UNIVERSIDADE FEDERAL DO PAMPA

Lucas Alves São Mateus

CLASSIFICAÇÃO DE ATIVIDADES DO USUÁRIO DO TRANSPORTE PÚBLICO POR MEIO DE *SMARTPHONES*

Alegrete-RS
2021

Lucas Alves São Mateus

CLASSIFICAÇÃO DE ATIVIDADES DO USUÁRIO DO TRANSPORTE PÚBLICO POR MEIO DE *SMARTPHONES*

Projeto de Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Marcelo Resende Thielo

Alegrete-RS
2021

LUCAS ALVES SÃO MATEUS

CLASSIFICAÇÃO DE ATIVIDADES DO USUÁRIO DO TRANSPORTE PÚBLICO ATRAVÉS DE SMARTPHONES

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Trabalho de Conclusão defendido e aprovado em: 08/10/2021

Banca examinadora:

_____

Prof. Dr. Marcelo Resende Thielo

Orientador

UNIPAMPA

_____

Profa. Dra. Amanda Meincke Melo

UNIPAMPA

_____

Prof. Dr. Claudio Schepke

UNIPAMPA

Assinado eletronicamente por **CLAUDIO SCHEPKE**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 08/10/2021, às 14:38, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **MARCELO RESENDE THIELO**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 08/10/2021, às 14:44, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **AMANDA MEINCKE MELO**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 08/10/2021, às 14:54, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

A autenticidade deste documento pode ser conferida no site https://sei.unipampa.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **0635382** e o código CRC **8C572F92**.

AGRADECIMENTOS

Ao meu Deus, que participou como uma variável independente, interferindo para mudar a minha série temporal. Para isso, utilizavam os meus pais, que desde criança acreditaram nos meus objetivos, a minha esposa que sempre compreendeu as minhas ausências. Além disso, não posso deixar de registrar o sorriso encantador da Leila. Obrigado filha, por ser um motivador para minha vida.

Antes tem o seu prazer na
lei do Senhor, e na sua lei
medita de dia e de noite.
(Bíblia Sagrada, Salmos 1:2)

RESUMO

Mobilidade deveria ser um direito universal. Porém, os conceitos de cidade e mobilidade não estão completamente em harmonia. No Brasil, boa parte do tempo dos trabalhadores é desperdiçado no trânsito caótico das cidades. O transporte público está longe de ser eficiente e o poder público está aquém deste desafio. Diante disso, vários trabalhos propõem soluções tecnológicas em busca da mobilidade urbana com foco na predição do tempo de viagem em tempo real. Para atingir esse objetivo é necessário um aprofundamento no estudo de classificação das atividades relacionadas com os meios de locomoção utilizados. Atualmente, há soluções com grande potencial utilizando redes móveis e *smartphones* por meio de vários algoritmos que possibilitam a classificação com acurácia. Porém, ainda assim, cidades brasileiras estão distantes de um transporte público inteligente. O objetivo deste trabalho é aprofundar o conhecimento sobre classificação das atividades relacionadas ao transporte público, com apoio do algoritmo J48 implementado no Weka, usando os dados do acelerômetro, com vistas a preservar a privacidade do usuário.

Palavras chaves: Classificação. J48.Transporte público.

ABSTRACT

Mobility should be a universal right. However, the concepts of city and mobility are not completely in harmony. In Brazil, much of the workers' time is wasted in chaotic city traffic. Public transport is far from being efficient and the public authorities fall short of this challenge. Therefore, several works propose technological solutions in search of urban mobility with a focus on real-time travel time prediction. To achieve this goal, it is necessary to deepen the study of classification of activities related to the means of locomotion used. Currently, there are solutions with great potential using mobile networks and *smartphones* through several algorithms that enable the classification with accuracy. However, even so, Brazilian cities are far from intelligent public transport. The objective of this work is to deepen the knowledge about the classification of activities related to public transport, through the J48 algorithm implemented in Weka, using accelerometer data, in order to preserve user privacy.

Keywords: Classification. J48.Public transport.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

J48	Algoritmo baseado no C4.5 implementado no Weka
MAPE	Erro Absoluto Médio Percentual
FK	Filtro de Kalman
GPS	Global Positioning System (Sistema de Posicionamento Global)
MH	Média histórica
MMQ	Método dos Mínimos Quadrados
RN	Rede Neural
RNA	Rede Neural Artificial
SOMArt	*System Bus Automatically Monitors in real time*
FPR	Taxa de Falsos Positivos
Weka	*Waikato Environment for Knowledge Analysis*

SUMÁRIO

1 INTRODUÇÃO

Cidade e mobilidade são indissociáveis. Porém, no Brasil, a política do transporte público é insuficiente e insustentável. Além disso, nos últimos anos, o uso de transporte individual sofreu um aumento significativo - o exemplo maior aconteceu na cidade do Rio de Janeiro cuja frota, em apenas 12 anos, cresceu 73%. Esse fenômeno provocou uma imobilização urbana generalizada (PAULA; BARTELT, 2016).

Um dado que ilustra essa imobilização é o aumento do tempo médio de deslocamento em dez regiões do Brasil (de 38,1 para 43,3 minutos) em um período compreendido de 2008 até 2016 (PAULA; BARTELT, 2016). Os grandes eventos que ocorreram no Brasil recentemente, como a Copa do Mundo e os Jogos Olímpicos, contribuíram com projetos para mobilidade urbana. Todavia, de acordo com PAULA e BARTELT (2016), as estratégias adotadas podem ter sido equivocadas pois enfatizaram mais o monitoramento do cidadão do que o planejamento urbano.

> ...os modelos e estratégias territoriais escolhidas deixam dúvida se esses investimentos serão capazes de reverter em tempo razoável o caos e os efeitos negativos sobre as condições de vida da população decorrentes dos anos de ausência de ações no campo da mobilidade urbana, em especial do transporte público de massa."(PAULA; BARTELT, 2016)

Apesar do cenário negativo, PAULA e BARTELT (2016) afirma que os usuários de transporte individual usariam os meios de locomoção públicos, se os mesmos, fossem de qualidade. Diante disso, esses autores apontam alguns problemas levantados pelos usuários:

> ...os principais problemas apontados pelos os entrevistados, de modo geral, foram a lotação e o tempo de espera, mas com índices superiores no começo do trajeto, ou seja, nos bairros de moradia. Essa etapa também têm maiores níveis de problemas não relacionados diretamente com o sistema de transporte, mas sim com infraestrutura urbana, como insegurança,falta de urbanização e iluminação. (PAULA; BARTELT, 2016)

Nas cidades de pequeno porte, como Alegrete-RS, o transporte público encontra as mesmas dificuldades em relação ao resto do Brasil. Faltam políticas públicas, as frotas de ônibus são antigas e sem acessibilidade no seu conceito amplo (EMQUESTAO, 2016).

As informações essenciais do transporte público devem ser de fácil acesso aos passageiros, bem como os horários do transporte. Todavia, essas informações básicas nem sempre estão acessíveis. Em várias cidades esse problema persiste e, de modo que não é diferente na 3º Capital Farroupilha. Um exemplo disso é que, ao acessar o site da prefeitura, simplesmente os dados não foram carregados. Nos dias de hoje, informação em tempo real é crucial para qualquer setor.

Diante dos problemas levantados, devem ser levados em conta aspectos cruciais para melhorar o serviço prestado. Primeiro, o conforto, e o segundo, previsibilidade de acordo com PAULA e BARTELT (2016). Neste trabalho a previsibilidade será dividida em duas fases: classificação da atividade e predição da hora da chegada.

No trabalho proposto, a contribuição é no primeiro enfoque - classificação da atividade. Para isso, será demonstrado no decorrer desse trabalho, que o algoritmo J48 tem boa eficiência na classificação das atividades(correr, andar de ônibus e etc) dos usuários.

Para alcançar o objetivo, foram aproveitadas as tecnologias presentes no dia a dia dos passageiros, como *smartphones* e redes móveis. Porém, com objetivo de preservar a privacidade do usuário, o dado fornecido é somente do acelerômetro. Além disso, sem a necessidade de intervenção física (como ocorreu em Porto Alegre-RS (LADEIRA; MICHEL; SENNA, 2011)) e sem a participação de órgãos públicos ou empresas.

Os celulares dos usuários são múltiplas fontes de dados que podem ser transformados em informações cruciais para atender com eficiência o transporte coletivo - intensidade do tráfego, velocidade média, predição da hora de chegada e outras informações que podem ser fornecidas sem a criação de uma nova estrutura.

1.1 OBJETIVO

O objetivo deste trabalho é analisar a eficiência do algoritmo J48 implementado no Weka(*Waikato Environment for Knowledge Analysis*). Esse algoritmo é capaz de realizar a classificação das atividades em cidades de pequeno porte, como Alegrete-RS.

O conceito de atividade neste trabalho é um conjunto de tarefas relacionadas ao deslocamento do indivíduo portando um celular. Neste Trabalho, foram definidas oito atividades que possuem correlação com o transporte - Caminhar, correr, utilizar um carro, um ônibus, um trem, um metrô, uma bicicleta ou simplesmente, ficar ”parado”(na espera de um ponto de ônibus, dentro de casa ou prédio).

Dado o objetivo geral, será necessário alcançar os seguintes objetivos específicos:

- *Identificar as características básicas que compõem o transporte público, visando verificar as condições que podem atrair o usuário.*;
- *Selecionar, a partir de uma revisão, projetos com o mesmo objetivo*;
- *Selecionar o algoritmo J48 e adicioná-lo ao uma aplicação no Android Studio*;
- *Realizar a análise dos resultados de classificação a fim de verificar se o desempenho corresponde à necessidade do projeto*;

1.2 ORGANIZAÇÃO DO TRABALHO

Este trabalho está organizado da seguinte forma: O capítulo 2 apresentam conceitos e técnicas que são importantes para a compreensão deste trabalho - conceitos matemáticos, como séries temporais e modelos de previsão. Neste último, serão abordadas as técnicas de regressão linear, Filtro de Kalman e Redes Neurais. Com essa abordagem será possível a compreensão dos trabalhos relacionados. Em seguida, são apresentadas as ferramentas úteis - *Firebase* e *Android Studio* e Weka - *Waikato Environment for Knowledge Analysis*.

Após a explanação teórica, no capítulo 3 é apresentada uma síntese dos trabalhos que contribuíram, devido à sua relação com o tema.

Logo após, é apresentado o desenvolvimento do trabalho no capítulo 4, descrevendo sua proposta com mais detalhes e a utilização dos dados com suas especificações e a divisão dos grupos de testes.

Posteriormente, o capítulo 5 demonstra os resultados e suas implicações para ambos os conjuntos de testes. Finalmente, o capítulo 6 apresenta as conclusões e novos campos para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICO E METODOLÓGICA

Neste capítulo, são apresentados os assuntos utilizados como base para a produção deste trabalho: fundamentos matemáticos e ferramentas utilizadas.

2.1 FUNDAMENTOS MATEMÁTICOS

Nesta seção são apresentados os fundamentos teóricos necessários para o entendimento deste trabalho. Para isso, a organização está da seguinte maneira. Na seção 2.1.1 são introduzidos conceitos básicos para compreensão deste trabalho; na seção 2.1.2 é demonstrado o conceito do método paramétrico na construção da trajetória; na seção 2.1.3 são abordados algoritmos que têm capacidade de predição de tempo de viagem; na seção 2.1.4 é abordado o algoritmo escolhido neste trabalho para classificação da atividade - J48; Por último, na seção 2.2 são abordadas outras ferramentas essenciais para este trabalho.

2.1.1 CONCEITOS BÁSICOS

Neste tópico são abordados conceitos básicos relacionados a estatísticas e modelos de previsão: séries temporais e técnicas de modelos de previsão. O primeiro é um grupo de dados observados em determinado período, a partir dos quais é possível obter informações sobre estados futuros (ESTEVES, 2003). Podem ser classificadas como discretas, contínuas, determinísticas, estocásticas, multivariadas e multidimensionais. Neste trabalho, as séries temporais são analisadas no domínio do tempo, com aplicação de modelos paramétricos.

Ainda sobre séries temporais, Esteves (2003) afirma que a análise da série é bastante facilitada quando se determinam as estruturas internas do modelo. Um exemplo seria o fator de desconto: sua aplicação está relacionada ao fato de determinar o grau de importância a um dado observado na construção da série temporal. O trabalho de Esteves (2003) comenta essa afirmação com clareza:

> ...os dados mais recentes possuem um conteúdo informativo muito mais importante do que os dados mais antigos, e isso precisa ser, de alguma forma, modelado. (ESTEVES, 2003, pp. 41)

Já os modelos de previsão, de acordo com Esteves (2003) são modelos matemáticos capazes de representar o comportamento e as características da série temporal que se deseja prever. São três tipos: modelos univariados, causais e multivariados. Para explicar esses modelos, um cenário hipotético se faz necessário: Um veículo tem posição coletada todos os dias a cada 1 minuto, transformando-se, nesse caso, os dados em uma série temporal.

O primeiro modelo usará somente as posições geográficas para a previsão da posição em determinado tempo. Já o segundo adiciona uma série que, de alguma forma, tem influência nas previsões (nesse caso, uma série de tempo climático). Por último, os multivariados que, além de prever a posição utilizando um conjunto de série temporais,

poderiam oferecer informações sobre comodidades do transporte público, como quantidade de usuários utilizando o meio de locomoção em dado momento.

2.1.2 MÉTODO PARAMÉTRICO NA CONSTRUÇÃO DA TRAJETÓRIA

Este assunto está intrinsecamente relacionado ao método dos Mínimos Quadrados (MMQ). Para isso, serão feitas observações sobre MMQ.

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ \vdots \\ Y_n \end{bmatrix}_{n\times 1} = \begin{bmatrix} 1 & X_{11} & X_{21} & \cdots & X_{k1} \\ 1 & X_{12} & X_{22} & \cdots & X_{k2} \\ \vdots & \vdots & \vdots & \cdots & \\ \vdots & \vdots & \vdots & \cdots & \\ 1 & x_{1n} & X_{2n} & \cdots & X_{kn} \end{bmatrix}_{n\times k} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \vdots \\ \beta_n \end{bmatrix}_{n\times k} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \vdots \\ \epsilon_n \end{bmatrix}_{n\times 1} \tag{2.1}$$

A matriz X são dados sobre variáveis independentes para n observações. A matriz Y é observada sobre a variável dependente. Os termos β são parâmetros desconhecidos, que serão estimados. Já o termo ϵ é a representação de distúrbios ou erros (MICHAEL J. ROSENFELD, 2013). A expressão 2.1 pode ser reescrita da seguinte maneira:

$$y = X\beta + \epsilon$$

Esta expressão busca representar, com melhor aproximação, o comportamento complexo do mundo real. O termo ϵ se refere a influências que não podem ser diretamente observadas, diferentemente do conceito de resíduos, os quais podem ser observados Michael J. Rosenfeld (2013). O objetivo desse modelo é obter as estimativas para os parâmetros do coeficiente β. Para isso, é necessário minimizar o somatório dos resíduos $\sum e_i^2$.

A expressão 2.2 representa o vetor residual.

$$e = y - X\hat{\beta} \tag{2.2}$$

A representação por matrizes do somatório dos resíduos pode ser vista na equação 2.3 ou, de forma mais compacta, como $e'e$. Com essa representação, podemos minimizar o somatório aplicando derivadas (MICHAEL J. ROSENFELD, 2013).

$$\begin{bmatrix} e_1 & e_2 & \cdots & e_n \end{bmatrix}_{1\times n} \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}_{n\times 1} = \begin{bmatrix} e_1^2 + e_2^2 + \cdots + e_n^2 \end{bmatrix}_{1\times 1} \tag{2.3}$$

Dessa maneira pode-se escrever o somatório residual da seguinte maneira:

$$e'e \;=\; (y - X\hat{\beta})(y - X\hat{\beta})$$

$$\begin{aligned} &= y'y - \hat{\beta}'X'y - y'X\hat{\beta}' + \hat{\beta}X'X\hat{\beta} \\ &= y'y - 2\hat{\beta}X'y + \hat{\beta}'X'X\hat{\beta} \end{aligned}$$

A partir desse momento, podemos aplicar a derivada:

$$\frac{\partial e'e}{\partial \hat{\beta}} = -2X'y + 2X'X\hat{\beta} = 0 \tag{2.4}$$

Derivando mais uma vez, seria obtida a expressão $2X'X$, levando à seguinte afirmação: foi minimizada a soma dos resíduos se X for uma matriz de classificação completa (MICHAEL J. ROSENFELD, 2013). O resultado da equação 2.4 é uma equação normal:

$$(X'X)\hat{\beta} = X'y$$

Multiplicando o inverso da expressão $(X'X)$ em ambos os lados, depois utilizando a propriedade de matriz identidade, chega-se à seguinte expressão:

$$\hat{\beta} = (X'X)^{-1}X'y \tag{2.5}$$

Observa-se que não foram feitas suposições no decorrer das expressões. Além disso, fica evidente que as variáveis x e y têm certas propriedades diretas (MICHAEL J. ROSENFELD, 2013). Portanto, pode-se achar facilmente os valores estimados para $\hat{\beta}$, bastando substituir os dados na fórmula.

Portanto, o MMQ atende os critérios de minimizar a soma dos resíduos quadrados. Existem outras propriedades derivadas da equação 2.5.

- $X'e = 0$, ou seja, X não estão correlacionados com os resíduos
- $\hat{y}'e = 0$, Y não está correlacionado com os resíduos.

Estas observações do funcionamento MMQ se fizeram necessárias para entender o modelo paramétrico, seu funcionamento e sua relação com dados coletados, visto que as observações precisam ser ajustadas para um modelo matemático, como afirma o autor Fernando Nogueira (2009): "O conceito de regressão pode ser entendido como uma maneira de 'ajustar' um dado modelo matemático a um conjunto de dados (geralmente observados ou mensurados)."

De acordo com Fernando Nogueira (2009), o método paramétrico baseia-se em observações diretas que geram as incógnitas, porém ajustadas, com objetivo de representar os dados. O modelo de expressão dos dados observados é dado pelas equações lineares.

$$y = Ax + B; y_1 = Ax_1 + B$$

$$y_2 = Ax_2 + B; y_n = Ax_n + B$$

As variáveis (x, y) são coordenadas do plano cartesiano, A e B são os coeficientes angular e linear respectivamente. Cada observação coletada fornece uma equação linear. O n é a quantidade de dados coletados.

Para a construção da reta gerada pelos pontos coletados com base no método paramétrico, é necessário seguir os procedimentos demonstrados a seguir.

O primeiro, é a construção de um matriz das derivadas parciais (Jacobianas) que têm sua coluna com base no número de parâmetros utilizados e suas linhas pelo número de dados observados, dada por A_{2n}:

$$A_{2n} = \begin{bmatrix} \frac{\partial Y_1}{\partial a} & \frac{\partial Y_1}{\partial b} \\ \frac{\partial Y_2}{\partial a} & \frac{\partial Y_2}{\partial b} \\ \vdots & \vdots \\ \vdots & \vdots \\ \frac{\partial Y_n}{\partial a} & \frac{\partial Y_n}{\partial b} \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix}$$

O segundo passo:

$$L_{n1} = L_0 + L_b \tag{2.6}$$

De acordo com autor Fernando Nogueira (2009) o vetor L_0 é nulo no caso linear. Já o vetor L_b são valores observados do eixo y. A equação acima, com essas informações, é demonstrada a seguir:

$$L_{n1} = L_0 + L_b = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} -y_1 \\ -y_2 \\ \vdots \\ -y_n \end{bmatrix}$$

O próximo passo foi dividido em duas etapas, devido à expressão matemática resultante. O vetor correção X é obtido através da expressão:

$$X_{21} = -(A^t P A)^{-1} A^t P L$$

$$N_{22} = A^t P A$$

$$U_{21} = A^t P L$$

Conforme o autor Fernando Nogueira (2009) o vetor P é uma matriz de pesos dos dados observados.

$$N_{22} = A^t P A = \begin{vmatrix} x_1 & x_2 & \cdots & x_n \\ 1 & 1 & \cdots & 1 \end{vmatrix} \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} 1 = n \end{bmatrix}$$

$$U_{21} = A^t L = \left| \begin{array}{cccc} x_1 & x_2 & \cdots & x_n \\ 1 & 1 & \cdots & 1 \end{array} \right| \left[\begin{array}{c} -y_1 \\ -y_2 \\ \vdots \\ -y_n \end{array} \right] = \left[\begin{array}{c} -\sum_{i=1}^{n} x_i y_i \\ -\sum_{i=1}^{n} y_i \end{array} \right]$$

O resultado da expressão $X_{21} = -(N)^{-1} A^t P L$, no sistema linear, é o vetor dos parâmetros ajustados(X_a), representado abaixo:

$$Xa_{21} = \left| \begin{array}{c} a11 \\ a12 \end{array} \right|$$

De acordo com Fernando Nogueira (2009), após obter a_{11} e a_{12}, pode-se construir a equação da reta que simboliza o conjunto dos dados coletados de um segmento da rota total. Portanto, ao concluir esses passos para toda a rota, o trajeto estará linearizado:

$$Y = a_{11} X + a_{12}$$

Para a utilização prática desse conceito e construção de um trajeto linearizado, é necessário dividir a rota total em vários segmentos.

Esses conceitos abordados até aqui, servem para demonstrar que se faz necessário identificar, em que modelo matemático, os dados podem ser representados. Na análise dos resultados, é possível identificar qual melhor modelo matemático se encaixa para um conjunto de dados.

2.1.3 MODELOS PREDIÇÃO DE VIAGEM

Nesta seção são abordados conceitos dos modelos que podem prever o comportamento de uma variável ao longo do tempo. O primeiro modelo será univariado, baseado na média da velocidade. Já os demais, são multivariados.

2.1.3.1 MODELO UNIVARIADO

O modelo da média histórica, de acordo com Gurmu Zegeye Kebede e Fan (2014), fornece a previsão da duração do tempo da viagem atual com base nos tempos das viagens anteriores. Esse modelo se encaixa na definição de modelos univariados (ESTEVES, 2003).

O autor Weigang et al. (2002) utilizou o modelo descrito na equação 2.7, para prever as posições futuras de ônibus. A grande vantagem é que o tempo computacional demandado por esse modelo é relativamente pequeno.

$$v = \frac{\sum_{i=1}^{n-1} v_{ai} + v_r}{n} \tag{2.7}$$

onde:

v = é a velocidade usada para ser aplicada na equação $t = \frac{\Delta S}{v}$

v_{ai} = média histórica por segmento

v_r = velocidade atual

n = o número de segmentos até o ponto de ônibus de interesse

Porém, o fator de desconto, conceito já abordado na seção 2.1.1, não foi levado em consideração, gerando uma perda de acurácia. A variável, v_{ai} possui um maior peso, comparando com v_r. Analisando um caso hipotético, na equação 2.7 tendo dez registros de velocidades, 9(nove) sendo sendo do somatório e 1(uma) da velocidade atual, fica evidente que a predição da posição geográfica terá maior influência da variável v_{ai} do que v_r , ou seja, a média histórica da velocidade do segmento terá maior influência do que a velocidade atual.

Os autores Sun et al. (2007) e Gurmu Zegeye Kebede e Fan (2014) afirmam que a influência da velocidade atual(v_r) é um fator preponderante. Além disso, esse método pode prever uma velocidade diferente de zero, mesmo quando $v_r = 0$. Indo além, o fator de desconto pode ser aplicado a outros dados coletados dos sensores em uma aplicação real.

2.1.3.2 MODELO MULTIVARIADO BASEADO EM MÉDIA HISTÓRICA

O modelo analisado nesta seção consiste em um melhoramento da média histórica. O autor Gurmu Zegeye Kebede e Fan (2014) afirma que o modelo de regressão seria uma solução viável e adotada por vários autores. Uma possível definição seria:

> Os modelos de regressão requerem uma função matemática linear para explicar uma variável dependente com um conjunto de variáveis independentes (GURMU ZEGEYE KEBEDE E FAN, 2014).

Ao analisar a definição, fica evidente que a previsão do tempo da viagem do ônibus está relacionada a outros fatores. Além da velocidade, como é o caso da média histórica. Embarque, desembarque, quantidade de passageiros e condições climáticas são variáveis que influenciam no tempo de duração de um deslocamento. Portando, o modelo de regressão é um modelo multivariado que, para prever o comportamento de uma variável, utiliza variáveis independentes.

O autor Sun et al. (2007) propôs um algoritmo com base no modelo, que é dividido em duas equações (Eqs 2.8 e 2.9).

$$v_i = \frac{av_r + bv_{ai}}{a + b} \tag{2.8}$$

onde:

i = trajetória ou rota

v_i = velocidade prevista do segmento na rota i

v_r = velocidade atual do barramento derivada dos dados do GPS

v_{ai} = velocidade média histórica do segmento da rota i no tempo atual do período

ab = variáveis relacionadas à posição do veículo dentro do segmento

Nesta equação, a velocidade atual e a média possuem o mesmo peso. Porém quando observamos a equação 2.9, o v_i tem uma maior contribuição na previsão.

$$T = \frac{S_i}{v_i} + t_{i+1} + t_{i+2} \cdots + t_{i+n} + t_d \tag{2.9}$$

onde:

T = duração da viagem do ônibus

S_i = distância da localização do barramento atual até o final do segmento da rota i

t_{i+1} = tempo de viagem de cada segmento, estimado com base nas velocidades médias dos segmentos e a velocidade atual.

t_d = soma do tempo médio de permanência em cada ponto de parada

n = número de segmentos antes de chegar ao ponto de interesse

A variável t_d contribui na previsão da duração da viagem, devido ao fato de computar o tempo de embarque e desembarque nos ônibus, tornando a predição mais precisa. Vale ressaltar que essa é uma variável independente. O trabalho proposto por Patnaik, Chien e Bladikas (2004) desenvolve essa técnica, ao adicionar algumas variáveis interessantes: hora do dia, número de paradas, passageiros na rota do trajeto e outras.

2.1.3.3 MODELO MULTIVARIADO: FILTRO DE KALMAN

O Filtro de Kalman foi baseado no método dos mínimos quadrados. Quando foi descoberto, logo foi aplicado em vários segmentos de problemas, devido que o algoritmo de filtragem realmente funcionava e sua implementação era fácil. Sua popularidade, se espalhou ao ponto, que o autor Paul Zarchan, escreveu a seguinte frase:

> A filtragem de Kalman é provavelmente a técnica algorítmica mais importante já criada.

Houve muitas implementações, variações diferentes do Filtro Kalman original. Todavia, com o avanço do Hardware, desde a década de 60 do século XX, essa inovações não são determinantes para aplicação do filtro.

Com foco de explicar o funcionamento do filtro, será inicialmente exposta a teoria e logo após será demonstrado o seu funcionamento através de um exemplo em Java.

Todavia, o foco será na demonstração prática, de como o filtro funciona. Vale ressaltar que o programa em Java foi construído com base nos códigos em Fortran escritos no texto de Zarchan e Musoff (2013).

O modelo de filtro de Kalman deve ser descrito como um conjunto de equações diferenciais e expressas na forma de matriz. Sua equação geral é:

$$\dot{x} = Fx + Gu + w \tag{2.10}$$

onde:

$\dot{x}$ = primeira derivada temporal de x
F = a matriz de dinâmica do sistema
Gu = é um vetor conhecido, que às vezes é chamado de vetor de controle
w = é um processo de ruído branco, que também é expresso como um vetor

A equação geral, com algumas transformações matemáticas que serão abordadas em sua maioria, pode ser representada da seguinte forma:

$$\hat{x}_k = \Phi_k \hat{x}_{k-1} + G_k u_{k-1} + K_k(z_k - H\Phi_k \hat{x}_{k-1} - HG_K u_{k-1}) \tag{2.11}$$

onde:

$\hat{x}$ é a estimativa de um estado no tempo "k", logo, $\hat{x}_{k-1}$ segue a mesma linha.
Φ_k é a matriz dinâmica do sistema.
G_k é obtida pela fórmula $\int_0^{T_s} \Phi_t G d_\tau$.
K_k representa a matriz de ganho de Kalman.
z_k é medição, ou o sinal real mais um ruído, ou seja, $z = Hx_k + v_k$
H é matriz de medição dos dados

O valor de K é calculado através das equações de Riccati, que na prática são um conjunto de equações recursivas.

$$M_k = \Phi_k P_{k-1} \Phi_k^T + Q_k \tag{2.12}$$
$$K_k = M_k H^T (H M_K H^T + R_K)^{-1} \tag{2.13}$$
$$P_K = (I - K_k H) M_K \tag{2.14}$$

Onde P_K é a matriz de covariância que representa os erros dos estados estimados após uma atualização.

Para melhor compreensão, será abordado somente filtro de Kalman sem perturbação determinística ou vetor de controle e com sinais polinomiais, $u_k = 0$. Sendo assim, a equação 2.11 sem vetor de controle fica representada da seguinte maneira:

$$\hat{x}_k = \Phi_k \hat{x}_{k-1} + K_k(z_k - H\Phi_k \hat{x}_{k-1}) \tag{2.15}$$

Para compreender o funcionamento na prática, será abordado um sinal com se fosse uma reta, $x = a_0 + a_1 t$. Realizando derivações de primeira e segunda ordem.

$$\dot{x} = a_1 \tag{2.16}$$

$$\ddot{x} = 0 \tag{2.17}$$

Observando essas equações, pode-se representá-las através de espaço de estados, ou seja, na forma de matrizes.

$$\begin{vmatrix} \dot{x} \\ \ddot{x} \end{vmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

Comparando essa expressão com a expressão 2.10, pode-se concluir a seguinte afirmação matemática (polinômio de primeira ordem).

$$F = \begin{vmatrix} 0 & 1 \\ 0 & 0 \end{vmatrix}$$

O autor, (ZARCHAN; MUSOFF, 2013), demonstram passo a passo cada etapa, para encontrar as matrizes para cada ordem de polinômio, resumidas na Figura1. O caminho para chega-se a matriz F , de polinômio de ordem 1, foi demonstrado acima. Na Figura1 estão todas as matrizes que se deve usar para situações onde o Filtro de Kalman tenha zero ruído de processo.

Substituindo as matrizes da Figura1 na fórmula 2.15 de segunda ordem, obtém-se a eq 2.18

$$\begin{bmatrix} \dot{x} \\ \hat{\dot{x}} \\ \hat{\ddot{x}} \end{bmatrix} = \begin{bmatrix} 1 & T_s & 0.5T_s^2 \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_{k-1} \\ \hat{\dot{x}}_{k-1} \\ \hat{\ddot{x}}_{k-1} \end{bmatrix} + \begin{bmatrix} K_{1_K} \\ K_{2_k} \\ K_{3_k} \end{bmatrix} \left[X_k^* - \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & T_s & 0.5T_s^2 \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_{k-1} \\ \hat{\dot{x}}_{k-1} \\ \hat{\ddot{x}}_{k-1} \end{bmatrix} \right] \tag{2.18}$$

Para melhor compreensão, foi escrita essa equação 2.18, em linguagem Java, (apêndice A) para melhor compreender o funcionamento do filtro, e aplicá-lo no seguinte problema proposto no livro (ZARCHAN; MUSOFF, 2013). O enunciado do problema:

> *O objeto está inicialmente a 400.000 pés acima do radar e tem uma velocidade de 6.000 pés / s em direção ao radar, que está localizado na superfície de uma*

Figura 1 – Matrizes

Order	Systems dynamics	Fundamental	Measurement	Noise
0	$\boldsymbol{F} = 1$	$\Phi_k = 1$	$\boldsymbol{H} = 1$	$\boldsymbol{R}_k = \sigma_n^2$
1	$\boldsymbol{F} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$	$\Phi_k = \begin{bmatrix} 1 & \boldsymbol{T}_s \\ 0 & 1 \end{bmatrix}$	$\boldsymbol{H} = [1 \quad 0]$	$\boldsymbol{R}_k = \sigma_n^2$
2	$\boldsymbol{F} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\Phi_k = \begin{bmatrix} 1 & T_s & 0.5T_s^2 \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix}$	$\boldsymbol{H} = [1 \quad 0 \quad 0]$	$\boldsymbol{R}_k = \sigma_n^2$

Matrizes importantes para filtros Kalman polinomiais de ordem diferente

> *Terra plana. Neste exemplo, estamos negligenciando o arrasto ou a resistência do ar para que apenas a gravidade g (ou seja, $g = 32,2$ ft / s^2) atue no objeto. Vamos fingir que o radar mede o alcance do radar até o alvo (ou seja, a altitude do alvo) com uma precisão de medição de desvio padrão de 1000 pés. O radar faz medições 10 vezes por segundo durante 30s.* (ZARCHAN; MUSOFF, 2013)

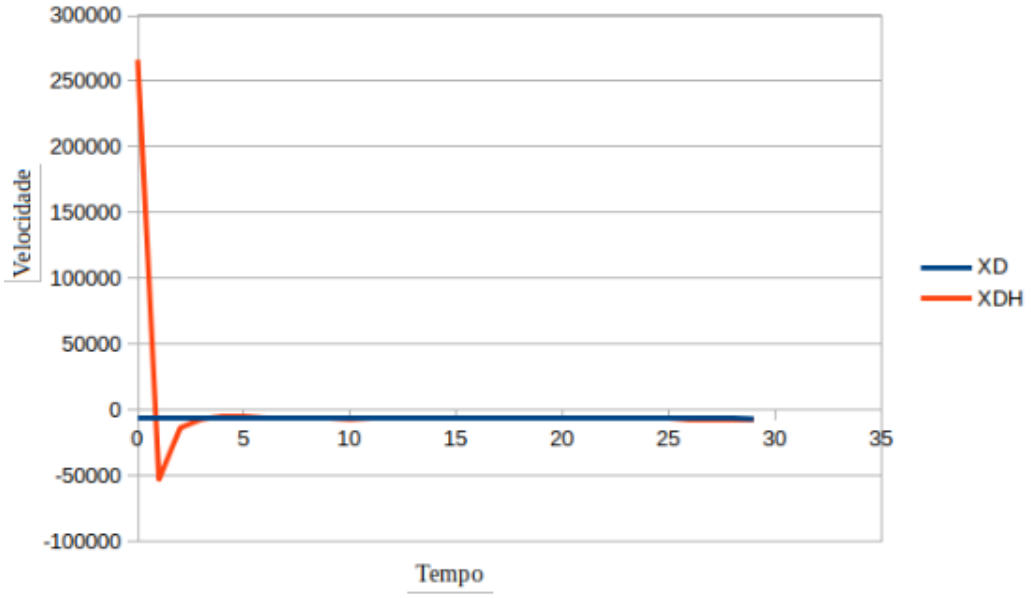
Ao analisar o Apêndice A, observa-se que as estimativas dos estados iniciais estão com um erro considerável, $XH = 0$ e $A_0 = 400000$. Isso está relacionado ao fato que a priori não possui nenhuma informação relativa à altitude ou velocidade inicial do alvo. Observe que as variáveis nas linhas 20, 21 e 22, são estimativas. Sendo que a letra D e DD representam derivadas de primeira e segunda ordem, respectivamente.

Ao observar o valor da XD(velocidade do objetivo) e XDH(velocidade estimada do objeto), na linha 140 e 145, respectivamente, no apêndice A, fica claro que os valores iniciais estão com erro considerável, o que irá impactar no primeiros segundos do filtro. Porém, depois de aproximadamente 5 segundos, já é possível verificar que as estimativas estão próximas dos valores reais. A Figura 2, ilustra que, depois de poucos segundos, foi possível obter precisão satisfatória. Vale lembrar que a variável XD inicialmente era zero, portanto, tinha um erro considerável em relação à velocidade do objeto, $A1 = -6000$.

Todavia, os autores Zarchan e Musoff (2013) afirmam que para validar o algoritmo não é somente comparar as estimativas de estado com os estados reais. Portanto, mesmo que na Figura 3 demonstra precisão, como se ver, os estados reais e os estimados praticamente são idênticos ao longo do tempo K, pois as linhas estão praticamente uma sobre a outra, todavia, há uma diferença entre os dados estimados e os reais. Essa diferença é crucial para poder observar se o sistema está de acordo com a teoria do Filtro de Kalman.

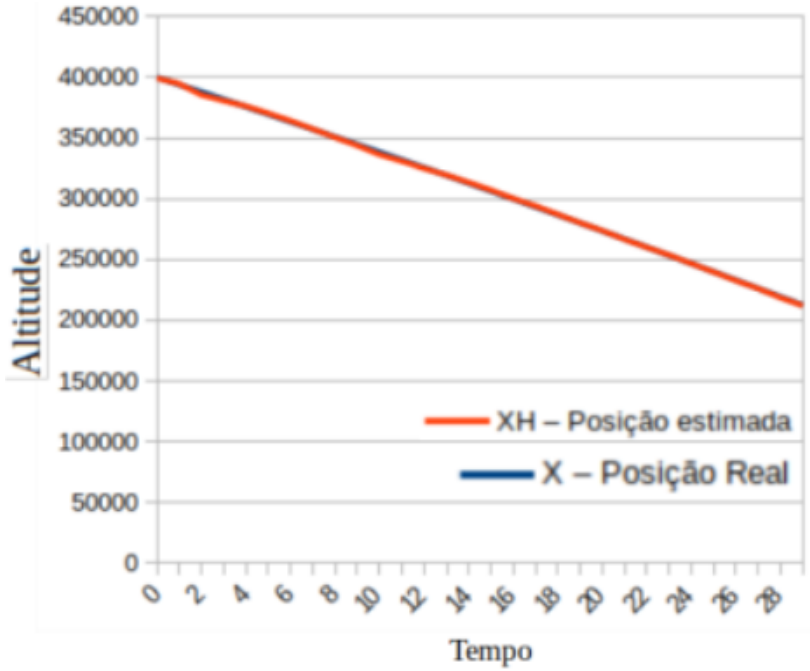
Para obter a certeza que o filtro esteja funcionando, deve-se examinar os erros $(X - XH)$ a cada interação e compará-los com as respostas teóricas obtidas a partir da

Figura 2 – Predição da Velocidade



Observa-se que são necessários aproximadamente 5 s para obter uma alta precisão dos dados

Figura 3 – Prever a Posição

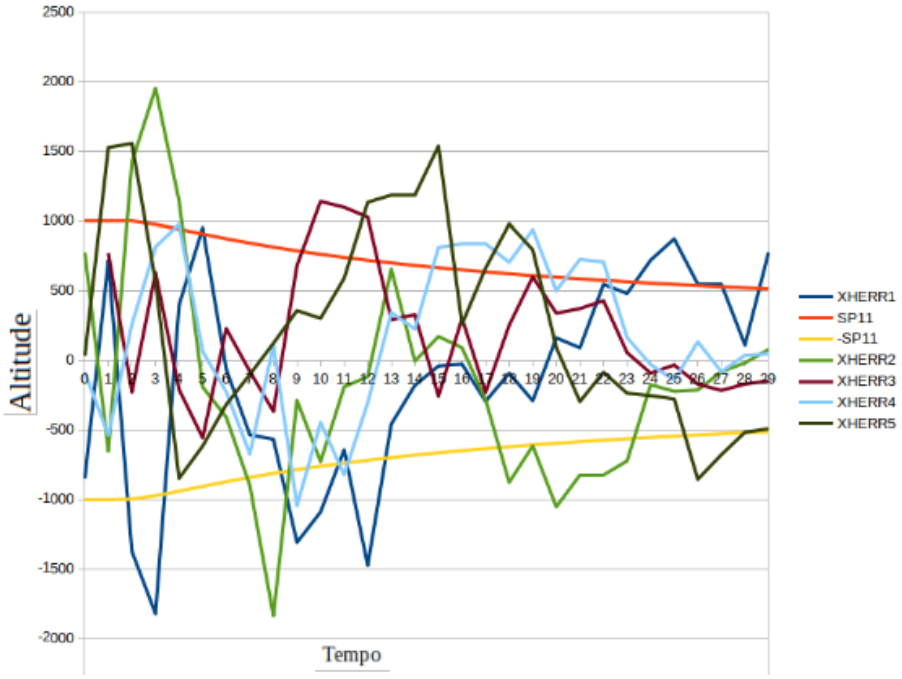


Os estados estimados praticamente são idênticos aos estados reais, o que na prática demonstra a precisão do algoritmo.

matriz de covariância(ZARCHAN; MUSOFF, 2013). A variável no código que representa é $XHERR$, ou seja, a diferença entre o real menos o estimado, para todas as medições.

Caso os resultados, em sua maioria, fiquem dentro do intervalo da variável SPP, que representa a variação do erro, o algoritmo é validado, como fica evidente na Figura 4.

Figura 4 – Estimativa de altitude



Os resultados de cinco execuções do Filtro Polinomial Kalman de segunda ordem parecem coincidir com a teoria dos erros na estimativa de altitude.

Neste trabalho, foi observado que a implementação do algoritmo é de fácil compreensão na linguagem Java.

2.1.3.4 MODELO MULTIVARIADO: REDES NEURAIS ARTIFICIAIS

Surgiu da observação de que o cérebro processa informações de maneira diferente da computação convencional. Nesta, normalmente, formula-se um modelo matemático e, depois, valida-se com dados reais coletadas. Já a Rede Neural é baseada diretamente nos dados do mundo real, ou seja, bem semelhante à forma como o cérebro processa informações de interesse: aprende através da experiência, realiza associações entre padrões diferentes possuindo capacidade de generalizar (HAYKIN, 2001).

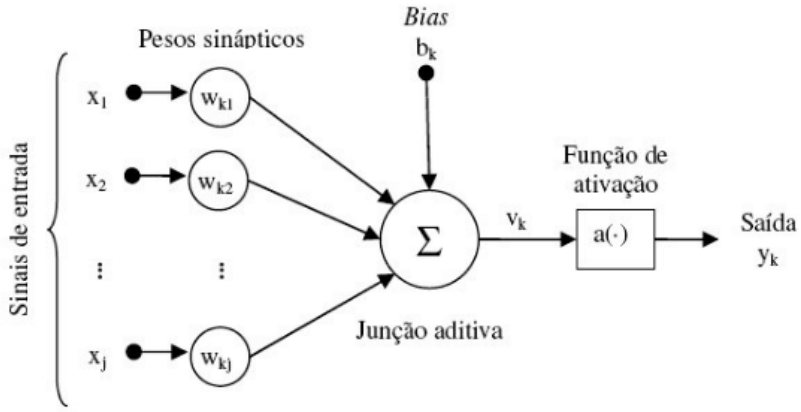
O autor Haykin (2001) define que uma rede conexionista é uma máquina projetada para modelar a maneira como o cérebro realiza uma tarefa específica ou uma função de interesse. Por isso, possui características como adaptabilidade (alterar pesos sinápticos), informação contextual (todos neurônios podem afetar a todos), tolerância a falhas e não linearidade.

Sobre essas características, é importante destacar a tolerância a falhas. Esta se baseia no fato que a informação é distribuída na rede, ou seja, se algumas conexões ou

neurônios ficarem danificados, a degradação do desempenho será suave, devido que a informação está distribuída na rede neural como afirma o autor Haykin (2001).

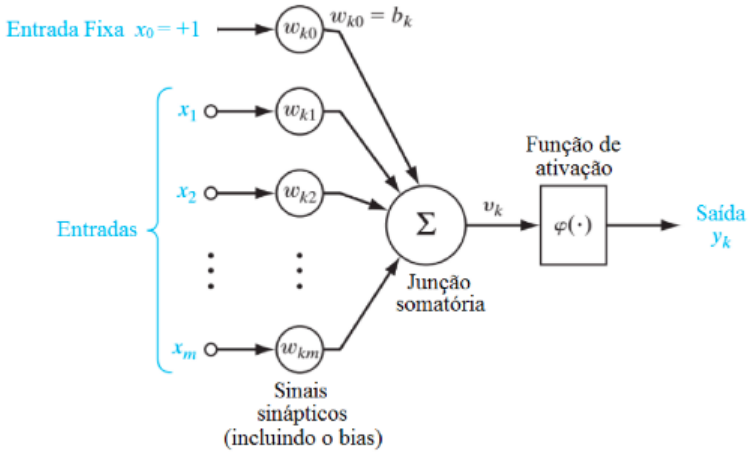
A RN é constituída na sua essência por neurônio(s), ou seja, é um ingrediente comum a todas as redes neurais. Na prática, é uma unidade de processamento de informação. Ou seja, os neurônios são elementos básicos desse sistema que, quando usados em paralelo, se tornam uma RNA (FACURE, 2017). Ao receber sinais das variáveis, passa adiante uma versão ponderada e tratada.

Figura 5 – Modelo de Neurônio com "bias"Interna



Adaptada de (HAYKIN, 2001)

Figura 6 – Modelo de Neurônio com "bias"Externa



Adaptada de (HAYKIN, 2001)

Nas Figura 5 e 6 têm se dois modelos de neurônios. Porém, com os mesmos elementos e com equivalência matemática.

O primeiro elemento, os sinais de entrada são os elos de conexão com seus respectivos pesos. Já os elementos X_j são sinais enviados de outros neurônios ou de dados do mundo externo. O w_{kj} é o peso da sinapse em questão, onde o k está relacionado ao neurônio que recebe as entradas e o j ao peso referente aquela entrada. O segundo elemento, um somador que é responsável pelo somatório dos sinais das entradas. As operações no somatório constituem um combinador linear.

O terceiro elemento é a função de ativação que tem como responsabilidade restringir a amplitude de saída de um neurônio, fundamental para a RNA. Com a sua implementação, a RNA pode representar qualquer função, dado um número suficiente de neurônios, esse elemento ainda será tratado no decorrer deste trabalho. O último elemento, a *bias* pode ser tanto interna como externa e tem como efeito aumentar ou diminuir a entrada líquida da função de ativação.

A representação matemática da Figura 5 pode ser vista nas eqs 2.19 e 2.20:

$$u_k = \sum_{j=1}^{n} W_{kj} x_j \tag{2.19}$$

$$y = \phi(u_k + b_k) \tag{2.20}$$

- j é um índice vinculado a uma entrada
- x_j representa os dados de entrada
- w_{kj} representa os pesos sinápticos
- ϕ representa a função de ativação
- $\sum_{j=1}^{n} w_{kj}$ representa os somatórios dos pesos multiplicados pelas entradas.
- y_k é o sinal de saída de um neurônio.

Se a *bias* for um parâmetro externo, o j deve começar com zero e definir que a entrada $x_0 = 1$ e $w_k0 = b_k$ é adicionado como um novo peso sináptico. A *bias* é fundamental, tanto interna como externa, para que a função de ativação venha a ter capacidade de oferecer uma saída aceitável. Essa saída, o autor Haykin (2001) dar o nome de saída induzida, na *bias* interna a representação matemática seria a seguinte:

$$v_k = u_k + b_k \tag{2.21}$$

Vale ressaltar que a saída v_k é a entrada para função ativação na equação 2.20.

As funções de ativação são fundamentais para a RNA. Com a sua implementação, a RNA pode representar qualquer função, dado um número suficiente de neurônios. O autor Haykin (2001) define três tipos - Limiar ou *Heaviside*, linear por partes e a sigmoide.

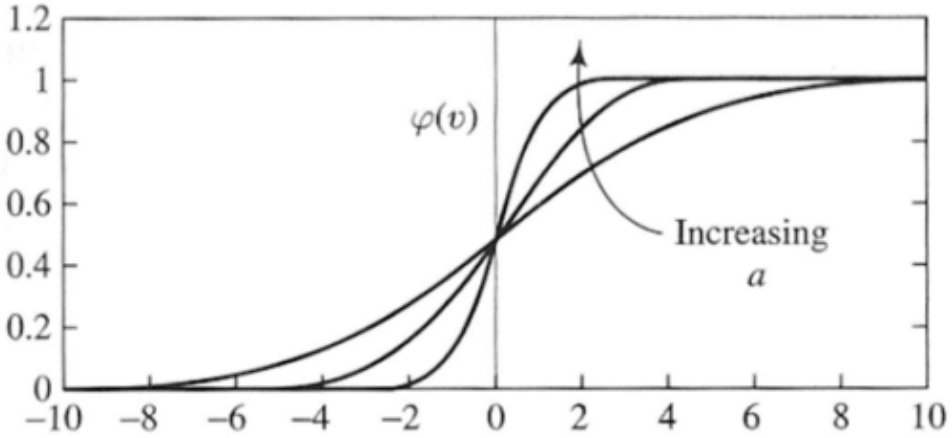
Entretanto, será abordado somente a sigmoide, devido a ter um balanceamento adequado entre linear e não linear.

A função sigmoide (*bias* externa) é representada pela expressão matematicamente 2.22 e tem sua representação gráfica com a Figura 7. Ao observar seu comportamento, pode-se concluir: gráfico é um S, função crescente, comportamento balanceado adequado entre linear e não linear quando o parâmetro de inclinação se aproxima do infinito, a função tem comportamento linear.

Essa função, inicialmente era a mais utilizada (HAYKIN, 2001), devido ao seu comportamento similar ao do neurônio biológico, todavia, devido dificuldade no treinamento, relacionado com a sua derivada (saturação entre -5 e 5), ou seja, gradiente desvanece nessas regiões, o que provoca dificuldade no treinamento.

$$y = \frac{1}{1 + e^{-\sum_{j=0}^{n} W_{kj} x_j}} \tag{2.22}$$

Figura 7 – Função sigmoide



(HAYKIN, 2001) Função sigmoide para parâmetros de inclinação a variável

O entendimento do funcionamento de um neurônio é fundamental - suas entradas, pesos sinápticos, integração (Somatório) e sua função ativação - para a compreensão de como a RNA realiza a questão do aprendizado.

O autor Haykin (2001), afirma que a característica primordial de RNA é aprender a partir do seu ambiente e melhorar o seu desempenho através das relações.

Para isso, deve-se ter em mente que o processo de aprendizado implica na seguinte sequência:

- A rede neural é estimulada por um ambiente;
- A rede neural sofre modificações nos seus parâmetros livres como resultados desta estimulação;

- A rede neural responde de uma maneira nova ao ambiente, devido às modificações ocorridas na sua estrutura;

Há uma série de algoritmos de aprendizagem que utilizam um conjunto de regras bem definidas para a solução de problemas, na sua maioria das vezes, um problema bem específico. Neste trabalho será abordado somente aprendizagem por correção de erro.

Com objetivo de exemplificar, a Figura 8 demonstra todo o processo relacionado a esse algoritmo de aprendizado de um neurônio. Há três variáveis ainda não citadas, a $d_k(n)$, que tem como significado a resposta desejada ou saída alvo, $e_k(n)$ que é o sinal de erro e a última n que está relacionada a o tempo, ou seja, aos instantes de cada operação.

Diante disso, a aplicação desse algoritmo está condicionada que a resposta desejada seja fornecida por alguma fonte externa e que seja acessível ao neurônio k. Antes de abordar as expressões matemáticas, vale ainda comentar, que as soluções do aprendizado por correção de erro são de natureza local, ou seja, as modificações acontecem somente no entorno do neurônio k.

Sobre as expressões matemáticas, quando a função ativação fornece a resposta $y_k(n)$, ela é comparada com a resposta saída alvo. Caso seja diferente,

$$e_k = d_k(n) - y_k(n) \tag{2.23}$$

chega -se ao valor de e_k. Nesse instante, o algoritmo aciona o mecanismo de ajuste dos pesos sinápticos do neurônio em questão, ou seja, encontra o valor de ε, que é o valor instantâneo da energia de erro.

$$\varepsilon = \frac{1}{2}e_k^2(n) \tag{2.24}$$

Esses ajustes somente encerram se no neurônio k no momento em que o sistema atingir um estado estável. Para o sistema "encerrar", é necessário a minimização da função de custo "ε", ou seja, pode se aplicar a regra *Widrow-Holf*,

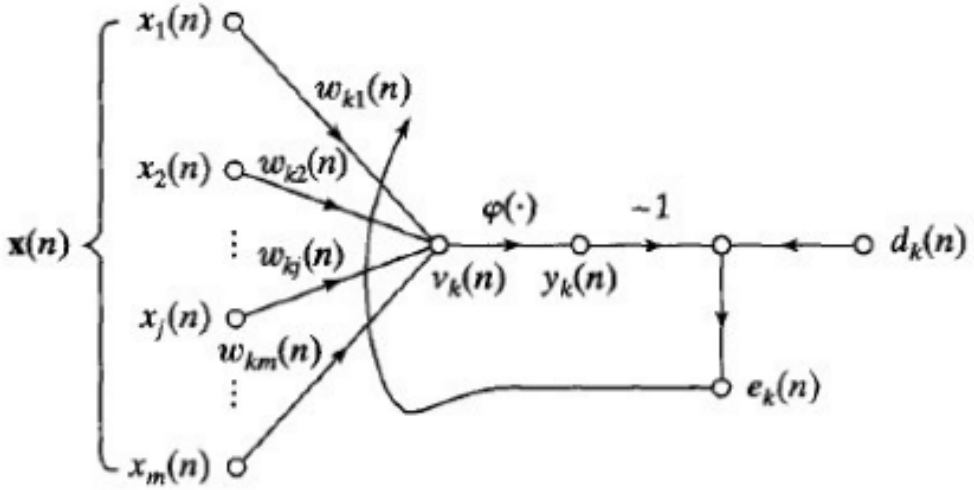
$$\Delta w_{kj} = \eta e_k(n) x_j(n) \tag{2.25}$$

onde η é uma constante positiva que determina a taxa de aprendizado. Ou seja, a regra *Widrow-Holf* é proporcional ao produto do sinal de erro pelo sinal de entrada da sinapse em questão. Vale ressaltar que o valor de η deve ser escolhido com objetivo de obter a estabilidade ou a convergência do sistema. Portanto, o η tem o papel chave nesse algoritmo.

Tendo alcançado o valor de Δw_{kj} pode se agora atualizar os pesos sinápticos da entrada do neurônio k.

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj} \tag{2.26}$$

Figura 8 – Aprendizado por correção de erro



(HAYKIN, 2001) Grafo de um fluxo de sinal do neurônio de saída

O valor $w_{kj}(n+1)$ é novo peso sináptico do neurônio k, com entrada j. Observe que, $w_{kj}(n)$ é o peso sináptico no momento n e o peso sináptico $w_{kj}(n+1)$ é no momento $n+1$.

Após essas etapas, se o sinal y_k ainda não for a resposta desejada (d_k), o algoritmo ainda continua seu "treinamento"enquanto o sistema está estável.

2.1.4 ÁRVORES DE DECISÃO: C4.5(J48)

O entendimento do funcionamento de algoritmos baseados em árvores de decisão é de fácil compreensão, de acordo com o autor Fonseca (1994), pois a estratégia adotada é dividir para conquistar. Na prática essa solução de problema baseia-se na sucessiva divisão do problema em vários subproblemas de proporções menores, até que um subproblema represente uma classe ou uma folha contendo a classe majoritária não sendo possível novas divisões, de forma justificada. As árvores de decisão possuem vantagens que as tornam aplicáveis em vários cenários. O autor Fonseca (1994) enumerou cinco delas.

- Podem ser aplicadas a qualquer tipo de dados;
- Manipula de uma forma muito eficiente informação condicional subdividindo o espaço em sub-espaços que são manipulados individualmente;
- Revelam-se normalmente robustos e insensíveis a erros de classificação no conjunto de treino;

- As árvores resultantes são normalmente bastante compreensíveis podendo ser facilmente utilizadas para se obter uma melhor compreensão do fenômeno em causa. Esta é talvez a mais importante de todas as vantagens enunciadas;

Existem vários algoritmos que utilizam esses conceitos, este trabalho irá abordar especificamente o C4.5, que inclusive é usado pelo software Weka para implementar o J48 tendo como base esse algoritmo. Antes de explicar o funcionamento do J48, se faz necessário explicar o conceito de indução de árvores de decisão.

O processo de indução de árvores de decisão tem como objetivo particionar um conjunto de forma recursiva até obter subconjuntos divididos, contendo somente uma classe. Para isso, utiliza critério de divisão e conquista na construção da árvore tem como princípio realizar escolhas localmente, ou seja, é um algoritmo "guloso".

Utilizando a nomenclatura de árvores, o ponto de partida é chamado de nó raiz, que pode ser um subconjunto (treinamento), pode ser terminal ou não terminal. Na prática um nó terminal é quando não há mais divisão, e o nó já representa uma classe.

De acordo com o Barbosa, Carneiro e Tavares (2012a) um algoritmo que usa árvore de classificação precisa responder às seguintes questões:

1. Como escolher as condições para dividir cada nó?
2. Que critério devemos usar para dividir um nó pai em seus nós filhos?
3. Como vamos decidir quando um nó se tornar um nó terminal (parar a divisão)?
4. Como vamos atribuir uma classe a esse nó terminal?

Para esclarecer as questões acima e adentrar no funcionamento do C4.5, o pseudocódigo 1 explica o funcionamento em linhas gerais do algoritmo, todavia, será detalhado o funcionamento **EscolheParticao(), EscolheAtributo(), CriterioParada(), EscolheClasse() e PodaArvore()**.

Inicialmente, o algoritmo C4.5 tem capacidade de realizar uma divisão não binária. Diante disso, a função **EscolheAtributo()** faz uma busca gulosa, com objetivo de maximizar a divisão dos dados por meio da entropia.

O conceito de entropia está relacionado à heterogeneidade de um conjunto de dados. Na prática a entropia está relacionada com o ganho de informação, que nesse caso é inverso da probabilidade. Quanto maior a probabilidade de um evento acontecer, menor é o ganho de informação.

$$E(S) = \sum_{i=1}^{n} p_i log p_i \tag{2.27}$$

onde E é a entropia, baseado no conjunto S, com "c"classes distintas. Além disso, o p_i é a proporção de dados em S que pertence a classe i.

Algorithm 1 INDUÇÃO C4.5 (exemplos, subatributos)

```
1: if Critério de Parada(exemplos) then
2:     Escolhe Classe(exemplos)
3: else
4:     melhor = Escolher Atributo(subatributos,exemplos)
5:     árvore= nova árvore com nó raiz = melhor
6:     partição = Escolher Partição (melhor)
7:     while partição do
8:         ex_p = elementosdeexemploscommelhor = p
9:         subArv = INDUCAOC4.5(ex_p, subA − melhor)
10:        ADICIONA_RAMO_ARVORE(p, subAvr)
11:    end while
12: end if
13: PODA árvore(arvore)
```

Em relação aos atributos, se a escolha for o Atributo A, ela representa a melhor entropia "local", no momento da partição do conjunto S, onde x é um elemento desse conjunto S_x um subconjunto de S, formado pelos dados onde $A = x$. Portando, se obtém a entropia particionando S em função de A, como fica evidente na equação 2.28.

$$E(A) = \sum_{x \epsilon P(A)} \frac{|S_x|}{|S|} EntropiaS_x \tag{2.28}$$

Com essas duas equações é possível calcular o ganho:

$$Ganho(S, A) = Entropia(S)E(A) \tag{2.29}$$

onde ambas são medidas de não homogeneidade do conjunto S. Todavia, a E(A) é estimativa caso utilize o atributo "A"para fazer a próxima partição.

Em relação ao método **EscolherPartição()** o algoritmo determina cada valor do atributo em um ramo próprio. A desvantagem dessa abordagem é a criação de um número de ramos desnecessários, produzindo árvores, muitas das vezes, exageradas. Essa situação é corrigida com poda da árvore. A vantagem é obter as características do seu conteúdo informativo.

No tocante ao **CriterioParada()**, somente acontece, quando a folha contém uma única classe ou quando os dois nós internos têm os mesmos atributos, mas pertencem a classes diferentes.

Sobre o método **EscolhaClasse()** o C4.5 escolhe o nó terminal a classe de maior probabilidade dentro dos exemplos.

Por último, a **PodaArvore()** é baseada no erro, que utiliza o conjunto treino, para efetuar a poda da árvore . Um nó, que classifica N casos do conjunto treino, sendo K deles incorretos. Devido a isso, o cálculo do erro aparente será calculado da seguinte forma - $\frac{K}{N}$.

O erro aparente é útil devido ao conceito de probabilidade de distribuição, que pode ser representado por um par de limites de confiança. O texto (BARBOSA; CARNEIRO; TAVARES, 2012b) afirma que para um dado fator de confiança α o limite superior desta probabilidade pode ser encontrado através dos limites de confiança da distribuição binomial, que podem ser presentados p_i e p_s, onde possuem valores de probabilidade real de um acontecimento($\frac{K}{N}$), está fora do intervalo definido por estes valores é de $1 - \alpha$.

Com essa informação, é possível calcular o valor do limite superior de cada nó. A poda da árvore acontecerá na seguinte situação: quando a soma dos erros do nó seja inferior à soma dos erros dos seus descendentes, é podado, sendo substituído por uma folha cuja classificação seja a mais provável.

2.2 FERRAMENTAS ÚTEIS - FIREBASE E ANDROID STUDIO E WEKA

Neste trabalho, cresce a importância de conhecer ferramentas úteis disponíveis com capacidade de transformar o transporte público. Gerenciar dados em redes, gerir ambiente de desenvolvimento integrado e aplicar algoritmos de aprendizado de máquina de forma rápida e eficiente.

As três ferramentas, Firebase, Android Studio e Weka, tem condições de gerar uma ”sintonia”para criação de soluções para o transporte público.

2.2.1 *FIREBASE*

Firebase é uma plataforma que disponibiliza diversos serviços diferentes que auxiliam em várias áreas. Seu uso se popularizou no ano de 2014, quando foi adquirido pela empresa Google. Na prática, é uma plataforma dedicada e SDK para a construção de aplicativos. Atualmente, o serviço suporta desenvolvimento nas linguagens de programação C++, Java, Javascript, Node.js, Objective-C e Swift (MASTERTECH, 2017).

Suas funcionalidades que poderão ser aplicados no referido trabalho:

- Autenticação – suporte para autenticação de usuários via *e-mail, Facebook, GitHub, Google, Sign-In e Twitter;*
- Base de dados – um banco de dados NoSQL utilizado para armazenar dados JSON;
- *Offline* – possibilita a armazenagem de dados na memória cache local, permitindo assim o funcionamento da aplicação em estado *offline*;
- *Real time* – os dados são armazenados em tempo real no banco de dados;

A ferramenta permite que os dados sejam sincronizados em todos os dispositivos conectados com eficiência. Além disso, os dados permanecem disponíveis caso o aplicativo fique *off-line*. Isso, na prática, possibilita coleta dos dados nos *smartphones* para o banco de dados quando obtiver conexão novamente.

As demais plataformas se enquadram no modelo open source. O Parse Server é um exemplo disso. Possui fácil implementação, utiliza Node.js e suporta *Express Web App Framework*. A grande vantagem é que os desenvolvedores têm controle total sobre o código-fonte (IMASTERS, 2016), o que não ocorre com *Firebase*.

As duas plataformas podem ser configuradas para o Android Studio, todavia, a *Firebase* possui interações que facilitam essa integração.

2.2.2 ANDROID STUDIO

O Android Studio (DEVELOPERS, 2018) é o ambiente de desenvolvimento integrado (IDE) para o desenvolvimento de aplicativos Android. Possui recursos cruciais para o referido trabalho:

- Um sistema de compilação flexível baseado no Gradle
- Um emulador rápido com inúmeros recursos
- Ferramentas de verificação de código suspeito para detectar problemas de desempenho, usabilidade, compatibilidade com versões e outros
- Compatibilidade embutida com o Google Cloud Platform, facilitando a integração do Google Cloud Messaging e do App Engine

A escolha dessa ferramenta está relacionada a dois fatores. O primeiro, compatibilidade com o *Google Cloud Platform*, tornando a comunicação da aplicação com *Firebase* um problema trivial. Essa conexão possibilita adicionar serviços como o *Google Analytics*, autenticação e notificações. O segundo fator, possuir um emulador com variedades de recursos de emulação de *hardware*, como por exemplo localização de GPS, latência de rede, sensores de movimento e entrada multitoque.

2.2.3 WEKA - WAIKATO ENVIRONMENT FOR KNOWLEDGE ANALYSIS

O *Waikato Environment for Knowledge Analysis* - Weka é um pacote de software de código aberto emitido sob a *General Public License* - GNU que possuem séries de algoritmos de aprendizado de máquina, com foco na mineração de dados.

As ferramentas disponíveis podem ser úteis para classificação, regressão, agrupamento, mineração de regras de associação e visualização. Na prática, o objetivo é agregar algoritmos para diferentes abordagens na área de inteligência artificial.

Para atingir o objetivo deste trabalho foi usada a ferramenta de classificação através do algoritmo J48.

3 TRABALHOS RELACIONADOS

Nesta seção é descrito o processo de seleção dos trabalhos relacionados com uma breve explanação dos textos selecionados. Para tal finalidade foram formuladas as questões de pesquisa com o intuito de direcionar o processo de busca.

Após as questões de pesquisa, é apresentada uma breve síntese de cada produção que serviu de alicerce para construção deste trabalho. As pesquisas foram divididas em duas frentes. A primeira, é referente ao transporte público no Brasil. A segunda, consiste em comparações entre os algoritmos e novas técnicas de previsão e identificação do veículo.

3.1 REVISÃO BIBLIOGRÁFICA

O primeiro questionamento,**"Quais características os usuários mais prezam no transporte público?"**, tem como objetivo selecionar trabalhos que visam conhecer as necessidades dos usuários. O segundo, **É possível prever a posição veicular com base no GPS dos usuários dentro do veículo?** Com base nesse filtro de pesquisa, foi possível encontrar trabalhos que abordassem a questão crucial que esse trabalho propõe resolver - **Classificar qual é a atividade sendo executada no momento?**

A base de dados selecionada para a consulta foi o Scopus, devido a mesma ter indexação de outras bases, como por exemplo: IEEE(Institute of Electrical and Electronics Engineers).

3.2 TRANSPORTE PÚBLICO NO BRASIL

As cidades brasileiras estão distantes do conceito de cidade inteligente. O trabalho PAULA e BARTELT (2016) comenta que o recurso destinado ao sistema de transporte inteligente tem como foco o controle dos cidadãos, o que fica evidente no direcionamento dos recursos na compra de câmaras de HD que são usadas nos grandes centros de controle. Esse fato aconteceu tanto nos eventos de grande circulação (Copa do Mundo e Olimpíadas) como no projeto implementado em Porto Alegre (LADEIRA; MICHEL; SENNA, 2011).

Em relação à capital gaúcha, já está implementado o sistema AVL- Automatic Vehicle Location através do projeto SOMArt (LADEIRA; MICHEL; SENNA, 2011). Esse sistema baseia-se em antenas fixas, espalhadas pelas vias da cidade e etiquetas RFID(Radio Frequency Identification) para os ônibus. O conteúdo das mensagens entre os ônibus e as antenas possuem ID e localização. Com informações obtidas, é possível determinar horário previsto de chegada, velocidade e faltas nas linhas. De acordo com os autores Ladeira, Michel e Senna (2011), os mesmos avaliam como positiva a implementação através da diminuição de viagens fora do horário previsto.

> ... observa-se que a redução do número de viagens fora do intervalo de 5,42 % em 2008 para 2,71% em 2010. (LADEIRA; MICHEL; SENNA, 2011)

Como dito na introdução, essas informações poderiam ser obtidas através dos próprios usuários. No projeto SOMArt, foi necessária uma intervenção física no espaço urbano. Todavia, em relação aos congestionamentos, um projeto centralizado, como é o caso do SOMArt, tende a contribuir para a organização e mobilidade urbana das grandes cidades.

A importância dos usuários terem essas informações acessíveis está relacionado ao fato que o tempo do deslocamento dos trabalhadores corresponde a mais de 10% do seu dia. Como PAULA e BARTELT (2016) evidência com essa afirmação:

> ...o fato de que milhões de pessoas perdem todos os dias três à seis horas para se deslocar ida e volta entre a sua residência e o lugar do seu trabalho equivale a uma violação ou minimamente a uma ameaça dos seus direitos humanos.

Diante disso, uma das características que o usuário mais preza é a previsibilidade do transporte público. Além desta, o conforto foi outro aspecto levantado. Para que ocorra a migração do transporte individual para o coletivo é necessário que esse item(conforto) seja adotado. Todavia, como PAULA e BARTELT (2016) cita os meios de locomoção públicos que, além de serem ineficientes, não possuem conforto. Nesse caso, um conforto mínimo seria o usuário ter acesso sobre a lotação do veículo.

O autor Singh, Bansal e Sofat (2017) observa que, além das comodidades, o usuário irá levar em conta os valores econômicos: custo de manutenção do veículo e combustível. Desses dois fatores, o transporte público tem ligeira vantagem. Portanto, caso o transporte público apresente condições de eficiência(previsibilidade e conforto), será natural a migração para o transporte coletivo.

3.3 PESQUISAS SOBRE A IDENTIFICAÇÃO DO TIPO DE TRANSPORTE E PREDIÇÃO DE POSIÇÃO VEICULAR

O trabalho Singh, Bansal e Sofat (2017) defende que o desempenhos dos algoritmos de predição estão relacionadas às características de trânsito, como por exemplo: se os motoristas são disciplinados, semáforos, interseções e outros. Essa afirmação é comprovada através de comparações entre os resultados utilizando diferentes algoritmos - de regressão, FK e Rede Neural Artificial - em dois cenários: Países desenvolvidos e em desenvolvimento.

O primeiro: essas soluções já são uma realidade com a aplicação de alguns algoritmos com desempenho satisfatório. Já no segundo, devido à heterogeneidade do trânsito, a precisão dos algoritmos falha no objetivo de oferecer serviços de predição de posição aos usuários. De acordo com os resultados obtidos por Singh, Bansal e Sofat (2017), é constatado que os algoritmos têm sua eficiência alterada de acordo com o trânsito da região.

”No entanto, essas técnicas foram implementadas nos países desenvolvidos

> sob condições de tráfego homogêneo, onde os resultados foram promissores, mas quando o mesmo é implementado em condições de tráfego na Índia, os resultados podem não ser eficazes."(SINGH; BANSAL; SOFAT, 2017)

Diante dessa constatação, a escolha do algoritmo proposto teve como foco o trânsito da cidade de porte pequeno, como a cidade do Alegrete-RS, visto que o trabalho tem como foco atender a comunidades onde soluções tecnológicas não estão presentes.

Na construção da solução deste trabalho, o artigo Gurmu Zegeye Kebede e Fan (2014) contribuiu através das comparações entre a Média Histórica (MH), Filtragem de Kalman (FK) e Rede Neural Artificial (RNA) usando o modelo MAPE *(Mean Absolute Percentage Error)* para definir qual teria a maior eficiência.

De acordo com o Gurmu Zegeye Kebede e Fan (2014) vantagem do modelo MH é que o tempo de computação é relativamente pequeno, todavia, o desempenho não é satisfatório. Já o FK possui a capacidade de filtrar ruídos, o que é fundamental para prever a precisão dos veículos. Já o RNA, além de filtrar ruídos, é capaz de lidar com dados complexos e relacionar variáveis dependentes com independentes. Apesar da eficiência do RNA, o autor fez comentários mencionando que as condições do tráfego influenciam nos resultados. Além disso, observou a necessidade de um banco de dados suficiente para aplicação do RNA.

> "No entanto, sua desvantagem é que os resultados obtidos usando esses modelos para um local podem não ser transferíveis para o próximo devido a circunstâncias específicas do local (geometria, controle de tráfego, etc.)". (GURMU ZEGEYE KEBEDE E FAN, 2014)

> "No entanto, para obter o máximo benefício da rede neural, deve haver dados ou observações suficientes". (GURMU ZEGEYE KEBEDE E FAN, 2014)

Os resultados obtidos pelo MAPE para verificar o desempenho foram divididos em três tipos de trajeto: curto, médio e longo. O algoritmo que apresentou o melhor desempenho foi RNA. No quesito curto, os três algoritmos apresentaram praticamente os mesmos resultados, contudo, em alguns momentos MH apresentou melhores resultados que o RNA. Inclusive Gurmu Zegeye Kebede e Fan (2014) cita que o MH poderia ser aprimorado.

> "Modelos históricos de previsão médios poderiam ser melhorados considerando-se erros ou variações e correlações entre diferentes valores da variável sob consideração [21]. Por exemplo, se o ônibus leva mais tempo durante a primeira seção da viagem, é provável que o barramento também levará mais tempo na segunda seção da viagem, o que significa que os tempos de viagem das sucessivas sessões de viagem para um ônibus podem ser correlacionados."(GURMU ZEGEYE KEBEDE E FAN, 2014)

O autor Sun et al. (2007) propôs uma solução mais vantajosa que MH, através das relações das variáveis independentes, como o tempo médio das paradas dos ônibus e a posição dentro do segmento. Além dessa contribuição, utilizou uma máquina de estados finitos para evitar o problema *backward data*. O desempenho do algoritmo proposto por Sun et al. (2007) teve um resultado com erro menor de 5%.

Todos os modelos de previsão vistos anteriormente utilizam dados de GPS, porém o trabalho de Zhou, Zheng e Li (2012) propôs uma alternativa visando proteger a privacidade do usuário e evitar custo de energia envolvendo as operações com GPS.

O autor Zhou, Zheng e Li (2012) propôs utilizar as torres de celulares e os áudios coletados dos *smartphones* e dados dos sensores para obterem posição geográfica do veículo e a previsão do tempo da viagem.

O primeiro passo da proposta Zhou, Zheng e Li (2012), foi identificar quando o usuário estará dentro do veículo. A solução foi usar o som emitido pelo aparelho mostrado na Figura 9 quando utilizado pelos usuários. O resultado foi uma precisão de 95%, considerando a captação do som a cerca de três metros. Além dessa identificação, o autor mostrou ainda que é possível identificar quais dos passageiros estão compartilhando o mesmo ônibus, através dos intervalos dos áudios emitidos pelo aparelho da Figura 9.

Contudo, o aparelho da Figura9 é utilizado para outros serviços, como trens, provocando falsos positivos para identificação do transporte. Para resolver esse impasse, o autor Zhou, Zheng e Li (2012) usou o acelerômetro do *smartphone* para distinguir esses tipos de transporte. O comportamento da aceleração do ônibus possui características que possibilitam sua identificação. A precisão do resultado foi de 90%.

O autor Zhou, Zheng e Li (2012) demonstra que é possível identificar o segmento pelas torres do celular. A Figura10 demonstra o deslocamento do ponto A até o ponto B. O autor Zhou, Zheng e Li (2012), usou a seguinte estratégia: um banco de dados com três torres de maior intensidade para cada segmento. Ao comparar o banco de dados com informações das torres conectadas pelo smartphone(7,8,4,5) da Figura10 seria possível determinar qual o segmento atual do ônibus, com base no histórico das torres conectadas. Porém, essa simples consulta do banco de dados, para cenários complexos, mostrou que essa abordagem não foi eficiente.

Porém, foi possível aumentar a acurácia com os seguintes passos. O primeiro, envolve identificar três torres para cada segmento do trajeto para compor o banco de dados, coletar as torres que os usuários utilizaram em um trajeto e aplicar esses dados no algoritmo Smith-Waterman para gerar a localização do segmento. O resultado foi de aproximadamente 90% de precisão. Essa precisão foi "baixa"devido à sobreposição de rotas no caso analisado de acordo com Zhou, Zheng e Li (2012).

Os resultados obtidos pelo trabalho Zhou, Zheng e Li (2012) sobre detecção do tipo de transporte poderiam ter precisão maior, caso os autores tivessem usado a combinação de sensores presente em um *smartphone*. O trabalho Ballı e Sağbaş (2017) aplicou essa

Figura 9 – Máquina leitora de cartão

máquinas leitoras de cartão, que ao realizarem essa tarefa emitem som que é utilizado na identificação do transporte. (ZHOU; ZHENG; LI, 2012)

Figura 10 – Um trajeto dividido por segmentos

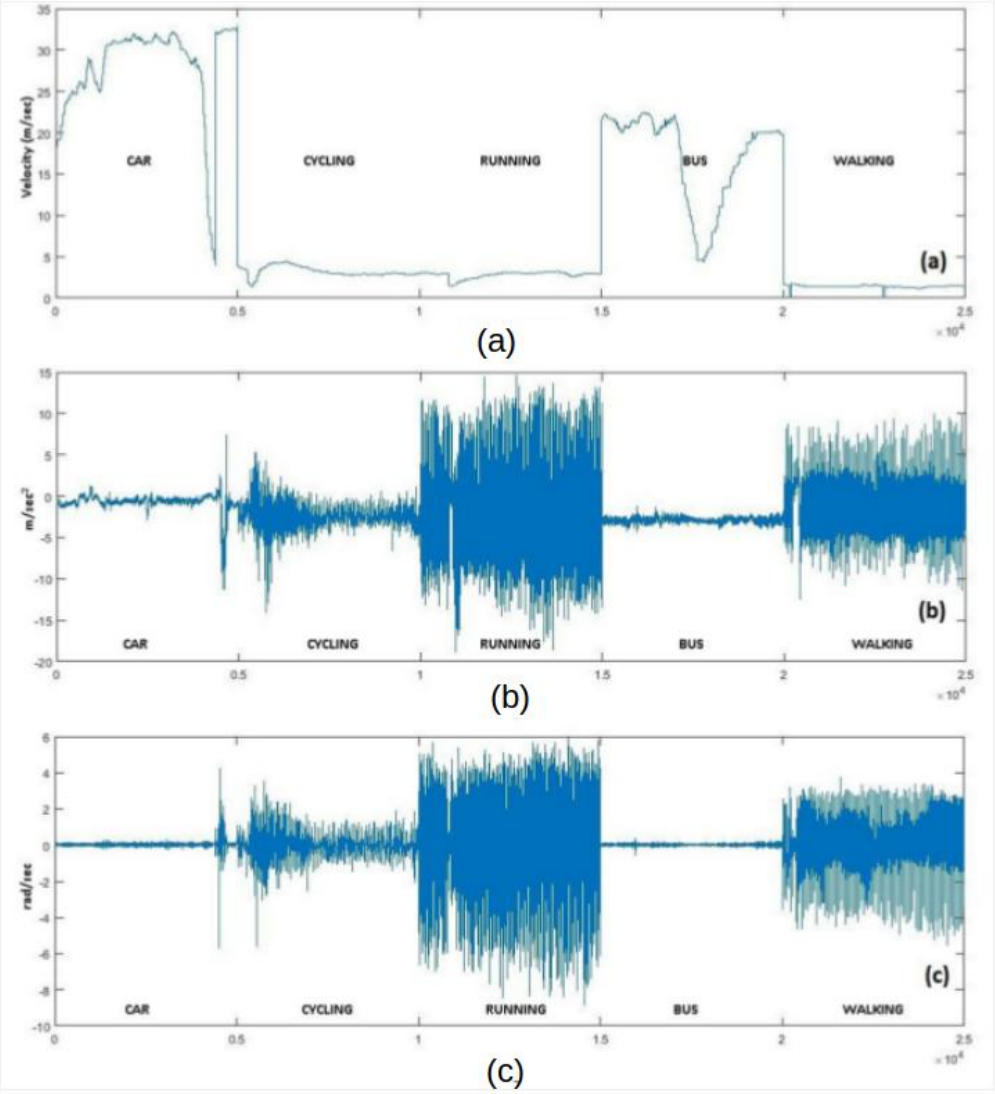
Prever a posição com base na conexão com as antenas da rede móvel. (ZHOU; ZHENG; LI, 2012)

possibilidade e obtive resultados expostos a seguir.

Os autores se concentraram na identificação de cinco tipos de locomoção: caminhar, correr, andar de bicicleta, carro e ônibus. Para isso, utilizou dados de acelerômetro, giroscópio e sensores de GPS coletados através de um aplicativo em celular. A Figura11 demonstra que cada tipo de locomoção tem suas características em relação aos tipos de

dados coletados. Em relação ao GPS, cada tipo de transporte têm velocidades médias diferentes, porém, em condições de tráfego intenso podem apresentar as mesmas velocidades. O acelerômetro e o giroscópio conseguem distinguir facilmente transportes não motorizados. Em relação ao carro e ônibus, embora se vejam semelhanças entre ambos, as velocidades angulares são diferentes.

Figura 11 – Amostra de dados do sensor

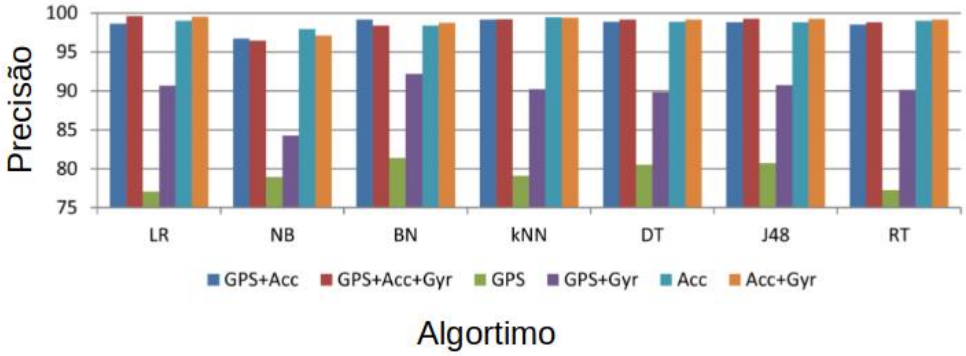


(a) Velocidades - (b) Acelerômetro(eixo-X) -(c) Giroscópio(eixo-Y) (BALLI; SAĞBAŞ, 2017)

Com os dados coletados, o autor aplicou alguns métodos supervisionados de aprendizado de máquina: Bayes Ingênuo(NB), rede Bayesiana(BN), *K- nearest neighbour*(KNN), regressão logística(LR), J48, tabela de decisão(DT) e árvore aleatória(RT).

Os resultados do trabalho Ballı e Sağbaş (2017) na Figura12 mostram que a

Figura 12 – Taxas de precisão de combinações dos sensores



(BALLI; SAĞBAŞ, 2017) Acc (acelerômetro) e Gyr(giroscópio)

combinação dos três sensores e o método de regressão logística alcançaram uma precisão de 99,6%. Contudo, a precisão utilizando somente o acelerômetro e o giroscópio chegou a 99,5%, o que é um resultado considerável, devido ao consumo de energia das operações do GPS e questões sobre privacidade do usuário.

Para finalizar esta seção, o Algoritmo Rede Neural Artificial na maioria dos cenários demonstrou maior eficiência. Porém, os modelos baseados em RNAs consomem muita energia e recurso computacional (FACURE, 2017).

4 DESENVOLVIMENTO DO TRABALHO

O objetivo central deste trabalho é contribuir com a sociedade em um setor que pudesse conciliar a área de computação com uma melhoria da qualidade de vida da população.

No momento da apresentação do TCC 1, o objetivo era abrangente e com uma certa complexidade, que ficou acentuada ainda mais devido às dificuldades impostas pela pandemia. Devido a isso, se fez necessário uma adequação do tema, que será apresentado no decorrer deste trabalho. A proposta inicial está presente no texto, visando demonstrar que há uma conexão entre as propostas.

Neste capítulo, é apresentada a proposta do TCC inicial e a adequação do tema junto com a proposta final. Em seguida, será abordado o desenvolvimento do trabalho propriamente dito: dados utilizados, algoritmo de classificação e o conjunto de testes com suas especificações e por último os resultados obtidos.

4.1 PROPOSTA DE TRABALHO INICIAL

Em vista dos problemas apresentados no transporte público das cidades brasileiras, em especial a cidade de Alegrete-RS, este trabalho tem como objetivo implementar um modelo de regressão linear baseado no trabalho (SUN et al., 2007), a fim de oferecer informações importantes aos passageiros da 3º Capital Farroupilha sobre tempo de viagem, localização em tempo real dos ônibus.

Os dados utilizados foram coletados dos celulares dos usuários, como longitude, latitude e áudio, acelerômetro e giroscópio.

Visando manter a privacidade do usuário e reduzir o consumo de energia do GPS, os dados deste sensor serão coletados somente quando a detecção do ônibus for feita. Para isso, será necessário utilizar três tipos de dados, vistos nos trabalhos relacionados: o áudio e o sensores(acelerômetro e giroscópio). Nesse sentido, esses métodos serão ”gatilhos”utilizados para obter a localização do GPS do usuário ou desativar o envio de dados para o sistema. Com isso, a posição geográfica do usuário será coletada somente dentro do ônibus.

Após a identificação do transporte, será usado o método paramétrico (FERNANDO NOGUEIRA, 2009) para criação de segmentos de até 80 metros. Esse valor poderá sofrer alteração.

Com a rota estabelecida e segmentos identificados, é possível aplicar o algoritmo (SUN et al., 2007) com o incremento da variável tempo climático. Essa informação será fornecida pelos usuários.

$$T = \frac{S_i}{v_i} + t_{i+1} + t_{i+2} \cdots + t_{i+n} + t_c \tag{4.1}$$

A variável t_c será ativada no momento em que informações sobre condições climáticas estiverem disponíveis. A variável S_i é a distância da localização do barramento

atual até o final do segmento da rota i, já a variável t_{1+n} é o tempo de viagem de cada segmento, estimado com base nas velocidades médias dos segmentos.

O algoritmo desenvolvido para resolução do problema proposto será implementado na linguagem Java. A escolha aconteceu em razão dos aplicativos para dispositivos *mobile*, em sua maioria, serem desenvolvidos nessa linguagem. Já o *Android Studio* foi escolhido em virtude da compatibilidade com a plataforma *Firebase*, que será usada para o tráfego de dados e processamento. Os *smartphones* irão processar dados coletados dos sensores, com o objetivo de identificar o tipo de transporte. Sendo positiva a verificação para ônibus, acontecerá o tráfego de dados da posição geográfica do usuário.

Após realizar a implementação do algoritmo, será feita a coleta de dados e análise dos resultados como base no modelo de desempenho *Mean Absolute Percentage Error(MAPE)* utilizado no trabalho Sun et al. (2007).

4.2 ADEQUAÇÃO DO ESCOPO - PROPOSTA FINAL

Apesar da adequação do tema, o propósito permanece inalterado - contribuir com a sociedade fornecendo uma solução tecnológica para um problema real, com uma perspectiva diferente: Verificar a precisão do algoritmo J48, utilizando o para a determinação da atividade que está sendo executada(correr, andar, ônibus e etc...). Além disso, identificar a relação entre a quantidade de informações fornecidas e a sua precisão. Para esse fim, cenários de testes foram criados.

Sobre a adequação do escopo, um dos motivos está relacionado ao cenário atual - Covid-19. Seria necessário obter um conjunto considerável de dados, em cidades de pequeno porte, como Alegrete-RS, com objetivo de prever a posição do veículo em determinada situação. Portanto, seria um risco à saúde dos usuários que estariam no transporte público somente para a coleta de dados.

4.3 NOVA PROPOSTA DE TRABALHO

Uma das premissas do trabalho inicial, era que cada usuário fosse participante contribuindo para solução colaborativa do problema apresentado. Essa premissa ainda é válida. Os usuários irão fornecer os dados para o algoritmo.

A proposta é que o Algoritmo J48 identifique qual o meio de transporte ou atividade sendo utilizada pelo usuário. As atividades são as seguintes: ficar parado, andando, correndo, bicicleta, carro, ônibus, trem e metrô. Para o algoritmo, cada atividade representa uma classe.

Para alcançar a proposta, é utilizado um conjunto de dados produzidos pela Universidade *Sussex Huawei Locomotion*. Os dados coletados através dos sensores do celular tendem a representar o comportamento do indivíduo ao realizar uma atividade. Existem variedades de informações obtidas dos sensores do *smartphone*, porém foi usado

somente o acelerômetro(aceleração linear), devido aos excelentes resultados obtidos no trabalho de Ballı e Sağbaş (2017).

Os dados utilizados para o treinamento foram particionados, nas seguintes proporções: 80%, 60%, 40% e 20%. O objetivo dessa redução é visualizar o impacto na acurácia do J48.

Dentro de cada proporção, será fornecido ao algoritmo duas informações: a média e o desvio padrão com base nos dados do acelerômetro. Com esses dois parâmetros é possível identificar qual a atividade.

Para atingir a proposta, foram realizados 90 testes, distribuídos com as seguintes amostragens para o cálculo da média - 100, 200, 300, 400, 500, 1000, 1500, 2000 e 2500.

Cada média foi testada com 6 e 8 classes. A primeira é constituída pelas atividades ficar "parado", andando, correndo, bicicleta, carro e ônibus. Já com oito classes é adicionado metrô e trem. Esse número de classes diferentes, visa atender cidades de portes diferentes, ou seja, que tem mais variedades de transporte público.

Ao fim do trabalho, pretende-se enriquecer a literatura existente e contribuir para um transporte público mais eficiente.

4.4 DADOS UTILIZADOS

Buscando se adaptar à realidade dos dias atuais - a pandemia - não foi possível criar dados próprios. Contudo, a universidade de *Sussex Huawei Locomotion*, realizou um trabalho de coleta de dados em parceria com a empresa *Huawei*, registrando durante 7 meses do ano de 2017, várias atividades, relacionadas a locomoção do usuário, através dos sensores do celular (THE UNIVERSITY OF SUSSEX, 2017). Todavia, somente 3 dias de dados estão disponibilizados. Todos os dados disponíveis, sobre acelerômetro, foram utilizados neste trabalho.

O conjunto de dados possui 4 *smartphones*, transportados simultaneamente em locais típicos do corpo. A imagem 14 representa essa afirmação. Os dados possuem mais de 3.000 horas no total. Na imagem 13 é possível identificar as horas para cada atividade. Vale salientar que há uma diferença considerável em relação aos dados coletados da atividade correndo em relação às outras. Todavia, não houve impacto nos resultados. Essa afirmação será elucidada no capítulo 5.

Dentro da coleta de dados por atividade (ao total 8 classes), ainda tem as seguintes subdivisões listadas abaixo, sendo que o número no final de cada item representa a classe no algoritmo. Ao total são 18 subtipos de atividades. Um exemplo, na atividade "ônibus", as 4 divisões representam somente um conjunto - atividade transporte público dentro de um ônibus.

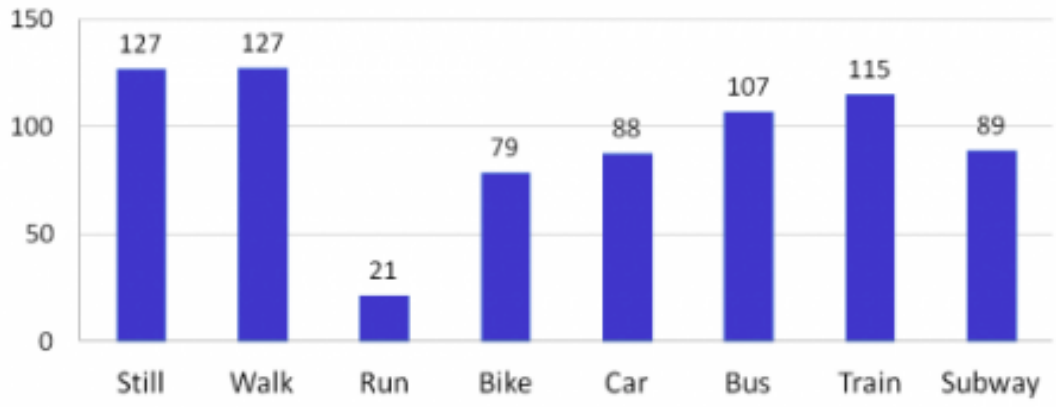
- Parado: em pé ou sentado; dentro ou fora de um prédio;
- Andando;

- Corre;
- Bicicleta;
- Carro: como motorista ou como passageiro;
- Ônibus: em pé ou sentado e convés inferior ou convés superior;
- Trem: em pé ou sentado;
- Metrô: em pé ou sentado;

Essas subdivisões têm como principal contribuição representar a realidade. Visto que os sensores reagem ao movimento do usuário, se o mesmo está em pé, sentado e até mesmo se está dirigindo ou indo de carona.

A lista de dados coletados é significativa - giroscópio, magnetômetro, orientação, gravidade, pressão,temperatura derivada do senso de pressão - porém, para fins deste trabalho foi utilizado somente a aceleração linear. A escolha deste sensor está relacionada ao trabalho Ballı e Sağbaş (2017), visto que os resultados foram relevantes.

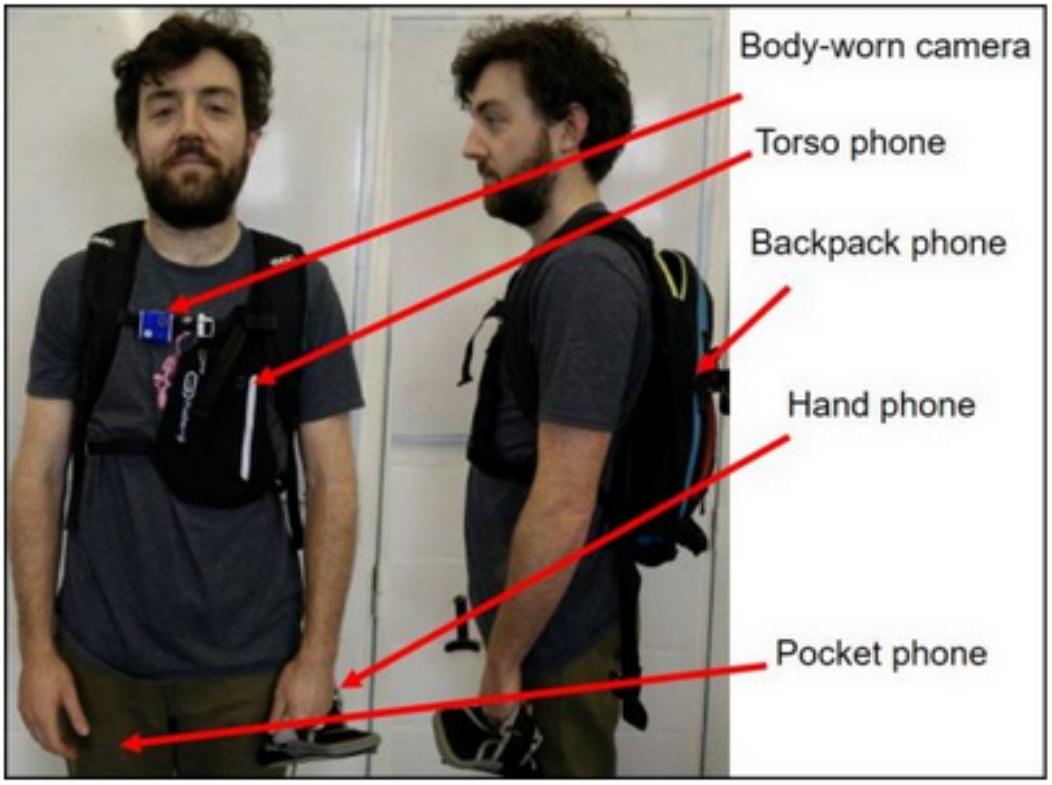
Figura 13 – Horas por atividade



(THE UNIVERSITY OF SUSSEX, 2017)A corrida foi atividade que coletou menor quantidade de dados.

O conjunto de dados disponibilizados contém 3 usuários. O total de dados computados para análise foram 27,4 GB. Cada usuário carrega consigo 4 celulares em posições diferentes, como está demonstrado na Figura14. Essas posições tendem a representar o comportamento do usuário no transporte público. Além disso, os artigos analisados neste trabalho não fazem referência à posição do aparelho eletrônico para coleta de dados. Na prática, a posição influencia a leitura dos sensores embutidos nos *smartphone*, consequentemente na capacidade de classificação do algoritmo.

Figura 14 – Posição do celular



(THE UNIVERSITY OF SUSSEX, 2017) Posições usuais de um celular em uma pessoa

4.4.1 UTILIZAÇÃO DOS DADOS

O *dataSet* possui inúmeras informações e um volume de dados considerável para realizar o processamento. Para tal tarefa, foram utilizados dois equipamentos: um celular Moto G e um notebook com processador Intel Core™ i3-4005U CPU 1.70GHzx4 e memória 3,8 GiB. Devido à quantidade de informações, os dois equipamentos não conseguiram realizar algumas tarefas de processamento, principalmente na questão treinamento do algoritmo J48 com oito classes, como será abordado nos resultados.

Devido esse fato, foi necessário adotar os seguintes procedimentos com objetivo de minimizar a capacidade de processamento dos equipamentos.

O primeiro passo (etapa 1 da Figura15)foi desmembrar os arquivos que continham as atividades em arquivos menores. Esses arquivos estavam agrupados por posição do celular e não por atividade. Portanto, de 36 arquivos(3 - usuários * 4 - posições) foram gerados 128 arquivos, cada um contendo aproximadamente 1 milhão de linhas.

O segundo passo (etapa 2) , após essa divisão, foi possível realizar o processamento para identificar qual atividade e separar em um arquivo por tipo. Ou seja, dos 128 arquivos, foram gerados oito arquivos, cada arquivo representando uma atividade. Na etapa 3, foi necessário desmembrar os dados novamente, pois não foi possível processar diretamente os

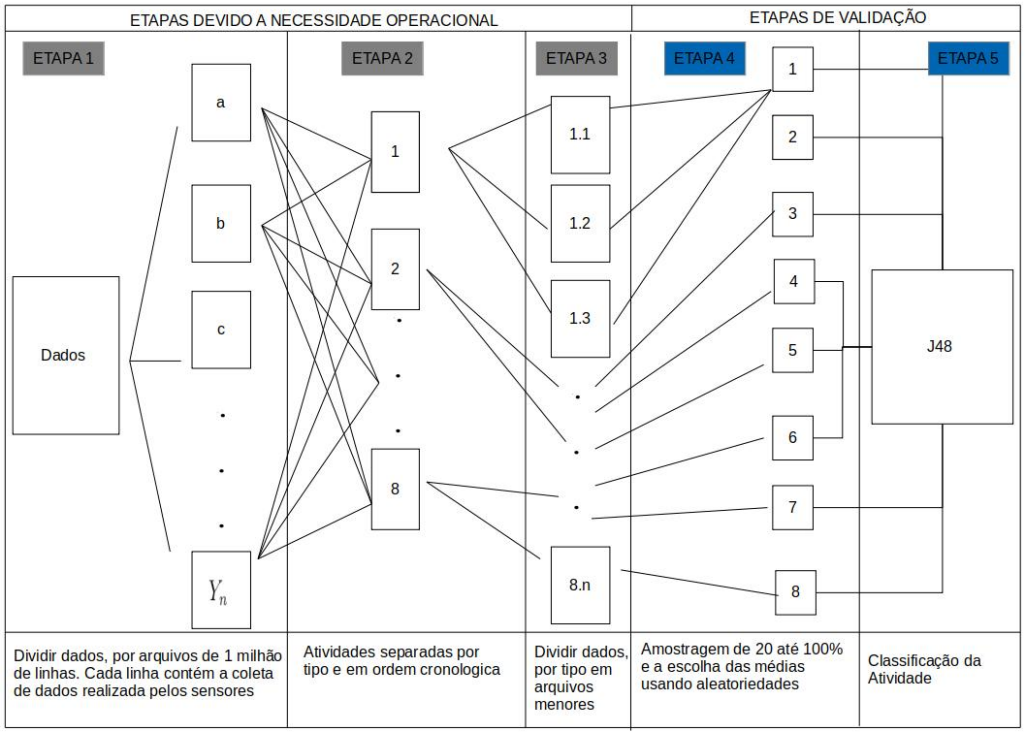
arquivos dos tipos, devido à quantidade de informações contidas nos mesmos. Devido a isso, foi necessário realizar mais uma divisão com o mesmo procedimento.

Na etapa 4 foram processados quatro reduções (20%, 40%, 60% e 80%) do tamanho do arquivo original, com objetivo de compreender o impacto da quantidade de informações na classificação do algoritmo. Além disso, para cada redução, foram criadas as seguintes amostragens das médias - 100, 200, 300, 400, 500, 1000, 1500, 2000 e 2500.

Por último, na etapa cinco, foi realizada a classificação usando o algoritmo J48 utilizando validação cruzada, que será abordada no momento da apresentação dos testes.

Com esses procedimentos, foi possível fornecer dados para a classificação do J48 na maioria dos casos. As exceções são os dados com média 100 e 200 para oito classes.

Figura 15 – Forma como os dados foram trabalhados



As informações foram desmembradas para possibilitar o processamento nos equipamentos disponíveis neste trabalho

4.5 TESTES

O objetivo principal dos testes foi encontrar quais são os melhores parâmetros que podem ser fornecidos ao algoritmo para buscar um resultado aceitável para classificação da atividade. Para isso, os testes buscaram determinar a eficiência do J48 na classificação de uma determinada atividade realizada.

Para isso, foi utilizada a técnica de validação cruzada com objetivo de avaliar a capacidade de generalização. Nesse aspecto buscou-se garantir que o algoritmo J48 tenha o mesmo desempenho para um novo conjunto de dados. Neste trabalho, foi utilizado a técnica *K-fold*, portanto, o conjunto de dados disponíveis foi dividido em 10 subconjuntos exclusivos do mesmo tamanho, sendo nove para o treinamento e um para o teste.

Já em relação às classes, o projeto inicial era não utilizar 8 classes, visto que no escopo deste projeto, o objetivo é atender cidades de pequeno porte, ou seja, sem trem ou metrô. Porém, como os dados estavam disponíveis, estas classes foram incluídas.

No algoritmo J48, no momento de classificação, as atividades são "visualizadas"como números *double*. Sendo assim, as atividades têm as seguintes numerações: parado(1.0), andando(2.0), correndo(3.0), bicicleta(4.0), carro(5.0), ônibus(6.0), trem(7.0) e metrô(8.0).

A divisão dos testes pelo número de atividades, tem como premissa, alcançar desde cidades de pequeno porte até cidades de grande circulação. Vale ressaltar que o conjunto de dados usados foi de um país desenvolvido, a Inglaterra.

Em relação ao número de atividades, previstas nos testes, são as seguintes.

- Seis classes - parado, andando, bicicleta, correndo, carro e ônibus - Experimento 1;
- Oito Classes - todos da anterior mais o trem e o metrô - Experimento 2;

O número total de testes realizados foi 90, sendo que metade para seis classes(cidade de pequeno porte) e a outra metade para oito classes(cidades grande porte).

Os experimentos 1 e 2 seguem o mesmo raciocínio. Para cada nível de amostragem(20% até 100% dos dados) são realizados testes para as amostragem(médias) escolhidas, que vão de 100 até o valor de 2500, ou seja, 9 testes por amostragem. Como são cinco amostragens, $5 * 9 = 45$ testes por experimento.

Devido às limitações dos equipamentos, não foi possível realizar o processamento dos seguintes parâmetros do experimento 2: Primeiro, com média 100 para oito classes com as seguintes amostragens 40%, 60%, 80% e 100%. Segundo, com média 200 para oito classes com as seguintes amostragens 80% e 100%.

Nesses casos, o processamento era interrompido no momento do treinamento do algoritmo J48 acusando falta de memória para processamento.

4.5.1 AMBIENTE DE TESTES

Para realizar os testes, foi utilizado um celular moto e(7) plus, que possui as seguintes configurações listadas abaixo. O processamento ocorre em linguagem Java, através do Android Studio, utilizando pacotes disponíveis do Weka para a classificação.

- Processador :4x 1.8 GHz Kryo 260 + 4x 1.6 GHz Kryo;

- chipSet: Snapdragon 636 Qualcomm SDM636;
- 64 Bit;
- RAM 4 GB;
- Android 10;

A Figura15, apresenta as cinco etapas que foram realizadas no celular. Porém, quando o processamento envolvia médias de 100 e 200 para os dois casos, com 6 e 8 classes, não era possível realizar o processamento devido à falta de memória no celular. Devido a essas limitações, um notebook foi utilizado para obter os resultados dos experimentos através do software Weka 3.8.5 para linux. Nesse caso, foi possível reduzir o tempo de processamento e obter resultados indisponíveis por limitações do celular. Nesse contexto, somente a etapa 5, da Figura15, foi processada no notebook, que tem as seguintes configurações:

- Processador : Intel® Core™ i3-4005U CPU @ 1.70GHz $\times$ 4;
- OS type: 64-bit;
- Memória : 3,8 GiB;
- Ubuntu 18.04.5 LTS;

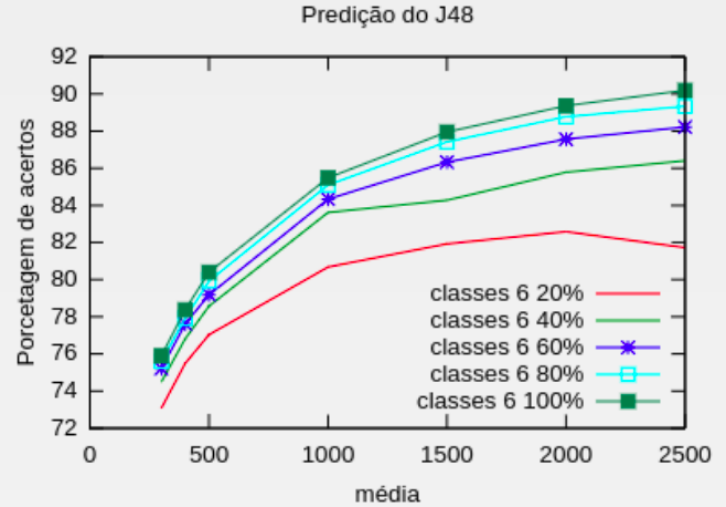
Vale destacar que as limitações envolviam o treinamento do algoritmo. Portanto, deve ser observado que há uma diferença no tempo de processamento do treinamento e classificação de uma atividade. Na linguagem Java, é possível serializar um objeto e ainda enviar o objeto serializado ao Firebase. A implicação desta afirmação é que ao serializar o objeto de uma classe Java, nesse caso J48, o processamento no celular do usuário será somente de classificação da atividade.

5 RESULTADOS

Neste capítulo serão apresentados os resultados obtidos com os 90 testes realizados para classificação da atividade utilizando o J48. Foram avaliados os impactos do número de elementos utilizados na média, inclusive no momento de diminuição da taxa de classificação, taxa de falsos positivos (FPR), juntamente com a matriz de confusão.

A Figura15 ilustra a estratégia utilizada para obter os resultados. Cada atividade possui um comportamento que é expresso na variável escolhida - acelerômetro. Os experimentos 1 e 2 possuem o mesmo método para atingir o resultado, sendo que a diferença entre ambos é o número de atividades. As figuras 16 e 17 demonstram que o comportamento das curvas do gráfico é similar, aproximando-se de uma função $F(x) = log(x)$. A semelhança vai além, desse aspecto, observando que quando há um aumento do número de elementos na média, há uma maior porcentagem de acertos do J48. Observe na Figura17, que com amostragem dos dados de 60% (linha roxa) a correta classificação das instâncias aumenta com o número de elementos utilizados na média - (nº 1000 - 73.37%) < (nº 2000 - 76.27%) até um ponto crítico.

Figura 16 – Predição do J48 - seis classes

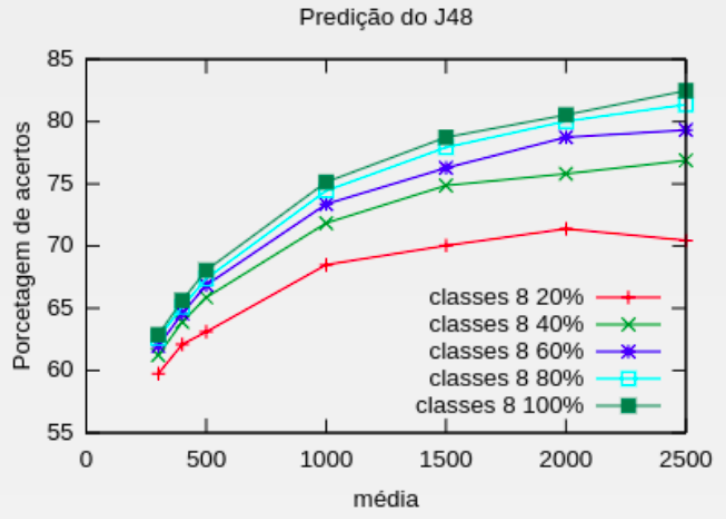


(THE UNIVERSITY OF SUSSEX, 2017)Apresentam o mesmo comportamento em relação ao eixo da média

Neste trabalho o ponto crítico é definido quando a taxa de ganho de precisão torna-se negativa, reduzindo a acurácia do algoritmo do J48. Nesse contexto, o ponto crítico está relacionado com os dados que são fornecidos ao algoritmo, por isso, cada amostragem - 20% até 100% - tem pontos críticos diferentes.

Pode-se analisar o ponto crítico ao observar a Figura20, que compara o ganho de acertos em relação à média anterior. A taxa de acertos está crescendo a cada aumento do número de elementos usados na média (nº 500 para nº1000 elementos aumentou em 3,64% a taxa de acertos). Para exemplificar esses números, a precisão na Figura19 com

Figura 17 – Predição do J48 - oito classes



(THE UNIVERSITY OF SUSSEX, 2017)Apresentam o mesmo comportamento em relação ao eixo da média

500 elementos é de 77,04%, já com 1000 elementos é de 80,68%, ou seja, a diferença é de 3,64%. Esses dados são importantes para encontrar o ponto crítico que, nesse caso, ocorreu 2216 elementos, onde a taxa de acertos tornou-se negativa, a partir desse momento, o aumento da amostragem para o cálculo da média reduz a acurácia do J48. Vale ressaltar que esse comportamento está relacionado ao conjunto de dados fornecidos ao J48. Além disso, todos os testes tendem a ir ao encontro deste ponto, como fica claro na Figura18, com aumento no eixo x há uma diminuição da taxa de ganho em relação ao número de elementos na média anterior.

5.0.1 J48 - SEIS CLASSES(EXPERIMENTO 1)

Nessa seção serão abordados os resultados com os dados do Experimento 1, apresentando as tabelas 1 e 2 que demonstram a acurácia do J48.

O trabalho Ballı e Sağbaş (2017) não analisou duas diferenças que impactaram no resultado da acurácia deste trabalho. A primeira, é a atividade ”parado”, ou seja, quando o usuário encontrasse dentro de um prédio, casa e etc. O J48 encontrou dificuldades com essa atividade, ao ponto que apresentou o maior taxa de falso positivo (FPR), como consta na tabela 1. Ou seja, na maioria das classes a atividade ”parado”impactou no resultado.

Essa atividade, ao observar a tabela 1, na coluna Falsos positivos(FPR), apresentou um resultado acima da média(soma de todas FPR dividido por 5). Com esse dado, pode-se afirmar que a mesma provocou o maior impacto na taxa de acurácia do algoritmo.

O segundo fator é relacionado às 4 posições do celular. Não há referências no trabalho Ballı e Sağbaş (2017) sobre esse fato. Cada posição, como consta na Figura14,

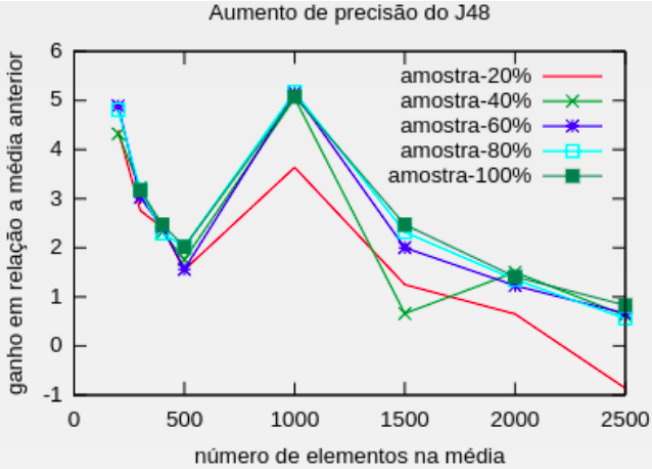


Figura 18 – Precisão do J48

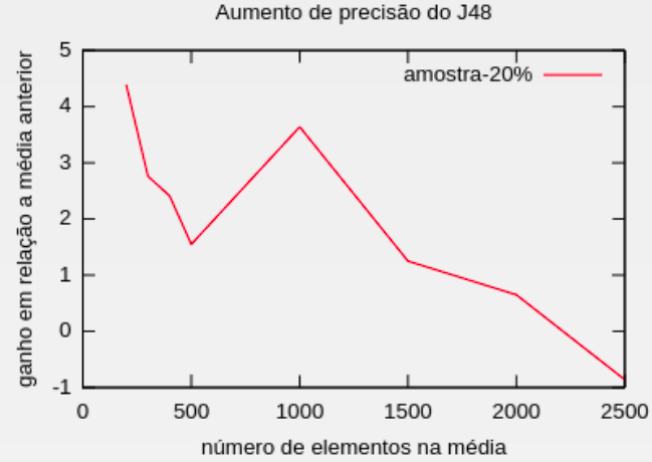


Figura 19 – Ponto crítico - quando eixo y é zero

Figura 20 – Fonte: Elaborado pelo autor.

Tabela 1 – Matriz de Confusão - 6 classes

Classes	Parado	Caminhando	Correndo	Bicicleta	Carro	Ônibus	FPR
Parado	**5238**	74	0	171	190	186	0,034
Caminhando	66	**5453**	2	123	39	25	0,015
Correndo	0	2	**580**	0	0	0	0,000
Bicicleta	195	136	0	**3687**	91	54	0,023
Carro	149	31	0	95	**4001**	185	0,028
Ônibus	195	27	0	59	213	**2275**	0,022

tem impacto no sensor de acelerômetro. Todavia, não é o foco deste trabalho aprofundar com mais detalhes as diferenças dos resultados.

Sobre a tabela 1, a matriz de confusão foi construída com 100% dos dados disponíveis. A escolha de demonstrar somente essa matriz de confusão foi devido que a mesma apresentou os melhores resultados. Em relação ao número de elementos utilizados na média foi o caso com 2500 elementos. Observe que os maiores números estão na diagonal,

Tabela 2 – Acurácia por Classe - 6 classes

Classes	**Nº elementos para o cálculo da média**				
	500	**1000**	**1500**	**2000**	**2500**
Parado	80,30%	84,60%	87,10%	88,10%	89,60%
Caminhando	90,70%	93,40%	94,50%	94,80%	95,30%
Correndo	99,60%	99,70%	99,70%	99,50%	99,70%
Bicicleta	77,90%	84,10%	86,80%	88,10%	89,20%
Carro	75,50%	82,80%	85,60%	87,80%	88,20%
Ônibus	66,80%	74,50%	79,40%	82,90%	83,50%
Precisão Geral	**80,40%**	**85,50%**	**87,90%**	**89,30%**	**90,20%**

o que mostra a precisão do algoritmo. Nas atividades, caminhando e correndo, foram as que representaram as maiores precisões com 5453 e 580 acertos, respectivamente. Esses números representam uma acurácia de 95,3% e 99,7% (Tabela 2). Essas duas atividades apresentaram os FPR com resultados significativos, 0,015 e zero(0.0).

No que diz respeito à atividade ”correndo”, apesar dos limites dos dados, o J48 determinou com uma acurácia significativa. Ao observar a Figura11, fica notória a diferença dos dados do acelerômetro da atividade ”correndo”para outras atividades. Por isso, o J48 apresentou, para essa atividade, o seu melhor resultado. Portanto, apesar da limitação dos dados em termos de quantidade, este não foi um fator limitador.

Sobre as outras quatro atividades, todas apresentaram, em todos os cenários, resultados menores de 90% conforme demonstrado na tabela 2. Contudo, todas as atividades aumentaram sua precisão quando o número de elementos para cálculo da média sofreu alteração, ao ponto da atividade ”ônibus”iniciar com 66% e terminar com 83%.

A atividade ”ônibus”, foco deste trabalho, apresentou resultados relevantes. O primeiro, é que sua precisão alcançou 83%, sendo possível identificar uma relação de ganho entre precisão e número de elementos usados na média. O segundo, é em relação ao FPR, pois este apresentou resultado inferior (0,022),sendo que a média dessa variável é 0,024. Os resultados usando o J48 sofreram impactos devido às altas taxas dos falsos positivos das classes - bicicleta(54), carro(185) e principalmente, ”parado”(186).

Sobre a detecção da atividade carro, demonstrou-se que é possível alcançar resultados importantes, como a precisão 88,2% na última coluna da tabela 2. Porém, encontrou-se os mesmos impasses dos falsos positivos nas atividades ”parado”, ônibus e bicicleta, todavia, com melhores resultados.

Já atividade bicicleta encontrou os mesmos impasses, das duas últimas atividades citadas, porém com resultados melhores - 89,2%.

Os dados do sensor acelerômetro demonstraram que podem ser usados com precisão considerável para classificar qual é atividade que está sendo executada.

Tabela 3 – Matriz de Confusão - 8 classes

Classes	Parado	Caminhando	Correndo	Bicicleta	Carro	Ônibus	Trem	Metrô	FPR
Parado	**4438**	54	0	187	97	124	472	487	0,047
Caminhando	52	**5446**	2	116	22	12	41	17	0,01
Correndo	0	2	**580**	0	0	0	0	0	0
Bicicleta	185	118	0	**3631**	91	35	83	20	0,019
Carro	86	21	0	84	**3870**	154	103	143	0,022
Ônibus	120	14	0	52	167	**2121**	130	165	0,02
Trem	446	56	0	111	137	134	**4205**	452	0,043
Metrô	448	11	0	27	146	155	383	**3762**	0,044

5.0.2 J48 - 8 CLASSES (EXPERIMENTO 2)

Nessa seção, a diferença para os resultados apresentados na seção anterior, é que houve um incremento de duas atividades para classificação - trem e metrô. Observe que essas atividades estão relacionadas a cidades de maior porte como São Paulo ou Rio de Janeiro.

As mesmas dificuldades que o J48 encontrou com 6 seis classes foram potencializadas no experimento 2. Ao observar a tabela 3, o FPR da atividade ”parado”apresentou o maior índice, como ocorreu no experimento 1. Na atividade ”metrô”, 487 casos de falsos positivos ocorreram, demonstrando que o impacto na acurácia do J48 é significativo.

Ainda sobre o FPR, as duas atividades adicionadas apresentaram números semelhantes da atividade ”parado”, como fica evidente ao observar as últimas duas linhas da última coluna da tabela 3. Esses números impactaram a acurácia geral do J48.

Na tabela 4, o maior resultado foi com 2500 elementos para o cálculo da média - 82,4%, que na prática representou uma diminuição da acurácia de 8,6% aproximadamente, comparando com experimento 1.

O experimento 2 também apresentou um aumento da acurácia com o aumento do número de elementos para o cálculo da média. Na tabela 4, a atividade ”ônibus”teve um ganho de 20,2% com o aumento de 500 para 2500 elementos, portanto apresentando o mesmo comportamento com seis classes.

Contudo, as três classes que envolvem transporte público apresentaram resultados inferiores a 80% de acurácia. Esse fato tem como consequência direta os altos números dos falsos positivos da atividade ”parado”. Ao observar a tabela 3, quase todas as atividades tiveram um grande impacto na sua acurácia devido à atividade ”parado”, exceto caminhando e correndo. Ao analisar a porcentagem dos erros, no caso do metrô, o valor chegou a aproximadamente 10%.

Sobre a classe correndo, mesmo com poucos dados, esta continuou a apresentar os melhores resultados - 99,7%.

Portanto, o comportamento do J48 perante o experimento 2 apresentou uma eficiência razoável, devido ao alto índice do FPR da atividade ”parado”.

Tabela 4 – Acurácia por Classe - 8 classes

Classes	**Nº elementos para o cálculo da média**				
	500	**1000**	**1500**	**2000**	**2500**
Parado	62,2%	68,6%	72,0%	73,8%	76,8%
Caminhando	89,0%	91,7%	93,7%	94,4%	95,2%
Correndo	99,7%	99,8%	99,6%	99,6%	99,7%
Bicicleta	73,7%	80,9%	83,8%	85,6%	86,3%
Carro	67,7%	77,7%	80,9%	84,1%	85,4%
Ônibus	56,4%	67,1%	73,1%	75,1%	77,6%
Trem	60,4%	67,5%	72,7%	74,5%	77,6%
Metrô	57,2%	66,4%	70,6%	72,4%	74,6%
Precisão Geral	**68,0%**	**75,1%**	**78,7%**	**80,5%**	**82,4%**

6 CONSIDERAÇÕES FINAIS

Buscando fomentar novos trabalhos e buscando criar uma relação de conhecimento com mudança de vida população, este trabalho teve como objetivo geral a demonstração da eficiência do J48 para classificação de qual atividade está sendo executada com objetivo de determinar se um indivíduo está usando o transporte público. Com esse conhecimento é possível realizar a criação de aplicativos que possam contribuir com um transporte público que atenda as necessidades do usuário, principalmente adicionando previsibilidade a esta.

Para alcançar este objetivo principal, foi utilizado o algoritmo J48 do Weka - C4.5 - juntamente com um conjunto de dados considerável, da *Sussex Huawei Locomotion*. Vale ressaltar que a solução se baseia em algoritmos livres com desempenho satisfatório. A partir dos testes constatou-se que a solução para o problema proposto é viável, utilizando a variável acelerômetro, com algumas limitações. Porém, é possível adicionar outros dados no algoritmo J48 com objetivo alcançar melhores resultados.

Através da metodologia usada para buscar a literatura do tema, constatou-se a notoriedade do trabalho Zhou, Zheng e Li (2012) citado por 601 trabalhos relacionados ao tema. Complementando esse trabalho, na escolha do algoritmo e da variável acelerômetro, utilizou-se o trabalho de Ballı e Sağbaş (2017). Durante os testes, analisaram-se questões relativas às limitações dos equipamentos disponíveis e, com isso, alguns procedimentos foram adotados com vistas a contornar essas limitações e algumas renúncias foram feitas, como testes que exigiam memória além da capacidade dos equipamentos disponíveis.

Durante o desenvolvimento do trabalho, analisaram-se questões relativas à eficiência do J48 perante cenários semelhantes à realidade do usuário do transporte público, principalmente na questão dos dados que representam as possibilidades do cotidiano. Para isso, os dados utilizados possuem informações úteis, como posições do celular em relação ao corpo, simulando o modo de uma pessoa carregar o mesmo. Esse fato contribuiu e muito com este trabalho, mostrando que o algoritmo sofreu impacto ao analisar esses dados, quando comparamos os resultados limitando a posição do celular.

Para avaliar a qualidade da J48, foram realizados testes (90 ao total) utilizando a ferramenta Weka, em uma aplicação desenvolvida no Android Studio e em uma aplicação no próprio Weka. Comparando os resultados de Ballı e Sağbaş (2017) com o trabalho desenvolvido, ficou evidente o impacto da atividade "parado"nos resultados dos testes. Os valores do FPR são altíssimos, principalmente, no experimento com oito classes - 0,047. Conclui-se que o J48 encontrou dificuldades para classificar essa atividade. Uma solução para essa situação é adicionar novos dados obtidos através dos sensores, como por exemplo do giroscópio.

Ao analisar os resultados dos testes, foi possível identificar que há uma relação entre a amostragem utilizada na média e a acurácia. Utilizando como referência a tabela 1 conclui-se que a amostragem da média deve ser dinâmica, pois há um aumento na acurácia quando se modifica a amostragem da média. A função logarítmica ,$f(x) = log(x)$, é a

que melhor se aproxima dessa relação. Embora ainda seja necessária uma validação com conjuntos de dados diferentes, esta constatação possibilita que futuros trabalhos possam a partir dessa constatação encontrar melhores resultados.

Ao adentrar na acurácia do J48 este trabalho, apresentou uma redução considerável nos dois grupos de experimento se comparado ao trabalho Ballı e Sağbaş (2017). No primeiro grupo(seis classes) e no segundo grupo(oito classes), a redução foi de 8.6% e 18.4% respectivamente. Ao comparar os resultados, devem ser levadas em consideração as diferenças dos trabalhos como o número de classes. Porém, mesmo com incremento do número de classes, é possível observar que o J48 apresentou resultados significativos para seis classes - 90.2% de acurácia geral.

Ao observar os dados disponíveis da universidade *Sussex Huawei Locomotion*, é importante salientar que é possível adicionar outros dados, aumentando as informações sobre um mesmo evento.

A partir dos resultados da seção 5, constatou-se que o J48 tem sua acurácia para a atividade "correndo"surpreendentemente alta, em ambos os grupos apresentou o mesmo resultado - 99.7%. Os dados do acelerômetro nessa atividade possuem diferenças significativas em relação às outras atividades.

Durante o desenvolvimento deste trabalho, várias questões técnicas e de privacidade surgiram. Quanto às limitações dos equipamentos, foram utilizadas algumas estratégias para reduzir o impacto. Já em relação à privacidade, as escolhas foram direcionadas a preservar a intimidade do usuário, buscando soluções conciliadoras. Para trabalhos futuros, pretende-se utilizar a validação da proposta com outras variáveis e adicionando outros algoritmos nos testes. Também vislumbra-se a possibilidade de desenvolver o trabalho da proposta inicial, estudar formas de otimizar a gerência da memória, que foi um dos aspectos limitadores e estudar formas de otimizar o tempo de execução.

REFERÊNCIAS

BALLI, S.; SAĞBAŞ, E. A. Diagnosis of transportation modes on mobile phone using logistic regression classification. *IET Software*, IET, v. 12, n. 2, p. 142–151, 2017. Citado 8 vezes nas páginas 50, 52, 53, 57, 58, 64, 69 e 70.

BARBOSA, J. M.; CARNEIRO, T. G. de S.; TAVARES, A. I. Métodos de classificação por árvores de decisão disciplina de projeto e análise de algoritmos. *UFOP–Universidade Federal de Ouro Preto Ouro Preto, Minas Gerais–MG*, 2012. Citado na página 42.

BARBOSA, J. M.; CARNEIRO, T. G. de S.; TAVARES, A. I. Métodos de classificação por árvores de decisão disciplina de projeto e análise de algoritmos. *UFOP–Universidade Federal de Ouro Preto Ouro Preto, Minas Gerais–MG*, 2012. Citado na página 44.

DEVELOPERS. *Tudo que você precisa para construir no Android*. 2018. Disponível em: ⟨https://developer.android.com/studio/features/⟩. Acesso em: 19 nov. 2018. Citado na página 45.

EMQUESTAO. *Vaucher critica falta de política de transporte público em Alegrete*. 2016. Disponível em: ⟨http://emquestao.com.br/2016/08/25/vaucher-critica-falta-de-politica-de-transporte-publico-em-alegrete/⟩. Acesso em: 19 nov. 2018. Citado na página 21.

ESTEVES, G. R. T. *Modelos de Previsão de Carga de Curto Prazo*. Dissertação (Mestrado) — PUCRio, Brasil, 2003. Citado 2 vezes nas páginas 25 e 29.

FACURE, M. *Introdução às Redes Neurais Artificiais*. 2017. Disponível em: ⟨https://matheusfacure.github.io/2017/03/05/ann-intro/⟩. Acesso em: 21 nov. 2018. Citado 2 vezes nas páginas 37 e 53.

FERNANDO NOGUEIRA. *Modelagem e Simulação - Modelos de Previsão*. 2009. Disponível em: ⟨http://www.ufjf.br/epd042/files/2009/02/previsao1.pdf⟩. Acesso em: 16 nov. 2018. Citado 4 vezes nas páginas 27, 28, 29 e 55.

FONSECA, J. da. *Indução de Árvores de decisão*. Tese (Doutorado) — Dissertação de Mestrado, Universidade Nova de Lisboa, Lisboa, 1994. Citado na página 41.

GURMU ZEGEYE KEBEDE E FAN, W. D. *Predicting Bus Arrival Time on the Basis of Global Positioning System Data*. [S.l.], 2014. Citado 3 vezes nas páginas 29, 30 e 49.

HAYKIN, S. *Redes Neurais Artificiais: Princípios e Práticas. 2ª edição*. [S.l.]: Editora Bookman, Porto Alegre, 900p, 2001. Citado 5 vezes nas páginas 36, 37, 38, 39 e 41.

IMASTERS. *Firebase x Parse Server*. 2016. Disponível em: ⟨https://imasters.com.br/desenvolvimento/firebase-x-parse-server⟩. Acesso em: 19 nov. 2018. Citado na página 45.

LADEIRA, M.; MICHEL, F.; SENNA, L. Public transport monitoring and control: The case of porto alegre, brazil. In: *ICTIS 2011: Multimodal Approach to Sustained Transportation System Development: Information, Technology, Implementation*. [S.l.: s.n.], 2011. p. 275–281. Citado 2 vezes nas páginas 22 e 47.

MASTERTECH. *O que é e como funciona a plataforma*. 2017. Disponível em: ⟨https://blog.mastertech.com.br/tecnologia/google-firebase-for-dummies-o-que-e-e-como-funciona-plataforma/⟩. Acesso em: 19 nov. 2018. Citado na página 44.

MICHAEL J. ROSENFELD. *OLS in Matrix Form*. 2013. Disponível em: ⟨https://web.stanford.edu/~mrosenfe/soc_meth_proj3/matrix_OLS_NYU_notes.pdf⟩. Acesso em: 20 nov. 2018. Citado 2 vezes nas páginas 26 e 27.

PATNAIK, J.; CHIEN, S.; BLADIKAS, A. Estimation of bus arrival times using apc data. *Journal of public transportation*, v. 7, n. 1, p. 1, 2004. Citado na página 31.

PAULA, M. d.; BARTELT, D. D. Mobilidade urbana no brasil: desafios e alternativas. *Rio de Janeiro: Fundação Heirich Boll*, 2016. Disponível em: ⟨https://br.boell.org/sites/default/files/mobilidade_urbana_boll_brasil_web_.pdf⟩. Citado 4 vezes nas páginas 21, 22, 47 e 48.

SINGH, G.; BANSAL, D.; SOFAT, S. Eta htc: Estimating time of arrival under heterogeneous traffic conditions using crowdsensing. In: IEEE. *2017 International Conference on Inventive Computing and Informatics (ICICI)*. [S.l.], 2017. p. 175–179. Citado 2 vezes nas páginas 48 e 49.

SUN, D. et al. Predicting bus arrival time on the basis of global positioning system data. *Transportation Research Record: Journal of the Transportation Research Board*, Transportation Research Board of the National Academies, n. 2034, p. 62–72, 2007. Citado 4 vezes nas páginas 30, 50, 55 e 56.

THE UNIVERSITY OF SUSSEX. *The University of Sussex-Huawei Locomotion (SHL) dataset - a versatile annotated dataset for multimodal locomotion analytics of mobile users*. 2017. Disponível em: ⟨http://www.shl-dataset.org⟩. Acesso em: 06 de Agosto 2021. Citado 5 vezes nas páginas 57, 58, 59, 63 e 64.

WEIGANG, L. et al. Algorithms for estimating bus arrival times using gps data. In: IEEE. *Intelligent Transportation Systems, 2002. Proceedings. The IEEE 5th International Conference on*. [S.l.], 2002. p. 868–873. Citado na página 29.

ZARCHAN, P.; MUSOFF, H. *Fundamentals of Kalman filtering: a practical approach*. [S.l.]: American Institute of Aeronautics and Astronautics, Inc., 2013. Citado 4 vezes nas páginas 32, 33, 34 e 35.

ZHOU, P.; ZHENG, Y.; LI, M. How long to wait?: predicting bus arrival time with mobile phone based participatory sensing. In: ACM. *Proceedings of the 10th international conference on Mobile systems, applications, and services*. [S.l.], 2012. p. 379–392. Citado 3 vezes nas páginas 50, 51 e 69.

APÊNDICE A – EXEMPLO DE FILTRO KALMAN DE 2º ORDEM, RASTREANDO UM OBJETO EM QUEDA - JAVA

```java
package filterkalmanrecursivo;

import Jama.Matrix;
import java.util.Random;

/**
 * Example of Kalman Filter Tracking a Falling Object
 * pag 159, listing 4.3
 * @author Paul Zarchan, Howard Musoff
 * O aluno Lucas, somente transformou de Fortran para JAVA.
 */
public class secondOrder {

    int ORDER = 3;
    double PHIS = 0.; // ruido de processo
    int TS = 1;
    double A0 = 400000;
    double A1 = -6000;
    double A2 = -16.1;
    double XH = 0; // estimativa
    double XDH = -5000; //estimativa  1 derivada
    double XDDH = 0; // estimativa 2 derivada
    double SIGNOISE = 1000.;


    Matrix M = new Matrix(3, 3); // formula - covariancia antes
       update
    Matrix P = new Matrix(3, 3); // formula - covariancia depois
       update
    Matrix K = new Matrix(3, 1); // ganho
    Matrix PHI = new Matrix(3, 3); // forumula
    Matrix H = new Matrix(1, 3); // ordem
    Matrix R = new Matrix(1, 1); // foruma no K
    Matrix PHIT = new Matrix(3, 3); // tranposta do PHI

    Matrix PHIP = new Matrix(3, 3); // PHI * P
    Matrix HT = new Matrix(3, 1); // transposta de H
    Matrix KH = new Matrix(3, 3); // K * H
    Matrix IKH = new Matrix(3, 3);
```

```
38
39      Matrix MHT = new Matrix(3, 1); // M * HT
40      Matrix HMHT = new Matrix(1, 1);
41      Matrix HMHTR = new Matrix(1, 1); // HMHT* R
42      Matrix HMHTRINV = new Matrix(1, 1); // HMHTR* INV
43      Matrix IDN = new Matrix(3, 3); // matriz identidade
44
45      Matrix Q = new Matrix(3, 3);
46      Matrix PHIPPHIT = new Matrix(3, 3);
47
48      secondOrder() {
49
50
51          double X = 0;
52          double XD = 0;
53          double XDD = 0;
54          double XS = 0;
55          double RES = 0;
56          double XH = 0;
57          double XDH = 0;
58          double XDDH = 0;
59
60          double SP11 = 0;
61          double SP22 = 0;
62          double SP33 = 0;
63          double XHERR = 0;
64          double XDHERR = 0;
65          double XDDHERR = 0;
66
67          System.out.println("T XHERR SP11 -SP11 XDHERR SP22 -SP22
               XDDHERR SP33 -SP33");
68          //System.out.println("T X XH XD XDH XDD XDDH");
69          for (int i = 0; i < 3; i++) {
70              for (int j = 0; j < 3; j++) {
71                  PHI.set(i, j, 0);
72                  P.set(i, j, 0);
73                  IDN.set(i, j, 0);
74                  Q.set(i, j, 0);
75              }
76          }
77          IDN.set(0, 0, 1);
78          IDN.set(1, 1, 1);
```

```
79          IDN.set(2, 2, 1);
80
81          P.set(0, 0, 999999999);
82          P.set(1, 1, 999999999);
83          P.set(2, 2, 999999999);
84
85          PHI.set(0, 0, 1);
86          PHI.set(0, 1, TS);
87          PHI.set(0, 2, 0.5 * TS * TS);
88
89          PHI.set(1, 1, 1);
90          PHI.set(1, 2, TS);
91          PHI.set(2, 2, 1);
92
93          for (int i = 0; i < 3; i++) {
94              H.set(0, i, 0);
95          }
96
97          H.set(0, 0, 1);
98          HT = H.transpose();
99          R.set(0, 0, Math.pow(SIGNOISE, 2));
100         PHIT = PHI.transpose();
101
102         /*
103         Q esta relacionado a ruido, como
104         esse processo e ruido zero, o PHIS
105         tem valor zero.
106          */
107         Q.set(0, 0, PHIS * (Math.pow(TS, 5.0)) / 20.0);
108         Q.set(0, 1, PHIS * (Math.pow(TS, 4.0)) / 8.0);
109         Q.set(0, 2, PHIS * (Math.pow(TS, 3)) / 6.0);
110
111         Q.set(1, 0, Q.get(0, 1));
112         Q.set(1, 1, PHIS * (Math.pow(TS, 3)) / 3.0);
113         Q.set(1, 2, PHIS * (TS * TS) / 2.0);
114
115         Q.set(2, 0, Q.get(0, 2));
116         Q.set(2, 1, Q.get(1, 2));
117         Q.set(2, 2, PHIS * TS);
118
119         for (int T = 0; T < 30; T = T + TS) {
120
```

```
121             PHIP = PHI.times(P); // obsever que PHI*P = PHIP
122             PHIPPHIT = PHIP.times(PHIT);
123             M = PHIPPHIT.plus(Q); // valor de M
124 
125             this.MHT = M.times(HT);
126             HMHT = H.times(this.MHT);
127             this.HMHTR.set(0, 0, this.HMHT.get(0, 0) + this.R.get
                    (0, 0));
128             this.HMHTRINV.set(0, 0, 1.0 / (this.HMHTR.get(0, 0)))
                    ;
129             //HMHTRINV  = (H*MHT + R)* inversa
130             // calculo da inversa
131             K = this.MHT.times(HMHTRINV); // valor de k
132 
133             KH = K.times(H);
134             IKH = IDN.minus(KH);
135             P = IKH.times(M); // valor de P , matriz de
                    covariancia
136 
137             Random var = new Random();
138             double XNOISE = var.nextGaussian() * this.SIGNOISE;
139 
140             X = A0 + A1 * T + A2 * T * T;
141             XD = A1 + 2 * A2 * T;
142             XDD = 2 * A2;
143             XS = X + XNOISE;
144             RES = XS - XH - TS * XDH - 0.5 * TS * TS * XDDH;
145           //XH=  XH +  XDH* TS + 0.5 * TS * TS * XDDH + K(1,1)
                      *  RES
146             XH = XH + XDH * TS + 0.5 * TS * TS * XDDH + K.get(0,
                    0) * RES;
147           //XDH = XDH + XDDH * TS + K(2,1)       *RES
148             XDH = XDH + XDDH * TS + K.get(1, 0) * RES;
149 
150             XDDH = XDDH + K.get(2, 0) * RES;
151 
152             SP11 = Math.sqrt(P.get(0, 0)); //  variacoes de erros
153             SP22 = Math.sqrt(P.get(1, 1)); //  variacoes de erros
154             SP33 = Math.sqrt(P.get(2, 2)); //  variacoes de erros
155             XHERR = X - XH; //  diferenca do real - estimativa
                    nesse caso altitude
156             XDHERR = XD - XDH; // diferenca do real derivado par
```

```
                ao derivado da estimativa
157             XDDHERR = XDD - XDDH; // (2 derivida)diferenca do
                real derivado par ao derivado da estimativa
158
159             //System.out.println(T + " " + (float) X + " " + (
                float) XH + " " + (int) XD + " " + (int) XDH + " "
                + (float) XDD + " " + (float) XDDH);
160             System.out.println(T + " " + (int) XHERR + " " + (int
                ) SP11 + " " + (int) -SP11 + " " + (float) XDHERR +
                 " " + (float) SP22 + " " + (float) -SP22 + " " + (
                float) XDDHERR + " " + (float) SP33 + " " + (float)
                 -SP33);
161         }
162
163     }
164 }
```