CityTracks-RT: Uma aplicação para detecção do modo de transporte em tempo real nos centros urbanos

Alternative Title: CityTracks-RT: An application for real-time travel mode detection in urban centers

Elton F. de S. Soares
Universidade Federal do Estado do Rio de Janeiro
Av. Pasteur, 458
Rio de Janeiro, Brasil
elton.soares@uniriotec.br

Carlos A. M. S. Quintella
Universidade Federal do Estado do Rio de Janeiro
Av. Pasteur, 458
Rio de Janeiro, Brasil
caquintella@gmail.com

Carlos A. V. Campos
Universidade Federal do Estado do Rio de Janeiro
Av. Pasteur, 458
Rio de Janeiro, Brasil
beto@uniriotec.br

RESUMO

Sistemas para detecção do modo de transporte são cada vez mais necessários para aplicações sensíveis ao contexto em sistemas de transporte inteligente. No entanto, poucas aplicações permitem esta detecção em tempo real através do uso de *smartphones*. Neste trabalho, apresentamos uma proposta de aplicação para inferência do modo de transporte em tempo real com base em *traces* de GPS, utilizando uma técnica de mineração de dados através da qual esses *traces* são pré-processados, agrupados em segmentos de movimento e classificados por meio de algoritmos de aprendizado de máquina supervisionado. Foi implementado um protótipo da aplicação na plataforma Android, usada em smartphones, para coleta do movimento e detecção do modo de transporte do usuário utilizando a API do WEKA em Java. Por fim, para avaliação do desempenho da aplicação em um ambiente real foram realizados testes de campo com voluntários da região metropolitana do Rio de Janeiro, através dos quais, 1338 inferências de modos de transporte foram obtidas por quatro técnicas de aprendizado de máquina e os resultados foram avaliados e comparados através dos indicadores da matriz de confusão. Assim, através da avaliação de desempenho realizada foi possível verificar que a aplicação proposta é útil para a detecção do modo de transporte em tempo real nos centros urbanos.

Palavras-Chave

sensoriamento móvel, cidades inteligentes, detecção do modo de transporte.

ABSTRACT

Context-aware applications in intelligent transport systems have a growing need for travel mode detection systems. However, few applications allow real-time travel mode detection through the use of smartphones. In this paper, we propose a real-time travel mode detection application based on GPS traces using a data mining technique through which these traces are preprocessed, grouped in motion segments and classified by supervised machine learning algorithms. An application prototype was implemented on the Android platform, used by smartphones, for movement data collection and user travel mode detection using the WEKA API in Java. Finally, to evaluate the performance of the application in a real environment, field tests were carried out with dozens of volunteers in the metropolitan area of Rio de Janeiro through which 1338 travel mode inferences were obtained by four machine learning techniques and the results were evaluated and compared through the indicators of the confusion matrix. Thus, through the performance evaluation carried out, it was possible to verify that the proposed application is useful for real-time travel mode detection in urban centers.

SBSI 2017 June 5^{th} – 8^{th}, 2017, Lavras, Minas Gerais, Brazil

CCS Concepts

•Information systems → Mobile information processing systems; •Human-centered computing → Smartphones; Empirical studies in ubiquitous and mobile computing;

Keywords

mobile sensing, smart cities, travel mode detection.

1. INTRODUÇÃO

Em 2011, cerca de cinquenta por cento da população mundial vivia em áreas urbanas e em 2050 esta porcentagem será de aproximadamente setenta por cento da população. Dado isso, a maior onda de urbanização ainda está por vir e junto com ela virá uma grande oportunidade de se melhorar os estilos de vida das pessoas e grandes desafios na economia, saúde, meio ambiente e planejamento urbano [3]. Entender melhor como as cidades funcionam permite melhorias na prestação de serviços e a criação de cadeias de comunicação com os habitantes para reduzir, por exemplo, o consumo de energia e impacto ambiental causado pelas cidades. Além disso, os telefones móveis ou *smartphones* estão rapidamente se tornando o dispositivo central de computação

e comunicação na vida das pessoas e estão sendo vendidos com um conjunto cada vez maior de sensores programáveis embutidos [8]. Tendo em vista este cenário, as aplicações de sensoriamento urbano utilizam os sensores programáveis presentes nos *smartphones* para coleta de dados estatísticos sobre os centros urbanos e geração de informações úteis para se entender melhor o funcionamento destes centros, com o intuito de permitir melhorias nos serviços oferecidos.

Algoritmos de detecção do modo de transporte possibilitam uma gama de aplicações sensíveis ao contexto para cidades inteligentes, principalmente no que diz respeito à criação de sistemas de transportes inteligentes, preservação do meio ambiente e monitoramento da qualidade de vida da população. Uma vez que seja possível identificar o modo de transporte utilizado pelo usuário, é possível coletar e fornecer informações sobre o modo de transporte utilizado, calcular o total de calorias gasto e a quantidade de dióxido de carbono emitida durante o deslocamento diário [12]. Os principais beneficiários destas aplicações seriam os planejadores urbanos, que poderiam utilizar os dados coletados para tornar os sistemas de transporte público mais eficientes, agentes de saúde, que poderiam monitorar os hábitos diários de pacientes sob observação, órgãos de proteção ao meio ambiente, que poderiam estimar a emissão de gases poluentes causada por meios de transporte motorizados com maior precisão, e usuários comuns, que poderiam se beneficiar de todas estas informações para melhorar suas tomadas de decisão quanto aos meios de transporte utilizados no dia-a-dia [2].

Diversos algoritmos de detecção do modo de transporte foram desenvolvidos ao longo da ultima década, no entanto, poucos estudos encontrados nesta área apresentaram a implementação e avaliação destes algoritmos em uma aplicação de detecção do modo de transporte em tempo real. Neste trabalho, apresentamos uma proposta de aplicação para inferência do modo de transporte em tempo real com base em *traces* de localização, utilizando uma técnica de mineração de dados através da qual os *traces* são pré-processados, agrupados em segmentos de movimento e classificados por um algoritmo de aprendizado de máquina em duas etapas. A primeira utilizando *Support Vector Machine* para detectar os segmentos motorizados de não-motorizados e a segunda utilizando *Decision Table* para classificar os segmentos motorizados em ônibus, carro e moto, e *Decision Table* e *Bayesian Networks* para classificar os segmentos não motorizados em a pé ou de bicicleta. Além disso, é utilizada a técnica de *Multilayer Perceptron* para classificação nas cinco classes abordadas com o objetivo de comparar o desempenho do algoritmo em duas etapas com uma técnica de redes neurais. Um protótipo da aplicação foi implementado na plataforma Android utilizando a *FusedLocationAPI* para coleta dos *traces* de localização e a API do WEKA em Java para execução dos algoritmos de aprendizado de máquina. O desempenho do protótipo foi avaliado através de testes de campo com 18 voluntários que coletaram mais de 120.000 localizações durante os seus deslocamentos pela cidade do Rio de Janeiro. Durante esse teste de campo foram realizadas 1338 inferências de modos de transporte, através de quatro técnicas de aprendizado de máquina, e os resultados de cada técnica foram avaliados e comparados através dos indicadores da matriz de confusão obtida. As contribuições deste trabalho são:

- Proposta de uma aplicação para detecção do modo de transporte em tempo real e a implementação de um protótipo da aplicação proposta na plataforma Android.
- Avaliação da aplicação em um ambiente real através da realização de testes de campo com 18 usuários na cidade do Rio de Janeiro.

O restante deste artigo está estruturado da seguinte forma. A Seção 2 resume e compara os principais trabalhos relacionados. A Seção 3 descreve o funcionamento da aplicação CityTracks-RT. A Seção 4 descreve a implementação do protótipo da aplicação na plataforma Android. A Seção 5 descreve a avaliação do protótipo em um teste de campo. A Seção 6 apresenta a conclusão e trabalhos futuros.

2. TRABALHOS RELACIONADOS

Nesta seção, descreveremos algumas das soluções para detecção do modo de transporte já existentes, seus resultados e limitações.

Em [15] foi utilizada a técnica de redes neurais para inferir o modo de transporte em cada segmento da viagem. Como resultado, foi detectado um nível de precisão na inferência de mais de 86%. Além disso, a precisão da inferência para o deslocamento de ônibus foi maior do que a precisão em qualquer outro estudo. As limitações deste estudo foram o fato de que os autores não consideraram uma deficiência do algoritmo de rede neural tradicional, chamada de *local optimum*, e o fato de que a comparação de resultados com outros estudos tem pouco valor, uma vez que diferentes estudos utilizam dados de qualidade distinta, o que gera um grande impacto nos resultados. Em estudos realizados com dados de GPS de alta precisão os resultados tendem a ser melhores que em estudos onde os dados coletados não contém alta precisão no posicionamento [13].

Já em [14] foram utilizadas diferentes técnicas de aprendizado de máquina para inferência do modo de transporte e foi detectado que a acurácia das redes neurais era superior às demais técnicas utilizadas. Além disso, eles identificaram cinco atributos relevantes para a inferência do modo de transporte: velocidade média, velocidade mediana, aceleração absoluta média, distância percorrida, e 95% da velocidade. No entanto, para distinguir segmentos de deslocamento de ônibus e carro foi necessário adicionar um atributo adicional, chamado de taxa de pontos de baixa velocidade, e foi utilizado o teste Kolmogorov-Smirnov de duas amostras para obter o valor desta taxa para estas duas classes de modo de transporte. Os resultados mostraram uma acurácia da inferência de 95.81% no subconjunto de treinamento e de 94,44% no subconjunto de teste. Apesar dos resultados promissores, a técnica proposta não foi implementada em uma aplicação para detecção de modo transporte em tempo real, o que dificulta a avaliação do desempenho do algoritmo proposto para este tipo de aplicação.

Em [9] foi utilizada a técnica de *random forest* para classificar os modos de transporte. Neste trabalho, foram identificados alguns atributos valiosos para a inferência, como: velocidade, precisão do GPS, variação de direção, variação de velocidade, aceleração e variação de aceleração. Os autores desenvolveram um modelo de *random forest* e utilizaram 30% dos traces de GPS como amostras de teste e 70% como amostras de treinamento. Os resultados mostraram uma precisão na detecção do modo de transporte de 96,91%. Apesar da alta precisão detectada, algumas limitações deste

trabalho foram o fato de não terem sido amplamente aplicadas técnicas de reconhecimento de erros de dados e a falta de uma base científica para a seleção do modelo de *random forest* utilizado [13].

Em [5] foi implementada uma aplicação, que teve como principal objetivo realizar a coleta do modo de transporte utilizado pelo usuário através de sensoriamento participativo e oportunista, sendo o modo de transporte utilizado inserido manualmente pelo usuário e a informações de movimento coletadas automaticamente através do *framework Core Location* do *IOS* que utiliza dados de sensor GPS, WiFi e redes celulares. Os dados coletados foram utilizados para o desenvolvimento de um algoritmo de mineração de dados completo que realiza o pré-processamento, sumarização de atributos e aplicação de técnicas de aprendizado de máquina para realizar a inferência do modo de transporte. Este algoritmo de mineração foi aplicado após a realização da coleta (detecção *offline*), através da utilização dos software R e *Weka Explorer*.

Com uma metodologia similar, [2] coletou dados de GPS, acelerômetro e giroscópio, e o modo de transporte utilizado pelos usuários para elaboração de uma técnica de detecção de modo de transporte em tempo real baseada em *Cascading* [1] de classificadores. A técnica proposta foi implementada em uma aplicação na plataforma Android, no entanto, uma das limitações deste trabalho foi a ausência de uma avaliação do desempenho da detecção do modo de transporte em tempo real através da aplicação implementada, uma vez que a etapa de análise dos algoritmos de classificação foi realizada antes do desenvolvimento do protótipo.

Já [11], coletou dados de Acelerômetro, Sensor de Gravidade, Giroscópio, Magnetômetro e Barômetro para elaboração de uma técnica de detecção do modo de transporte em tempo real com o menor consumo de energia possível. A não utilização do GPS, permitiu uma grande redução no consumo de energia e a técnica proposta apresentou um bom desempenho. No entanto, alguns dos sensores utilizados, como Barômetro e Sensor de Gravidade, não estão disponíveis em grande parte dos *smartphones* comercializados, o que restringe a utilização desta técnica em um cenário real. Além disso, o desempenho da técnica foi avaliado de forma *offline*, não tendo sido implementado um protótipo para avaliação do algoritmo proposto em um cenário real.

A detecção do modo de transporte através de *smartphones* permite a utilização de uma grande variedade de sensores, no entanto, a maior parte das soluções do estado da arte encontradas utilizou somente dados de GPS. Algumas soluções utilizaram leituras do acelerômetro e dados de redes celulares com o objetivo de compensar as perdas de sinal do GPS e outras utilizaram dados do giroscópio, magnetômetro e outros sensores menos convencionais, como barômetro e sensor de gravidade, para reduzir o consumo de energia. Na solução proposta neste trabalho foi utilizada uma combinação de dados de GPS, WiFi e redes celulares, pois estes sensores estão disponíveis na maior parte dos *smartphones* fabricados e permitem a obtenção de posições GPS com boa precisão, mesmo em ambientes onde há perda de sinal GPS. Ao melhor do nosso conhecimento, nenhum dos trabalhos relacionados apresentou a avaliação de uma aplicação para detecção do modo de transporte *online* em um cenário real, o que evidencia a originalidade do presente trabalho.

3. APLICAÇÃO CITYTRACKS-RT

A aplicação CityTracks-RT utiliza o algoritmo proposto em [5] para realizar a detecção do modo de transporte em tempo real, uma vez que este recurso não foi implementado em [5].

A aplicação CityTracks-RT utiliza dados de GPS, WiFi e redes celulares para coletar as posições dos usuários de smartphones. Durante 90 [5] segundos são capturados traces de localização do usuário contendo altitude, latitude, longitude, precisão do GPS e o *timestamp* da medição. Estes dados são utilizados para calcular a velocidade instantânea a partir de 2 segundos, e aceleração a partir de 3 segundos. Através da velocidade instantânea são identificadas as paradas no movimento. Traces com velocidade inferior a 0,4 m/s são considerados paradas. Além disso, traces com precisão de GPS superior a 200 metros ou diferença de *timestamp* da medição inferior a um segundo são descartados [5]. Traces com diferença de tempo superior a um segundo são inseridos através de uma técnica de interpolação na qual são gerados pontos intermediários com base nas características do trace anterior, mantendo velocidade constante e aceleração zero [5]. Ao fim dos 90 segundos de coleta são extraídos os atributos de sumarização do conjunto de traces, denominado *chunk*. Os atributos de sumarização extraídos são velocidade máxima, aceleração máxima e número de mudanças de direção. A partir destes atributos os chunks são classificados em deslocamento motorizado e não motorizado por SVM (*Support Vector Machine*) e em deslocamento a pé, de bicicleta, ônibus, carro e moto por *Multilayer Perceptron*. Em seguida, os *chunks* classificados como não motorizados por SVM são classificados em deslocamento a pé ou de bicicleta por *Bayesian Net* e *Decision Table*. Já os *chunks* classificados como motorizados por SVM são classificados em deslocamento de ônibus, carro ou moto por *Decision Table*. Após a coleta e classificação dos *chunks* em tempo real, os *chunks* serão persistidos juntamente com as classificações para permitir uma análise posterior e comparação dos resultados de cada técnica de inferência. O funcionamento da aplicação é representado pelo diagrama de atividade ilustrado na Figura 1.

4. IMPLEMENTAÇÃO DE PROTÓTIPO NA PLATAFORMA ANDROID

Nesta seção, será apresentada a implementação de um protótipo da aplicação CityTracks-RT na plataforma Android, utilizando a FusedLocationAPI[1] para coleta das posições por GPS, WiFi e por redes celulares, e a API do Weka em Java[2] para utilização dos algoritmos de aprendizado de máquina.

Foi criado um conjunto de classes em Java baseadas no paradigma de orientação a objetos para realização da etapa de pré-processamento e sumarização dos *chunks*, além de uma classe que realiza o treinamento dos algoritmos de aprendizado de máquina e classificação dos *chunks* através destes algoritmos.

Para o algoritmo de classificação Multilayer Perceptron, foram utilizados os parâmetros *default* da API do Weka, com excessão de 4 parâmetros que deveriam ser obrigatoriamente configurados: *learning rate*, que foi configurado em 0,1; *momentum* que foi configurado em 0,2; tempo de treinamento, que foi configurado em 2 segundos e número de *hidden layers*, que foi configurado em 3. Estes valores fo-

[1]Disponível em https://developers.google.com/
[2]Disponível em https://weka.wikispaces.com/

ram selecionados com base na referência da API do Weka em Java supracitada. Para o algoritmo SVM foi utilizada a classe SMO da API do Weka, configurada com os parâmetros *default*: constante de complexidade = 1, normalização de atributos = *true*, tolerância = 1.0e-3, *epsilon* = 1.0e-12, *seed* de número aleatório = 1, precisão de casas decimais = 2. Para o algoritmo *Decision Table*, também foram utilizados os parâmetros *default* da API do Weka: método de busca = *Best First*, *number of folds* = 1, medida de avaliação de desempenho = *accuracy* para classe discreta e *rmse* para classe numérica. Por fim, para o algoritmo *Bayesian Net* foram utilizados os parâmetros *default* da API do Weka: *number of folds* = 10, *seed* de número aleatório = 1 e algoritmo de busca = K2.

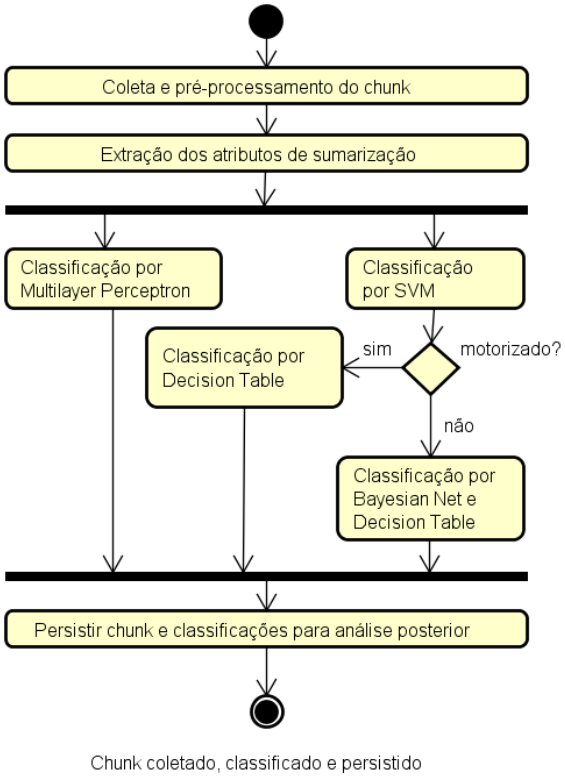

Figura 1: Funcionamento da aplicação CityTracks-RT.

Uma aplicação em Android, consiste basicamente de uma Activity. Uma Activity tem seu layout especificado através de um arquivo xml. Na implementação do CityTracks-RT, foi implementada uma Activity denominada MainActivity que será responsável por controlar toda a interação do usuário com o aplicativo. Foi definido um layout simples através do arquivo *main_activity.xml*. A interface projetada permite que o usuário selecione o modo de transporte que está sendo utilizado para que este seja registrado juntamente com os modos detectados pelas técnicas de aprendizado de máquina permitindo assim, uma avaliação das inferências realizadas. Além disso, a interface projetada permite que o usuário visualize o *timestamp*, latitude e longitude da ultima posição capturada permitindo ao usuário identificar se as posições estão sendo capturadas corretamente.

O controle da FusedLocationAPI em conjunto com as demais classes da aplicação foi feito através da classe CityTracksRTService que implementa um Service da plataforma Android. Uma vez que um Service é acionado por uma Activity ele é executado em *backgroud* de forma assíncrona a Activity que o acionou, mesmo que a aplicação CityTracks-RT seja minimizada pelo usuário. Todo o pré-processamento dos *chunks*, sumarização dos atributos e classificação do modo de transporte ocorrem dentro desta classe, o que garante que a coleta e detecção do modo de transporte não sejam interrompidas caso o usuário precise utilizar outra aplicação.

A persistência dos *chunks* foi feita no próprio *smartphone* em três formatos: ARFF (*Attribute-Relation File Format*), CSV (*Comma Separated Value*) e JSON (*JavaScript Object Notation*). A persistência em ARFF foi adotada por sua compatibilidade com a API do Weka [7]. Já o formato CSV, foi adotado por permitir fácil visualização e processamento dos dados através de scripts em programação e o formato JSON foi utilizado por ser altamente recomendável para persistência e transmissão de dados entre aplicações web e móveis [4]. Nos formatos ARFF e CSV foram salvos somente os dados de sumarização de cada *chunk*. Já no formato JSON foram salvas todas as informações do *chunks*, contendo todas as localizações que foram coletadas e atributos de sumarização extraídos, o que permitiu a fácil recuperação dos dados de coleta e inferência através da API Gson[3].

O código-fonte e o APK do protótipo implementado estão disponíveis em https://bitbucket.org/eltonfss/citytracks-rt.

5. AVALIAÇÃO DO PROTÓTIPO

Nesta seção, será apresentada uma avaliação do protótipo da aplicação CityTracks-RT na plataforma Android por meio de testes de campo realizados por voluntários que se locomoveram na região metropolitana do Rio de Janeiro. Assim, serão apresentadas a metodologia utilizada nos testes de campo e a análise dos resultados obtidos. A análise dos resultados se dará em duas etapas. Na primeira, analisaremos os dados coletados pelos usuários, identificando os principais modos de transporte utilizados, quantidade de pontos coletados por faixa de precisão GPS, total de pontos capturados e total de pontos gerados por interpolação para cada modo de transporte. Na segunda etapa, analisaremos o desempenho dos algoritmos de classificação, discutindo os fatores relevantes para a compreensão dos resultados obtidos.

5.1 Métodos utilizados

Para a realização dos testes de campo foram utilizados *smartphones* de 18 alunos do curso de sistemas de informação da UNIRIO, de diferentes fabricantes, rodando diferentes versões do sistema operacional Android. Foi disponibilizado o APK da aplicação CityTracks-RT com 86 chunks de treinamento coletados através de um dispositivo Samsung Galaxy S3 Mini rodando a versão 4.3 do sistema Android, sendo 20 destes chunks coletados através de deslocamento a pé, 45 de ônibus e 21 de bicicleta. Foi disponibilizado também um manual de instalação e coleta com ilustrações para guiar os voluntários na utilização da ferramenta. O compartilhamento dos traces coletados pelos alunos foi realizado via email. Os traces foram coletados majoritariamente na cidade do Rio de Janeiro, nas regiões do Centro, Aterro do Flamengo, Botafogo e Urca. Foram coletados também, traces na região de Duque de Caxias, Nilópolis e Nova Iguaçu,

[3]Disponível em https://github.com/google/gson

havendo coleta de traces em todo o trajeto de deslocamento entre estas cidades e o Centro do Rio de Janeiro, passando pela Av. Brasil. Nas Figuras 2 e 3 é possível visualizar os traces de mobilidade coletados no mapa da cidade.

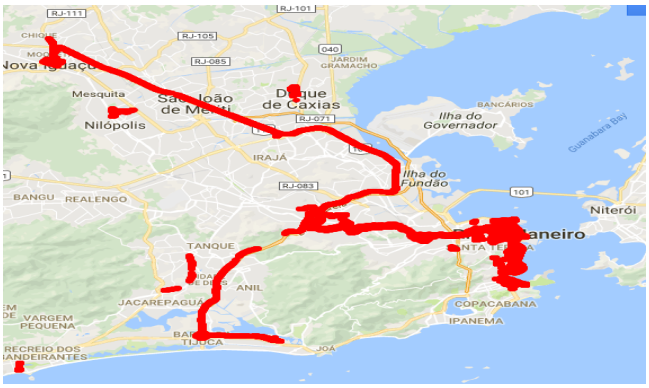

Figura 2: Gráfico de mobilidade com todas as localizações coletadas na região metropolitana do Rio de Janeiro.

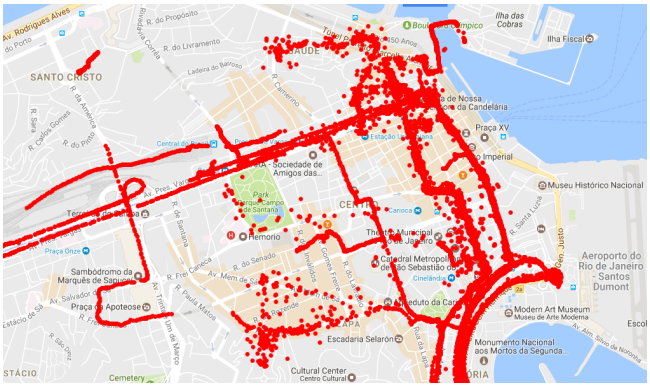

Figura 3: Gráfico de mobilidade com todos as localizações coletadas na região do Centro.

Para análise dos dados, foi desenvolvido um script de processamento dos traces em Java que se baseou nos arquivos JSON gerados pela aplicação, pois estes permitiram recuperar todos os dados dos chunks coletados. Cada linha do arquivo JSON gerado pelo aplicativo armazena um *chunk* completo, com todas as localizações coletadas, atributos de sumarização e modo de transporte coletado, e inferido por cada técnica de aprendizado de máquina. O script de processamento analisa se o modo de transporte coletado, informado pelo usuário, é igual ao modo de transporte inferido por cada uma das técnicas gerando os dados necessários para a elaboração da matriz de confusão de cada algoritmo e, a partir dela, são calculadas as métricas *accuracy*, *precision*, *recall* e *F-Measure*, utilizadas na avaliação e comparação dos algoritmos.

5.2 Análise dos dados coletados

Durante os testes de campo realizados pelos voluntários foi possível capturar traces de localização com precisão média de 25,1 metros com frequência entre um e dois segundos. Foi coletado um total de 1338 *chunks* e 120176 localizações, sendo 53639 localizações capturadas pelos sensores, aproximadamente 45%, e 66537 localizações geradas sinteticamente através da técnica de interpolação utilizada, aproximadamente 55%. Esta interpolação foi necessária quando a localização não foi coletada corretamente, conforme explicado na Seção 3.

Na Tabela 1 é possível visualizar o total de *chunks* e localizações GPS coletadas por dispositivo (colunas Chunks Colet. e Localiz. GPS) assim como a precisão média de posição GPS das localizações coletadas por cada dispositivo (coluna Precis. Média) e os respectivos desvios padrão (coluna Desvio Padrão), além do percentual de localizações capturadas e geradas por interpolação (colunas % Gerado e % Capt.). Conforme podemos observar nesta tabela, ocorreu uma grande variação no número de *chunks* coletados por cada dispositivo. Isso se deveu ao fato de haver um engajamento menor por parte de alguns voluntários e isso pode ter influenciado de alguma forma os resultados obtidos. No entanto, mantivemos todos os *chunks* coletados para garantir uma maior variedade de fontes de coleta. Outro fator relevante para os resultados deste estudo foi a precisão das localizações obtidas. Devido à utilização de diversos modelos de *smartphones*, alguns deles coletaram localizações com uma precisão média superior a 40 metros. Apesar disso, todas as localizações coletadas estavam abaixo do limiar de precisão de 200 metros estabelecido com base em [5]. Durante a análise dos dados, posterior a coleta, observamos que se reduzíssemos o limiar de posição para valores inferiores, à 200 metros, diminuiríamos consideravelmente a quantidade de localizações obtidas, o que por sua vez geraria um grande espaçamento entre as localizações capturadas dificultando, ainda mais, a detecção do modo de transporte utilizado.

ID	Chunks Colet.	Localiz. GPS	Precis. Média	Desvio Padrão	% Gerado	% Capt.
1	219	19710	32,3 m	25,0 m	60%	40%
2	52	4680	26,5 m	15,9 m	77%	23%
3	7	630	31,0 m	4,2 m	39%	61%
4	68	6120	9,4 m	6,0 m	36%	64%
5	87	7830	25,7 m	40,0 m	33%	67%
6	40	3600	37,3 m	30,2 m	60%	33%
7	123	11070	20,4 m	23,5 m	36%	64%
8	34	3060	34,3 m	43,7 m	38%	62%
9	222	19980	13,1 m	7,1 m	38%	62%
10	16	1440	48,3 m	47,4 m	67%	33%
11	83	7470	42,7 m	38,7 m	84%	16%
12	22	1980	7,4 m	7,7 m	2%	98%
13	13	1170	10,1 m	7,0 m	32%	68%
14	51	4590	47,6 m	43,1 m	82%	18%
15	11	990	4,1 m	4,4 m	36%	64%
16	7	630	7,0 m	3,7 m	35%	65%
17	30	2700	20,1 m	35,6 m	41%	59%
18	253	22770	34,1 m	24,4 m	76%	24%
Média	74,3	6690	25,1 m	22,6 m	48,4 %	51,1%

Tabela 1: Dados de coleta com a aplicação CityTracks-RT por dispositivo.

Entre os modos de transporte coletados, aproximadamente 51% das localizações foram coletadas em deslocamento por ônibus e 41% a pé. Apenas 7% das localizações foram coletadas em deslocamento de carro e 1% andando de bicicleta. Nenhuma localização foi coletada por deslocamento de moto. Os totais de localizações coletadas para cada modo de transporte e os percentuais de localizações capturadas, pela FusedLocationAPI e geradas por interpolação para cada um deles, são representados no gráfico da Figura 4.

Outro aspecto relevante é a precisão das localizações coletadas. Apenas 8% tinha precisão de até 5 metros. Cerca de 19% apresentou precisão entre 5 e 10 metros, e 28% entre 10 e 20 metros. Entre 20 e 40 metros cerca de 25%, e 19% acima de 40 metros. As frequências de localizações capturadas pela FusedLocationAPI e geradas por interpolação para

cada faixa de precisão, além da frequência cumulativa das localizações coletadas por faixa de precisão são representadas no gráfico da Figura 5. Uma observação importante sobre os *traces* coletados é em relação ao nível de precisão da localização coletada, pois essa precisão influencia as aplicações sensíveis ao modo de transporte que poderão surgir.

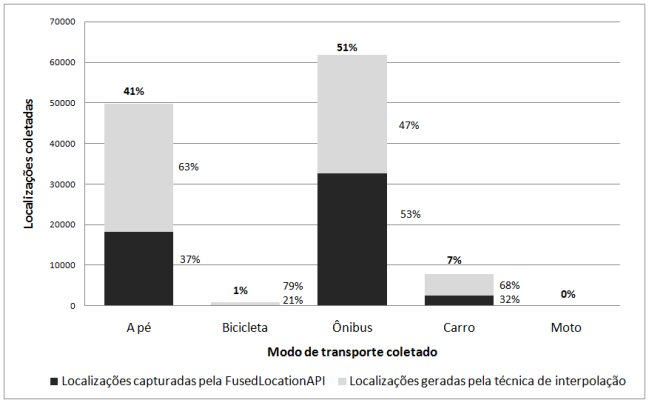

Figura 4: Localizações capturadas x localizações geradas por modo de transporte.

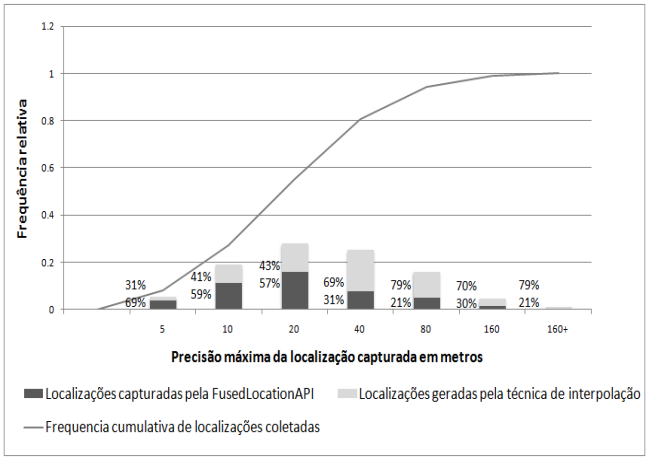

Figura 5: Localizações capturadas x localizações geradas por faixa de precisão em metros e frequência cumulativa das localizações coletadas por faixa de precisão.

5.3 Análise dos algoritmos de classificação

Para avaliação dos algoritmos de classificação utilizados na detecção do modo de transporte pela aplicação proposta, foram analisadas três métricas baseadas na matriz de confusão de cada classificador: *accuracy*, *precision* e *recall* [6]. Estas métricas foram escolhidas por apresentarem um conjunto de informações relevantes para a avaliação dos classificadores e por serem as principais métricas utilizadas para a avaliação de algoritmos de aprendizado de máquina nos trabalhos relacionados. Enquanto que a *accuracy* permite uma visão superficial sobre o percentual de acerto do algoritmo, *precision* e *recall* fornecem informações mais precisas sobre a especificidade e a sensibilidade das inferências realizadas [10]. Uma *precision* alta indica que a maioria dos chunks classificados pelo algoritmo como pertencentes a uma classe de modo de transporte eram de fato pertencentes a ela. Uma *recall* alta indica que a maioria dos chunks que eram pertencentes a uma classe de modo de transporte foram classificados corretamente pelo algoritmo como pertencentes a ela. A *accuracy* (A) de cada classe é calculada somando-se o número de *chunks* classificados corretamente como pertencentes (*True positives*) e não pertencentes (*True negatives*) àquela classe, e dividindo esta soma pelo total de inferências realizadas (*Total population*) [6]. O cálculo desta métrica é definido pela equação:

$$A = \frac{True\ positives + True\ negatives}{Total\ population} \quad (1)$$

A métrica *precision*(P) é obtida através da razão entre o número de *chunks* classificados corretamente como pertencentes (*True positives*) e o total de inferências realizadas (*True positives + False positives*) para àquela classe [6]. Assim, P é definida pela equação:

$$P = \frac{True\ positives}{True\ positives + False\ positives} \quad (2)$$

Já a métrica *recall*(R) é obtida pela razão entre o número de *chunks* classificados corretamente como pertencentes (*True positives*) e o total de *chunks* capturados pertencentes àquela classe (*True positives + False negatives*) [6]. R é definida pela equação:

$$R = \frac{True\ positives}{True\ positives + False\ negatives} \quad (3)$$

Além destas métricas calculamos também a *F-Measure*(F_1), também conhecida como *F-score* ou F_1*-score*, que é calculada com base nos valores de *precision* e *recall* conforme a equação [6]:

$$F_1 = 2 \times \frac{P \times R}{P + R} \quad (4)$$

Após calcular os valores de *accuracy*, *precision* e *recall* para cada uma das classes de modo de tranporte consideradas é calculada a média de cada uma destas métricas para cada classificador utilizado.

5.3.1 Classificação por Multilayer Perceptron

A *accuracy* média observada pela inferência por *Multilayer Perceptron* foi de aproximadamente 69,8%, enquanto que o *precision* média foi de 68,9% e o *recall* média foi de 46,4%. A F-Measure foi calculada em 55,4%. Devido à ausência de dados de treinamento para as classes de moto e carro, as mesmas foram desconsideradas no cálculo destas métricas. A Tabela 2 representa a matriz de confusão das classes de modo de transporte consideradas para este classificador e a Tabela 3 exibe as métricas calculadas para cada classe de modo de transporte.

	Coletado a pé	Coletado bicicleta	Coletado ônibus
Classificado a pé	256	9	73
Classificado bicicleta	73	0	120
Classificado ônibus	223	1	496

Tabela 2: Matriz de confusão das classes de modo de transporte a pé, bicicleta e ônibus para o classificador *Multilayer Perceptron*

Podemos observar que mesmo com zero inferências corretas realizadas para deslocamento de bicicleta, este modo de transporte obteve o maior valor para a métrica *accuracy*, o

	A pé	Bicicleta	Ônibus
Accuracy	0,698	0,838	0,667
Precision	0,757	0	0,689
Recall	0,464	0	0,720
F-Measure	0,575	-	0,704

Tabela 3: *Accuracy*, *precision*, *recall* e *F-Measure* para as classes de modo de transporte a pé, bicicleta e ônibus para o classificador *Multilayer Perceptron*

que demonstra a importância de calcularmos os valores de *precision* e *recall* para validar os resultados desta métrica. O maior valor para a métrica *precision* foi obtido para o deslocamento a pé e para a métrica *recall* o maior valor obtido foi para o deslocamento de Ônibus. O modo de transporte que apresentou o maior valor para *F-Measure* foi ônibus, o que juntamente com a *accuracy* demonstra uma performance melhor para este modo de transporte através do classificador *Multilayer Perceptron*.

5.3.2 Classificação por Support Vector Machine

Para a inferência por *Support Vector Machine* a *accuracy* média observada foi de aproximadamente 65,5%. *Precision* e *recall* gerais foram de aproximadamente 64,7% e 62,4%, respectivamente. A *F-Measure* foi calculada em 63,5%. Conforme observamos na Tabela 5, não houve grande diferença entre os valores de *precision* para inferência de chunks motorizados e não motorizados, mas o *recall* registrado para chunks motorizados foi quase o dobro dos não motorizados, o que indica uma sensibilidade maior dos resultados das inferências de chunks motorizados. Na Tabela 4 é possível visualizar a matriz de confusão para este algoritmo.

	Coletado motorizado	Coletado não motorizado
Classificado motorizado	632	318
Classificado não motorizado	144	244

Tabela 4: Matriz de confusão das classes modo de transporte motorizado e não motorizado para o classificador *Support Vector Machine*

	Motorizado	Não motorizado
Accuracy	0,655	0,655
Precision	0,665	0,629
Recall	0,814	0,434
F-Measure	0,732	0,514

Tabela 5: *Accuracy*, *precision*, *recall* e *F-Measure* para as classes modo de transporte motorizado e não motorizado para o classificador *Support Vector Machine*.

Com base na F-Measure, podemos concluir que a classificação para o modo de transporte motorizado obteve um desempenho superior ao não motorizado. Este resultado foi altamente influenciado pela diferença entre os valores da métrica *recall* entre estas classes.

5.3.3 Classificação por Decision Table para chunks motorizados

Dos 1338 chunks, somente 632 foram classificados corretamente como motorizados pelo algoritmo SVM. A inferência do modo de transporte motorizado pelo algoritmo de *Decision Table* obteve uma *accuracy* média de 87,5%, *precision* média de 93,4% e *recall* média de 100%. Devido à ausência de dados de treinamento para as classes de modo de transporte carro e moto as mesmas foram desconsideradas no cálculo das métricas. Foram mantidas somente as informações sobre os *chunks* coletados em deslocamento de carro para permitir o cálculo dos falsos positivos para a inferência do modo de transporte ônibus. Na Tabela 6 é possível visualizar a *confusion matrix* para este algoritmo. Na Tabela 7 são apresentadas as métricas *accuracy*, *precision*, *recall* e *F-Measure* para cada classe de modo de trasporte considerada.

	Coletado ônibus	Coletado carro
Classificado ônibus	590	42

Tabela 6: Matriz de confusão de modo de transporte motorizado para o classificador *Decision Table*.

	Ônibus
Accuracy	0,875
Precision	0,933
Recall	1
F-Measure	0,903

Tabela 7: *Accuracy*, *precision*, *recall* e *F-Measure* para detecção de modo de transporte ônibus com o classificador *Decision Table*.

5.3.4 Classificação por Decision Table e Bayesian Net para chunks não motorizados

Já para os chunks não motorizados, o algoritmo SVM foi capaz de identificar 244 chunks corretamente. Destes chunks, aproximadamente 43,8% foi classificado corretamente por *Bayesian Net* e 42,2% corretamente por *Decision Table*. Para *Bayesian Net* os valores de *precision* e *recall* gerais foram de 46,5% e 22,7%, respectivamente. Na Tabelas 8 e 9 é possível visualizar as matrizes de confusão para as duas técnicas, considerando as classes de modo de transporte a pé e de bicicleta, e os valores de *accuracy*, *precision*, *recall* e *F-Measure* para as duas classes de modo de transporte de cada técnica, respectivamente.

Apesar das classes de modo de transporte a pé e de bicicleta possuírem praticamente a mesma quantidade de *chunks* de treinamento, a baixa quantidade de *chunks* coletados para o deslocamento de bicicleta não nos permite concluir se o desempenho da inferência para o deslocamento a pé e tão superior à inferência de deslocamento de bicicleta como as métricas nos levam a crer, pois houve muito mais oportunidades de detecção de modo de transporte a pé do que de bicicleta.

	Decision Table		*Bayesian Net*	
	Coletado a pé	Coletado bicleta	Coletado a pé	Coletado bicleta
Classificado a pé	103	8	107	8
Classificado bicleta	133	0	129	0

Tabela 8: Matriz de confusão de modos de transporte não motorizados para os classificadores *Decision Table* e *Bayesian Net*.

	Decision Table		*Bayesian Net*	
	A pé	Bicicleta	A pé	Bicicleta
Accuracy	0,422	0,422	0,438	0,438
Precision	0,928	0	0,930	0
Recall	0,436	0	0,453	0
F-Measure	0,594	-	0,610	-

Tabela 9: *Accuracy*, *precision*, *recall* e *F-Measure* para detecção classe de modo de transporte a pé e bicicleta com os classificadores *Decision Table* e *Bayesian Net*

5.4 Discussão sobre os resultados obtidos

Através do experimento realizado, foi possível constatar o potencial da aplicação CityTracks-RT para detecção do modo de transporte utilizado por usuários de dispositivos móveis em tempo real nos centros urbanos, com um total

de 1338 chunks coletados por 18 usuários de *smartphones* Android no Rio de Janeiro.

Foram utilizados diversos modelos de *smartphones Android* de diferentes fabricantes, o que ocasionou uma variação na precisão média dos *chunks* coletados e pode ter influenciado os resultados obtidos. No entanto, todos as localizações coletadas respeitaram o limiar de 200 m/s e a precisão média por dispositivo não passou dos 50 metros. Durante a análise dos dados, posterior a coleta, observamos que ao reduzir o limiar de precisão para valores inferiores a 200 metros diminuiríamos consideravelmente a quantidade de localizações obtidas, o que por sua vez geraria um grande espaçamento entre as localizações capturadas, dificultando ainda mais a detecção do modo de transporte utilizado.

No que diz respeito aos algoritmos de aprendizado de máquina, foi possível avaliar e comparar o desempenho das técnicas de *Multilayer Perceptron*, *Support Vector Machine*, *Decision Table* e *Bayesian Net* para detecção do modo de transporte em tempo real. No entanto, os dados coletados não foram suficientes para subsidiar a classificação em todos os modos de transporte, sendo necessária a realização de novos testes de campo, com dados de treinamento para todas as classes de modo de transporte suportadas pela aplicação e um conjunto de usuários com perfis de mobilidade diferenciados que permitam a avaliação da detecção de modo de transporte de uma forma mais abrangente.

No que diz respeito ao consumo de recursos dos dispositivos durante a inferência, a técnica utilizada demonstrou ser eficiente, permitindo inclusive a realização da coleta de dados em *background*. Apesar do *feedback* positivo dos usuários quanto ao consumo de bateria, uma sugestão de trabalho futuro seria um estudo mais profundo sobre o consumo de recursos dos dispositivos pela aplicação CityTracks-RT e a implementação de técnicas de otimização para tornar a aplicação mais leve e energeticamente eficiente.

6. CONCLUSÃO E TRABALHOS FUTUROS

No presente artigo abordamos um importante problema que é a detecção do modo de transporte em centros urbanos. Esta detecção é necessária para diversas aplicações sensíveis ao contexto em cidades inteligentes como, por exemplo, aplicações para sistemas de transporte inteligentes.

Assim, este artigo propõe uma aplicação que detecta, em tempo real, o modo de transporte de usuários de *smartphone*. Para isso, foi implementado um protótipo na plataforma Android que faz uso de *data mining* com técnicas de aprendizado de máquina supervisionado. Esse protótipo foi avaliado por voluntários em testes de campo e os resultados obtidos na avaliação evidenciam a utilidade da aplicação proposta.

Como trabalhos futuros, consideramos a realização de testes de campo com uma quantidade maior de usuários, uma análise mais profunda das técnicas de mineração de dados utilizadas e a implementação e avaliação da aplicação proposta na plataforma iOS para comparação com os resultados obtidos na plataforma Android.

Além disso, o desenvolvimento de técnicas para redução do consumo de recursos durante o sensoriamento e detecção do modo de transporte, a criação de um algoritmo para transferência oportunista de traces através de comunicação D2D e a proposição de novas aplicações sensíveis ao contexto do modo de transporte são problemas pesquisa que podem ser explorados em trabalhos futuros.

7. REFERÊNCIAS

[1] E. Alpaydin. *Introduction to machine learning*. MIT press, 2014.

[2] L. Bedogni, M. Di Felice, and L. Bononi. Context-aware android applications through transportation mode detection techniques. *Wireless Communications and Mobile Computing*, 16(16):2523–2541, 2016.

[3] F. Calabrese, L. Ferrari, and V. D. Blondel. Urban sensing using mobile phone network data: A survey of research. *ACM Comput. Surv.*, 47(2):25:1–25:20, Nov. 2014.

[4] D. Crockford. The application/json media type for javascript object notation (json). 2006.

[5] C. A. de MS Quintella, L. C. Andrade, and C. A. V. Campos. Detecting the transportation mode for context-aware systems using smartphones. In *Intelligent Transportation Systems (ITSC)*, pages 2261–2266. IEEE, 2016.

[6] T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[7] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[8] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell. A survey of mobile phone sensing. *IEEE Communications Magazine*, 48(9):140–150, Sept 2010.

[9] Z. A. Lari and A. Golroo. Automated transportation mode detection using smart phone applications via machine learning: Case study mega city of tehran. In *Proceedings of the Transportation Research Board 94th Annual Meeting, Washington, DC, USA*, pages 11–15, 2015.

[10] D. M. Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2011.

[11] X. Su, H. Caceres, H. Tong, and Q. He. Online travel mode identification using smartphones with battery saving considerations. *IEEE Transactions on Intelligent Transportation Systems*, 17(10):2921–2934, 2016.

[12] P. J. Troped, M. S. Oliveira, C. E. Matthews, E. K. Cromley, S. J. Melly, and B. A. Craig. Prediction of activity mode with global positioning system and accelerometer data. *Medicine and science in sports and exercise*, 40(5):972–978, 2008.

[13] L. Wu, B. Yang, and P. Jing. Travel mode detection based on gps raw data collected by smartphones: a systematic review of the existing methodologies. *Information*, 7(4):67, 2016.

[14] G. Xiao, Z. Juan, and C. Zhang. Travel mode detection based on gps track data and bayesian networks. *Computers, Environment and Urban Systems*, 54:14–22, 2015.

[15] F. Yang, Z. Yao, and P. J. Jin. Gps and acceleration data in multimode trip data recognition based on wavelet transform modulus maximum algorithm. *Transportation Research Record: Journal of the Transportation Research Board*, (2526):90–98, 2015.