tempo total de trajeto serão expressados em porcentagem do erro MAPE, e conforme explicado no item 2.3.3, quanto menor a porcentagem do erro, mais preciso se torna o modelo de previsão.

O primeiro experimento da RNA foi realizado apenas inserindo todas as entradas e todas as saídas em uma única RNA, ou seja, foi realizado o treino e teste de uma única RNA para realizar as previsões de tempo de trajeto entre todos os pontos de ônibus intermediários e o tempo total de viagem, cada valor previsto de tempo de viagem é uma saída da RNA.

Apesar da RNA treinar e o erro convergir, o erro MAPE ficou no valor de 190,72%, muito maior que o erro da previsão ingênua ou da média móvel. Significando que existem muitas saídas para uma única RNA e que o ideal é separar as saídas criando uma RNA para cada tempo de viagem ser previsto. Podendo ser utilizado o mesmo conjunto de dados, ou apenas os mais relevantes para cada trecho.

Separando as saídas da RNA o erro MAPE reduziu para 26,61%. Essa queda no erro já era esperada, pois os neurônios de cada RNA separada podem se especializar em uma única saída e não correr o risco de interferência entre múltiplas saídas. Pois em cada interação para recálculo de peso pode melhorar a precisão de uma determinada previsão de um trecho da linha enquanto pode prejudicar a previsão de outro trecho da linha.

O método ingênuo ainda possui um erro menor, podendo confirmar que, dependendo dos hiperparâmetros utilizados, pode influenciar nos resultadas das previsões.

4.15.2 Análise Combinatória de Hiperparâmetros

Em seguida, outros hiperparâmetros foram alterados, porém com por conta da aleatoriedade dos resultados da RNA, praticamente essas alterações foram imperceptíveis. Ao alterar a largura de neurônios da RNA entre 5 e 50 praticamente não há influência nos resultados. A utilização de 1 ou 2 camadas escondidas também não influenciou os resultados. A utilização de otimizadores desde que sejam adaptativos (Nadam, Adagrad, Adadelta, Adam

e Adamax) tiveram bom desempenho, porém os outros otimizadores (RMSprop e SGD) nem sempre tiveram bons resultados e demoraram um pouco mais até convergir.

O resultado da RNA quando se utiliza a padronização dos dados foi comparado com normalização, e caso, o erro MAPE da RNA foi menor para a utilização da normalização, alcançando uma margem de erro MAPE de 10,83%. O qual já começou a ficar menor do que a previsão pelo método Naïve.

As funções de perda utilizadas foram MSE, MAE e MAPE, não houve diferenciação notável entre a utilização do MAE e do MAPE como funções de perda. Porém houve uma diferença para o uso do MSE, que é mais influenciado por grandes erros, pois é uma função quadrática, enquanto o MAE reduz o erro absoluto. Como a intenção dessa pesquisa é diminuir o MAPE, a função de perda MAPE foi a escolhida.

As seguintes funções de ativações foram testadas: “relu”; “sigmoid”; “softmax”; “tanh”; “exponential”; “linear”. Dentre essas funções, as que tiveram uma performance melhor foram a “relu” e “sigmoid”, com pequenas diferenças de comportamento. A função “sigmoid” performou melhor com a saída normalizada e a “relu” com a saída não normalizada.

Em todos os casos testados, a parada antecipada (*early stop*) foi utilizada, separando 10% dos dados do treino para ser testados durante o treino. Conforme a evolução das épocas, a validação é verificada para não causar o sobre-ajuste (*overfitting*). O parâmetro de paciência da interrupção do treino foi alterado para verificar seu reflexo nos resultados, e na maioria dos testes, valores de paciência entre 2 e 3 obtiveram melhores respostas. O diagrama da construção deste modelo pode ser observado na Figura 27:

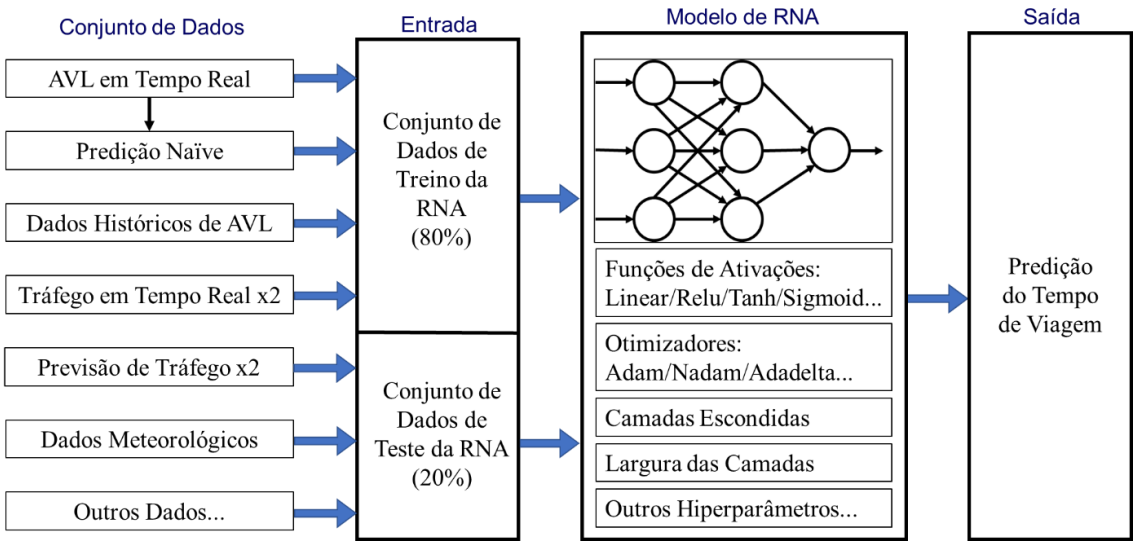


Figura 27 – Diagrama da Arquitetura para Combinações de Dados de Entrada e Hiperparâmetros da RNA

Fonte: Autor, elaborado através do PowerPoint

Após a análise hiperparamétrica, pode-se identificar uma combinação de hiperparâmetros que deixasse a RNA mais estável e que entregasse resultados mais constantes. Pois a próxima fase é manter os hiperparâmetros constantes e apenas variar os dados de entrada para analisar a influência de cada um. Os hiperparâmetros adotados daqui por diante estão listados na Tabela 7:

Tabela 7 – Hiperparâmetros Fixados para Análise da Influência dos Dados de Entrada

Hiperparâmetro	**Valor adotado**
Largura de neurônios	10
Camadas escondidas	1
Função de ativação	Sigmoid
Otimizador	Nadam
Erro a ser otimizado (*Loss*)	MAE
Interrupção do Treino	Sim
Paciência	4
Tipo de Normalização	Normalização
Normalização da entrada	Sim
Normalização da saída	Sim

Fonte: Autor

4.16 Análise dos Conjuntos de Dados de Entrada

Para poder analisar o conjunto de dados de entrada mais relevantes ao modelo, será realizado várias combinações dos dados descritos na Tabela 6, para ver a influência na precisão da previsão ao remover ou incluir determinados dados. Como a RNA elaborada até o momento possui 16 dados de entrada, uma análise combinatória completa resultaria em 65536 possibilidades. Para diminuir essa quantidade de combinações, algumas entradas serão agrupadas, seguindo os grupos formados na Tabela 8.

Tabela 8 – Grupos de Dados de Entrada da RNA

Nº do Grupo	**Nome do Grupo**	**Dados de Entrada**
A	Minuto do dia para Despacho	Minuto do dia
B	Datas	Dia Semana Feriado
C	Quantidade de Viagens de Veículos	Veículos Motorizados Transporte Coletivo Transporte Individual
D	Meteorológicos	Temperatura Humidade Vento Precipitação
E	Velocidade Média	Velocidade Mediana da linha por Hora do Último Mês
F	Previsão Ingênua (Naïve)	Tempo de Trajeto do último Ônibus
G	Google Maps	Tempo de Trajeto Atual Google Maps
H	Google Maps Pessimista	Tempo de Trajeto Atual Google Maps Pessimista
I	Google Maps Previsão	Tempo de Trajeto Previsto (5 - 20 min) Google Maps
J	Google Maps Previsão Pessimista	Tempo de Trajeto Previsto (5 - 20 min) Google Maps Pessimista

Fonte: Autor

Para exemplificar a interpretação da Tabela 8, o grupo “D” corresponde aos dados meteorológicos coletados no item 4.8 e compõe as informações de temperatura, humidade, vento e precipitação.

O dado do grupo A: “minuto do dia” é o único dado que precisa ser mantido em todos os casos, pois é o horário de partida do ônibus. Antes de realizar a análise combinatória dos outros dados de entrada, foi avaliado a aleatoriedade da RNA. Os pesos iniciais adotados por uma RNA são aleatórios, e dependendo dos pesos inicializados por uma RNA, pode-se obter valores diferentes de erro ao final do treinamento. Para poder observar a influência que os dados de entrada geram no modelo foi necessário remover a influência estocástica da RNA. Uma maneira de amenizar essa influência é realizando uma série de treinos e comparar a média dos erros. Portanto cada valor de erro calculado para as combinações será uma média de 20 treinos, para que o efeito da aleatoriedade do treino da RNA seja minimizado e a influência de cada dado de entrada seja mais perceptível.

Em seguida, foi realizada uma variação dos entrada dos grupos na RNA e o resultado MAPE, RMSE e coeficiente de correlação de Pearson (CCP) foi calculado em cada uma das opções, gerando a Tabela 9. A coluna “N” refere-se ao número sequencial da iteração, as colunas de “A” a “J” informam a presença ou não do grupo do dado, sendo valor 1 para quando o grupo do dado está presente e 0 para quando não está presente, e as colunas “MAPE” e “RMSE” informam o valor respectivo do erro para cada iteração.

Os erros MAPE, RMSE e o CCP são valores médios de erro dos 20 treinos de cada agrupamento, mesmo realizando essa média, os valores obtidos podem sofrer leves variações por conta da aleatoriedade da RNA.

Tabela 9 – Análise do Erro MAPE para as Diferentes Combinações de Dados de Entrada

Nº	A	B	C	D	E	F	G	H	I	J	MAPE (%)	RMSE (s)	CCP
1	1	0	0	0	0	0	0	0	0	0	20,83	577,86	0,390
2	1	1	1	1	1	1	1	1	1	1	9,10	297,32	0,875
3	1	0	1	1	1	1	1	1	1	1	9,32	304,61	0,870
4	1	1	0	1	1	1	1	1	1	1	9,57	314,00	0,862
5	1	1	1	0	1	1	1	1	1	1	9,21	306,06	0,871
6	1	1	1	1	0	1	1	1	1	1	9,19	306,70	0,869
7	1	1	1	1	1	0	1	1	1	1	9,25	314,46	0,861
8	1	1	1	1	1	1	0	1	1	1	9,21	300,95	0,873
9	1	1	1	1	1	1	1	0	1	1	9,16	297,47	0,875
10	1	1	1	1	1	1	1	1	0	1	9,31	301,77	0,872
11	1	1	1	1	1	1	1	1	1	0	9,26	301,78	0,872
12	1	1	1	1	1	1	0	0	0	0	9,88	313,91	0,861
13	1	1	1	1	1	1	1	0	1	0	9,28	302,07	0,870
14	1	1	1	1	1	1	1	0	0	0	9,43	308,33	0,866
15	1	1	1	1	1	1	0	0	1	0	9,37	300,90	0,871
16	1	1	1	1	1	1	0	1	0	1	9,64	307,44	0,866
17	1	1	1	1	1	1	1	1	0	0	9,35	304,13	0,870
18	1	1	0	1	0	1	1	1	1	1	9,50	311,84	0,863
19	1	1	0	0	0	1	0	0	0	0	11,52	334,93	0,836
20	1	1	1	1	1	1	0	0	1	1	9,22	295,97	0,876
21	1	1	1	0	1	1	0	0	0	0	9,92	318,26	0,859
22	1	1	1	1	1	0	0	0	0	0	12,68	374,07	0,799
23	1	1	0	1	0	0	1	1	1	1	9,65	336,58	0,843
24	1	1	0	1	0	0	0	0	0	0	18,28	519,26	0,540
25	1	1	0	0	0	0	0	0	0	0	19,71	541,54	0,474
26	1	1	1	0	1	0	0	0	0	0	11,09	375,38	0,817
27	1	0	0	0	0	0	1	1	1	1	10,01	354,67	0,825
28	1	0	0	0	0	1	0	0	0	0	11,61	339,15	0,833
29	1	1	0	0	1	0	0	0	0	0	11,42	385,49	0,805

Fonte: Autor

A seguir será apresentado o objetivo de cada agrupamento:

- **1:** Apenas o dado de “minuto do dia” foi inserido no modelo, correspondendo ao quanto a RNA é capaz de se adaptar a uma média histórica, e o valor do erro revela o quanto o tempo de viagem pode variar dependendo de outros dados e fatores;
- **2:** Todos os dados foram inseridos no modelo para o treinamento da RNA, foi a iteração que gerou o melhor resultado, mostrando o quanto a RNA é capaz de apender com os dados informados para realizar as previsões de tempo de viagem;
- **3 a 11:** Foi removido grupo a grupo do modelo que possui todos os dados, com isso é possível observar qual dado possui maior influência para o modelo. A influência que o grupo possui para o modelo é diretamente proporcional ao incremento do erro quando o grupo for removido;
- **12:** Corresponde ao aprendizado da RNA se dados de tráfego do Google Maps não fossem coletados;
- **13:** Corresponde a um teste sem dados de tráfego no modo pessimista;
- **14:** Corresponde a um teste sem dados de tráfego no modo pessimista e sem dados de previsão de tráfego do Google Maps;
- **15:** Corresponde a um teste sem dados de tráfego no modo pessimista e tráfego em tempo real;
- **16:** Corresponde a um teste sem dados de tráfego no modo melhor estimativa;
- **17:** Corresponde a um teste sem dados de previsão de tráfego do Google Maps;
- **18:** Corresponde a um teste se dados de quantidade de viagens de veículos e velocidade média dos ônibus do mês anterior não estivessem presentes no modelo;
- **19:** Corresponde a um teste ao inserir apenas dados sobre datas e a predição Naïve na RNA;
- **20:** Corresponde ao aprendizado se dados de tráfego em tempo real não fossem coletados;
- **21:** Corresponde ao aprendizado se nenhum dado de tráfego do Google Maps ou dados meteorológicos fossem coletados;

- **22:** Corresponde ao aprendizado sem dados de tráfego do Google Maps e predição Naïve;
- **23:** Corresponde ao modelo apenas com dados meteorológicos, dados de tráfego e informações sobre as datas;
- **24:** Corresponde ao modelo apenas com dados meteorológicos e datas;
- **25:** Corresponde ao modelo apenas com as informações sobre as datas;
- **26:** Corresponde ao modelo apenas com dados sobre as datas, quantidade viagens de veículos e velocidades médias dos ônibus do mês anterior;
- **27:** Corresponde ao modelo apenas com dados de tráfego;
- **28:** Corresponde a um teste ao inserir apenas dados sobre a predição Naïve na RNA, e resultou um leve ganho em relação ao que o próprio Naïve entrega como resultado (MAPE=12,89%);
- **29:** Corresponde a um teste ao inserir apenas dados sobre datas e velocidades médias dos ônibus do mês anterior.

4.17 Comparação das Metodologias de Outros Autores

Comparando com o fruto do trabalho dos outros autores, descrito no item 2.5, pode-se perceber que os valores de erro MAPE foram obtidos próximos ao estudados na revisão bibliográfica. A grande diferença entre o trabalho proposto e a bibliografia atual é que esta metodologia possibilita a análise dos ônibus nas vias mais afetadas pelos automóveis e pelos horários de pico. O qual gera um desvio padrão maior nos tempos de viagem dos ônibus por conta da irregularidade do comportamento do trânsito.

A metodologia sugerida apresentou benefícios da aplicação de dados de trânsito em tempo real, junto a um método de previsão que utilizasse uma RNA. Ao comparar com outros métodos existentes, como filtro de Kalman, Floresta Aleatória e SVM, os dados de trânsito em tempo real poderiam contribuir com a Floresta Aleatória, por conta das árvores de decisões, uma vez que alguns dos ramos poderiam estar baseados em valores limites dos dados de trânsito ou previsão de trânsito. Quanto ao SVM, dependendo de como foi criado, este também pode-se beneficiar de novos dados além dos fornecidos pelo AVL. Como o filtro de Kalman analisa apenas uma única variável, não seria possível atualmente a aplicação de dados de trânsito em tempo real para este método.

4.18 Impacto da Previsão do Tempo Total de Trajeto

A partir dos resultados obtidos, é possível analisar o ganho em termos de tempo de espera dos passageiros nos pontos de ônibus caso um controle preventivo utilize esse resultado de previsão de tempo total de trajeto para regularizar o headway nas viagens dinâmicas.

Para que esse ganho seja levantado, seria necessário testar a solução em campo, ou utilizar um simulador de tráfego. Porém para ambos os testes, demandaria um trabalho considerável. Testar a solução em campo poderia atrapalhar o controle operacional atual, pois durante a fase de protótipo, algumas falhas podem ocorrer.

Para testar a solução usando simulador de tráfego demandaria coletar dados precisos das vias de toda a extensão da linha para cada faixa horária (volume de veículos entrando e saindo nas vias, tempos semafóricos, outras linhas de ônibus, demanda de passageiros em cada ponto de ônibus), e ainda calibrar rede. Por esses motivos não será estimado o possível ganho de controle operacional da linha.

4.19 Aplicação Previsão utilizando o Modelo construído em RNA

Após a verificação dos dados mais relevantes presentes e o modelo computacional baseado em RNA estar devidamente treinado e testado, o modelo pode ser utilizado em uma aplicação em tempo real. Existem diversas maneiras para se implementar esse modelo (através de um programa, plugin, API). A seguir será apresentado uma forma de se implementar o modelo desenvolvido que poderá beneficiar a agência responsável pelo transporte e os usuários com previsões de tempos de viagem mais precisas.

Ao finalizar o treino da RNA, um arquivo de extensão .h5 é gerado. Esse arquivo contém as informações sobre os pesos e funções de ativação da RNA, através dele é possível calcular o resultado do modelo computacional. Para poder utilizar o arquivo devidamente, deve-se coletar os dados em tempo real e aplicar a mesma normalização adotada no modelo, o mesmo processo se aplica para a saída após a utilização do modelo.

Recomenda-se criar uma função ou rotina dentro de um programa ou servidor que seja capaz de receber os dados de entrada, aplicar a normalização nos dados informados, utilizar o arquivo .h5 para o cálculo da previsão, realizar a desnormalização da saída para retornar o valor

da previsão. Como foi elaborado 35 RNA diferentes para prever o tempo de viagem entre os pontos intermediários, pode-se utilizar as 35 funções diferentes.

Outra forma comum de implementar o algoritmo gerado por uma RNA é pela utilização de APIs. Um novo servidor conectado a nuvem pode ser instanciado e configurado para receber solicitações HTTP com os dados de entrada para responder em XML ou JSON o valor da previsão, a Figura 28 exemplifica uma forma simplificada de um comando GET para a URL do servidor onde está o algoritmo de previsão e alguns parâmetros da API. Para implementar esse servidor deve-se criar uma URL nova, para redirecionamento da API e criar uma documentação para explicar a forma de utilização de cada dado de entrada em forma de parâmetro da API.

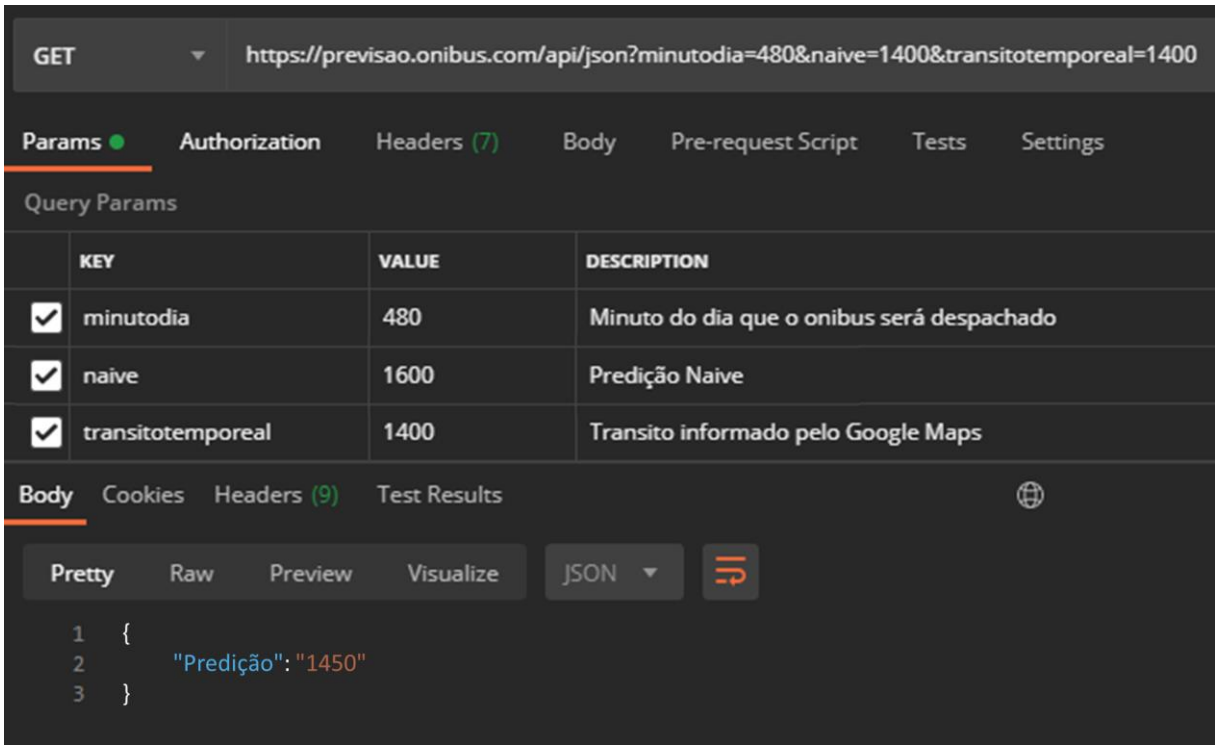


Figura 28 – Exemplo de utilização de uma API para consultar a previsão de viagem

Fonte: Autor, elaborado utilizando Postman

Para que este processo de utilização da RNA por meio da API funcione devidamente, o servidor deve ficar monitorando a entrada de solicitações através das portas da rede, em seguida um código deve ser executado para realizar todo o processamento e verificação dos dados recebidos pela solicitação da rede e ao finalizar a execução deve enviar a resposta ao endereço solicitante.

Para poder realizar as consultas das previsões do modelo desenvolvido é necessário informar os dados em tempo real e os outros dados históricos referente ao horário da solicitação. Para coletar os dados de tráfego em tempo real, deve-se configurar uma máquina ou servidor para coletar ciclicamente e consultar os últimos dados quando for utilizar a função de predição, ou pode-se realizar a coleta dos dados apenas quando for realizar a predição. O mesmo pode ser utilizado para os dados meteorológicos.

Para obter os dados da predição Naïve de forma automática será necessário um processamento de dados logo após a coleta da geolocalização dos ônibus da linha. Esse processamento envolve os cálculos geodésico descritos no item 3.2, e realizar a limpeza dos dados coletado de forma automatizada, conforme explicado no item 3.4 e implementado no item 4.9. Ao final da coleta e tratamento dos dados de AVL, pode-se estimar os tempos de viagens do último ônibus e com isso, disponibilizar esse dado para o sistema de previsão. A agência responsável pelo transporte é capaz de obter este dado com maior facilidade, pois ela já possui um sistema de monitoramento e os dados dos AVL ficam armazenados em um banco de dados.

Por conta que a previsão está sendo realizada em uma série temporal, a precisão do modelo pode diminuir com o tempo. Isto é decorrente da mudança do padrão de comportamento ao longo do tempo e os dados históricos começam a ficar irrelevantes e desatualizados. Para manter a performance ao longo do tempo, é necessário realizar novos treinos e testes da RNA a partir dos novos dados coletados. Em seguida deve-se atualizar os programas ou servidores com o novo arquivo .h5, correspondente aos novos pesos para os dados. Esta pesquisa não avaliou a perda de performance da RNA ao longo do tempo.

5 CONCLUSÃO

Esta pesquisa propõe uma metodologia para realizar previsões de tempos de viagem de ônibus com alta precisão. A motivação é decorrente da dificuldade de realizar as previsões do ônibus principalmente em vias que não sejam corredores ou faixas exclusivas de ônibus, onde o tempo de viagem é afetado pelo trânsito gerado pelos automóveis.

A lacuna de pesquisa foi identificada com a hipótese de que os dados de trânsito e meteorológicos podem ser coletados e informados a um sistema que realize previsões mais precisas. Para executar tal tarefa, foi necessário criar um sistema que colete e processe automaticamente os dados de posição dos ônibus e dados do trânsito em tempo real, e os envie para um programa que realize as previsões de tempo de viagem. Construir um modelo de regressão dessa função não linear, onde há a presença de várias fontes e conjuntos de dados é um problema para o qual as Redes Neurais Artificiais estão entre as técnicas de ponta atuais.

A metodologia proposta foi testada e comprovou que é possível realizar a coleta automatizada de vários tipos de dados que possam afetar o no tempo total de trajeto dos ônibus. Foi possível desenvolver uma RNA que utilizasse esses dados para realizar as previsões.

Os resultados entregados pela RNA mostraram-se satisfatórios, alcançando um erro MAPE de 9,10% e RMSE de 297 segundos para uma linha de ônibus que trafega em vias onde outros automóveis trafegam ao mesmo tempo. O resultado MAPE alcançou um resultado próximo ao de outros autores e com dados de entrada menos correlacionados. Se não houvesse dados de previsão de trânsito e em tempo real, o erro MAPE seria de 9,88%. Isto mostra a relevância que estes dados têm para a previsão. Outros dados também tiveram sua contribuição, mostrando o quanto a RNA está preparada para receber diversos tipos de dados.

Como proposta para continuidade dessa pesquisa, outros dados deverão ser analisados, como por exemplo, dados de APC, táxi, redes sociais ou o que estiver disponível, para que seja analisado a sua correlação e influência nos tempos de viagem, outras técnicas de aprendizado de máquina também podem ser usados para comparação de eficiência, como por exemplo, o aprendizado profundo (*deep learning*).

Outros focos na pesquisa também seriam enriquecedores para fundamentar e para analisar outras possibilidades. Como é o caso do impacto da previsão de tempos de viagem na regularidade do headway e como informação ao usuário. Pode-se mensurar o quão a precisão da previsão reflete em termos de qualidade do transporte público e o quanto isso poderia diminuir em tempo de espera dos passageiros nos pontos de ônibus.

6 BIBLIOGRAFIA

ACHAR, A. et al. Bus Arrival Time Prediction: A Spatial Kalman Filter Approach. **IEEE Transactions on Intelligent Transportation Systems**, v. 21, n. 3, p. 1298–1307, mar. 2020. Disponível em: <https://ieeexplore.ieee.org/document/8691701/>.

ADEWALE, A. E.; HADACHI, A. Neural Networks Model for Travel Time Prediction Based on ODTravel Time Matrix. n. May, 8 abr. 2020. Disponível em: <http://arxiv.org/abs/2004.04030>.

ALTINKAYA, MEHMET, ZONTUL, M. Urban bus arrival time prediction: A review of computational models. **International Journal of Recent Technology and Engineering**, n. 24, p. 164–169, 2013. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.683.2536&rep=rep1&type=pdf>.

AMAZON. **Elastic Compute Cloud - Amazon EC2 - AWS**. Disponível em: <https://aws.amazon.com/pt/ec2/>. Acesso em: 25 out. 2019.

ANTP. **SÉRIE CADERNOS TÉCNICOS volume 20 - Big Data para análise de métricas de qualidade de transporte: metodologia e aplicação**. [s.l: s.n.]

ANTRIM, A.; BARBEAU, S. THE MANY USES OF GTFS DATA – OPENING THE DOOR TO TRANSIT AND MULTIMODAL APPLICATIONS Aaron. **ITS America's 23rd Annual Meeting & Exposition**, 2013. Disponível em: <http://medcontent.metapress.com/index/A65RM03P4874243N.pdf%5Cnhttp://www.locationaware.usf.edu/wp-content/uploads/2010/02/The-Many-Uses-of-GTFS-Data-–-ITS-America-submission-abbreviated.pdf>.

ASSAF, M. H.; WILLIAMS, K. M. RFID for optimisation of public transportation system. In: 2011 Seventh International Conference on Intelligent Sensors, Sensor Networks and Information Processing, **Anais**...IEEE, dez. 2011. Disponível em: <http://ieeexplore.ieee.org/document/6146604/>.

AZURE, M. **VMs (Máquinas Virtuais) para Linux e Windows | Microsoft Azure**. Disponível em: <https://azure.microsoft.com/pt-br/services/virtual-machines/>. Acesso em: 25 out. 2019.

BAPTISTA, A. T.; BOUILLET, E. P.; POMPEY, P. Towards an uncertainty aware short-term travel time prediction using GPS bus data: Case study in Dublin. In: 2012 15th International IEEE Conference on Intelligent Transportation Systems, **Anais**...IEEE, set. 2012. Disponível em: <http://ieeexplore.ieee.org/document/6338633/>.

BARBEAU, S. J. Quality Control - Lessons Learned from the Deployment and Evaluation of GTFS-Realtime Feeds. **Transportation Research Board**, 2018. Disponível em: <http://www.ghbook.ir/index.php?name=فرهنگ و رسانه های نوین&option=com_dbook&task=readonline&book_id=13650&page=73&chkhashk=ED9C9491B4&Itemid=218&lang=fa&tmpl=component>.

BERREBI, S. J. et al. Comparing bus holding methods with and without real-time predictions. **Transportation Research Part C: Emerging Technologies**, v. 87, p. 197–211, fev. 2018. Disponível em: <https://doi.org/10.1016/j.trc.2017.07.012>.

BERREBI, S. J.; WATKINS, K. E.; LAVAL, J. A. A real-time bus dispatching policy to minimize passenger wait on a high frequency route. **Transportation Research Part B: Methodological**, v. 81, p. 377–389, nov. 2015. Disponível em: <http://dx.doi.org/10.1016/j.trb.2015.05.012>.

BOX, G. E. et al. **Time series analysis: forecasting and control**. [s.l.] John Wiley & Sons, 2015.

CATHEY, F. W.; DAILEY, D. J. A prescription for transit arrival/departure prediction using automatic vehicle location data. **Transportation Research Part C: Emerging Technologies**, v. 11, n. 3–4, p. 241–264, jun. 2003. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0968090X03000238>.

CHEN, C. et al. Bus travel time prediction based on deep belief network with back-propagation. **Neural Computing and Applications**, v. 0123456789, n. 1, 2 nov. 2019. Disponível em: <https://doi.org/10.1007/s00521-019-04579-x>.

CHEN, J. et al. Design of limited-stop bus service with capacity constraint and stochastic travel time. **Transportation Research Part E: Logistics and Transportation Review**, v. 83, p. 1–15, nov. 2015. Disponível em: <http://dx.doi.org/10.1016/j.tre.2015.08.007>.

CHEN PENG; YAN XIN-PING; LI XU-HONG. Bus Travel Time Prediction Based on Relevance Vector Machine. In: 2009 International Conference on Information Engineering and Computer Science, 3, **Anais**...IEEE, dez. 2009. Disponível em: <http://ieeexplore.ieee.org/document/5367101/>.

CHEN, X. et al. Evaluating the Effects of Traffic Congestion and Passenger Load on Feeder Bus Fuel and Emissions Compared with Passenger Car. **Transportation Research Procedia**, v. 25, p. 616–626, 2017. Disponível em: <http://dx.doi.org/10.1016/j.trpro.2017.05.446>.

CHIEN, S. I. J.; DING, Y.; WEI, C. Dynamic Bus Arrival Time Prediction with Artificial Neural Networks. **Journal of Transportation Engineering**, v. 128, n. 5, p. 429–438, set. 2002. Disponível em: <http://ascelibrary.org/doi/10.1061/%28ASCE%290733-947X%282002%29128%3A5%28429%29>.

CHOUDHARY, R.; KHAMPARIA, A.; GAHIER, A. K. Real time prediction of bus arrival time: A review. In: 2016 2nd International Conference on Next Generation Computing Technologies (NGCT), October, **Anais**...IEEE, out. 2016. Disponível em: <http://ieeexplore.ieee.org/document/7877384/>.

DELGADO, F.; MUNOZ, J. C.; GIESEN, R. How much can holding and/or limiting boarding improve transit performance? **Transportation Research Part B: Methodological**, v. 46, n. 9, p. 1202–1217, nov. 2012. Disponível em: <http://dx.doi.org/10.1016/j.trb.2012.04.005>.

DOU, M. et al. Predicting Passengers in Public Transportation Using Smart Card Data. (M. A. Sharaf, M. A. Cheema, J. Qi, Eds.) In: Databases Theory and Applications, Cham. **Anais**... Cham: Springer International Publishing, 2015.

ERTEL, W. **Introduction to Artificial Intelligence**. Cham: Springer International Publishing, 2017. v. 57

ESTRADA, M. et al. Bus control strategies in corridors with signalized intersections. **Transportation Research Part C: Emerging Technologies**, v. 71, p. 500–520, out. 2016. Disponível em: <http://dx.doi.org/10.1016/j.trc.2016.08.013>.

FAN, W.; GURMU, Z. Dynamic Travel Time Prediction Models for Buses Using Only GPS Data. **International Journal of Transportation Science and Technology**, v. 4, n. 4, p. 353–366, 2015. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S204604301630168X>.

FAYYAZ S., S. K.; LIU, X. C.; ZHANG, G. An efficient General Transit Feed Specification (GTFS) enabled algorithm for dynamic transit accessibility analysis. **PLOS ONE**, v. 12, n. 10, p. e0185333, 5 out. 2017. Disponível em: <http://dx.plos.org/10.1371/journal.pone.0185333>.

FERRIS, B.; WATKINS, K.; BORNING, A. OneBusAway. In: Proceedings of the 28th international conference on Human factors in computing systems - CHI '10, September 2010, New York, New York, USA. **Anais**... New York, New York, USA: ACM Press, 2010. Disponível em: <http://portal.acm.org/citation.cfm?doid=1753326.1753597>.

FREEPIK. **Stem cell diagram on white background Vector | Free Download**.

Disponível em: <https://www.freepik.com/free-vector/stem-cell-diagram-white-background_2480958.htm#page=1&query=neuron&position=2>. Acesso em: 11 jan. 2020.

GOLDSTEIN, B.; DYSON, L. **Beyond Transparency**. [s.l: s.n.]

GONÇALVES, E. S. **Análise de padrões operacionais da frota de ônibus de transporte público no municí-pio de São Paulo e a influência de fatores climáticos em sua dinâmica.** 2018. Universidade de São Paulo, São Paulo, 2018. Disponível em: <http://www.teses.usp.br/teses/disponiveis/3/3138/tde-12032018-134651/>.

GOOGLE. **GTFS Reference**. Disponível em: <https://developers.google.com/transit/gtfs/reference>. Acesso em: 24 out. 2019a.

GOOGLE. **Produtos de computação em nuvem em grande escala | Cloud Compute Products | Google Cloud**. Disponível em: <%5C>. Acesso em: 25 out. 2019b.

GOOGLE. **Google Maps Platform Documentation**. Disponível em: <https://developers.google.com/maps/documentation>. Acesso em: 19 out. 2019c.

GOOGLE. **Google Maps Directions API Documentation**. Disponível em: <https://developers.google.com/maps/documentation/directions/overview>.

GOOGLE MAPS. **Sobre - Google Maps**. Disponível em: <https://www.google.com/maps/about/>. Acesso em: 28 out. 2019a.

GOOGLE MAPS. **Google Maps Platform Documentation**. Disponível em: <https://developers.google.com/maps/documentation>. Acesso em: 25 out. 2019b.

GOOZE, A.; WATKINS, K. E.; BORNING, A. Benefits of Real-Time Transit Information and Impacts of Data Accuracy on Rider Experience. **Transportation Research Record: Journal of the Transportation Research Board**, v. 2351, n. 1, p. 95–103, jan. 2013. Disponível em: <http://journals.sagepub.com/doi/10.3141/2351-11>.

GRAUPE, D. **PRINCIPLES OF ARTIFICIAL NEURAL NETWORKS**. [s.l: s.n.]

HAYKIN, S. **Neural Networks and Learning Machines**. [s.l: s.n.]v. 3

HE, P. et al. Learning heterogeneous traffic patterns for travel time prediction of bus journeys. **Information Sciences**, v. 512, p. 1394–1406, fev. 2020. Disponível em: <https://doi.org/10.1016/j.ins.2019.10.073>.

HE, S.-X. An anti-bunching strategy to improve bus schedule and headway reliability by making use of the available accurate information. **Computers & Industrial Engineering**, v. 85, p. 17–32, jul. 2015. Disponível em: <http://dx.doi.org/10.1016/j.cie.2015.03.004>.

HE, Z.; CAO, J.; LIU, X. High quality participant recruitment in vehicle-based crowdsourcing using predictable mobility. In: 2015 IEEE Conference on Computer

Communications (INFOCOM), **Anais**...IEEE, abr. 2015. Disponível em: <http://ieeexplore.ieee.org/document/7218644/>.

HENSHER, D. A. Future bus transport contracts under a mobility as a service (MaaS) regime in the digital age: Are they likely to change? **Transportation Research Part A: Policy and Practice**, v. 98, p. 86–96, abr. 2017. Disponível em: <http://dx.doi.org/10.1016/j.tra.2017.02.006>.

HICKMAN, M. Bus Automatic Vehicle Location (AVL) Systems. (D. Gillen, D. Levinson, Eds.) In: Assessing the Benefits and Costs of ITS, Boston, MA. **Anais**... Boston, MA: Springer US, 2004. Disponível em: <https://link.springer.com/chapter/10.1007/1-4020-7874-9_5>.

HICKMAN, M. D. An Analytic Stochastic Model for the Transit Vehicle Holding Problem. **Transportation Science**, v. 35, n. 3, p. 215–237, ago. 2001. Disponível em: <http://pubsonline.informs.org/doi/abs/10.1287/trsc.35.3.215.10150>.

HOSSEINI-MOTLAGH, S. M.; AHADPOUR, P.; HAERI, A. Proposing an approach to calculate headway intervals to improve bus fleet scheduling using a data mining algorithm. **International Journal of Industrial and Systems Engineering**, v. 8, n. 4, p. 72–86, 2015. Disponível em: <http://www.jise.ir/article_11225.html>.

HU, J.; PARK, B. B.; LEE, Y.-J. Coordinated transit signal priority supporting transit progression under Connected Vehicle Technology. **Transportation Research Part C: Emerging Technologies**, v. 55, p. 393–408, jun. 2015. Disponível em: <http://dx.doi.org/10.1016/j.trc.2014.12.005>.

JEONG, R.; RILETT, L. Prediction Model of Bus Arrival Time for Real-Time Applications. **Transportation Research Record: Journal of the Transportation Research Board**, v. 1927, n. 1927, p. 195–204, jan. 2005. Disponível em: <http://trrjournalonline.trb.org/doi/10.3141/1927-23>.

JEONG, R.; RILETT, R. Bus arrival time prediction using artificial neural network model. In: Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No.04TH8749), **Anais**...IEEE, 2004. Disponível em: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1399041>.

JÚNIOR, W. J. de A. **Métodos de otimização hiperparamétrica: um estudo comparativo utilizando árvores de decisão e florestas aleatórias na classificação binária**. 2018. Universidade Federal de Minas Gerais, 2018. Disponível em: <http://hdl.handle.net/1843/BUBD-AX2NLF>.

KUMAR, B. A. et al. Real time bus travel time prediction using k -NN classifier. **Transportation Letters**, v. 11, n. 7, p. 362–372, 31 jul. 2019. Disponível em: <http://dx.doi.org/10.1080/19427867.2017.1366120>.

KUMAR, B. A.; VANAJAKSHI, L.; SUBRAMANIAN, S. C. Bus travel time prediction using a time-space discretization approach. **Transportation Research Part C: Emerging Technologies**, v. 79, p. 308–332, jun. 2017. Disponível em: <http://dx.doi.org/10.1016/j.trc.2017.04.002>.

LAI, Y. et al. Bus Arrival Time Prediction Using Wavelet Neural Network Trained by Improved Particle Swarm Optimization. **Journal of Advanced Transportation**, v. 2020, p. 1–9, 13 jan. 2020. Disponível em: <https://www.hindawi.com/journals/jat/2020/7672847/>.

LI, Y.; VOEGE, T. Mobility as a Service (MaaS): Challenges of Implementation and Policy Required. **Journal of Transportation Technologies**, v. 07, n. 02, p. 95–106, 2017. Disponível em: <http://www.scirp.org/journal/doi.aspx?DOI=10.4236/jtts.2017.72007>.

LIN, W.-H.; ZENG, J. Experimental Study of Real-Time Bus Arrival Time Prediction with GPS Data. **Transportation Research Record: Journal of the Transportation Research Board**, v. 1666, n. 1, p. 101–109, jan. 1999. Disponível em: <http://journals.sagepub.com/doi/10.3141/1666-12>.

LIN, Y. et al. Real-Time Bus Arrival Time Prediction: Case Study for Jinan, China. **Journal of Transportation Engineering**, v. 139, n. 11, p. 1133–1140, nov. 2013. Disponível em: <http://ascelibrary.org/doi/10.1061/%28ASCE%29TE.1943-5436.0000589>.

LIU, N. et al. Mobility Crowdsourcing: Toward Zero-Effort Carpooling on Individual Smartphone. **International Journal of Distributed Sensor Networks**, v. 9, n. 2, p. 615282, fev. 2013a. Disponível em: <http://journals.sagepub.com/doi/10.1155/2013/615282>.

LIU, Z. et al. Bus stop-skipping scheme with random travel time. **Transportation Research Part C: Emerging Technologies**, v. 35, p. 46–56, out. 2013b. Disponível em: <http://dx.doi.org/10.1016/j.trc.2013.06.004>.

LU, H. et al. The impact of real-time information on passengers' value of bus waiting time. **Transportation Research Procedia**, v. 31, n. 2016, p. 18–34, 2018. Disponível em: <https://doi.org/10.1016/j.trpro.2018.09.043>.

MANDHARE, P. A.; KHARAT, V.; PATIL, C. Y. Intelligent Road Traffic Control System for Traffic Congestion A Perspective. **International Journal of Computer Sciences and Engineering**, v. 6, n. 7, p. 908–915, 31 jul. 2018. Disponível em: <http://www.ijcseonline.org/full_paper_view.php?paper_id=2534>.

MENG, Q.; QU, X. Bus dwell time estimation at bus bays: A probabilistic approach. **Transportation Research Part C: Emerging Technologies**, v. 36, p. 61–71, nov. 2013. Disponível em: <http://dx.doi.org/10.1016/j.trc.2013.08.007>.

METRÔ SÃO PAULO. **Pesquisa OD 2017**. [s.l: s.n.]. Disponível em: <http://www.metro.sp.gov.br/pesquisa-od/arquivos/Ebook Pesquisa OD 2017_final_240719_versao_4.pdf>.

MILKOVITS, M. N. Modeling the Factors Affecting Bus Stop Dwell Time. **Transportation Research Record: Journal of the Transportation Research Board**, v. 2072, n. 1, p. 125–130, jan. 2008. Disponível em: <http://journals.sagepub.com/doi/10.3141/2072-13>.

MOHANTY, S. P.; CHOPPALI, U.; KOUGIANOS, E. Everything you wanted to know about smart cities: The Internet of things is the backbone. **IEEE Consumer Electronics Magazine**, v. 5, n. 3, p. 60–70, jul. 2016. Disponível em: <http://ieeexplore.ieee.org/document/7539244/>.

MONTGOMERY, D. C.; JENNINGS, C. L.; KULAHCI, M. **Introduction to time series analysis and forecasting**. [s.l.] John Wiley & Sons, 2015.

MONTGOMERY, D. C.; RUNGER, G. C. **Estatística Aplicada E Probabilidade Para Engenheiros**. [s.l.] LTC, 2013.

MORI, U. et al. A review of travel time estimation and forecasting for Advanced Traveller Information Systems. **Transportmetrica A: Transport Science**, v. 11, n. 2, p. 119–157, 7 fev. 2015. Disponível em: <https://doi.org/10.1080/23249935.2014.932469>.

MUGHAL, M. J. H. Data Mining: Web Data Mining Techniques, Tools and Algorithms: An Overview. **International Journal of Advanced Computer Science and Applications**, v. 9, n. 6, p. 208–215, 2018. Disponível em: <http://thesai.org/Publications/ViewPaper?Volume=9&Issue=6&Code=ijacsa&SerialNo=30>
.

PADMANABAN, R. P. S.; VANAJAKSHI, L.; SUBRAMANIAN, S. C. Estimation of bus travel time incorporating dwell time for APTS applications. In: 2009 IEEE Intelligent Vehicles Symposium, **Anais**...IEEE, jun. 2009. Disponível em: <http://ieeexplore.ieee.org/document/5164409/>.

PAHL, C. Containerization and the PaaS Cloud. **IEEE Cloud Computing**, v. 2, n. 3, p. 24–31, maio 2015. Disponível em: <http://ieeexplore.ieee.org/document/7158965/>.

PARAISO, F. et al. A Federated Multi-cloud PaaS Infrastructure. In: 2012 IEEE Fifth

International Conference on Cloud Computing, **Anais**...IEEE, jun. 2012. Disponível em: <http://ieeexplore.ieee.org/document/6253530/>.

PECAR, M.; PAPA, G. Transportation problems and their potential solutions in smart cities. In: 2017 International Conference on Smart Systems and Technologies (SST), **Anais**...IEEE, out. 2017. Disponível em: <http://ieeexplore.ieee.org/document/8188694/>.

PETERSEN, N. C.; RODRIGUES, F.; PEREIRA, F. C. Multi-output bus travel time prediction with convolutional LSTM neural network. **Expert Systems with Applications**, v. 120, p. 426–435, abr. 2019. Disponível em: <https://doi.org/10.1016/j.eswa.2018.11.028>.

RAHMANI, M.; KOUTSOPOULOS, H. N.; JENELIUS, E. Travel time estimation from sparse floating car data with consistent path inference: A fixed point approach. **Transportation Research Part C: Emerging Technologies**, v. 85, n. October, p. 628–643, dez. 2017. Disponível em: <http://dx.doi.org/10.1016/j.trc.2017.10.012>.

RAJBHANDARI, R.; CHIEN, S. I.; DANIEL, J. R. Estimation of Bus Dwell Times with Automatic Passenger Counter Information. **Transportation Research Record: Journal of the Transportation Research Board**, v. 1841, n. 1, p. 120–127, jan. 2003. Disponível em: <http://journals.sagepub.com/doi/10.3141/1841-13>.

RASHIDI, S.; RANJITKAR, P. Approximation and Short-Term Prediction of Bus Dwell Time using AVL Data. **Journal of the Eastern Asia Society for Transportation Studies**, v. 10, n. 0, 2013.

ROTHFELD, R. et al. Analysis of European airports' access and egress travel times using Google Maps. **Transport Policy**, v. 81, n. January, p. 148–162, set. 2019. Disponível em: <https://doi.org/10.1016/j.tranpol.2019.05.021>.

SCIPOPULIS. **Quem é a Scipopulis**. Disponível em: <https://www.scipopulis.com/>. Acesso em: 28 out. 2019a.

SCIPOPULIS. **Painel do Ônibus**. Disponível em: <http://smt.scipopulis.com/>. Acesso em: 26 out. 2019b.

SHALABY, A.; FARHAN, A. Prediction Model of Bus Arrival and Departure Times Using AVL and APC Data. **Journal of Public Transportation**, v. 7, n. 1, p. 41–61, mar. 2004. Disponível em: <http://scholarcommons.usf.edu/jpt/vol7/iss1/3/>.

SINGH, B.; GUPTA, A. Recent trends in intelligent transportation systems: a review. **Journal of Transport Literature**, v. 9, n. 2, p. 30–34, abr. 2015. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S2238-10312015000200030&lng=en&tlng=en>.

SINNOTT, R. W. Sky and telescope. **Virtues of the Haversine**, v. 68, n. 2, p. 159, 1984.

SMT E SPTRANS. **SISTEMA DE MONITORAMENTO E GESTÃO OPERACIONAL**, 2018. . Disponível em: <https://www.prefeitura.sp.gov.br/cidade/secretarias/transportes/edital/index.php?p=247319>.

SPTRANS. **Indicadores do Sistema de Transporte**. Disponível em: <https://www.prefeitura.sp.gov.br/cidade/secretarias/transportes/institucional/sptrans/acesso_a_informacao/index.php?p=152415>. Acesso em: 14 out. 2019a.

SPTRANS. **API OLHO VIVO**. Disponível em: <http://www.sptrans.com.br/desenvolvedores/>. Acesso em: 19 out. 2019b.

SPTRANS. **OlhoVivo SPTrans**. Disponível em: <http://olhovivo.sptrans.com.br/>. Acesso em: 11 maio. 2020.

SUN, A.; HICKMAN, M. The Real–Time Stop–Skipping Problem. **Journal of Intelligent Transportation Systems**, v. 9, n. 2, p. 91–109, 26 abr. 2005. Disponível em: <https://www.tandfonline.com/doi/full/10.1080/15472450590934642>.

TRANSITLAND. **Transitland**. Disponível em: <https://transit.land/>. Acesso em: 25 out. 2019.

TSAMARDINOS, I.; RAKHSHANI, A.; LAGANI, V. Performance-Estimation Properties of Cross-Validation-Based Protocols with Simultaneous Hyper-Parameter Optimization. **International Journal on Artificial Intelligence Tools**, v. 24, n. 05, p. 1540023, 19 out. 2015. Disponível em: <https://www.worldscientific.com/doi/abs/10.1142/S0218213015400230>.

ULUSOY, Y. Y.; CHIEN, S. I.-J.; WEI, C.-H. Optimal All-Stop, Short-Turn, and Express Transit Services under Heterogeneous Demand. **Transportation Research Record: Journal of the Transportation Research Board**, v. 2197, n. 1, p. 8–18, 22 jan. 2010. Disponível em: <http://trrjournalonline.trb.org/doi/10.3141/2197-02>.

WANG, P. et al. Provision of Bus Real-Time Information: Turning Passengers from Being Contributors of Headway Irregularity to Controllers. **Transportation Research Record: Journal of the Transportation Research Board**, v. 2672, n. 8, p. 143–151, dez. 2018. Disponível em: <http://journals.sagepub.com/doi/10.1177/0361198118798722>.

WATKINS, K. E. et al. Where Is My Bus? Impact of mobile real-time information on the perceived and actual wait time of transit riders. **Transportation Research Part A: Policy and Practice**, v. 45, n. 8, p. 839–848, out. 2011. Disponível em:

<http://dx.doi.org/10.1016/j.tra.2011.06.010>.

WILLIAMS, B. **Intelligent Transport Systems Standards**. [s.l: s.n.]

WU, W.; LIU, R.; JIN, W. Modelling bus bunching and holding control with vehicle overtaking and distributed passenger boarding behaviour. **Transportation Research Part B: Methodological**, v. 104, p. 175–197, out. 2017. Disponível em: <http://dx.doi.org/10.1016/j.trb.2017.06.019>.

XINGHAO, S. et al. Predicting Bus Real-time Travel Time Basing on both GPS and RFID Data. **Procedia - Social and Behavioral Sciences**, v. 96, n. Cictp, p. 2287–2299, nov. 2013. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S1877042813023847>.

XUAN, Y.; ARGOTE, J.; DAGANZO, C. F. Dynamic bus holding strategies for schedule reliability: Optimal linear control and performance analysis. **Transportation Research Part B: Methodological**, v. 45, n. 10, p. 1831–1845, dez. 2011. Disponível em: <http://dx.doi.org/10.1016/j.trb.2011.07.009>.

YAI, A. K. **Análise e visualização de dados do transporte público de ônibus da cidade de São Paulo**. 2015. Universidade de São Paulo, 2015. Disponível em: <https://linux.ime.usp.br/~andreky/AndreYai_monografia_revisada.pdf>.

YEGNANARAYANA, B. **Artificial Neural Networks**. 19. ed. New Delhi: PHI Learning, 2012.

YIN, T. et al. A prediction model of bus arrival time at stops with multi-routes. **Transportation Research Procedia**, v. 25, p. 4623–4636, 2017. Disponível em: <http://dx.doi.org/10.1016/j.trpro.2017.05.381>.

YOON, H. et al. Smart Itinerary Recommendation Based on User-Generated GPS Trajectories. In: YU, Z. et al. (Ed.). **Ubiquitous Intelligence and Computing**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 19–34.

YU, B. et al. Dynamic Extra Buses Scheduling Strategy in Public Transport. **PROMET - Traffic&Transportation**, v. 27, n. 3, p. 205–216, 26 jun. 2015. Disponível em: <https://traffic.fpz.hr/index.php/PROMTT/article/view/1547>.

YU, B.; YANG, Z.; LI, S. Real-time partway deadheading strategy based on transit service reliability assessment. **Transportation Research Part A: Policy and Practice**, v. 46, n. 8, p. 1265–1279, out. 2012. Disponível em: <http://dx.doi.org/10.1016/j.tra.2012.05.009>.

ZANINI, A. **REDES NEURAIS E REGRESSÃO DINÂMICA: UM MODELO HÍBRIDO PARA PREVISÃO DE CURTO PRAZO DA DEMANDA DE GASOLINA AUTOMOTIVA NO BRASIL**. 2000. PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO

DE JANEIRO, Rio de Janeiro, Brazil, 2000. Disponível em: <http://www.maxwell.vrac.puc-rio.br/Busca_etds.php?strSecao=resultado&nrSeq=7457@1>.

ZHANG, S.; LO, H. K. Two-way-looking self-equalizing headway control for bus operations. **Transportation Research Part B: Methodological**, v. 110, p. 280–301, abr. 2018. Disponível em: <https://doi.org/10.1016/j.trb.2018.02.012>.

ZHU, T. et al. The prediction of bus arrival time using global positioning system data and dynamic traffic information. In: 2011 4th Joint IFIP Wireless and Mobile Networking Conference (WMNC 2011), **Anais**...IEEE, out. 2011. Disponível em: <http://ieeexplore.ieee.org/document/6097232/>.

APÊNDICE A - CÓDIGOS EM R PARA REALIZAR A COLETA CÍCLICA DE DADOS PELAS APIS OLHOVIVO E GOOGLE MAPS

1. Código R para Coleta das posições dos AVLs a cada 30s da linha 4708-10 pela API da SPTrans

```
setwd("C:/ondrv/OneDrive - usp.br/AWSGCP/Data/pos")
direc <- "C:/ondrv/OneDrive - usp.br/AWSGCP/Data/pos"
require("rjson")
require("httr")
require("lubridate")
url_aut_txt <- "http://api.olhovivo.sptrans.com.br/v2.1/Login/Autenticar?token=bb6cf"
linhas <- c("332","33100")
ds <- now()
dta <- Sys.Date()
subdirec <- file.path(direc, dta,"/")
setwd(subdirec)
for (i in 1:100000){
  for (l in linhas){
    url_get_txt <- paste("http://api.olhovivo.sptrans.com.br/v2.1/Posicao/Linha?codigoLinha=",l,sep="")
    result <- tryCatch({
      POST(url_aut_txt)
    }, warning = function(war){
      #print("WARNING")
    }, error = function(err){
      print(date())
    }, finally = {
      dt <- Sys.Date()
      if (dt != dta) {
         dta <- dt
         subdirec <- file.path(direc, dt,"/")
         dir.create(file.path(subdirec))
         setwd(subdirec) }
      file_name <- paste(l,"_",dt,"_pos.json",sep="")
      file_path <- file.path(subdirec, file_name)
      jsdt <- RETRY("GET",url_get_txt,times = 10)
      jsdt <- content(jsdt, "text",encoding = "utf-8")
      jsdt <- fromJSON(jsdt)
      if (length(jsdt) > 1){
        dados_coleta <- jsdt[[2]]
        qtd <- length(dados_coleta)
        if (qtd > 0) {
          for (ip in 1:length(dados_coleta)){
            campo <- dados_coleta[[ip]]
            campo_1 <- toString(campo[1])
            campo_2 <- toString(campo[2])
            campo_3 <- toString(campo[3])
            campo_4 <- toString(campo[4])
            campo_5 <- toString(campo[5])
            txt <- paste(i,ip,date(),l,jsdt[[1]],campo_1,campo_2,campo_3,campo_4,campo_5,sep=";")
            write(txt,paste(subdirec,l,"_",dt,"_posicao.txt",sep=""),append=TRUE)
          }} else {
            write(paste(i,0,date(),l,jsdt[[1]],"No Bus",sep =
";"),paste(subdirec,l,"_",dt,"_posicao.txt",sep=""),append=TRUE)}
      }else{
        write(paste(i,0,date(),l,jsdt,"No Bus",sep =
";"),paste(subdirec,l,"_",dt,"_posicao.txt",sep=""),append=TRUE)
      }})}
  if((i %% 1)== 0){
    dsd <- ds + 30*i
    dn <- now()
    dd <- dsd - dn
    if (dd<0){dd=0}
    Sys.sleep(dd)
    warnings()}}
```

2. Código R para Coleta das previsões de chegada dos AVLs nos pontos de ônibus a cada 30s da linha 4708-10 pela API da SPTrans (Resultado nulo para esta linha específica)

```
setwd("C:/ondrv/OneDrive - usp.br/AWSGCP/Data/prev")
direc <- "C:/ondrv/OneDrive - usp.br/AWSGCP/Data/prev"
require("rjson")
require("httr")
require("lubridate")
url_aut_txt <- "http://api.olhovivo.sptrans.com.br/v2.1/Login/Autenticar?token=bb6cf"
linhas <- c("332","33100")
ds <- now()
dta <- Sys.Date()
subdirec <- file.path(direc, dta,"/")
setwd(subdirec)
for (i in 1:100000){
  for (l in linhas){
    url_get_txt <- paste("http://api.olhovivo.sptrans.com.br/v2.1/Previsao/Linha?codigoLinha=",l,sep="")
    result <- tryCatch({
      POST(url_aut_txt)
    }, warning = function(war){
    }, error = function(err){
      print(date())
    }, finally = {
      dt <- Sys.Date()
      if (dt != dta) {
        dta <- dt
        subdirec <- file.path(direc, dt,"/")
        dir.create(file.path(subdirec))
        setwd(subdirec)}
      file_name <- paste(l,"_",dt,"_prev.json",sep="")
      file_path <- file.path(subdirec, file_name)
      jsdt <- RETRY("GET",url_get_txt,times = 10)
      jsdt <- content(jsdt, "text",encoding = "utf-8")
      jsdt <- fromJSON(jsdt)
      if(length(jsdt) > 1){
        hora_coleta <- jsdt[[1]] #hora API
        qtdpt <- length(jsdt[[2]]) #qtd de pontos
         if(qtdpt>0){
          for (pt in 1:qtdpt){
            codpt <- jsdt[[2]][pt][[1]][1][[1]] #cod ponto
            locpt <- paste(jsdt[[2]][pt][[1]][3][[1]],jsdt[[2]][pt][[1]][4][[1]], sep=" ") #loc
            qtdbus <- length(jsdt[[2]][pt][[1]][5][[1]]) #qtd onibus a chegar no ponto
            if(qtdbus>0){
              for (bs in 1:qtdbus){
                codbus <- jsdt[[2]][pt][[1]][5][[1]][bs][[1]][1][[1]] #cod bus
                horachegada <- jsdt[[2]][pt][[1]][5][[1]][bs][[1]][2][[1]] #hr chegda bus
                abus <- jsdt[[2]][pt][[1]][5][[1]][bs][[1]][3][[1]] #a bus
                horabus <- jsdt[[2]][pt][[1]][5][[1]][bs][[1]][4][[1]] #hora bus
                locbus <-
paste(jsdt[[2]][pt][[1]][5][[1]][bs][[1]][5][[1]],jsdt[[2]][pt][[1]][5][[1]][bs][[1]][6][[1]], sep = " ")
                txt <-
paste(i,date(),hora_coleta,codpt,locpt,qtdbus,codbus,horachegada,abus,horabus,locbus,sep=";")
write(txt,paste(subdirec,l,"_",codpt,"_",dt,"_previsao.txt",sep=""),append=TRUE)
                }}else{
                codbus <- "No Bus"
                horachegada <- "No Bus"
                abus <- "No Bus"
                horabus <- "No Bus"
                locbus <- "No Bus"
                txt <-
paste(i,date(),hora_coleta,codpt,locpt,qtdbus,codbus,horachegada,abus,horabus,locbus,sep=";")
write(txt,paste(subdirec,l,"_",dt,"_",codpt,"_previsao.txt",sep=""),append=TRUE)
                }}else{
                  txt <- paste(i,date(),hora_coleta,qtdpt,"No Prevision",sep=";")
write(txt,paste(subdirec,l,"_",dt,"_previsao.txt",sep=""),append=TRUE)}
      }else{
        txt <- paste(i,date(),jsdt,"No Prevision",sep=";")
        write(txt,paste(subdirec,l,"_",dt,"_previsao.txt",sep=""),append=TRUE)}}))}
  if((i %% 1)== 0){
    dsd <- ds + 30*i
    dn <- now()
    dd <- dsd - dn
    if (dd<0){dd=0}
    Sys.sleep(dd)
    warnings()}}
```

3. Código R para coleta dos tempos de trajeto entre cada trecho da linha 4708-10-0 a cada 1 minuto pela API do Google Maps *directions*

```
setwd("C:/ondrv/OneDrive - usp.br/AWSGCP/Data/maps")
direc <- "C:/ondrv/OneDrive - usp.br/AWSGCP/Data/maps"
require("rjson")
require("httr")
require("lubridate")
key <- "&key=AIzaSyDfNCJSPkU14FmtMY3U"
trechos <- c("-23.6398248770259,-46.607308893331","-23.6370503433708,-46.6071614565125","-
23.6352490309203,-46.6060593591848","-23.6328908584029,-46.6050373925733","-23.6304977514816,-
46.6038687602859","-23.6294293937468,-46.6047248879112","-23.6272805616842,-46.6063075256665","-
23.6249038312799,-46.6070155404753","-23.6230255317148,-46.6076101830839","-23.6209552785041,-
46.6082185361581","-23.6183546829382,-46.6094826699595","-23.618161790278,-46.6096885741109","-
23.6174264740262,-46.6102740696322","-23.614298869195,-46.6091920964885","-23.6137285082604,-
46.6103523570251","-23.6121143257974,-46.6106921074617","-23.6102606030973,-46.6120476310923","-
23.6085093702536,-46.6134921187606","-23.6051821025611,-46.6144398611718","-23.6033780034802,-
46.6147345882342","-23.6013352406393,-46.6151920792878","-23.5997563636149,-46.6158792116795","-
23.5982853004579,-46.6158408373559","-23.5964170809876,-46.6165202977249","-23.5955448934394,-
46.617134202324","-23.5954838261225,-46.6174758224958","-23.5942830576835,-46.6197471042453","-
23.5927969162167,-46.6198686148975","-23.592821821463,-46.6213078936006","-23.5951072360281,-
46.6237893177635","-23.5950202045737,-46.6268998697663","-23.5931874507308,-46.6297230843116","-
23.5926269886946,-46.6305273188054","-23.5904938971378,-46.6322156913943","-23.5906284496643,-
46.6326373943149","-23.5893,-46.6336")
ds <- now()
dta <- Sys.Date()
subdirec <- file.path(direc, dta,"/")
setwd(subdirec)
for (i in 1:100000){
  if (hour(dn) > 3) {
  for (tr in 1:(length(trechos)-1)){
    url_get_txt <-
paste("https://maps.googleapis.com/maps/api/directions/json?origin=",trechos[tr],"&destination=",trechos[tr
+1],"&departure_time=now", key, sep="")
      dt <- Sys.Date()
      if (dt != dta) {
        dta <- dt
        subdirec <- file.path(direc, dt,"/")
        dir.create(file.path(subdirec))
        setwd(subdirec)}
      file_name <- paste("332_53537_",dt,"_",tr,"_mapsdirect.json",sep="")
      file_path <- file.path(subdirec, file_name)
      jsdt <- RETRY("GET",url_get_txt,times = 10)
      jsdt <- content(jsdt, "text",encoding = "utf-8")
      jsdt <- fromJSON(jsdt)
      if (length(jsdt[[1]])>1){
          de <- paste(jsdt[2][[1]][[1]][[3]][[1]][[1]][[7]][[1]][1],jsdt[2][[1]][[1]][3][[1]][[1]][[7]][[1]][2],
sep = " ") #from
          para <-
paste(jsdt[2][[1]][[1]][3][[1]][[1]][5][[1]][1],jsdt[2][[1]][[1]][3][[1]][[1]][5][[1]][2], sep = " ") #to
          traf <- jsdt[2][[1]][[1]][3][[1]][[1]][3][[1]][2][[1]] #traffic
          distan <- jsdt[2][[1]][[1]][3][[1]][[1]][1][[1]][2][[1]] #dist
          dur <- jsdt[2][[1]][[1]][3][[1]][[1]][2][[1]][2][[1]] #duration
          txt <- paste(i,date(),trechos[tr],trechos[tr+1],de,para,distan,dur,traf,sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_",tr,"mapsdirect.txt",sep=""),append=TRUE)
      }else{
        txt <- paste(i,date(),trechos[tr],trechos[tr+1],jsdt[[1]],sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_",tr,"mapsdirect.txt",sep=""),append=TRUE)}
  file_name <- paste(dt,"all_mapsdirect.json",sep="")
  file_path <- file.path(subdirec, file_name)
  jsdt <- RETRY("GET",url_get_txt2,times = 10)
  jsdt <- content(jsdt, "text",encoding = "utf-8")
  jsdt <- fromJSON(jsdt)
  if (length(jsdt[[1]])>1){
  de<-paste(jsdt[2][[1]][[1]][3][[1]][[1]][7][[1]][1],jsdt[2][[1]][[1]][3][[1]][[1]][7][[1]][2],sep = " ")
  para<-paste(jsdt[2][[1]][[1]][3][[1]][[1]][5][[1]][1],jsdt[2][[1]][[1]][3][[1]][[1]][5][[1]][2],sep=" ")
  traf <- jsdt[2][[1]][[1]][3][[1]][[1]][3][[1]][2][[1]] #traffic
  distan <- jsdt[2][[1]][[1]][3][[1]][[1]][1][[1]][2][[1]] #dist
  dur <- jsdt[2][[1]][[1]][3][[1]][[1]][2][[1]][2][[1]] #duration
  txt <- paste(i,date(),de,para,distan,dur,traf,sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_","all_mapsdirect.txt",sep=""),append=TRUE)
  }else{
    txt <- paste(i,date(),trechos[tr],trechos[tr+1],jsdt[[1]],sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_",tr,"mapsdirect.txt",sep=""),append=TRUE)}
  if((i %% 1)== 0){
    dsd <- ds + 60*i
    dn <- now()
    dd <- dsd - dn
    if (dd<0){dd=0}
    Sys.sleep(dd)
    warnings()}}
```

4. Código R para coleta dos tempos de trajeto pessimista entre cada trecho da linha 4708-10-0 a cada 1 minuto pela API do Google Maps *directions*

```
setwd("C:/ondrv/OneDrive - usp.br/AWSGCP/Data/mapspess")
direc <- "C:/ondrv/OneDrive - usp.br/AWSGCP/Data/mapspess"
require("rjson")
require("httr")
require("lubridate")
key <- "&traffic_model=pessimistic&key=AIzaSyDfNCJSPkU14FmtMY3U"
trechos <- c("-23.6398248770259,-46.607308893331","-23.6370503433708,-46.6071614565125","-
23.6352490309203,-46.6060593591848","-23.6328908584029,-46.6050373925733","-23.6304977514816,-
46.6038687602859","-23.6294293937468,-46.6047248879112","-23.6272805616842,-46.6063075256665","-
23.6249038312799,-46.6070155404753","-23.6230255317148,-46.6076101830839","-23.6209552785041,-
46.6082185361581","-23.6183546829382,-46.6094826699595","-23.618161790278,-46.6096885741109","-
23.6174264740262,-46.6102740696322","-23.614298869195,-46.6091920964885","-23.6137285082604,-
46.6103523570251","-23.6121143257974,-46.6106921074617","-23.6102606030973,-46.6120476310923","-
23.6085093702536,-46.6134921187606","-23.6051821025611,-46.6144398611718","-23.6033780034802,-
46.6147345882342","-23.6013352406393,-46.6151920792878","-23.5997563636149,-46.6158792116795","-
23.5982853004579,-46.6158408373559","-23.5964170809876,-46.6165202977249","-23.5955448934394,-
46.617134202324","-23.5954838261225,-46.6174758224958","-23.5942830576835,-46.6197471042453","-
23.5927969162167,-46.6198686148975","-23.592821821463,-46.6213078936006","-23.5951072360281,-
46.6237893177635","-23.5950202045737,-46.6268998697663","-23.5931874507308,-46.6297230843116","-
23.5926269886946,-46.6305273188054","-23.5904938971378,-46.6322156913943","-23.5906284496643,-
46.6326373943149","-23.5893,-46.6336")
ds <- now()
dta <- Sys.Date()
subdirec <- file.path(direc, dta,"/")
setwd(subdirec)
for (i in 1:100000){
  if (hour(dn) > 3) {
  for (tr in 1:(length(trechos)-1)){
    url_get_txt <-
paste("https://maps.googleapis.com/maps/api/directions/json?origin=",trechos[tr],"&destination=",trechos[tr
+1],"&departure_time=now", key, sep="")
      dt <- Sys.Date()
      if (dt != dta) {
        dta <- dt
        subdirec <- file.path(direc, dt,"/")
        dir.create(file.path(subdirec))
        setwd(subdirec)}
      file_name <- paste("332_53537_",dt,"_",tr,"_mapsdirect.json",sep="")
      file_path <- file.path(subdirec, file_name)
      jsdt <- RETRY("GET",url_get_txt,times = 10)
      jsdt <- content(jsdt, "text",encoding = "utf-8")
      jsdt <- fromJSON(jsdt)
      if (length(jsdt[[1]])>1){
          de <- paste(jsdt[2][[1]][[1]][[3]][[1]][[1]][[7]][[1]][1],jsdt[2][[1]][[1]][[3]][[1]][[1]][[7]][[1]][2],
sep = " ")#de
          para <-
paste(jsdt[2][[1]][[1]][[3]][[1]][[1]][5][[1]][1],jsdt[2][[1]][[1]][[3]][[1]][[1]][5][[1]][2], sep = " ")#p
          traf <- jsdt[2][[1]][[1]][[3]][[1]][[1]][[3]][[1]][2][[1]] #traffic
          distan <- jsdt[2][[1]][[1]][[3]][[1]][[1]][1][[1]][2][[1]] #dist
          dur <- jsdt[2][[1]][[1]][[3]][[1]][[1]][2][[1]][2][[1]] #duration
          txt <- paste(i,date(),trechos[tr],trechos[tr+1],de,para,distan,dur,traf,sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_",tr,"mapsdirect.txt",sep=""),append=TRUE)
      }else{
        txt <- paste(i,date(),trechos[tr],trechos[tr+1],jsdt[[1]],sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_",tr,"mapsdirect.txt",sep=""),append=TRUE)}
  file_name <- paste(dt,"all_mapsdirect.json",sep="")
  file_path <- file.path(subdirec, file_name)
  jsdt <- RETRY("GET",url_get_txt2,times = 10)
  jsdt <- content(jsdt, "text",encoding = "utf-8")
  jsdt <- fromJSON(jsdt)
  if (length(jsdt[[1]])>1){
  de <- paste(jsdt[2][[1]][[1]][[3]][[1]][[1]][[7]][[1]][1],jsdt[2][[1]][[1]][[3]][[1]][[1]][[7]][[1]][2],sep=" ")
#from
  para <- paste(jsdt[2][[1]][[1]][[3]][[1]][[1]][5][[1]][1],jsdt[2][[1]][[1]][[3]][[1]][[1]][5][[1]][2],sep="
") #to
  traf <- jsdt[2][[1]][[1]][[3]][[1]][[1]][[3]][[1]][2][[1]] #traffic
  distan <- jsdt[2][[1]][[1]][[3]][[1]][[1]][1][[1]][2][[1]] #dist
  dur <- jsdt[2][[1]][[1]][[3]][[1]][[1]][2][[1]][2][[1]] #duration
  txt <- paste(i,date(),de,para,distan,dur,traf,sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_","all_mapsdirect.txt",sep=""),append=TRUE)
  }else{
    txt <- paste(i,date(),trechos[tr],trechos[tr+1],jsdt[[1]],sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_",tr,"mapsdirect.txt",sep=""),append=TRUE)}
  if((i %% 1)== 0){
    dsd <- ds + 60*i
    dn <- now()
    dd <- dsd - dn
    if (dd<0){dd=0}
    Sys.sleep(dd)
    warnings()}}
```

5. Código R para coleta da previsão dos tempos de trajeto entre cada trecho da linha 4708-10-0 a cada 1 minuto pela API do Google Maps *directions*

```
setwd("C:/ondrv/OneDrive - usp.br/AWSGCP/Data/mapsprev")
direc <- "C:/ondrv/OneDrive - usp.br/AWSGCP/Data/mapsprev"
require("rjson")
require("httr")
require("lubridate")
key <- "&key=AIzaSyDfNCJSPkU14FmtMY3U"
url_get_txt <- paste("https://maps.googleapis.com/maps/api/directions/json?origin=-23.6398248770259%2C-46.607308893331&destination=-23.5893%2C-46.6336&waypoints=via:-23.630497,%20-46.603868|via:-23.623025,%20-46.607610|via:-23.610260,%20-46.612047|via:-23.601335,%20-46.615192|via:-23.595483,%20-46.617475|via:-23.592626,%20-46.630527|via:-23.590628,%20-46.632637&departure_time=", sep="")
ds <- now()
dta <- Sys.Date()
subdirec <- file.path(direc, dta,"/")
setwd(subdirec)
as.integer(ds)
for (i in 1:100000){
  if (hour(dn) > 3) {
  dt <- Sys.Date()
  if (dt != dta) {
    dta <- dt
    subdirec <- file.path(direc, dt,"/")
    dir.create(file.path(subdirec))
    setwd(subdirec)}
  for (j in 0: 4){
    deptime <- as.integer(now()) + j*60*5
    get_txt <- paste(url_get_txt, deptime, key, sep="")
    file_name <- paste("332_53537_",dt,"_", j ,"_mapsdirect.json",sep="")
    file_path <- file.path(subdirec, file_name)
    jsdt <- RETRY("GET",get_txt,times = 10)
    jsdt <- content(jsdt, "text",encoding = "utf-8")
    jsdt <- fromJSON(jsdt)
    if (length(jsdt[[1]])>1){
      de <- paste(jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][7][[1]][[1],jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][7][[1]][2], sep = " ") #from
      para <- paste(jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][5][[1]][[1],jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][5][[1]][2], sep = " ") #to
      traf <- jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][3][[1]][[1]][2][[1]] #traffic
      distan <- jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][1][[1]][[2]][[1]] #dist
      dur <- jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][2][[1]][[2]][[1]] #duration
      txt <- paste(i,date(),j,deptime,de,para,distan,dur,traf,sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_",j,"mapsdirect.txt",sep=""),append=TRUE)
    }else{
      txt <- paste(i,date(),j,deptime,jsdt[[1]],sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_",j,"mapsdirect.txt",sep=""),append=TRUE)}}}
  if((i %% 1)== 0){
    dsd <- ds + 60*i
    dn <- now()
    dd <- dsd - dn
    if (dd<0){dd=0}
    Sys.sleep(dd)
    warnings()}}
```

6. Código R para coleta da previsão pessimista dos tempos de trajeto entre cada trecho da linha 4708-10-0 a cada 1 minuto pela API do Google Maps *directions*

```
setwd("C:/ondrv/OneDrive - usp.br/AWSGCP/Data/mapsprevpess")
direc <- "C:/ondrv/OneDrive - usp.br/AWSGCP/Data/mapsprevpess"
require("rjson")
require("httr")
require("lubridate")
key <- "&key=AIzaSyDfNCJSPkU14FmtMY3U"
url_get_txt <- paste("https://maps.googleapis.com/maps/api/directions/json?origin=-23.6398248770259%2C-
46.607308893331&destination=-23.5893%2C-46.6336&waypoints=via:-23.630497,%20-46.603868|via:-23.623025,%20-
46.607610|via:-23.610260,%20-46.612047|via:-23.601335,%20-46.615192|via:-23.595483,%20-46.617475|via:-
23.592626,%20-46.630527|via:-23.590628,%20-46.632637&traffic_model=pessimistic&departure_time=", sep="")
ds <- now()
dta <- Sys.Date()
subdirec <- file.path(direc, dta,"/")
setwd(subdirec)
as.integer(ds)
for (i in 1:100000){
  if (hour(dn) > 3) {
  dt <- Sys.Date()
  if (dt != dta) {
    dta <- dt
    subdirec <- file.path(direc, dt,"/")
    dir.create(file.path(subdirec))
    setwd(subdirec)}
  for (j in 0: 4){
    deptime <- as.integer(now()) + j*60*5
    get_txt <- paste(url_get_txt, deptime, key, sep="")
    file_name <- paste("332_53537_",dt,"_", j ,"_mapsdirect.json",sep="")
    file_path <- file.path(subdirec, file_name)
    jsdt <- RETRY("GET",get_txt,times = 10)
    jsdt <- content(jsdt, "text",encoding = "utf-8")
    jsdt <- fromJSON(jsdt)
    if (length(jsdt[[1]])>1){
      de <- paste(jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][7][[1]][[1],jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][7][[1]][2], sep
= " ") #from
      para <- paste(jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][5][[1]][[1],jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][5][[1]][2],
sep = " ") #to
      traf <- jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][3][[1]][[1]][2][[1]] #traffic
      distan <- jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][1][[1]][[1]][2][[1]] #dist
      dur <- jsdt[2][[1]][[1]][[1]][3][[1]][[1]][[1]][2][[1]][[1]][2][[1]] #duration
      txt <- paste(i,date(),j,deptime,de,para,distan,dur,traf,sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_",j,"mapsdirect.txt",sep=""),append=TRUE)
    }else{
      txt <- paste(i,date(),j,deptime,jsdt[[1]],sep=";")
write(txt,paste(subdirec,"332_53537_",dt,"_",j,"mapsdirect.txt",sep=""),append=TRUE)}}}
  if((i %% 1)== 0){
    dsd <- ds + 60*i
    dn <- now()
    dd <- dsd - dn
    if (dd<0){dd=0}
    Sys.sleep(dd)
    warnings()}}
```

APÊNDICE B - CÓDIGO PYTHON PARA TREINAMENTO DA RNA

```
from __future__ import absolute_import, division, print_function, unicode_literals
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import tensorflow as tf
from array import array
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.models import load_model
from tensorflow import keras
from tensorflow.keras import layers
from keras import optimizers
from keras import backend as K
from tensorflow import set_random_seed
set_random_seed(2)
import seaborn as sns

width = 10
# activation = tf.nn.relu
# activation = keras.activations.relu
# activation = keras.activations.sigmoid
# activation = 'relu'
activation = 'sigmoid'
# activation = 'softmax'
# activation = 'tanh'
# activation = 'exponential'
# activation = 'linear'

# optimizer = 'tf.keras.optimizers.RMSprop(0.001)'
# optimizer = 'RMSprop'
optimizer = 'Nadam'
# optimizer = 'SGD'
# optimizer = 'Adagrad'
# optimizer = 'Adadelta'
# optimizer = 'Adam'
# optimizer = 'Adamax'

# loss = tf.keras.metrics.mean_squared_error
# loss = 'mse'
loss = 'mae'
# loss = 'mape'
patience = 4
normentry = 1
normout = 1
cliper = 0

# config = tf.ConfigProto()
# config.gpu_options.allow_growth = True
# keras.backend.set_session(tf.Session(config=config))

losses = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
mae = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
mse = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
rmse = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
mape = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
correlations = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]


def plot_history(history):
    hist = pd.DataFrame(history.history)
    hist['epoch'] = history.epoch
    plt.figure()
    plt.xlabel('Epoch')
    plt.ylabel('Mean Abs Error [s]')
    plt.plot(hist['epoch'], hist['mean_absolute_error'],
             label='Train Error')
    plt.plot(hist['epoch'], hist['val_mean_absolute_error'],
             label='Val Error')
    # plt.title('Model Mean Square Error')
    # plt.ylim([0, 0.5])
    plt.legend()
    plt.savefig('hist_mae37', transparent=True, dpi=1500)
    plt.figure()
    plt.xlabel('Epoch')
    plt.ylabel('Root Mean Square Error [s]')
    plt.plot(hist['epoch'], hist['rmse'],
             label='Train Error')
    plt.plot(hist['epoch'], hist['val_rmse'],
             label='Val Error')
    # plt.ylim([0,20])
```

```
    plt.legend()
    plt.savefig('hist_rms37', transparent=True, dpi=1500)
    plt.show()

def norm(x):
    return (x - train_stats['min']) / (train_stats['max'] - train_stats['min'])
    # return (x - train_stats['mean']) / train_stats['std']

def normlab(x):
    return (x - train_stats_labels['min']) / (train_stats_labels['max'] - train_stats_labels['min'])
    # return (x - train_stats_labels['mean']) / train_stats_labels['std']

def denorm(y):
    # return (y * train_stats_labels['std']) + train_stats_labels['mean']
    return (y * (train_stats_labels['max'] - train_stats_labels['min'])) + train_stats_labels['min']

def root_mean_squared_error(y_true, y_pred):
    return K.sqrt(K.mean(K.square(y_pred - y_true)))

def rootMS(predictions, targets):
    return np.sqrt(((predictions - targets) ** 2).mean())

def mean_absolute_percentage_error(y_true, y_pred):
    y_true, y_pred = np.array(y_true), np.array(y_pred)
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100

def clip(df):
    t = train_stats['mean']
    return df.clip(t/3, t*3, axis=1)


dataset = pd.read_csv('37.csv', decimal='.', sep=',')

dataset.isna().sum()
train_dataset = dataset.head(1256)

test_dataset = dataset.drop(train_dataset.index)
test_dataset.to_csv('new_inputs37.csv', index=False, sep=';', decimal=',')


# sns_plot = sns.pairplot(
#     train_dataset[i][["Scip" + str(i - 1), "Posx" + str(i), "Mapsx" + str(i), "MapsPessx" + str(i), "y" +
str(i)]],
#     diag_kind="kde")
# sns_plot.savefig("seaborn" + str(i) + ".png", transparent=True, dpi=200)

train_stats = train_dataset.describe()

train_stats = train_stats.transpose()

if cliper == 1:
    train_dataset = clip(train_dataset)

train_stats = train_dataset.describe()

train_stats_labels = (train_stats.pop("y37"))
train_stats = train_stats.transpose()
train_stats_labels = train_stats_labels.transpose()
train_stats.to_csv('train_stats37.csv')

train_labels = (train_dataset.pop("y37"))
test_labels = (test_dataset.pop("y37"))
test_labels.to_csv('test_labels37.csv', index=False, sep=';', decimal=',')

normed_train_data = (norm(train_dataset))
normed_test_data = (norm(test_dataset))

if normout == 1:
    normed_train_labels = (normlab(train_labels))
    model = (Sequential([
        Dense(units=width, activation=activation, input_dim=len(train_dataset.keys())),
        # Dropout(0.2),
        # Dense(units=width, activation=activation),
        Dense(units=width, activation=activation),
        Dense(units=1, activation=activation)
    ]))
else:
    normed_train_labels = (train_labels)
    model = (Sequential([
        Dense(units=width, activation=activation, input_dim=len(train_dataset.keys())),
        # Dropout(0.2),
        # Dense(units=width, activation=activation),
        Dense(units=width, activation=activation),
        Dense(units=1)
    ]))
```

```
model.compile(loss=loss, optimizer=optimizer,metrics=['mean_absolute_error', 'mean_squared_error',
tf.keras.metrics.RootMeanSquaredError(name='rmse')])
early_stop = (keras.callbacks.EarlyStopping(monitor='val_loss', min_delta=0, patience=patience))

# model[i].summary()
# example_batch = normed_train_data[i][:10]
# example_result = model[i].predict(example_batch)
# example_result

for i in range(0, 20):
    history = (model.fit(normed_train_data, normed_train_labels, epochs=2000, validation_split=0.2,
verbose=0,callbacks=[early_stop]))
    # plot_history(history)

    hist = (pd.DataFrame(history.history))
    hist.to_csv('hist37.csv', index=False, sep=';', decimal=',')

    losses[i] = (model.evaluate(normed_test_data, test_labels, verbose=0)[0])
    mae[i] = (model.evaluate(normed_test_data, test_labels, verbose=0)[1])
    mse[i] = (model.evaluate(normed_test_data, test_labels, verbose=0)[2])
    rmse[i] = (model.evaluate(normed_test_data, test_labels, verbose=0)[3])

    test_predictions = (model.predict(normed_test_data).flatten())

    if normout == 1:
        test_predictions = denorm(test_predictions)

    test_predictions2 = (model.predict(normed_test_data))
    if normout == 1:
        test_predictions2 = denorm(test_predictions2)

    test_predictions2 = pd.DataFrame(test_predictions2)
    test_predictions2.to_csv('test_predictions37.csv', index=False, sep=';', decimal=',')
    correlations[i] = (np.corrcoef(test_labels, test_predictions)[0, 1])
    mape[i] = (mean_absolute_percentage_error(test_labels, test_predictions))

lossesm = np.mean(losses)
maem = np.mean(mae)
msem = np.mean(mse)
rmsem = np.mean(rmse)
mapem = np.mean(mape)
correlationsm = np.mean(correlations)

print('Testing set MSE: {:5.5f} s^2'.format(msem))
print("Testing set Mean Abs Error: {:5.5f} s".format(maem))
print("Testing set RMS: {:5.5f} s".format(rmsem))
print("Testing set corr: {:5.5f} s".format(correlationsm))

# plt.scatter(test_labels, test_predictions)
# plt.xlabel('True Values [s]')
# plt.ylabel('Predictions [s]')
# plt.axis('equal')
# plt.axis('square')
# plt.xlim([0, plt.xlim()[1]])
# plt.ylim([0, plt.ylim()[1]])
# _ = plt.plot([-5000, 5000], [-5000, 5000])
# plt.savefig('correlation37', transparent=True, dpi=1500)
# plt.show()
# error = (test_predictions - test_labels)
# plt.hist(error, bins=25)
# plt.xlabel("Prediction Error [s]")
# _ = plt.ylabel("Count")
# plt.savefig('error37', transparent=True, dpi=1500)
# plt.show()
# error.to_csv('error37.csv', index=False, sep=';', decimal=',')

# print(normed_test_data.columns.values[26:])
print("Testing set RMS: {:5.5f} s".format(rmsem))
print('corr 37 = ', correlationsm)
print('mape 37 = ', mapem)
model.save('calibration_nn37.h5')
jsonmodel = (model.to_json())
with open('model37.json', 'w') as json_file:
    json_file.write(jsonmodel)
# del model
np.savetxt('correlations.csv', (correlationsm,), fmt='%.20f')
np.savetxt('losses.csv', (lossesm,), fmt='%.20f')
np.savetxt('mse.csv', (msem,), fmt='%.20f')
np.savetxt('rmse.csv', (rmsem,), fmt='%.20f')
np.savetxt('mape.csv', (mapem,), fmt='%.20f')
rmse2 = rootMS(test_predictions, test_labels)
np.savetxt('rmse2.csv', (rmse2,), fmt='%.20f')
```

APÊNDICE C - REPOSITÓRIO DE DADOS ABERTOS

Para reproduzir os resultados dessa pesquisa, os dados coletados podem ser acessados em:

https://github.com/gustlarsen/ANNBusTravelTimePrediction

APÊNDICE D - APLICATIVO ANDROID PARA COLETA CÍCLICA DE DADOS DE GEOLOCALIZAÇÃO DOS ÔNIBUS DA SPTRANS

Após a finalização da pesquisa foi aplicado os conceitos aprendidos para elaborar um aplicativo funcional para smartphones Android que é capaz de coletar dados das geolocalizações dos ônibus da SPTrans ciclicamente, o aplicativo desenvolvido pode ser acessado no seguinte link:

https://play.google.com/store/apps/details?id=com.gustlarsen.AVLSPDataCollect