



UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

ESCOLA DE INFORMÁTICA APLICADA

**CLASSIFICAÇÃO E IMPLEMENTAÇÃO DE  
APLICAÇÕES DE SENSORIAMENTO MÓVEL  
EM CIDADES INTELIGENTES**

ELTON FIGUEIREDO DE SOUZA SOARES

**Orientador:** DR. CARLOS ALBERTO VIEIRA CAMPOS

**Coorientador:** ME. CARLOS ALVARO DE MACEDO SOARES  
QUINTELLA

RIO DE JANEIRO, RJ - BRASIL

JANEIRO DE 2017

ELTON FIGUEIREDO DE SOUZA SOARES

# **CLASSIFICAÇÃO E IMPLEMENTAÇÃO DE APLICAÇÕES DE SENSORIAMENTO MÓVEL EM CIDADES INTELIGENTES**

Projeto de Graduação apresentado à Escola de Informática Aplicada da Universidade Federal do Estado do Rio de Janeiro (UNIRIO) para obtenção do título de Bacharel em Sistemas de Informação.

Aprovada por:

---

**DR. CARLOS ALBERTO VIEIRA  
CAMPOS**

---

**ME. CARLOS ALVARO DE  
MACEDO SOARES QUINTELLA**

---

**DR. CARLOS EDUARDO RIBEIRO  
DE MELLO**

**RIO DE JANEIRO, RJ - BRASIL  
JANEIRO DE 2017**

# Agradecimentos

Agradeço a minha família, especialmente aos meus pais por me darem a educação básica necessária para chegar a uma universidade e o apoio necessário para concluir este curso.

Agradeço a UNIRIO, ao corpo docente e servidores pela excelência e qualidade no ensino e serviços prestados.

Agradeço ao meu orientador, Dr. Carlos Alberto Vieira Campos, pela paciência na orientação, dedicação e incentivo prestados durante este trabalho.

Agradeço ao criador da aplicação CityTracks, Me. Carlos Alvaro de Macedo Soares Quintella pelo apoio prestado durante a realização deste trabalho.

Agradeço ao professor membro da banca examinadora, Dr. Carlos Eduardo Ribeiro de Mello, por aceitar o convite e disposição em avaliar e contribuir com o trabalho.

Agradeço aos alunos de graduação que participaram dos testes de campo da aplicação desenvolvida neste trabalho.

Agradeço à minha eterna namorada e futura esposa, Carla Santos Franco, por estar do meu lado durante toda esta jornada.

# Resumo

O sensoriamento urbano através de telefones móveis tem se tornado um dos principais tópicos de pesquisa nas áreas de sensoriamento móvel e computação ubíqua. O termo sensoriamento, refere-se a utilização de componentes eletrônicos para coleta de dados do mundo real. Exemplos de sensores utilizados para sensoriamento urbano são, Sensor GPS, Acelerômetro, Câmera e Microfone. Como motivação do presente trabalho, foi encontrado um conjunto enorme de aplicações móveis para diversas tarefas, tanto de coleta de dados em ambientes urbanos, como de disseminação de informações. Entretanto, não foram encontrados trabalhos que classificassem estas aplicações em relação aos aspectos de sensoriamento, cidades inteligentes e *crowdsourcing*, o que dificultava o aprendizado da área como um todo. Apresentamos uma proposta de classificação na qual foram classificadas 62 aplicações de sensoriamento móvel para cidades inteligentes em sua maioria relacionadas aos componentes de *Smart Living* e *Smart Mobility*, trabalhando na escala de sensoriamento comunitário e utilizando os paradigmas de sensoriamento oportunista ou participativo. Uma outra motivação deste trabalho, foi o estudo sobre a aplicação CityTracks. Esta aplicação realiza a coleta de dados de movimento de pessoas através de *smartphones* para posteriormente inferir o modo de transporte utilizado. Neste trabalho, apresentamos uma implementação desta aplicação na plataforma Android com o recurso de inferência de modo de transporte em tempo real. Além disso, realizamos testes de campo com 18 voluntários do curso de informática da UNIRIO que coletaram mais de 120.000 posições de GPS durante seu deslocamento pela cidade do Rio de Janeiro. Durante a coleta foram realizadas 1338 inferências de modos de transporte através de três técnicas de aprendizado de máquina.

**Palavras-chave:** sensoriamento móvel. cidades inteligentes. inferência de modo de transporte.

# Abstract

Urban sensing through mobile phones is becoming one of the main research topics in mobile sensing and ubiquitous computing. The word sensing, refers to the use of electronic components for real world data collection. Examples of sensors used for urban sensing are, GPS Sensor, Accelerometer, Camera and Microphone. As a motivation for the present work, we've found a giant amount of mobile applications for different tasks. However, we couldn't find works that would classify these applications in the aspects of sensing, smart cities and crowdsourcing. The absence of a classification that would combine these aspects made it difficult to understand the relationship between these areas. In this work, we present a classification in which we classified 62 mobile sensing applications for smart cities. Most of them were related to the Smart Living and Smart Mobility components of smart cities, working at the communitary sensing scale and using the participatory or opportunistic sensing paradigm. Another motivation for this work, was the study of the CityTracks application. This application collects people movement data through smartphones in order to apply machine learning techniques for transport mode inference. In this work, we present an implementation of this application at the Android platform with the addition of real-time transport mode inference. We also present the results of the field test of this new implementation which was realized with 18 volunteers from the informatics course of the UNIRIO university that collected over 120,000 GPS positions during their commute in Rio de Janeiro. During the collection, 1338 transport mode inferences were made through three machine learning techniques.

**Keywords:** mobile sensing. smart cities. travel mode detection.

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>9</b>
	<b>Introdução</b>	<b>9</b>
<b>1.1</b>	<b>Motivação</b>	<b>9</b>
<b>1.2</b>	<b>Objetivos</b>	<b>10</b>
<b>1.3</b>	<b>Organização do Texto</b>	<b>10</b>
<b>2</b>	<b>REFERENCIAL TEÓRICO</b>	<b>11</b>
<b>2.1</b>	<b>Cidades Inteligentes</b>	<b>11</b>
2.1.1	Fundamentação	11
2.1.2	Digital City x Intelligent City	12
2.1.3	Virtual City x Ubiquitous City	13
2.1.4	Smart City	13
2.1.5	Características de uma Cidade Inteligente	14
2.1.6	Sensoriamento em Cidades Inteligentes	15
2.1.7	Estudos em Cidades Inteligentes	16
<b>2.2</b>	<b>Sensoriamento por dispositivos móveis</b>	<b>17</b>
2.2.1	Fundamentação	17
2.2.2	Sensoriamento pessoal	18
2.2.3	Sensoriamento em grupo	19
2.2.4	Sensoriamento comunitário	19
2.2.5	Paradigmas de sensoriamento	19
2.2.6	Estudos em Sensoriamento participativo	20
2.2.7	Estudos em Sensoriamento oportunista	21
2.2.8	Estudos em controle de privacidade e gestão de recursos	22
<b>2.3</b>	<b>Crowdsourcing</b>	<b>23</b>
2.3.1	Fundamentação	23
2.3.2	Estudos em Crowdsourcing	24
<b>3</b>	<b>CLASSIFICAÇÃO DE APLICAÇÕES DE SENSORIAMENTO MÓVEL COM <i>CROWDSOURCING</i> PARA CIDADES INTELIGENTES</b>	<b>27</b>
<b>3.1</b>	<b>Metodologia</b>	<b>27</b>
<b>3.2</b>	<b>Proposta de classificação</b>	<b>27</b>
<b>4</b>	<b>IMPLEMENTAÇÃO DE APLICAÇÃO CITYTRACKS NA PLATAFORMA ANDROID</b>	<b>32</b>

<b>4.1</b>	<b>Inferência do modo de transporte baseada em aprendizado de máquina e dados de GPS</b>	<b>32</b>
4.1.1	Inferência do modo de transporte em cidades inteligentes	32
4.1.2	Técnicas de aprendizado de máquina	32
4.1.2.1	Multilayer Perceptron	33
4.1.2.2	Support Vector Machine (SVM)	33
4.1.2.3	Decision Table	33
4.1.2.4	Bayesian Networks	34
4.1.3	Inferência do modo de transporte com base em dados de GPS	35
<b>4.2</b>	<b>Aplicação CityTracks</b>	<b>36</b>
<b>4.3</b>	<b>Implementação</b>	<b>38</b>
<b>4.4</b>	<b>Estrutura do Código</b>	<b>39</b>
<b>4.5</b>	<b>Interface Gráfica e Uso da Aplicação</b>	<b>40</b>
<b>5</b>	<b>TESTES DE CAMPO DA APLICAÇÃO CITYTRACKS NA PLATAFORMA ANDROID</b>	<b>44</b>
<b>5.1</b>	<b>Metodologia</b>	<b>44</b>
<b>5.2</b>	<b>Análise dos Resultados Obtidos</b>	<b>46</b>
5.2.1	Análise dos dados coletados	46
5.2.2	Análise dos algoritmos de classificação	47
5.2.2.1	Resultados da classificação por <i>Multilayer Perceptron</i>	48
5.2.2.2	Resultados da classificação por <i>Support Vector Machine</i>	49
5.2.2.3	Resultados da classificação por <i>Decision Table</i> para <i>chunks</i> motorizados	50
5.2.2.4	Resultados da classificação por <i>Decision Table</i> e <i>Bayesian Net</i> para <i>chunks</i> não motorizados	50
<b>6</b>	<b>CONCLUSÃO</b>	<b>52</b>
<b>6.1</b>	<b>Considerações sobre a classificação de aplicações de Sensoriamento Móvel para Cidades Inteligentes</b>	<b>52</b>
<b>6.2</b>	<b>Considerações sobre a Implementação da Aplicação CityTracks na plataforma Android</b>	<b>52</b>
<b>6.3</b>	<b>Trabalhos futuros</b>	<b>53</b>
	<b>REFERÊNCIAS</b>	<b>54</b>
	<b>APÊNDICE A – CLASSE DE INFERÊNCIA DE MODO DE TRANSPORTE - CLASSIFICADOR.JAVA</b>	<b>60</b>

# Lista de tabelas

Tabela 1	– Tabela de classificação de aplicações de Sensoriamento por Móvel para Cidades Inteligente com Crowdsourcing - Parte 1. . . . .	30
Tabela 2	– Tabela de classificação de aplicações de Sensoriamento por Móvel para Cidades Inteligente com Crowdsourcing - Parte 2. . . . .	31
Tabela 3	– Tabela de decisão de nível 2 adaptada de (BECKER, 1998) . . . . .	34
Tabela 4	– Tabela de decisão de nível 1 adaptada de (BECKER, 1998) . . . . .	34
Tabela 5	– Tabela de decisão de nível 0 adaptada de (BECKER, 1998) . . . . .	35
Tabela 6	– Dados de coleta com a aplicação CityTracks por dispositivo. . . . .	47
Tabela 7	– <i>Confusion Matrix</i> das classes de modo de transporte A pé, Bicicleta e Ônibus para o classificador <i>Multilayer Perceptron</i> . . . . .	49
Tabela 8	– <i>Accuracy, precision, recall</i> e <i>F-Measure</i> para as classes de modo de transporte A pé, Bicicleta e Ônibus para o classificador <i>Multilayer Perceptron</i> . . . . .	49
Tabela 9	– <i>Confusion Matrix</i> das classes de tipo de modo de transporte Motorizado e Não Motorizado para o classificador <i>Support Vector Machine</i> . . . . .	49
Tabela 10	– <i>Accuracy, precision, recall</i> e <i>F-Measure</i> para as classes de tipo de modo de transporte Motorizado e Não Motorizado para o classificador <i>Support Vector Machine</i> . . . . .	49
Tabela 11	– <i>Confusion Matrix</i> de modo de transporte motorizado para o classificador <i>Decision Table</i> . . . . .	50
Tabela 12	– <i>Accuracy, precision, recall</i> e <i>F-Measure</i> para detecção classe de modo de transporte Ônibus com o classificador <i>Decision Table</i> . . . . .	50
Tabela 13	– <i>Confusion Matrix</i> de modo de transporte não motorizado para o classificador <i>Decision Table</i> . . . . .	51
Tabela 14	– <i>Confusion Matrix</i> de modo de transporte não motorizado para o classificador <i>Bayesian Net</i> . . . . .	51
Tabela 15	– <i>Accuracy, precision, recall</i> e <i>F-Measure</i> para detecção classe de modo de transporte A pé e Bicicleta com o classificador <i>Bayesian Net</i> . . . . .	51
Tabela 16	– <i>Accuracy, precision, recall</i> e <i>F-Measure</i> para detecção classe de modo de transporte A pé e Bicicleta com o classificador <i>Decision Table</i> . . . . .	51



# Lista de ilustrações

Figura 1 – Caraterísticas de cidades inteligentes. Imagem adaptada de (CIDA-DES..., 2017). . . . .	14
Figura 2 – Escalas de Sensoriamento: Pessoal, de Grupo e Comunitário. . . . .	18
Figura 3 – Ilustração de Crowdsourcing obtida em (THINK..., 2017). . . . .	23
Figura 4 – Diagrama de funcionamento do algoritmo de classificação <i>Multilayer Perceptron</i> . . . . .	33
Figura 5 – Ilustração do funcionamento de uma SVM adptada de (FANG et al., 2016). . . . .	34
Figura 6 – Rede bayesiana representando influências causais entre cinco variáveis adaptada de (PEARL, 2011). . . . .	35
Figura 7 – Representação da informação do movimento coletado passo a passo no processo de transformação e sumarização dos dados obtida em (QUINTELLA; CAMPOS; ANDRADE, 2016). . . . .	37
Figura 8 – Diagrama de processo da inferência de modo de transporte pela aplicação CityTracks em Android. . . . .	41
Figura 9 – Diagrama de classes de modelo baseado na notação UML (LARMAN, 2012). . . . .	42
Figura 10 – Passos 1 e 2 - Tela inicial da aplicação e Seleção de modo de transporte.	42
Figura 11 – Passos 3 e 4 - Modo de transporte selecionado e iniciar coleta. . . . .	43
Figura 12 – Passos 5 e 6 - Coletando posições e finalizando coleta. . . . .	43
Figura 13 – Gráfico de mobilidade com todos os pontos coletados que não eram paradas. . . . .	45
Figura 14 – Gráfico de mobilidade com todos os pontos coletados que não eram paradas na região do Centro. . . . .	45
Figura 15 – Gráfico de mobilidade com todos os pontos coletados que não eram paradas na região da Urca, Botafogo e Aterro do Flamengo. . . . .	46
Figura 16 – Pontos capturados x Pontos gerados por modo de transporte. . . . .	47
Figura 17 – Pontos capturados x Pontos gerados por faixa de precisão em metros e frequência cumulativa de pontos coletados por faixa de precisão. . . . .	48

# 1 Introdução

Neste capítulo, serão apresentadas as principais motivações e objetivos deste trabalho, bem como a organização do texto desta monografia.

O sensoriamento urbano através de telefones móveis tem se tornado um dos principais tópicos de pesquisa nas áreas de sensoriamento móvel e computação ubíqua. O termo sensoriamento, refere-se a utilização de componentes eletrônicos para coleta de dados do mundo real. Exemplos de sensores utilizados para sensoriamento urbano são, Sensor GPS, Acelerômetro, Câmera, Microfone, entre outros.

Já em 2011, durante a conferência internacional de computação ubíqua (Ubicomp), o pesquisador Francesco Calabrese da IBM Research mencionou em sua palestra que cerca de cinquenta por cento da população mundial vive em áreas urbanas atualmente e que 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 um grande desafio para a economia, saúde e meio ambiente. Entender melhor como as cidades funcionam permitiria melhorias na prestação de serviços e a criação de cadeias de comunicação com os habitantes de centros urbanos para reduzir, por exemplo, o consumo de energia e impacto ambiental causado pelas cidades (CALABRESE; FERRARI; BLONDEL, 2014).

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 fabricados com um conjunto crescente de sensores programáveis embutidos (LANE et al., 2010).

Tendo em vista este cenário, as aplicações de sensoriamento urbano poderiam utilizar 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 dos centros urbanos e este entendimento permitiria melhorias nos serviços oferecidos através de dados coletados por sensores espalhados por diversos pontos das cidades.

## 1.1 Motivação

Como motivação do presente trabalho, foi encontrado um conjunto enorme de aplicações móveis para diversas tarefas, tanto de coleta de dados em ambientes urbanos, como de disseminação de informações. Entretanto, não foram encontrados trabalhos que classificassem estas aplicações em relação aos aspectos de sensoriamento, cidades inteligentes e *crowdsourcing*, o que dificultava o aprendizado da área como um todo.

Uma outra motivação deste trabalho, depois da classificação das aplicações móveis, foi o estudo sobre a aplicação CityTracks. Esta aplicação, que foi proposta em (QUINTELLA, 2013), realiza a coleta de dados de movimento de pessoas através de *smartphones* para posteriormente inferir o modo de transporte utilizado. Entretanto, a aplicação implementada em (QUINTELLA, 2013) não realizava a inferência em tempo real. Por causa disso, o seu uso aplicado em cidades inteligentes estava muito limitado.

## 1.2 Objetivos

Este trabalho tem como objetivo fornecer uma classificação das aplicações de sensoriamento móvel existentes aplicadas em cidades inteligentes, fazendo uso de técnicas de *crowdsourcing*, além da implementação da aplicação CityTracks para detectar o modo de transporte em tempo real, bem como o teste de campo na cidade do Rio de Janeiro.

## 1.3 Organização do Texto

A estrutura desta monografia está dividida em capítulos e, além da introdução, será desenvolvida da seguinte forma:

Capítulo 2: Apresenta uma fundamentação sobre Cidade Inteligentes, discutindo os principais conceitos, abordagens e estudos relacionados. Além disso, apresenta alguns conceitos básicos em Sensoriamento por Dispositivos Móveis, demonstrando as escalas e paradigmas de sensoriamento e revisando estudos relacionados. Por fim, é apresentado um resumo sobre *Crowdsourcing* com a revisão de alguns estudos relacionados.

Capítulo 3: É apresentada uma proposta de classificação de aplicações de Sensoriamento por Dispositivos Móveis para Cidades Inteligentes que utilizem técnicas de *Crowdsourcing*, onde foram classificadas 62 aplicações de 99 encontradas.

Capítulo 4: É apresentada a implementação da aplicação CityTracks na plataforma Android para inferência de modo de transporte em tempo real através de técnicas de aprendizado de máquina.

Capítulo 5: São apresentados os resultados do teste de campo da aplicação implementada e análises gráficas dos dados de movimento coletados.

Capítulo 6: São apresentadas as conclusões do estudo realizado, bem como possibilidades de extensão deste trabalho.

## 2 Referencial Teórico

Neste capítulo são apresentados o conceito de Cidades Inteligentes, através de uma fundamentação teórica para compreendermos as diferentes formas de interpretação dos termos relacionados com *Smart City*. Dentre esses termos destacam-se *Digital city*, *Intelligent City*, *Virtual City* e *Ubiquitous City*. Dado isso, será apresentada a visão de seis características que definem uma Cidade Inteligente, o sensoriamento em Cidades Inteligentes, e outros estudos relacionados. Além disso, falaremos sobre sensoriamento em dispositivos móveis apresentando uma fundamentação teórica sobre as escalas e paradigmas de sensoriamento, além de estudos relacionados a estes e outros aspectos de sensoriamento para dispositivos móveis. Por fim, apresentaremos uma fundamentação teórica sobre *Crowdsourcing* além de alguns estudos relevantes desta área de pesquisa.

### 2.1 Cidades Inteligentes

Apresentaremos nesta seção o conceito de Cidades Inteligentes, seus termos relacionados, suas características e estudos sobre o sensoriamento em Cidades Inteligentes, além de alguns estudos relacionados a este tópico.

#### 2.1.1 Fundamentação

Nesta seção, são apresentadas a definição de Cidade Inteligente, sua utilização desde seu surgimento até a contemporaneidade e também as suas diferentes formas de interpretação. Dado isso, é descrito como este termo deu significado as novas tecnologias de informação e comunicação (ICT), uma vez que, a presença de ICT apenas não garante que uma cidade seja inteligente. Dentro desta perspectiva, apresentamos teorias pós e contra sobre a utilização do termo para a classificação de cidades consideradas inteligentes, dado que alguns críticos questionam algumas cidades que não possuam atributos necessários para serem consideradas cidades inteligentes, mas se autodeclaram *smart cities*.

O termo cidade inteligente ou *smart city* foi utilizado pela primeira vez nos anos 90. Neste período, o objetivo era dar significado as novas tecnologias de informação e comunicação (ICT) no que dizia respeito às infraestruturas modernas dentro das cidades. Alguns anos depois a ideia de cidades inteligentes foi criticada por ser tecnicamente orientada demais. Segundo as críticas, a cidade inteligente deveria possuir uma forte abordagem orientada à governança que enfatizasse a importância do capital social e dos relacionamentos no desenvolvimento urbano (ALBINO; BERARDI; DANGELICO, 2015).

Apesar das críticas, o rótulo *smart city* se difundiu no começo do século XXI

com um fenômeno de rotulagem urbana. Há cerca de alguns anos atrás, pesquisadores questionaram a existência de cidades verdadeiramente inteligentes, dado que muitas das cidades autodeclaradas *smart cities* não apresentavam os atributos necessários para receber este rótulo (ALBINO; BERARDI; DANGELICO, 2015).

Em 2010 a IBM afirmou, em um documento corporativo, que o termo *smart city* denota uma cidade instrumentada, interconectada e inteligente. "Instrumentada" refere-se à capacidade de capturar e integrar dados vivos do mundo real através do uso de sensores, medidores, aparelhos, dispositivos pessoais, e outros sensores similares. "Interconectada" refere-se à integração destes dados em uma plataforma computacional que permita a comunicação dessa informação entre vários serviços da cidade. "Inteligente" refere-se à inclusão de análíticas, modelagem, otimização, e serviços de visualização complexos para tomar melhores decisões operacionais (HARRISON et al., 2010).

No campo de planejamento urbano, o termo "smartcity" é frequentemente tratado como uma dimensão ideológica segundo a qual ser mais inteligente implica orientações estratégicas, visando o desenvolvimento sustentável, crescimento econômico, melhor qualidade de vida para os cidadãos, e criação de felicidade (ALBINO; BERARDI; DANGELICO, 2015).

Em (CARAGLIU; BO; NIJKAMP, 2011) uma cidade é considerada inteligente, quando investimentos em capital humano e social, assim como infraestruturas de transporte e ICT alimentam o crescimento econômico sustentável e um alto nível de qualidade de vida, com um gerenciamento inteligente dos recursos naturais através de governança participativa.

Ao analisar as diferentes definições de *smartcity*, é possível perceber que o conceito de cidade inteligente não se limita somente a difusão de ICT, mas também aborda as necessidades das pessoas e comunidades. A presença de ICT por si só não garante que uma cidade seja inteligente. É necessário que a difusão de ICT nas cidades melhore a qualidade de vida das pessoas (ALBINO; BERARDI; DANGELICO, 2015).

Alguns trabalhos discutiram a diferença entre o termo *smart city* e outros termos relacionados, como *digital city*, *intelligent city*, *virtual city* e *ubiquitous city* (ALBINO; BERARDI; DANGELICO, 2015).

### 2.1.2 Digital City x Intelligent City

Nesta seção serão apresentados os conceitos de Digital City e Intelligent City e suas respectivas diferenças e similaridades.

O termo *Digital City* refere-se à uma comunidade conectada que combina infraestrutura de comunicações de banda larga para atender às necessidades de governos, cidadão e negócios, proporcionando um ambiente de compartilhamento de informações,

colaboração, interoperabilidade e experiências midiáticas em qualquer lugar da cidade (ALBINO; BERARDI; DANGELICO, 2015).

Já o termo *intelligent city* refere-se à cidade que utiliza a tecnologia da informação de maneira consciente para melhorar a vida e o trabalho, implicando no suporte ao aprendizado, desenvolvimento tecnológico, e inovação (ALBINO; BERARDI; DANGELICO, 2015).

Toda *intelligent city* pode ser considerada uma *digital city* mas nem toda *digital city* pode ser considerada uma *intelligent city*. A principal diferença entre a *intelligent city* e a *smart city* é o fato da *smart city* possuir um foco maior nas pessoas e comunidades, independentemente da utilização de ICT (ALBINO; BERARDI; DANGELICO, 2015).

### 2.1.3 Virtual City x Ubiquitous City

Nesta seção serão apresentados os conceitos de Virtual City e Ubiquitous City e suas respectivas diferenças e similaridades.

Uma *virtual city* é uma cidade cujas entidades físicas e habitantes possuem contrapartes virtuais representadas em um ciberespaço. Já a *ubiquitous city* estende o conceito de *digital city* no aspecto de ampla acessibilidade. A computação ubíqua deve estar presente em elementos urbanos espalhados por toda a cidade, proporcionando um ambiente onde qualquer cidadão pode obter qualquer serviço em qualquer lugar à qualquer hora através de qualquer dispositivo (ALBINO; BERARDI; DANGELICO, 2015).

### 2.1.4 Smart City

Nesta seção, será apresentado o conceito de Smart City bem como as características que o diferenciam dos demais conceitos citados anteriormente.

De maneira geral o termo *smart city* pode englobar características de todos estes termos e incluir aspectos adicionais, como a criação de comunidades inteligentes que solucionam problemas locais de maneira criativa e inovadora (ALBINO; BERARDI; DANGELICO, 2015).

A principal característica que diferencia uma *smart city* dos demais conceitos citados é o investimento nas pessoas e comunidades como forma de tornar a cidade mais inteligente. Devido à isso, espera-se que cidades inteligentes formem e atraiam pessoas inteligentes (ALBINO; BERARDI; DANGELICO, 2015).

Estudos concluíram que o possível motivo de não haver um consenso geral sobre o termo *smart city* seja o fato de ele ter sido utilizado em dois diferentes tipos de domínios. Nos domínios considerados complexos, como planejamento de prédios e redes de energia, gestão de recursos naturais e hídricos, gerenciamento de desperdício, mobilidade e logística, a definição de *smart city* utilizada prevê uma participação decisiva da ICT. Já nos

domínios considerados simples, como educação, cultura, inovações políticas, inclusão social e governo, as definições de *smart city* não valorizam tanto a utilização de ICT, uma vez que estes domínios permitem a implantação de melhorias independentes de ICT (ALBINO; BERARDI; DANGELICO, 2015).

### 2.1.5 Características de uma Cidade Inteligente

Nesta seção, são apresentadas as seis características que deveriam ser analisadas e que definem uma cidade Inteligente. Além disso, é apresentado como cada característica agrupa um conjunto de fatores, que podem ser medidos através de indicadores.

Uma das definições de cidade inteligente mais reconhecidas no meio acadêmico foi concebida em 2007 por pesquisadores da *Vienna University of Technology*, *University of Ljubljana* e *Delft University of Technology* durante a elaboração de um ranking das cidades europeias de médio porte (GIFFINGER; PICHLER-MILANOVIĆ, 2007).

Neste trabalho, foram definidas 6 características que deveriam ser analisadas em cidades inteligentes: *Smart Economy*, *Smart People*, *Smart Governance*, *Smart Mobility*, *Smart Environment*, *Smart Living* (GIFFINGER; PICHLER-MILANOVIĆ, 2007). Cada característica agrupa um conjunto de fatores, que podem ser medidos através de indicadores. No total foram definidos 31 fatores e 74 indicadores.

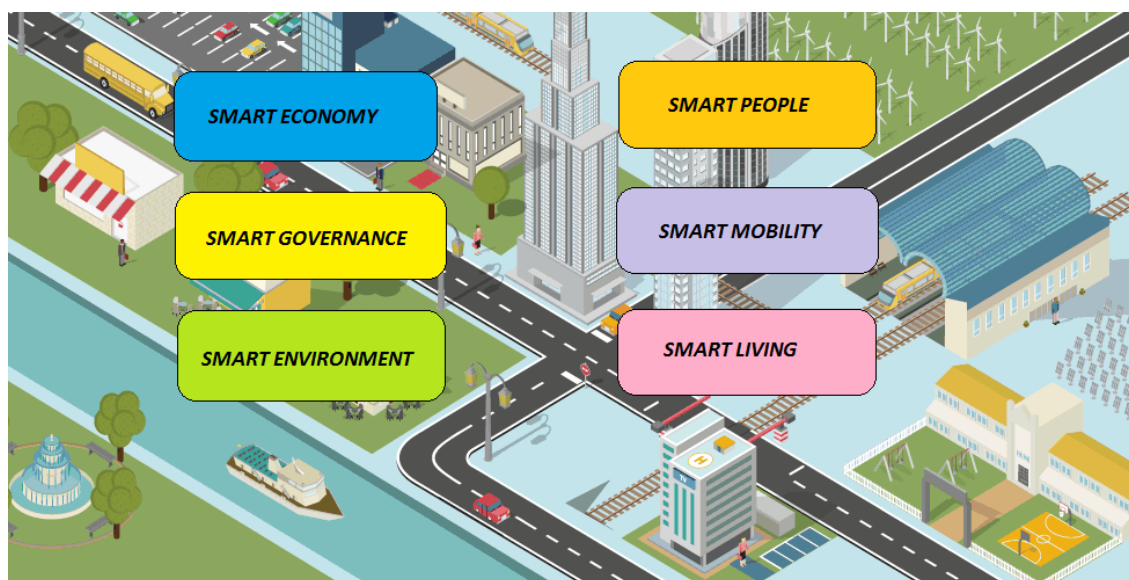


Figura 1 – Características de cidades inteligentes. Imagem adaptada de (CIDADES..., 2017).

Relacionados a característica de *Smart Economy* estão os fatores de espírito inovador, empreendedorismo, imagem econômica e marcas comerciais, produtividade, flexibilidade do mercado de trabalho, inserção internacional e habilidade de transformação.

Já no que diz respeito à característica de *Smart People* foram relacionados o nível de qualificação, afinidade ao aprendizado contínuo, pluralidade ética e social, flexibilidade, criatividade, cosmopolitismo ou espírito de abertura e participação na vida pública.

Em relação à *Smart Governance* os fatores considerados são a participação pública em tomadas de decisão, os serviços públicos e sociais, a governança transparente e as estratégias e perspectivas política.

Na característica de *Smart Mobility* são relevantes a acessibilidade local, a acessibilidade nacional e internacional, a disponibilidade de infraestruturas de ICT e sistemas de transporte seguros, inovadores e sustentáveis.

Já em relação à *Smart Environment* são relacionados a atratividade das condições naturais, o nível de poluição, a proteção ambiental e o gerenciamento sustentável de recursos.

Por fim, os fatores relacionados à característica de *Smart Living* foram as instalações culturais, as condições de saúde, a segurança individual, a qualidade das moradias, as instalações educacionais, a atratividade turística e a coesão social.

Podemos perceber que as cidades inteligentes possuem características nas quais devem apresentar bom desempenho. Desta forma, para que uma cidade seja considerada inteligente é necessário que a mesma apresente bom indicadores em cada uma destas características.

### 2.1.6 Sensoriamento em Cidades Inteligentes

Nesta seção é apresentada uma arquitetura de sensoriamento em Cidades Inteligentes em diferentes áreas e como o auxílio das tecnologias podem contribuir em sensoriamento para Cidades Inteligentes.

Em (HANCKE; JR et al., 2012) é apresentada uma revisão do estado da arte em sensoriamento para cidades inteligentes. Além disso, é proposta uma arquitetura de sensoriamento em cidades inteligentes que divide-se em sete áreas: *Smart Surveillance*, *Smart Electricity and Water distribution*, *Smart Buildings*, *Smart Healthcare*, *Smart Services*, *Smart Transportation* e *Smart Infrastructures*.

Na área de *Surveillance* estão compreendidas tecnologias de monitoramento e segurança, principalmente através de sistemas de câmeras. Aplicações de *Smart Surveillance* podem auxiliar na detecção de ações violentas e também na identificação das pessoas envolvidas. O sensoriamento nesta área permite contribuir no auxílio do monitoramento e segurança com o objetivo de auxiliar em supostas investigações que envolvam ações violentas. Desta forma, é de suma importância o sensoriamento em cidades inteligentes nesta área.



Na área de eletricidades e distribuição de água estão relacionadas as tecnologias para medição e previsão de demanda desses recursos. Aplicações de sensoriamento avançado para *Smart Electricity and Water distribution* permitem uma medição e previsão mais precisa em relação as técnicas tradicionais.

Em *Smart Buildings* desenvolvem-se técnicas de sensoriamento para auxiliar na redução do consumo de recursos. O sensoriamento tem papel fundamental para a esta tarefa pois para que se possa reduzir adequadamente o consumo de recursos é necessário avaliar de forma precisa o consumo atual.

Já em *Smart Healthcare* são consideradas todas as tecnologias que auxiliam o gerenciamento em hospitais ou clínicas. Nesta área, destacam-se as redes de área corporal sem fio que consistem de sensores interconectados localizados em diferentes partes do corpo humano. Estas redes de sensores permitem um monitoramento remoto dos sinais vitais do paciente, assim como o armazenamento e a transmissão das informações em tempo real.

Podemos perceber a importância do sensoriamento na área Smart Healthcare, tecnologias que permitem auxiliar no gerenciamento em hospitais, pois através deste recurso tecnológico é possível realizar o monitoramento remoto com intuito de detectar através de sensores interconectados em diferentes parte do corpo humano, problemas que podem comprometer a saúde de um paciente em tempo real.

No contexto de *Smart Services* são desenvolvidas tecnologias de suporte aos serviços públicos. Dois setores que possuem um grande potencial de melhoria através de aplicações de cidades inteligentes são os de aplicação de lei e combate a incêndios. Percebemos que através do sensoriamento nesta área é possível que seja feita a detecção de incêndios, uma vez que a qualquer sinal emitido para o sensor, seu dispositivo é acionado, sendo assim, permitindo que equipes de resgates possam estar presentes para combater incêndios. Assim, possibilitando melhorias no serviço prestado em cidades inteligentes.

Na área de *Smart Transportation* são abordadas tecnologias de monitoramento e gerenciamento do tráfego de veículos. Aplicações como os sistemas de navegação por GPS permitem que os motoristas escolham rotas mais eficientes para se locomover pela cidade, reduzindo assim o tempo de viagem até o destino e o tráfego nos grandes centros.

Por fim, na área de *Smart Infrastructure* são definidas tecnologias de sensoriamento de infraestruturas públicas, como pontes, estradas e prédios, permitindo uma manutenção e utilização de recursos mais eficiente com base nos dados coletados(PERERA et al., 2014).

### 2.1.7 Estudos em Cidades Inteligentes

Nesta seção, revisamos alguns dos trabalhos de destaque realizados na área de cidades inteligentes nos últimos anos.

Em 2013, (DORAN; GOKHALE; DAGNINO, 2013) propôs uma metodologia para extração de percepções através de sensoriamento humano com o objetivo de fornecer informações suplementares para cidades inteligentes. (VANOLO, 2013) fez uma análise de como a definição de cidade inteligente proposta pela União Europeia foi reclassificada ao longo dos anos e o papel dos atores privados e cidadãos no gerenciamento do desenvolvimento urbano. Também neste ano, (ANASTASI et al., 2013) propôs um sistema de sugestão de rotas automotivas baseado em sensoriamento urbano. Além disso, (GAURA et al., 2013) definiu *edge mining* como uma abordagem fundamental em IoT. A trabalho demonstra o uso desta abordagem para reduzir a transmissão de pacotes, consumo de energia e requisitos de armazenamento remoto através de três algoritmos: L-SIP, ClassAct, e BN e uma versão genérica: G-SIP. Além disso, é feita uma análise específica dos detalhes de implementação e uso das técnicas de *edge mining* propostas em aplicações reais.

Em 2014, (ROLIM et al., ) propôs uma arquitetura ubíqua para sensoriamento urbano que utiliza um modelo conceitual orientado a serviços e (SCHILDT et al., 2014) propôs um sistema de incentivo para redes tolerantes à atraso baseadas em *smartphones* para cidades inteligentes.

Em 2015, (SAGL; RESCH; BLASCHKE, 2015) estudou a integração de informações contextuais com dados de localização humanos e técnicos para cidades inteligentes. Primeiramente, é destacada a importância do contexto espaço temporal para cidades inteligentes. Em seguida, é introduzido um modelo de interações entre pessoas, meio ambiente e tecnologia, em cidades inteligentes. O modelo proposto é o primeiro a considerar o contexto espaço temporal como fator fundamental das interações. Por fim, é feita a análise e agrupamento das tecnologias de sensoriamento atuais em três grupos: sensores *in situ*, sensores remotos e sensores humanos. Estes grupos são relacionados a três dimensões de sensoriamento: geração de dados, fenômenos geográficos e tipo de sensoriamento.

## 2.2 Sensoriamento por dispositivos móveis

Será apresentado nesta seção a fundamentação teórica sobre sensoriamento por dispositivos móveis.

### 2.2.1 Fundamentação

Nesta seção, é apresentado a importância do sensoriamento por dispositivos móveis e os principais conceitos relacionados ao sensoriamento móvel.

O sensoriamento por dispositivos móveis, permite que os dados de sensoriamento sejam coletados através dos sensores dos *smartphones* e demais dispositivos móveis. Cada vez mais estes dispositivos estão se tornando o principal dispositivo de computação na vida das pessoas e isso aumenta ainda mais o seu potencial para aplicações de sensoriamento.

As aplicações de sensoriamento podem ser classificadas em três escalas: sensoriamento pessoal, sensoriamento em grupo e sensoriamento comunitário. Além disso podem ser classificadas em dois paradigmas: sensoriamento participativo e sensoriamento oportunista (LANE et al., 2010).

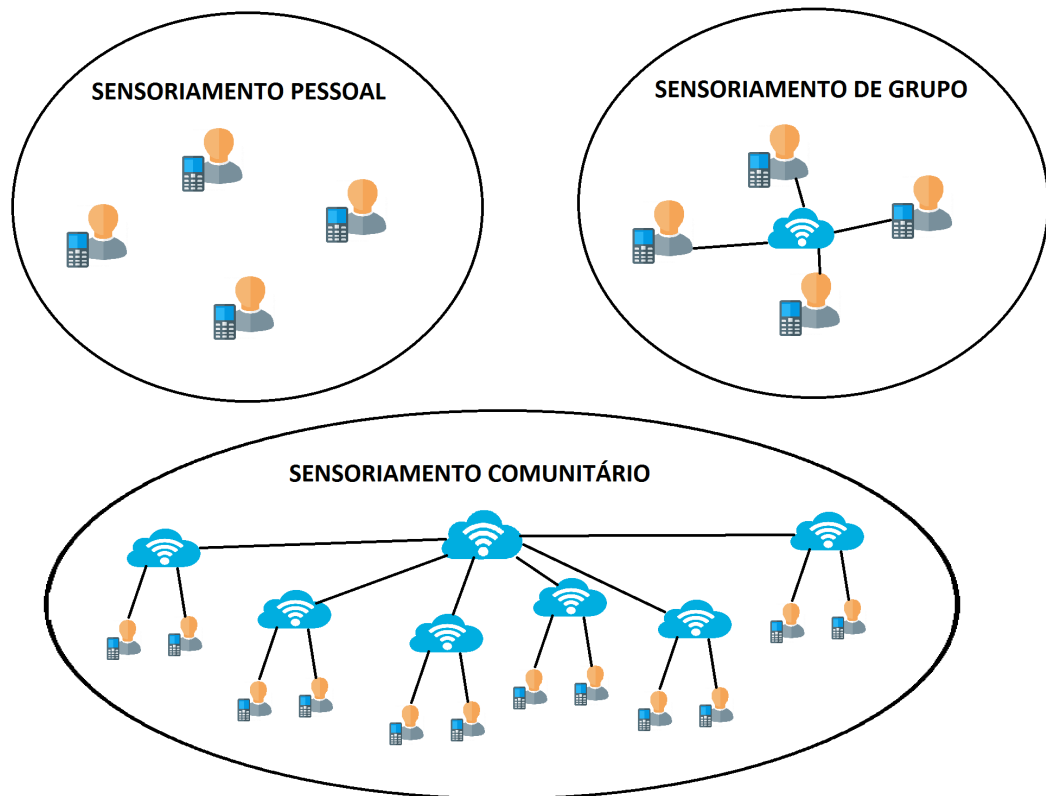


Figura 2 – Escalas de Sensoriamento: Pessoal, de Grupo e Comunitário.

### 2.2.2 Sensoriamento pessoal

Nesta seção, apresentaremos a definição de sensoriamento pessoal e seus benefícios para o usuário.

Aplicações de sensoriamento pessoal são designadas para um único indivíduo e são focadas em coleta de dados e análise. Típicamente, aplicações de sensoriamento pessoal geram dados para benefício único e exclusivo do usuário que os coletou não sendo compartilhados com nenhum outro usuário. Um cenário típico desse tipo de aplicação é o rastreamento das rotinas de exercícios de um indivíduo (LANE et al., 2010).

### 2.2.3 Sensoriamento em grupo

Nesta seção, apresentamos a definição de sensoriamento em grupo e seus benefícios para um grupo e são mostrados também exemplos da utilização desta aplicação.

Já as aplicações de sensoriamento em grupo são projetadas para que um grupo de indivíduos compartilhe dados de sensoriamento livremente ou com proteção de privacidade de modo a alcançar um objetivo comum a todos. Há um elemento de confiança em aplicações de sensoriamento de grupo que simplifica problemas complexos tais como a certeza de que os dados coletados são corretos e a redução do grau de privacidade que deve ser garantida ao agregar os dados individuais de cada usuário. Casos de uso comuns desse tipo de aplicação incluem segurança de vizinhanças, redes sociais orientadas a sensores e formas de ciência cidadã (LANE et al., 2010).

### 2.2.4 Sensoriamento comunitário

Nesta seção, apresentamos aplicações de sensoriamento comunitário e exemplos de cenários onde esse tipo de aplicação é útil.

A última escala de aplicação de sensoriamento são aplicações de sensoriamento comunitário que representam coleta de dados, análise e compartilhamento em larga escala para o bem comum de uma comunidade. Este tipo de aplicação implica na cooperação de pessoas que não necessariamente possuem uma relação de confiança entre si e consequentemente requer um grau de proteção de privacidade mais elevado bem como um baixo nível de comprometimento dos usuários. Exemplos de cenários onde esse tipo de aplicação seria útil são o rastreamento da propagação de uma doença pela cidade, rastreamento dos padrões de migração dos pássaros, rastreamento dos padrões de congestionamento pela cidade e a geração de uma mapa de poluição sonora da cidade (LANE et al., 2010).

### 2.2.5 Paradigmas de sensoriamento

Nesta seção, apresentamos a classificação das aplicações de sensoriamento em dois paradigmas, sensoriamento participativo e sensoriamento oportunista. Discutimos também, as vantagens e desvantagens na utilização cada um dos paradigmas de sensoriamento que aprofundaremos nesta seção.

Quanto ao paradigma de sensoriamento podemos classificar as aplicações em aplicações de sensoriamento participativo ou aplicações de sensoriamento oportunista. No sensoriamento participativo o usuário coleta dados de maneira voluntária enquanto que no sensoriamento oportunista a coleta de dados é feita automaticamente (LANE et al., 2010).

A vantagem de se utilizar o sensoriamento oportunista é o fato de exigir uma colaboração menor do usuário permitindo um alto nível de participação mesmo que a

aplicação não traga uma benefício pessoal ao usuário. Este aspecto é particularmente útil para o sensoriamento comunitário onde o benefício direto ao usuário geralmente só é percebido à longo prazo. No entanto, aplicações de sensoriamento oportunista são geralmente difíceis de construir, e o principal recurso, as pessoas, é subutilizado. Um dos principais desafios de se utilizar sensoriamento oportunista é o problema do contexto de sensoriamento. Por exemplo, uma aplicação de geração de mapa de poluição sonora deseja gravar sons somente quando o telefone estiver fora do(a) bolso(a). Esses tipos de problemas de contexto podem ser resolvidos utilizando outros sensores do telefone como o acelerômetro ou sensores de luz, por exemplo, que podem determinar se o telefone está dentro ou fora do(a) bolso(a) (LANE et al., 2010).

Diferentemente do sensoriamento oportunista, o sensoriamento participativo exige uma colaboração maior do usuário. Em aplicações de sensoriamento participativo o usuário será responsável por, por exemplo, selecionar manualmente os dados a serem coletados(e.g. menores preços de petróleo) e até mesmo coletá-los diretamente(e.g. tirar uma foto). A grande vantagem desse paradigma de sensoriamento é o fato de que os problemas de contexto existentes no sensoriamento oportunista podem ser resolvidos através da interação do usuário. Por exemplo, no cenário da coleta de dados para geração de um mapa de poluição sonora, o problema de contexto pode ser resolvido pela simples atitude do usuário retirar o telefone do(a) bolso(a) e coletar a amostra de som voluntariamente. Em contrapartida, um empecilho ao sensoriamento participativo é o fato de que a qualidade dos dados coletados depende diretamente da vontade do usuário de coletar os dados corretamente e da compatibilidade dos padrões de mobilidade do usuário com os objetivos da aplicação(e.g. coletar dados de poluição próxima à escolas) (LANE et al., 2010).

Claramente, oportunista e participativo representam pontos extremos no design de aplicações de sensoriamento. Cada paradigma possui vantagens e desvantagens. Até o momento há pouca experiência na criação de aplicações em larga escala para que se possa compreender completamente a relação de custo e benefício de cada uma. Existe uma necessidade de se desenvolver modelos para melhor entender os problemas de usabilidade e desempenho de cada um dos paradigmas. Além disso, é provável que muitas aplicações de sensoriamento se utilizarão de ambos os paradigmas em conjunto (LANE et al., 2010).

### 2.2.6 Estudos em Sensoriamento participativo

Nesta seção, é apresentado o conceito de sensoriamento participativo desde seu surgimento e uma arquitetura básica para aplicações desse tipo.

O conceito de sensoriamento participativo foi introduzido em 2006 por um grupo de pesquisadores do Center of Embedded Networked Sensing (CENS) da University of California, Los Angeles (BURKE et al., 2006). Neste trabalho eles propuseram uma arquitetura básica para aplicações desse tipo e um modelo para criação de aplicações para

campanhas de coleta de dados.

Em 2011, (GANTI; YE; LEI, 2011) introduziu o termo *mobile crowdsensing* (MCS) para se referir à um amplo número de paradigmas de sensoriamento comunitário. Além disso, os autores classificam as aplicações MCS em três categorias baseadas no tipo de fenômeno sendo medido ou mapeado. Em aplicações de MCS ambiental, os fenômenos são medidos estão relacionados ao meio ambiente. Aplicações de MCS de infraestrutura envolvem a medição de fenômenos de larga escala relacionados à infraestrutura pública. Já em aplicações de MCS social, indivíduos compartilham informações sensoriadas entre si. Em 2013, (PEREIRA, ) propôs um aplicação para coleta de dados orientada a mobilidade urbana em pequenas áreas, em uma escala espacial de bairro ou distrito.

Em 2015, (HU et al., 2015) desenvolveu um sistema de sensoriamento de estradas que utiliza sensoriamento participativo para detectar e identificar reguladores de trânsito, semáforos e sinais de parada. (MYLONAS; THEODORIDIS; MUÑOZ, 2015) descreveu a arquitetura do *SmartSantander*, um sistema que utiliza *smartphones* disponibilizados por voluntários para realizar testes de aplicações em larga escala. (SWENDEMAN et al., 2015) fez um estudo de seis semanas para avaliar a confiabilidade e viabilidade de uma aplicação de sensoriamento pessoal para pessoas soropositivas. (SILVA et al., ) discutiu os principais elementos das redes de sensoriamento participativo apresentando uma visão geral da área, desafios e oportunidades. Este trabalho mostra que estas redes de sensoriamento podem atuar como fontes de sensoriamento em larga escala, podendo inclusive complementar as redes de sensores sem fio tradicionais.

Em 2016, (SILVA; LOUREIRO, ) discutiu o conceito de computação urbana, dando prosseguimento ao seu trabalho anterior. Neste novo trabalho, destaca-se a relevância da área e motiva-se a construção de novas aplicações para tratar questões relacionadas com a dinâmica da cidade e do comportamento social urbano. Discutiu-se também o sensoriamento urbano com redes de sensoriamento participativo. E demonstrado que as RSPs oferecem oportunidades sem precedentes de acesso a dados de sensoriamento em escala planetária, dados que nos ajudam a entender melhor sociedades urbanas. Estudou-se também as principais técnicas utilizadas em trabalhos da área de computação urbana, bem como algumas das tecnologias e ferramentas comumente utilizadas para a análise de dados urbanos. Além disso, apresentou-se diversos desafios sobre tópicos de pesquisa atuais relacionados com computação urbana utilizando dados de RSPs. Ressaltou-se também várias oportunidades relacionadas ao uso de dados de RSPs em novos serviços e aplicações da área de computação urbana.

### 2.2.7 Estudos em Sensoriamento oportunista

Nesta seção, apresentamos o conceito de sensoriamento oportunista e alguns estudos relacionados a este paradigma de sensoriamento.

Em 2006, (CAMPBELL et al., 2006) propôs uma arquitetura de rede para sensoriamento centrado em pessoas em escala urbana com o objetivo de suportar diversos tipos de aplicações e sensores. Em 2012, (ANTONIOU et al., 2012) criou um sistema para monitoramento de grandes grupos de usuários utilizando uma combinação de infraestruturas de *IoT* estáticas e móveis, visando múltiplos domínios de aplicações. O sistema foi testado em dois cenários, um prédio empresarial e uma grande conferência (*FET'11*). (CHON et al., 2012) desenvolveu um protótipo de aplicação para caracterização automática de lugares através de sensoriamento oportunístico por *smartphones*. A abordagem proposta neste artigo baseia-se em tópicos e modelos para modelar lugares que possam combinar eficazmente uma variedade de classificadores baseados em sons e imagens (e.g., reconhecimento de cena, OCR, reconhecimento de fala, etc.) juntamente com sinais baseados em mobilidade de sensores GPS/WiFi. O projeto e avaliação indicam quais classificadores são eficazes para categorização de lugares com alguns classificadores sendo ajustados para esta aplicação especificamente. O protótipo foi avaliado em um experimento de sete semanas, utilizando *smartphones* comuns de 36 pessoas. A descoberta principal foi de que a aplicação proposta foi capaz de classificar 1,241 lugares encontrados pelos participantes do estudo em um total de sete categorias com um nível de precisão de 69%. Comparando o desempenho do protótipo com uma abordagem que utiliza somente estimativas de localização e bases de dados de localização de larga escala, os resultados mostraram que a aplicação proposta supera esta técnica em 40% na categorização de lugares. Em 2013, (HOSEINI-TABATABAEI; GLUHAK; TAFAZOLLI, 2013) estudou as técnicas de reconhecimento de contexto utilizadas em aplicações de sensoriamento oportunista. Em 2015, (CABERO et al., 2015) analisou a confiabilidade de *traces* de conectividade para caracterização de interações humanas.

### 2.2.8 Estudos em controle de privacidade e gestão de recursos

Nesta seção, apresentamos estudos e propostas de sistemas que tem como objetivo de melhorias de privacidade e gestão de recursos durante o sensoriamento, principalmente para sistemas Android. Em 2011, (PIDCOCK et al., 2011) propôs um sistema de notificação para sensoriamento urbano com melhorias de privacidade e (PRIYANTHA; LYMBEROPOULOS; LIU, 2011) propôs uma arquitetura inovadora de sensoriamento contínuo em dispositivos móveis para reduzir o consumo de energia. Em 2012, (MAN; NGAI; LIU, 2012) descreveu um *framework* unificado para suportar coleta de dados disparada por localização. É adotada uma abordagem que utiliza o GPS de maneira econômica para detectar localizações. Neste mesmo ano, (BAIER; DURR; ROTHERMEL, 2012) propôs técnicas para reduzir o consumo de energia no rastreamento de posição e trajetória através de dispositivos móveis. Já em 2013, (FRANK et al., 2013) apresentou ENRAPT, um algoritmo de sensoriamento de tráfego para smartphones que utiliza detecção automática de contexto baseada no acelerômetro para capturar informações de localização de desloca-

mento. É executado um estudo de validação em um ambiente controlado para testar a precisão do algoritmo de detecção de contexto.

Em 2015, Zhao et al. (ZHAO et al., 2015) implementou um sistema de monitoramento de acesso à recursos sensível à privacidade para sistemas Android e (BHATTACHARYA et al., 2015) propôs o sistema EnTrackedRT, que estende o sistema EnTrackedT para suportar rastreamento de trajetória robusto e permitir que usuários e aplicações especifiquem os níveis de robustez desejados. Propõe situational bounding como um mecanismo genérico para melhorar a robustez de estratégias de gerenciamento de sensores com economia de energia e analisa extensivamente a eficácia das extensões, focando em melhorias em robustez e tradeoffs com economia de energia. Além disso, demonstra através de experimentos extensivos multi-plataforma que as extensões propostas provêm melhorias significativas em robustez de rastreamento com pequeno impacto no consumo de energia.

## 2.3 Crowdsourcing

Nesta seção, será feita um aprofundamento teórico sobre Crowdsourcing, e a importância de sua utilização na aplicação proposta neste trabalho.

### 2.3.1 Fundamentação



Figura 3 – Ilustração de Crowdsourcing obtida em (THINK..., 2017).

Nesta seção, é apresentado o conceito de Crowdsourcing e os benefícios desta técnica em aplicações. Esta técnica permite que qualquer pessoa possa gerar informações úteis. A partir deste estudo percebemos os benefícios da utilização da técnica com base



em Crowdsourcing. Deve-se ressaltar a importância da utilização desta técnica para o desenvolvimento de uma aplicação de sensoriamento por dispositivos móveis para cidades inteligentes que propomos neste trabalho.

O termo *crowdsourcing* foi mencionado pela primeira vez em um artigo da *Wired Magazine* (HOWE, 2006). Neste trabalho, foi definido como o ato de uma empresa ou instituição tornar uma atividade antes realizada por empregados em uma atividade realizada por pessoas independentes de forma livre. Desde então, percebeu-se uma expansão desta prática para diversos aspectos e domínios. Cada vez mais negócios adotam *crowdsourcing* em seus processos de negócio e, em muitos dos casos, como diferencial dos seus concorrentes.

Conforme explicitado em Zhao, 2014 (ZHAO; ZHU, 2014) o sucesso desta técnica deve-se ao fato dela se basear em um conceito básico, mas poderoso, de que virtualmente qualquer pessoa pode gerar informações úteis. Alguns exemplos conhecidos de aplicações de *crowdsourcing* bem sucedidas são *Waze* (WAZE..., 2016), *Foursquare* (FOURSQUARE, 2016) e *Wikipedia* (WIKIPEDIA..., 2017).

### 2.3.2 Estudos em Crowdsourcing

Nesta seção, são descritos estudos realizados em Crowdsourcing. As diferenças existentes entre crowdsourcing e produção opensource. Segundos estudos realizados é possível afirmar que a utilização de crowdsourcing permite aumentar a participação dos cidadãos em projetos de planejamento urbano.

Em 2009, (BRABHAM, 2008) faz uma diferenciação entre *crowdsourcing* e produção *open source* e propõe uma agenda de pesquisa em *crowdsourcing*. Neste mesmo ano, (BRABHAM, 2009) defende a utilização de *crowdsourcing* como forma de aumentar a participação dos cidadãos em projetos de planejamento urbano. Segundo Daren a utilização desta técnica permite aumentar a contribuição dos cidadãos significativamente.

Em 2011, (YUEN; KING; LEUNG, 2011) apresentou uma revisão estruturada das pesquisas existentes em *crowdsourcing* e (KANHERE, 2011) faz uma revisão de trabalhos em sensoriamento participativo através de *crowdsourcing* onde são levantados os principais desafios em aberto nesta área. Neste mesmo ano, (BLANCO et al., 2011) faz a avaliação de um sistema de busca baseado em *crowdsourcing* e (QUINN; BEDERSON, 2011) classifica os sistemas de computação humana para ajudar a identificar similaridades entre diferentes sistemas e revelar oportunidades em trabalhos existentes para novas pesquisas. Além disso é feita uma análise da relação entre *crowdsourcing* e computação humana e suas diferenças.

Em 2012, (ESTELLÉS-AROLAS; GONZÁLEZ-LADRÓN-DE-GUEVARA, 2012) revisou as definições existentes de *crowdsourcing* e extraídos elementos comuns para estabelecer as características básicas de qualquer iniciativa de *crowdsourcing*. Por fim, os autores apresentaram uma nova definição de *crowdsourcing*. Neste mesmo ano,

(YANG et al., 2012) propôs dois modelos de mecanismos de incentivo para *crowdsourcing* e (CHATZIMILIOUDIS et al., 2012) propõe uma taxonomia para classificação de aplicações de *crowdsourcing* e três aplicações de otimização baseadas nesta técnica.

Em 2013, (LANE et al., 2013) propõe um sistema de coleta de dados por sensores de *smartphones* que diminui o consumo de energia através da participação direta dos usuários. Neste mesmo ano, (HETMANK, 2013) identificou 17 definições de sistemas de *crowdsourcing* que são categorizadas em quatro perspectivas: organizacional, técnica, de processo e centrada em indivíduos. Na perspectiva organizacional é destacado o papel dos sistemas de *crowdsourcing* na distribuição de tarefas requisitadas aos potenciais colaboradores. Já na perspectiva técnica são enumerados componentes de software, funções técnicas ou objetos de dados que são implementados em sistemas de *crowdsourcing*. A perspectiva de processo descreve detalhadamente as ações que são realizadas com os objetos de dados ou usuários do sistema de *crowdsourcing* e a perspectiva centrada em indivíduos enfatiza que o poder mental humano e a inteligência coletiva são os principais fatores em sistemas de *crowdsourcing*. Além disso, são apresentados quatro componentes que são implementados em um sistema de *crowdsourcing* e suas respectivas funcionalidades. No componente de gerenciamento de usuários são desenvolvidas funções de registro de usuários, avaliação de usuários, formação de grupos de usuários com diferentes propósitos e estabelecimento de mecanismos de coordenação entre os usuários. Já o componente de gerenciamento de tarefas fica responsável por receber as tarefas submetidas e distribuí-las para a multidão que irá realizar a tarefa. Para isso, implementa duas funções básicas: projetar tarefa e atribuir tarefa. No componente de gerenciamento de contribuição são abordadas funcionalidades de controle de qualidade como avaliação de contribuição e seleção de contribuição. Já no componente de gerenciamento de *workflow* são implementadas funções de definição de *workflow* e gerenciamento de *workflow* que são de crucial importância no projeto de tarefas complexas com requisitos e constantes globais.

Em 2014, (HE et al., 2014) é discutido o paradigma de *crowdsourcing* móvel como forma de prover respostas para buscas baseadas em localização em tempo real para as plataformas de *crowdsourcing* baseadas na web através da utilização de dispositivos móveis. É proposta uma arquitetura de *crowdsourcing* híbrida que permite a redução da carga de processamento dos dispositivos móveis através da utilização de um componente web central.

Em 2015, (HOSSAIN; KAURANEN, 2015) é feita uma análise estruturada de trabalhos acadêmicos na área de *crowdsourcing*. Os autores analisaram um total de 346 artigos em inglês e detectaram que as palavras-chave mais frequentes em trabalhos relacionados a *crowdsourcing* foram *social*, *Web*, *innovation*, *open*, *information*, *human*, *online*, *community*, *mechanical*, *collective* e *networks*. Além disso, eles identificaram que a maior parte dos trabalhos analisados foram escritos por autores dos Estados Unidos ou da

Alemanha e, dentre os países em desenvolvimento, China e Brasil tiveram uma presença notável na literatura.

### 3 Classificação de aplicações de Sensoriamento Móvel com *Crowdsourcing* para Cidades Inteligentes

Uma das contribuições deste trabalho consiste em uma nova proposta de classificação de aplicações móveis para *smartphones* com algumas especificidades. Esta classificação foi elaborada a partir de buscas exploratórias em sites da Internet e da revisão de trabalhos acadêmicos.

#### 3.1 Metodologia

Foram aplicadas técnicas de pesquisa exploratória, além de pesquisa bibliográfica e análise de aplicações. A principal ferramenta utilizada durante a pesquisa bibliográfica foi o Google Scholar (GOOGLE... , 2017c). Foram pesquisados trabalhos contendo os termos "urban sensing", "participatory sensing", "opportunistic sensing", "smart cities", "crowdsensing", "crowdsourcing", entre outros. Para o levantamento de aplicações foram feitas pesquisas exploratórias através do Google Search (GOOGLE... , 2017d) e das duas principais lojas de aplicativos: Google Play (GOOGLE... , 2017b) e iTunes (APPLE... , 2017). Foram utilizados os termos "urban sensing applications", "participatory sensing applications", "opportunistic sensing applications", "smart cities applications", "crowdsensing applications", "crowdsourcing applications", entre outros. As informações utilizadas na classificação basearam-se nas páginas de cada aplicação nas lojas de aplicativos e nos seus *websites* oficiais. Além disso, sempre que foi possível foi realizado um teste da aplicação para validar as funcionalidades implementadas por cada uma delas.

#### 3.2 Proposta de classificação

Após a revisão dos trabalhos encontrados, foi elaborada uma proposta de classificação considerando três aspectos principais: *Cidades Inteligentes*, *Sensoriamento Móvel* e *Crowdsourcing*.

No que diz respeito a cidades inteligentes, as aplicações são associadas a um dos componentes de cidades inteligentes baseando-se na abordagem criada em (GIFFINGER; PICHLER-MILANOVIĆ, 2007) que divide cidades inteligentes em 6 componentes: *Smart Economy*, *Smart Living*, *Smart Environment*, *Smart Mobility*, *Smart Government* e *Smart People*. A associação de cada aplicação com um destes componentes foi realizada através

da identificação de fatores e indicadores que poderiam ser influenciados ou analisados através do uso de cada aplicação.

Já em relação à sensoriamento, as aplicações foram caracterizadas em quatro aspectos: escala (pessoal, de grupo ou comunitário), paradigma (participativo ou oportunista), sensores utilizados e localização (utiliza ou não dados de localização obtidos através de GPS ou outras fontes). Estes aspectos foram escolhidos com base nos trabalhos de (LANE et al., 2010) e (CHATZIMILIOUDIS et al., 2012).

Em relação a crowdsourcing, foram analisados o público alvo (ESTELLÉS-AROLAS; GONZÁLEZ-LADRÓN-DE-GUEVARA, 2012; ZHAO; ZHU, 2014), a existência ou não de uma extensão web para a aplicação móvel (CHATZIMILIOUDIS et al., 2012) e a motivação dos usuários ao utilizar a aplicação (CHATZIMILIOUDIS et al., 2012; ESTELLÉS-AROLAS; GONZÁLEZ-LADRÓN-DE-GUEVARA, 2012). No que diz respeito à motivação, foram consideradas cinco opções:

- Altruísmo: O usuário contribui espontaneamente motivado pelo desejo de alcançar um objetivo comum a todos (QUINN; BEDERSON, 2011).
- Reputação: O usuário é motivado por um sistema de reputação presente na aplicação que dá destaque aos usuários que mais contribuem (QUINN; BEDERSON, 2011).
- Serviço: O usuário contribui enquanto utiliza o serviço disponibilizado pela aplicação (CHATZIMILIOUDIS et al., 2012).
- Recompensa: Os usuários participantes recebem algum tipo de prêmio ou pagamento pelas suas contribuições (CHATZIMILIOUDIS et al., 2012; QUINN; BEDERSON, 2011).
- Entretenimento: O usuário contribui através de algum tipo de jogo onde sua contribuição é feita através de forma lúdica (QUINN; BEDERSON, 2011).

As Tabelas 1 e 2 listam 62 das 99 aplicações encontradas durante a pesquisa classificando-as de acordo com os parâmetros definidos para cada aspecto analisado. Foram excluídas desta classificação as aplicações que não possuíam relação com ao menos dois dos três aspectos selecionados com objetivo de detectar o máximo possível de aplicações que fossem relevantes para as três áreas abordadas. Para fins de exibição, foram adotadas abreviações para alguns termos desta tabela. Na coluna Escala, as letras C, G e P representam as escalas Comunitário, de Grupo e Pessoal, respectivamente. Na coluna paradigma as letras P e O representam os paradigmas Participativo e Oportunista. Na coluna Sensores, as letras G, W, L, H, A, T e S representam os sensores GPS, WiFi, Luxímetro, Higrômetro, Acelerômetro, Termômetro e Sensor de Ozônio, respectivamente. As sílabas Ca, Co, Mi e Ma representam os sensores Câmera, Compasso, Microfone e

Magnetômetro, respectivamente. A coluna Local. indica a utilização ou não de dados de localização obtidos via GPS ou outras fontes como redes móveis e WiFi. A coluna Ext. Web indica existência ou não de uma extensão web da aplicação. Na coluna Motivação as abreviações Alt, Rec, Serv, Rep e Ent representam as motivações para contribuição dos usuários Altruísmo, Recompensa, Serviço, Reputação e Entretenimento, respectivamente.

Aplicação	Cidades Inteligentes	Sensoriamento Móvel			Crowdsourcing		
		Escala	Paradigma	Sensores	Local.	Público alvo	Ext. Web
Placemeter	Smart Economy	C	O	Ca	✓	Publico geral	✓
Foursquare	Smart Economy	C	P	G, W e Ca	✓	Publico geral	✓
Yelp	Smart Economy	C	P	G, Ca e Mi	✓	Publico geral	✓
Farmstand	Smart Economy	C	P	G e Ca	✓	Frequenteadores de feiras	✓
Gigwalk	Smart Economy	C	P/O	G e Ca	✓	Trabalhadores autonomos e empresas	✓
Crowd Translator	Smart Economy	G	P	Mi	✓	Publico geral	✓
CityExplorer	Smart Economy	C	P/O	G e Ca	✓	Publico geral	✓
Field Agent	Smart Economy	C	P	G, W, Ca e Mi	✓	Publico geral	✓
Whale Alert	Smart Environment	C	P	G e Ca	✓	Marinheiros e Navegantes	✓
Cicada Hunt	Smart Environment	C	P	G e Mi	✓	Frequenteadores da New Forest, UK	✓
Weather Signal	Smart Environment	C	O	G, L, Ma, H, A e T	✓	Publico geral	✓
Carma Carpooling	Smart Environment	C	P/O	G, W e Ca	✓	Motoristas	✓
Ear-Phone	Smart Environment	C	P	G e Mi	✓	Publico geral	✓
NoiseTube	Smart Environment	C	P	G e Mi	✓	Publico geral	✓
Creek Watch	Smart Environment	C	P	G e Ca	✓	Publico geral	✓
Gas Mobile	Smart Environment	C	O	G e S	✓	Publico geral	✓
Commonwealth Connect	Smart Government	C	P	G e Ca	✓	Publico geral	✓
StreetBump	Smart Government	C	O	G e A	✓	Motoristas	✓
1746 Rio	Smart Government	C	P	G e Ca	✓	Publico geral	✓
Rio Feiras	Smart Government	C	P	G e Ca	✓	Frequenteadores de feiras	✓
mPING	Smart Government	C	P	G	✓	Publico geral	✓
Disaster Reporter	Smart Government	C	P	G, W e Ca	✓	Publico geral	✓
Swarm	Smart Living	C	O	G, W e Ca	✓	Publico geral	✓
Gogobot	Smart Living	C	P	G, W e Ca	✓	Turistas	✓
Walkonomics	Smart Living	C	P/O	G e Ca	✓	Pedestres	✓
Stereopublic	Smart Living	C	P	G e Mi	✓	Publico geral	✓
Open Signal	Smart Living	C	O	G e W	✓	Publico geral	✓
SitOrSquat	Smart Living	C	P	G e Ca	✓	Publico geral	✓
Fogo Cruzado	Smart Living	C	P	G e Mi	✓	Publico geral	✓
Guadiões da Saúde	Smart Living	C	P	G e Ca	✓	Participantes dos jogos olímpicos	✓
Flickr	Smart Living	C	P	G, W, Ca e Mi	✓	Publico geral	✓

Tabela 1 – Tabela de classificação de aplicações de Sensoriamento por Móvel para Cidades Inteligente com Crowdsourcing - Parte 1.

Aplicação	Cidades Inteligentes	Sensoriamento Móvel			Crowdsourcing		Ext. Web	Motivação
		Escala	Paradigma	Sensores	Local.	Público alvo		
GasBuddy	Smart Living	C	P	G	✓	Motoristas	✓	Alt/Rec
DietSense	Smart Living	C	O	G, Ca e Mi	✓	Publico geral	✓	Alt/Serv
SmartSantander	Smart Living	C	P/O	G, W, Co, Mi, T e L	✓	Publico geral	✗	Alt/Serv
BeWell	Smart Living	P	O	G, A e Mi	✓	N/A	N/A	N/A
Ambulation	Smart Living	C	O	G e A	✓	Pacientes com doenças degenerativas	✓	Alt/Serv
AndWellness	Smart Living	C	P/O	G, W e Ca	✓	Publico geral	✓	Alt/Serv
LiveCompare	Smart Living	C	P	G e Ca	✓	Publico geral	✗	Alt/Serv
CenceMe	Smart Living	C	O	G, A, Ca e Mi	✓	Publico geral	✓	Alt/Serv
CompareEverywhere	Smart Living	C	P	G e Ca	✓	Publico geral	✗	Alt/Serv
ShopSavvy	Smart Living	C	P	G, W e Ca	✓	Publico geral	✓	Alt/Serv
StressSense	Smart Living	C	O	G, A e Mi	✓	Publico geral	✗	Alt/Serv
Waze	Smart Mobility	C	P/O	G, Ca e Mi	✓	Motoristas	✓	Alt/Rep/Serv
Here Maps	Smart Mobility	C	O	G e W	✓	Motoristas	✓	Alt/Serv
Go Mobile TomTom	Smart Mobility	C	P/O	G, W e Mi	✓	Motoristas	✗	Alt/Serv
BuddyTraffic	Smart Mobility	C	P	G	✓	Motoristas	✗	Rec/Rep
Moovit	Smart Mobility	C	P/O	G e W	✓	Passageiros de transporte público	✗	Alt/Serv
Jana	Smart Mobility	P	O	G	✓	Publico geral	✗	Rec
Vtrack	Smart Mobility	C	O	G e W	✓	Motoristas	✓	Alt/Serv
SignalGuru	Smart Mobility	C	O	G e Ca	✓	Motoristas	✗	Alt/Serv
Citymapper	Smart Mobility	C	O	G	✓	Passageiros de transporte público	✓	Alt/Serv
Google Maps	Smart Mobility	C	P/O	G, W, Ca e Mi	✓	Publico geral	✓	Alt/Serv
Nericell	Smart Mobility	C	O	G, A e Mi	✓	Motoristas	✗	Alt
Biketastic	Smart Mobility	C	P/O	G, Ca, A e Mi	✓	Ciclistas	✓	Alt/Serv
MetroTrack	Smart Mobility	C	O	G e Mi	✓	Publico geral	✗	Alt
VibN	Smart Mobility	C	P/O	G, A e Mi	✓	Publico geral	✗	Alt/Serv
AirPlace	Smart Mobility	C	O	W	✓	Publico geral	✗	Alt/Serv
SmartTrace	Smart Mobility	C	O	G e W	✓	Publico geral	✗	Alt/Serv
CityTracks	Smart Mobility	C	P/O	G e W	✓	Publico geral	✗	Alt/Serv
Learnist	Smart People	N/A	N/A	N/A	N/A	Publico geral	✓	Alt
Edmodo	Smart People	N/A	N/A	N/A	N/A	Membros de instituições de ensino	✓	Alt
Showme	Smart People	N/A	N/A	N/A	N/A	Professores e estudantes	✓	Alt/Serv

Tabela 2 – Tabela de classificação de aplicações de Sensoriamento por Móvel para Cidades Inteligente com Crowdsourcing - Parte 2.



## 4 Implementação de Aplicação CityTracks na plataforma Android

Neste capítulo, será apresentada uma fundamentação teórica sobre inferência de modo de transporte baseada em aprendizado de máquina e dados de GPS, bem como alguns trabalhos relacionados. Além disso, será descrito o funcionamento da aplicação CityTracks e detalhes sobre a implementação em Android. Por fim, serão apresentadas a estrutura do código e a interface gráfica desenvolvida nesta plataforma.

### 4.1 Inferência do modo de transporte baseada em aprendizado de máquina e dados de GPS

Nesta seção, discutiremos a importância da inferência de modo de transporte em cidades inteligentes e apresentaremos algumas técnicas de aprendizado de máquina utilizadas no desenvolvimento de aplicações com este objetivo. Além disso, descreveremos alguns trabalhos voltados para a inferência de modo de transporte baseada em dados de GPS coletados via *smartphones*.

#### 4.1.1 Inferência do modo de transporte em cidades inteligentes

A inferência do modo de transporte em tempo real, possibilita uma gama de aplicações sensíveis a contexto para cidades inteligentes. Uma vez que seja possível identificar o modo de transporte utilizado pelo usuário, é possível coletar e fornecer informações relevantes para tomada de decisão no que diz respeito ao modo de transporte utilizado. Além disso, aplicações de sensoriamento oportunista poderiam coletar automaticamente o tempo de deslocamento por meios de transporte motorizados e não motorizados, permitindo calcular o número de calorias gasto durante o deslocamento diário ou a quantidade de carbono emitida pela utilização de carro ou moto.

#### 4.1.2 Técnicas de aprendizado de máquina

Nesta seção apresentaremos as técnicas de aprendizado de máquina utilizadas na aplicação implementada neste trabalho. Além destas, outras técnicas podem ser utilizadas em outros trabalhos da área, uma vez que há uma grande variedade de técnicas que podem ser aplicadas e combinadas.

Neste trabalho, utilizaremos as seguintes técnicas de aprendizado de máquina: *Multilayer Perceptron*, *Support Vector Machine* (SVM), *Decision Table* e *Bayesian Net*. A seguir descrevemos de forma resumida cada uma destas técnicas.

#### 4.1.2.1 Multilayer Perceptron

A técnica de *Multilayer Perceptron* baseia-se em uma rede neural de múltiplas camadas de nós em um grafo direcionado com cada camada totalmente conectada a outra, conforme representado na Figura 4. Com exceção dos nós de entrada, cada nó é um neurônio com uma função de ativação não-linear. Este classificador utiliza uma técnica de aprendizado de máquina supervisionado para treinamento da rede neural. Mais informações sobre este classificador podem ser encontradas em (COLLOBERT; BENGIO, 2004).

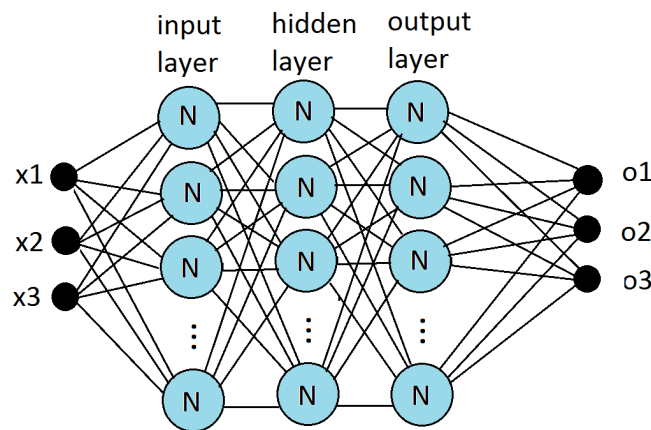


Figura 4 – Diagrama de funcionamento do algoritmo de classificação *Multilayer Perceptron*.

#### 4.1.2.2 Support Vector Machine (SVM)

O algoritmo SVM tenta encontrar a melhor função de classificação para distinguir membros de duas classes de dados de treinamento. A função gerada por esta técnica retornará um valor positivo para membros de uma classe e negativo para elementos da outra. Maiores explicações sobre esta técnica podem ser encontradas em (WU et al., 2008). A técnica de SMO, consiste em um algoritmo de treinamento das SVMs que utiliza a técnica de divisão e conquista para otimizar o cálculo da função de classificação, diminuindo a complexidade desta técnica de inferência. Segundo (PLATT et al., 1998) este algoritmo pode ser 1000 vezes mais rápido que os algoritmos de SVM tradicionais. A Figura 5 ilustra o funcionamento de uma SVM.

#### 4.1.2.3 Decision Table

O algoritmo de classificação de *Decision Table* consiste de uma tabela hierárquica na qual cada entrada em uma tabela de nível superior é dividida em um par de atributos

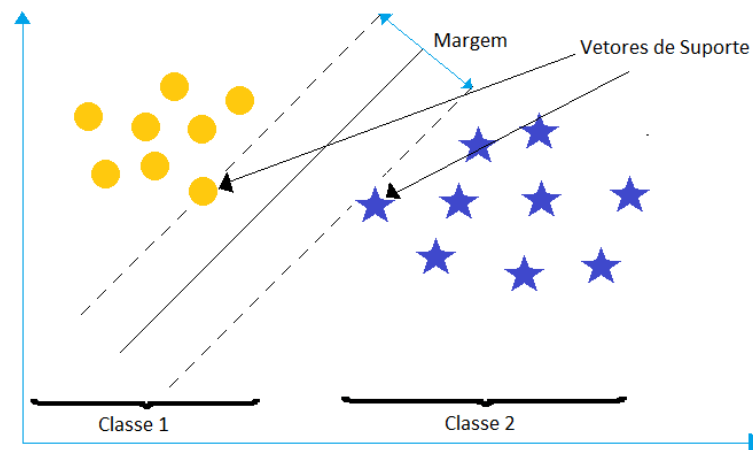


Figura 5 – Ilustração do funcionamento de uma SVM adaptada de (FANG et al., 2016).

adicionais para formar outra tabela. As Tabelas 3, 4 e 5 retiradas de (BECKER, 1998) ilustram o processo de classificação por *Decision Table*.

Atributo1	Atributo2	Atributo3	Atributo4	Peso	Probabilidade[]
a	10-20	4-9	sim	10	0,3; 0,6; 0,1
a	10-20	9-10	nao	34	0,61; 0,37; 0,02
a	20-30	9-10	sim	123	0,1; 0,6; 0,3
a	20-30	9-10	nao	1	1; 0; 0
a	20-30	10-18	sim	5	0,8; 0,2; 0
a	20-30	10-18	nao	23	0,2; 0,3; 0,5
:	:	:	:	:	:
d	4-9	4-9	sim	2	0,5; 0,5; 0
d	9-10	9-10	nao	7	0; 1; 0
d	10-18	10-18	nao	9	0,33; 0,33; 0,33

Tabela 3 – Tabela de decisão de nível 2 adaptada de (BECKER, 1998)

Atributo1	Atributo2	Peso	Probabilidade[]
a	10-20	44	0,61; 0,34; 0,05
a	20-30	850	0,23; 0,41; 0,36
a	30-40	230	0,35; 0,2; 0,45
a	40-50	56	0,12; 0,28; 0,60
:	:	:	:
d	40-50	120	0,17; 0,31; 0,52

Tabela 4 – Tabela de decisão de nível 1 adaptada de (BECKER, 1998)

#### 4.1.2.4 Bayesian Networks

O algoritmo de classificação *Bayesian Net*, baseia-se em uma estrutura de nós que representa variáveis de interesse proposicionais, como a temperatura de um dispositivo, o gênero de um paciente ou um modo de transporte, por exemplo. Estes nós são conectados de acordo com as dependências causais entre as variáveis. Estas dependências são quantificadas

Peso	Probabilidade
19,345	0,32; 0,33; 0,35

Tabela 5 – Tabela de decisão de nível 0 adaptada de (BECKER, 1998)

por probabilidades condicionais para cada nó e seus respectivos nós pais na rede (PEARL, 2011). Talvez o aspecto mais importante das redes bayesianas é o fato delas buscarem representar o mundo real ao invés de processos lógicos. As setas do diagrama representado na Figura 6, exemplificam conexões causais reais entre a estação do ano, o volume de chuvas, o número de profissionais de férias, o trânsito nas vias e o modo de transporte utilizado pelos habitantes de uma cidade.

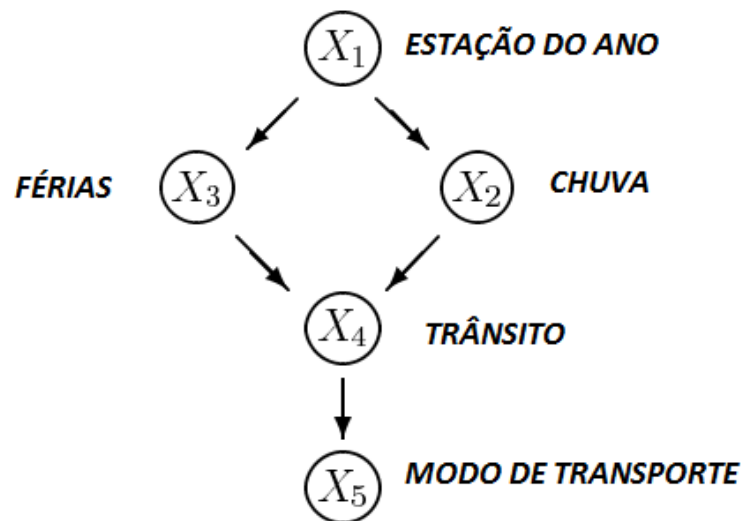


Figura 6 – Rede bayesiana representando influências causais entre cinco variáveis adaptada de (PEARL, 2011).

#### 4.1.3 Inferência do modo de transporte com base em dados de GPS

Nesta seção, descreveremos alguns trabalhos voltados para a inferência do modo de transporte com base em dados de GPS coletados via *smartphones*.

Em (YANG; YAO; JIN, 2015) 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 deslocamento de ônibus foi maior do que a precisão em qualquer outro estudo. As únicas 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

(WU; YANG; JING, 2016). 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.

Já em (XIAO; ZHICAI; JINGXIN, 2015) foram utilizados diferentes técnicas de aprendizado de máquina para inferência do modo de transporte e foi detectado que a acurácia das redes neurais era superior as demais técnicas utilizadas. No entanto, a deficiência dos algoritmos tradicionais desta técnica, chamada *local optimum* motivou os autores a utilizar *particle swarm optimization* (PSO) para encontrar um *global optimum*. Como resultado 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 garantir o valor desta taxa para estas duas classes de modo de transporte. Os dados de GPS foram divididos em dois subconjuntos distintos, um contendo 25% dos dados e o outro contendo 75%. O maior conjunto foi utilizado para treinamento dos classificadores e o menor conjunto para os testes. Os resultados mostraram uma acurácia da inferência de de 95.81% no subconjunto de treinamento e de 94,44% no subconjunto de teste.

Em (LARI; GOLROO, 2015) 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 inferência 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 (WU; YANG; JING, 2016).

## 4.2 Aplicação CityTracks

Nesta seção, serão apresentados os requisitos para a implementação da aplicação CityTracks (QUINTELLA, 2013) na plataforma Android.

A aplicação CityTracks, implementada em (QUINTELLA, 2013), teve como principal objetivo realizar a coleta do modo de transporte utilizado pelo usuário através coleta de dados de GPS. Os dados coletados foram utilizados para o desenvolvimento de um algoritmo de mineração de dados que realiza o pré-processamento, sumarização de atributos e aplicação de técnicas de aprendizado de máquina para inferência do modo de transporte.

Este algoritmo foi desenvolvido e avaliado após a coleta de dados através da utilização dos software R e *Weka Experimenter*. Uma das limitações da aplicação desenvolvida neste trabalho foi a ausência da inferência em tempo real, que permitiria a avaliação do algoritmo proposto em um teste de campo. A Figura 7 ilustra o processo de pré-processamento e sumarização dos chunks definido em (QUINTELLA, 2013).

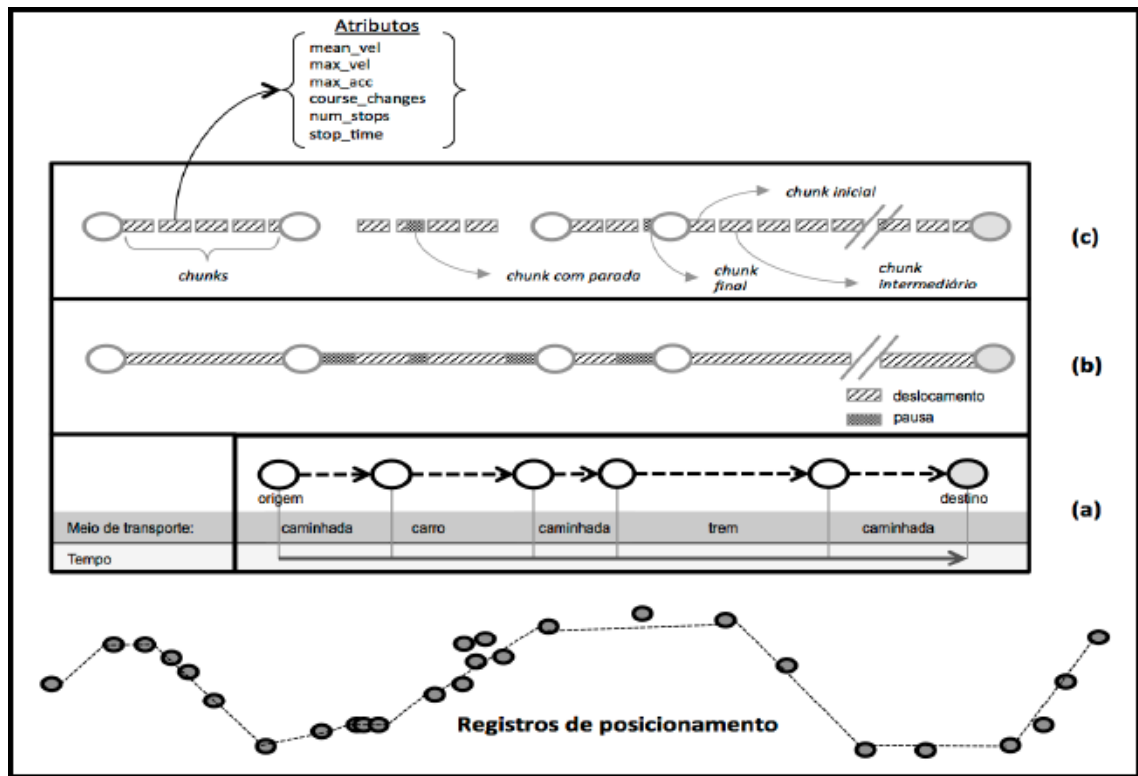


Figura 7 – Representação da informação do movimento coletado passo a passo no processo de transformação e sumarização dos dados obtida em (QUINTELLA; CAMPOS; ANDRADE, 2016).

No trabalho de (QUINTELLA, 2013) foi detectado que a divisão dos segmentos em 90 segundos gerou uma relação próxima do ideal entre custo de tempo e qualidade da inferência. Durante estes 90 segundos, são capturados traces de movimento 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. Traces com diferença de segundos superior a um são inseridos através de uma técnica de interpolação na qual são gerados traces intermediários com base nas características do trace anterior, mantendo velocidade constante e aceleração zero. Ao fim dos 90 segundos de coleta são calculadas as informações de sumarização do conjunto de traces, denominado *chunk*. Os campos

de sumarização são velocidade média, velocidade máxima, aceleração máxima, número de paradas, tempo de parada e número de mudanças de direção. Através destes campos de sumarização são utilizadas as técnicas de mineração de dados, *Multilayer Perceptron*, SVM, *Bayesian Net* e *Decision Table* para inferência do modo de transporte utilizado pelo usuário. Estas técnicas foram escolhidas com base no trabalho de (QUINTELLA, 2013), pois apresentaram os melhores resultados para este tipo de inferência.

## 4.3 Implementação

Nesta seção, será apresentada a implementação da aplicação CityTracks na plataforma Android, utilizando a FusedLocationAPI (DOCUMENTAÇÃO..., 2017) e a API do Weka em Java (HALL et al., 2009) para realização da coleta e inferência do modo de transporte em tempo real com base na técnica de mineração desenvolvida em (QUINTELLA, 2013).

No trabalho de (QUINTELLA, 2013) foi implementado um protótipo da aplicação CityTracks na plataforma IOS. Este protótipo permitia somente, que usuários coletassem os dados de movimento e informassem o modo de transporte utilizado, para que posteriormente fosse feita a inferência através do algoritmo proposto pelo autor. Permaneceu no entanto, a necessidade de implementar a inferência do modo de transporte em tempo real, com base nas técnicas de mineração de dados utilizadas. Além disso, a implementação da CityTracks somente na plataforma IOS oferece um certo nível de restrição para experimentos em larga escala, uma vez que os aparelhos desta plataforma tem um custo elevado, o que os torna menos populares perante ao mercado consumidor do país em que este estudo está sendo realizado.

Dados estes fatores, optou-se por implementar uma versão da aplicação CityTracks na plataforma Android, com o recurso de inferência em tempo real baseado na técnica desenvolvida em (QUINTELLA, 2013).

A implementação na plataforma Android, utilizou a FusedLocationAPI (DOCUMENTAÇÃO..., 2017) para captura dos traces a partir de dados de GPS, redes móveis celulares e WiFi. Foi criado um conjunto de classes em Java para realização da etapa de pré-processamento e sumarização dos segmentos de dados de posição, denominados *chunks* (QUINTELLA, 2013), além de uma classe que utiliza a API do WEKA em Java (WEKA..., 2017) para executar os algoritmos de classificação selecionados. Para o algoritmo de classificação Multilayer Perceptron, foram utilizados os parâmetros *default* do WEKA, com exceção do *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. 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* foram utilizados os parâmetros *default*: método de busca = *Best First*, *number of folds* = 1, medida de avaliação de desempenho = *accuracy* para classe discreta e *rmse* para classe numérica. Para o algoritmo *Bayesian Net* foram utilizados os parâmetros *default* do WEKA: *number of folds* = 10, *seed* de número aleatório = 1 e algoritmo de busca = K2.

## 4.4 Estrutura do Código

A implementação de uma aplicação em Android, consiste primariamente no desenvolvimento de uma Activity. Uma Activity tem seu layout especificado através de um arquivo xml (ACTIVITY... , 2017). Na implementação do CityTracks, foi implementada uma Activity denominada MainActivity que será responsável por controlar toda interação do usuário com o aplicativo. Foi definido um layout simples através do arquivo *main\_activity.xml*.

O controle da FusedLocationAPI e das demais classes da aplicação foi feito através da classe CityTracksService que implementa um Service da plataforma Android (SERIVCES... , 2017). Uma vez que um Service é acionado por uma Activity ele continua sendo executado em *background* de forma assíncrona a Activity que o acionou, o que incentivou a utilização deste componente para executar o processamento das traces de localização do usuário, mesmo que a aplicação CityTracks tenha sido minimizada pelo mesmo.

As demais classes foram organizadas em pacotes baseados no padrão de projetos Model-View-Controller (HUNT, 2013) de forma a reduzir o acoplamento, aumentar a coesão das mesmas e aumentar a legibilidade do código, seguindo as boas práticas de desenvolvimento de aplicações na linguagem Java (SIERRA; BATES; SIEBEN, 2005), utilizada para desenvolvimento de aplicações na plataforma Android.

No pacote de *modelo* foram definidos dois subpacotes. No subpacote *entidade* são definidas a classe Ponto, que representa um trace de localização capturado, a classe Chunk, que representa um conjunto de 90 pontos capturados e seus respectivos dados de sumarização, as classes ChunkBuilder e PontoFactory, que permitem a manipulação e criação de objetos das classes Chunk e Ponto e as enumerações ModosDeTransporte, ModosDeTransporteMotorizados, ModosDeTransporteNaoMotorizados e TiposDeModoDeTransporte, que definem os valores utilizados na classificação dos chunks. Já no subpacote *persistência* foram implementadas classes para persistência dos chunks em diferentes formatos de arquivos, baseadas no padrão de projeto DAO (*Data Access Object*) (HUNT, 2013) Foi implementada uma interface ChunkDAO para definir a assinatura dos métodos utilizados, e as classes ChunkDaoArff, para persistência em formato ARFF (*Attribute-Relation File Format*), ChunkDaoCsv, para persistência em formato



CSV (*Comma Separated Value*) e ChunkDaoJson, para persistência em formato JSON (*JavaScript Object Notation*).

A persistência em ARFF foi adotada por sua compatibilidade com a API do WEKA, que foi utilizada para executar os algoritmos de mineração de dados (HALL et al., 2009). 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 móvel (CROCKFORD, 2006). 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 e seus respectivos pontos, permitindo a fácil recuperação do objetos através da API Gson (GOOGLE..., 2017a).

No pacote *utils* foi armazenada a classe Utils, que possui métodos utilitários para as demais classes e a classe Classificador, que fica no subpacote *inferencia*. A classe Classificador é responsável por efetuar a inferência dos modos de transporte utilizados em cada chunk, utilizando três técnicas diferentes: *Multilayer Perceptron*, *SVM + Decision Table* e *SVM + Bayesian Net*. A Figura 8 ilustra o processo de inferência do modo de transporte realizado pela aplicação CityTracks em Android.

No pacote *controle* foram armazenadas as classes MainActivity e CityTracksService, responsáveis por responder as solicitações da interface e acionar as classes de modelo e as bibliotecas utilizadas para executar a coleta de traces e inferência do modo de transporte.

Como o layout de uma aplicação na plataforma Android é definido através de um arquivo XML, localizado por padrão na pasta *res/layout* não foi definido um pacote *visão*.

Na Figura 9 é possível visualizar o diagrama de classes da aplicação.

## 4.5 Interface Gráfica e Uso da Aplicação

Na interface gráfica da aplicação foi utilizado um layout simples, com dois botões: Um para iniciar a coleta e um para parar a coleta. O usuário pode informar o modo de transporte que está utilizando em uma caixa de seleção e os dados da coleta são exibidos de forma textual na tela do usuário, caso ele não minimize a aplicação. Se o usuário minimizar a aplicação após iniciar a coleta, a mesma continua sendo executada em *background* até que o usuário limpe a aplicação da memória ou selecione a opção parar coleta.

As Figuras 10, 11 e 12 demonstram o passo a passo da coleta utilizando a aplicação CityTracks em Android. Nos passos 1 e 2 o usuário acessa a aplicação e seleciona o modo de transporte que irá utilizar. Nos passos 3 e 4 o usuário inicia a coleta pressionando o botão "Iniciar Coleta". No 5 o usuário visualiza a latitude, longitude e hora da ultima posição GPS capturada e no passo 6 ele encerra a coleta pressionando o botão "Para

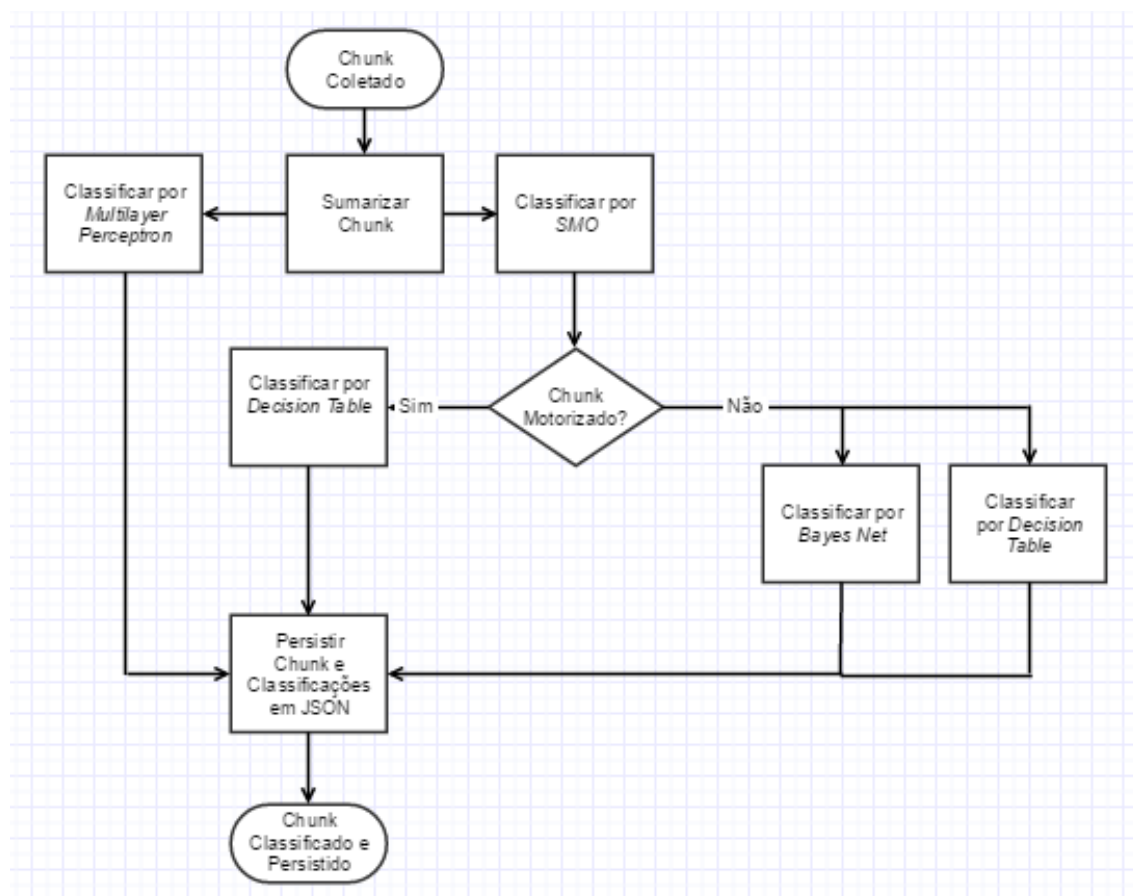


Figura 8 – Diagrama de processo da inferência de modo de transporte pela aplicação CityTracks em Android.

Coleta". O usuário pode selecionar uma outra opção de modo de transporte durante a coleta sendo que o chunk sempre será coletado com o modo de transporte selecionado no momento em que ele é sumarizado.

Os arquivos ARFF, CSV e JSON são gerados na pasta *Android/data/unirio.citytracks/files/Download* e podem ser acessados através de um aplicativo de gerenciamento de arquivos de terceiros como o Total Commander (TOTAL..., 2017). Na maioria das vezes não é possível acessar esta pasta através do gerenciador de arquivos nativo do Android.

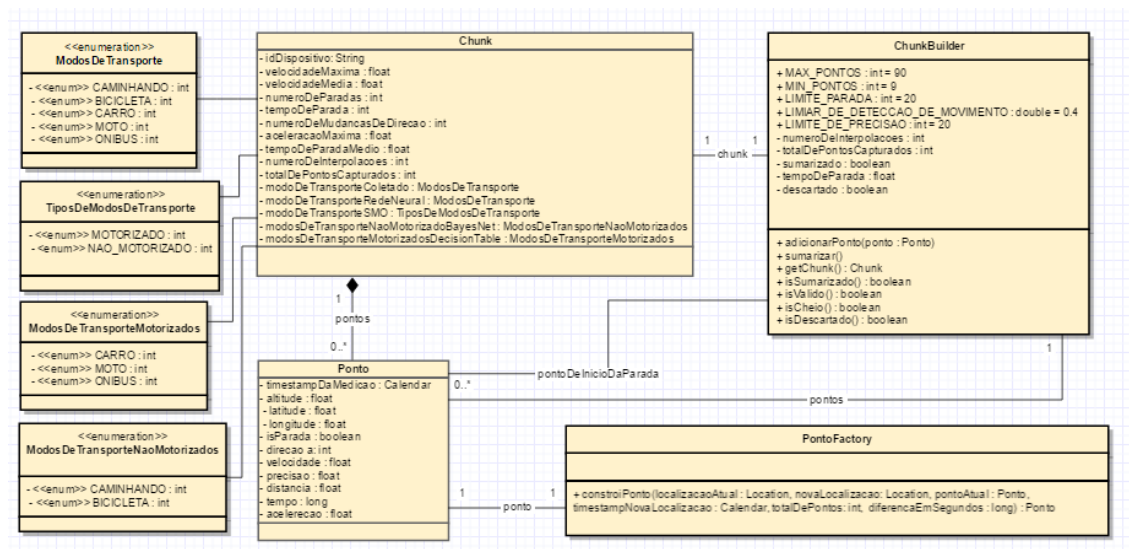


Figura 9 – Diagrama de classes de modelo baseado na notação UML (LARMAN, 2012).

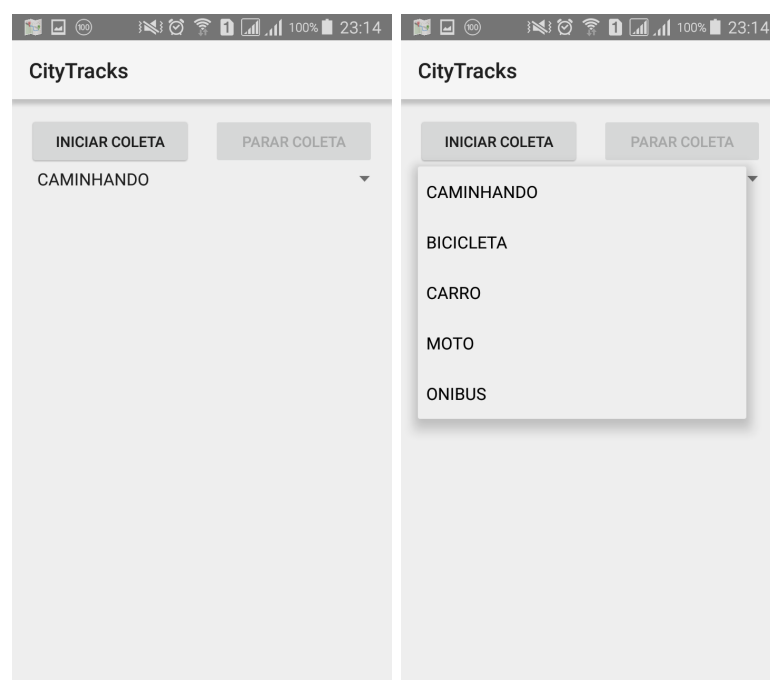


Figura 10 – Passos 1 e 2 - Tela inicial da aplicação e Seleção de modo de transporte.

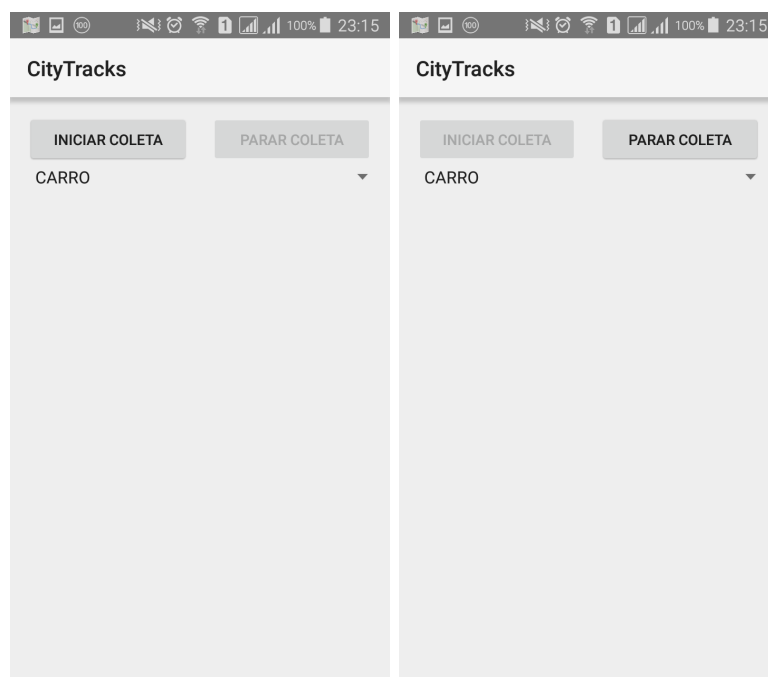


Figura 11 – Passos 3 e 4 - Modo de transporte selecionado e iniciar coleta.

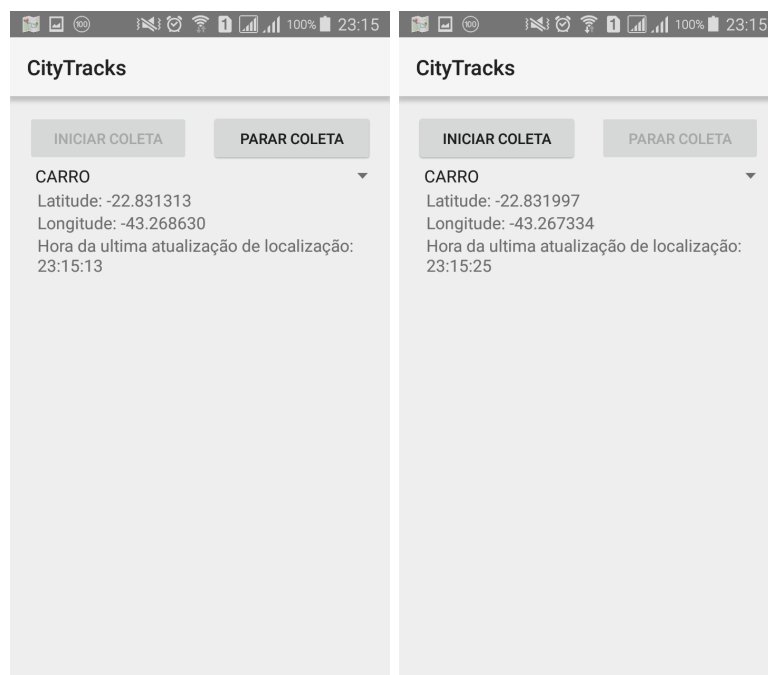


Figura 12 – Passos 5 e 6 - Coletando posições e finalizando coleta.

## 5 Testes de Campo da Aplicação CityTracks na plataforma Android

Neste capítulo, será apresentada uma avaliação do uso da aplicação implementada por meio de testes de campo realizados por voluntários do Rio de Janeiro. Assim, serão apresentadas a metodologia utilizada nos testes de campo da aplicação CityTracks e a análise dos resultados destes testes. 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, total de pontos capturados e 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 encontrados.

### 5.1 Metodologia

Para os testes de campo foram utilizados dispositivos de 18 alunos do curso informática da UNIRIO, de diferentes fabricantes, rodando diferentes versões do sistema operacional Android. Foi disponibilizado o APK 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 á 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 e Nova Iguaçu, 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 13, 15 e 14 é possível visualizar os traces de mobilidade coletados no mapa da cidade.

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 objeto da classe Chunk completo, com todos os pontos coletados, dados de sumarização e modo de transporte coletado e inferido, pelas quatro técnicas de aprendizado de máquina escolhidas. 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, com algumas exceções. Todos os chunks sofrem inferência por



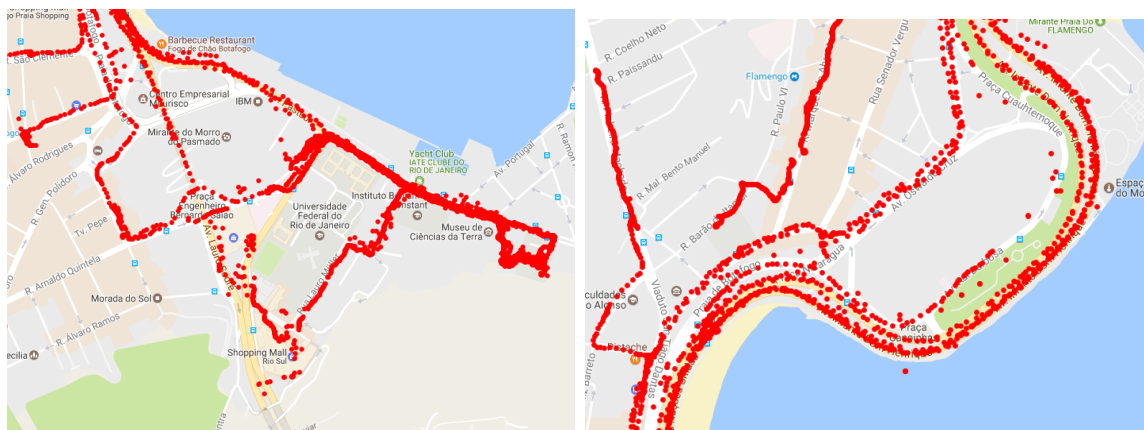


Figura 15 – Gráfico de mobilidade com todos os pontos coletados que não eram paradas na região da Urca, Botafogo e Aterro do Flamengo.

## 5.2 Análise dos Resultados Obtidos

Nesta seção, analisaremos os resultados dos testes de campo realizados em duas etapas. Na primeira etapa, analisaremos os dados coletados pelos usuários, identificando os principais modos de transporte utilizados, quantidade de pontos coletados por faixa de precisão e total de pontos capturados e 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 encontrados.

### 5.2.1 Análise dos dados coletados

Durante os testes de campo realizados pelos voluntários foi possível capturar em média traces de localização com precisão de 26.95 metros com frequência entre um e dois segundos. Foi coletado um total de 1338 chunks e 120176 pontos, sendo 53639 pontos capturados, aproximadamente 45% e 66537 pontos gerados sinteticamente através de uma técnica de interpolação baseada em (QUINTELLA, 2013), aproximadamente 55%. Na Tabela 6 é possível visualizar o total de chunks e pontos coletados por dispositivo assim como a precisão média de posição GPS dos pontos coletados por cada dispositivo e o percentual de pontos capturados e gerados por interpolação.

Entre os modos de transporte, aproximadamente 51% dos pontos foram coletados em deslocamento por ônibus e 41% a pé. Apenas 7% dos pontos foram coletados em deslocamento de carro e 1% andando de bicicleta. Nenhum ponto foi coletado para deslocamento de moto. Os valores totais de pontos coletados para cada modo de transporte e os percentuais de pontos capturados pela FusedLocationAPI e gerados por interpolação para cada um deles são representados no gráfico da Figura 16.

Outro aspecto relevante é a precisão dos pontos coletados. Apenas 8% dos pontos

ID Dispositivo	Chunks	Pontos	Precisão GPS Méd.	Desv. Pad.	% Gerado	% Capturado
45e146f29d39728	219	19710	32.3 m	25.0 m	60%	40%
e7e306fecc024def	52	4680	26.5 m	15.9 m	77%	23%
a575b02e7b59e37a	7	630	31.0 m	4.2 m	39%	61%
2f49f7e65fc7a1e	68	6120	9.4 m	6.0 m	36%	64%
b8c035ec310c0263	87	7830	25.7 m	40.0 m	33%	67%
319bf0d88ea98fe9	40	3600	37.3 m	30.2 m	60%	33%
8c2bb28c5b325f8c	123	11070	20.4 m	23.5 m	36%	64%
db45742cf7c44a6a	34	3060	34.3 m	43.7 m	38%	62%
d465c3491dcbcf99	222	19980	13.1 m	7.1 m	38%	62%
4c369cbb6b63e0af	16	1440	48.3 m	47.4 m	67%	33%
ee0e4d65037467f5	83	7470	42.7 m	38.7 m	84%	16%
315ed684283a2dbf	22	1980	7.4 m	7.7 m	2%	98%
ed7193e2f6d147ba	13	1170	10.1 m	7.0 m	32%	68%
6bf6a97f511fe481	51	4590	47.6 m	43.1 m	82%	18%
e474217e78cf5997	11	990	4.1 m	4.4 m	36%	64%
818761301263337	7	630	7.0 m	3.7 m	35%	65%
30abdd982abc5478	30	2700	20.1 m	35.6 m	41%	59%
a51dff5e74f3ee20	253	22770	34.1 m	24.4 m	76%	24%

Tabela 6 – Dados de coleta com a aplicação CityTracks por dispositivo.

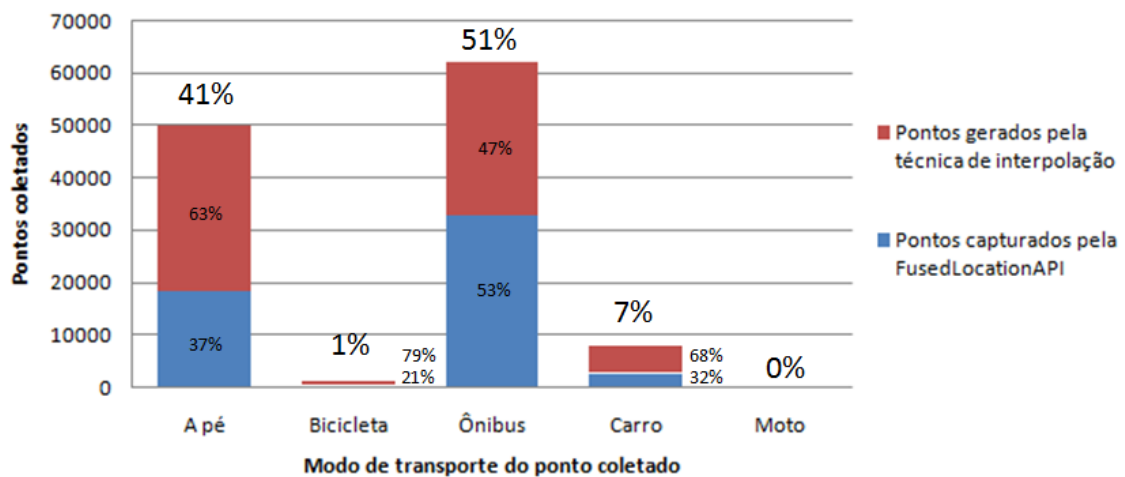


Figura 16 – Pontos capturados x Pontos gerados por modo de transporte.

coletados 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 pontos capturados pela FusedLocationAPI e gerados por interpolação para cada faixa de precisão e a frequência cumulativa de pontos coletados por faixa de precisão são representadas no gráfico da Figura 17.

### 5.2.2 Análise dos algoritmos de classificação

Para avaliação dos algoritmos de classificação utilizados na inferência do modo de transporte foram analisados três métricas: *accuracy*, *precision* e *recall*, os dois últimos calculados com base na *confusion matrix* de cada classificador. Estas métricas foram escolhidas por apresentarem um conjunto de informações relevantes para avaliação dos classificadores e por serem as principais métricas utilizadas para avaliação de algoritmos de



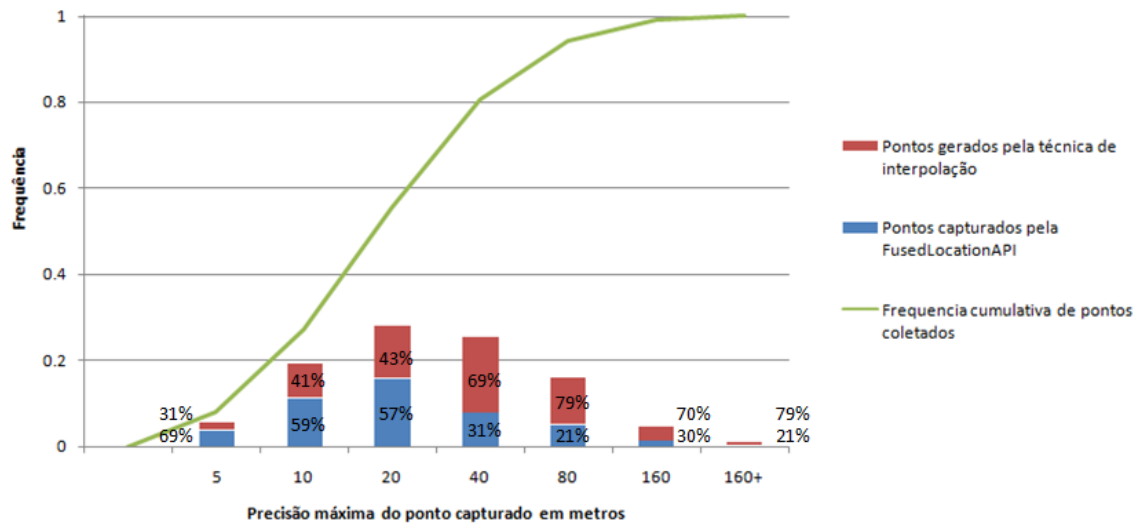


Figura 17 – Pontos capturados x Pontos gerados por faixa de precisão em metros e frequência cumulativa de pontos coletados por faixa de precisão.

aprendizado de máquina nos trabalhos relacionados. Enquanto que a *accuracy* nos informa o percentual de inferências corretas, *precision* e *recall* nos dão informações mais precisas sobre especificidade e sensibilidade das inferências realizadas (POWERS, 2011). *Precision* é calculada pela razão entre o total de inferências corretas para uma determinada classe e o total de inferências realizadas para aquela classe. *Recall* é obtida pela razão entre o total de inferências corretas para uma determinada classe e o total de instâncias capturadas pertencentes àquela classe (QUINTELLA, 2013). De forma geral, *precision* e *recall* nos dão uma noção maior de relevância dos resultados obtidos enquanto que a *accuracy* nos dá uma visão superficial do desempenho do algoritmo. Além destas métricas calculamos também a *F-Measure*, bastante utilizada para análise de algoritmos de classificação (QUINTELLA, 2013), ela é calculada conforme a fórmula  $F_1 = 2 \times \frac{precision \times recall}{precision + recall}$ .

#### 5.2.2.1 Resultados da classificação por *Multilayer Perceptron*

A *accuracy* geral observada pela inferência por *Multilayer Perceptron* foi de aproximadamente 69,8%, enquanto que o *precision* geral foi de 68,9% e o *recall* geral 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 7 representa a *confusion matrix* das classes de modo de transporte consideradas para este classificador e a Tabela 8 exibe as métricas calculadas para cada classe.

Podemos observar que mesmo com 0 inferências corretas realizadas a classe de modo de transporte Bicicleta obteve o maior valor para a métrica *accuracy*, o 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

	Coletado A pé	Coletado Bicicleta	Coletado Ônibus
<b>Classificado A pé</b>	256	9	73
<b>Classificado Bicicleta</b>	73	0	120
<b>Classificado Ônibus</b>	223	1	496

Tabela 7 – *Confusion Matrix* das classes de modo de transporte A pé, Bicicleta e Ônibus para o classificador *Multilayer Perceptron*

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

Tabela 8 – *Accuracy, precision, recall* e *F-Measure* para as classes de modo de transporte A pé, Bicicleta e Ônibus para o classificador *Multilayer Perceptron*

e o 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 melhor performance para este modo de transporte através do classificador *Multilayer Perceptron*.

#### 5.2.2.2 Resultados da classificação por *Support Vector Machine*

Para a inferência por SVM a *accuracy* geral 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 10, 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 relevância maior dos resultados das inferências de chunks motorizados. Na Tabela 9 é possível visualizar a *confusion matrix* para este algoritmo.

	Coletado Motorizado	Coletado Não Motorizado
<b>Classificado Motorizado</b>	632	318
<b>Classificado Não Motorizado</b>	144	244

Tabela 9 – *Confusion Matrix* das classes de tipo de modo de transporte Motorizado e Não Motorizado para o classificador *Support Vector Machine*

	Motorizado	Não Motorizado
<b>Accuracy</b>	0,655	0,655
<b>Precision</b>	0,665	0,629
<b>Recall</b>	0,814	0,434
<b>F-Measure</b>	0,732	0,514

Tabela 10 – *Accuracy, precision, recall* e *F-Measure* para as classes de tipo de 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 tipo de modo de transporte Motorizado obteve um desempenho superior ao da classificação para a classe Não Motorizado. Este resultado foi altamente influenciado pela diferença entre os valores da métrica *recall* entre estas classes.

### 5.2.2.3 Resultados da 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* geral de 87,5%, *precision* geral de 93,4% e *recall* geral 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 como modo de transporte carro para permitir o cálculo dos falsos positivos para a inferência do modo de transporte Ônibus. Na Tabela 11 é possível visualizar a *confusion matrix* para este algoritmo. Na Tabela 12 são apresentadas as métricas *accuracy*, *precision*, *recall* e *F-Measure* para cada classe de modo de transporte considerada.

	Coletado Ônibus	Coletado Carro
Classificado Ônibus	590	42

Tabela 11 – *Confusion Matrix* de modo de transporte motorizado para o classificador *Decision Table*.

	Ônibus
<i>Accuracy</i>	0,875
<i>Precision</i>	0,933
<i>Recall</i>	1
<i>F-Measure</i>	0,903

Tabela 12 – *Accuracy*, *precision*, *recall* e *F-Measure* para detecção classe de modo de transporte Ônibus com o classificador *Decision Table*

### 5.2.2.4 Resultados da 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 244 *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. Nas Tabelas 13 e 14 é possível visualizar as matrizes de confusão para as duas técnicas, considerando as classes de modo de transporte a pé e de bicicleta. Nas Tabelas 16 e 15 é possível visualizar os valores de *accuracy*, *precision*, *recall* e *F-Measure* para as duas classes de modo de transporte de cada técnica.

	Coletado A pé	Coletado Bicicleta
Classificado A pé	103	8
Classificado Bicicleta	133	0

Tabela 13 – *Confusion Matrix* de modo de transporte não motorizado para o classificador *Decision Table*.

	Coletado A pé	Coletado Bicicleta
Classificado A pé	103	8
Classificado Bicicleta	133	0

Tabela 14 – *Confusion Matrix* de modo de transporte não motorizado para o classificador *Bayesian Net*.

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 é tão superior à inferência de deslocamento de bicicleta como as métricas nos levam a crer.

	A pé	Bicicleta
<i>Accuracy</i>	0,438	0,438
<i>Precision</i>	0,930	0
<i>Recall</i>	0,453	0
<i>F-Measure</i>	0,610	-

Tabela 15 – *Accuracy*, *precision*, *recall* e *F-Measure* para detecção classe de modo de transporte A pé e Bicicleta com o classificador *Bayesian Net*

	Caminhando	Bicicleta
<i>Accuracy</i>	0,422	0,422
<i>Precision</i>	0,928	0
<i>Recall</i>	0,436	0
<i>F-Measure</i>	0,594	-

Tabela 16 – *Accuracy*, *precision*, *recall* e *F-Measure* para detecção classe de modo de transporte A pé e Bicicleta com o classificador *Decision Table*

## 6 Conclusão

Neste capítulo, são descritas as principais conclusões obtidas através desta pesquisa e os principais pontos de extensão do trabalho realizado.

### 6.1 Considerações sobre a classificação de aplicações de Sensoriamento Móvel para Cidades Inteligentes

Através do levantamento realizado foi possível concluir que a grande maioria das aplicações encontradas neste estudo está relacionada aos componentes Smart Living e Smart Mobility no que diz respeito a cidades inteligentes (Smart Cities) (GIFFINGER; PICHLER-MILANOVIĆ, 2007).

Na perspectiva de sensoriamento, quase todas as aplicações trabalham na escala de sensoriamento comunitário e utilizam dados de localização, principalmente através do sensor GPS. Em relação ao paradigma, há um equilíbrio entre aplicações de sensoriamento participativo e oportunista, com uma minoria utilizando os dois paradigmas ao mesmo tempo.

Já no que diz respeito à crowdsourcing a maior parte das aplicações não focou em um público específico e disponibilizou algum tipo de funcionalidade em uma extensão web. Além disso, a maioria esmagadora das aplicações não utilizou mecanismos de incentivo para atrair contribuições sendo em uma parcela significativa delas o serviço prestado pela aplicação a principal motivação para a participação dos usuários.

Outro aspecto observado adicionalmente é que a maior parte das aplicações foi implementada na plataforma Android ou iOS e uma grande parte delas foi implementada em ambas.

### 6.2 Considerações sobre a Implementação da Aplicação CityTracks na plataforma Android

Através dos testes realizados, foi possível observar o desempenho da aplicação CityTracks para inferência do modo de transporte utilizado por usuários de dispositivos móveis em tempo real, com um total de 1338 chunks coletados por 18 usuários de *smartphones* Android voluntários no Rio de Janeiro.

No que diz respeito ao desempenho dos algoritmos de aprendizado de máquina, foi possível observar um grande potencial das técnicas de *Multilayer Perceptron*, *Support*

*Vector Machine*, *Decision Table* e *Bayesian Net* para detecção do modo de transporte em tempo real. No entanto, a ausência de dados de treinamento para o deslocamento de carro e moto não permitiu avaliar completamente o desempenho da ferramenta implementada, sendo necessária a realização de novos testes de campo, com dados de treinamento para todas as classes de modo de transporte suportadas pela aplicação.

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, uma sugestão de trabalho futuro seria um estudo mais profundo sobre o consumo de recursos dos dispositivos pela aplicação CityTracks e a implementação de técnicas de otimização para tornar a aplicação mais leve e energeticamente eficiente.

## 6.3 Trabalhos futuros

Trabalhos futuros envolvem um levantamento estruturado de estudos e aplicações de sensoriamento por dispositivos móveis para cidades inteligentes utilizando técnicas de *crowdsourcing* e uma análise mais aprofundada de aplicações sensíveis a contexto baseadas na inferência de modos de transporte em tempo real, alcançada através da aplicação CityTracks.

Outro trabalho possível, é a realização de um novo teste de campo da aplicação CityTracks, com mais usuários e com dados de treinamento em maior quantidade e melhor distribuídos por todas as classes de modo de transporte suportadas pela aplicação.

Além disso, conforme mencionado anteriormente, faz-se necessária uma análise do consumo de recursos gerado pela utilização da aplicação CityTracks e a implementação de técnicas de otimização para torná-la mais eficiente.

Por fim, a implementação da transferência automática dos traces coletados pelos usuários para um servidor na nuvem e a implementação de melhorias no algoritmo de mineração de dados utilizado para inferência, através de modificações na parametrização das técnicas de aprendizado de máquina utilizadas.

# Referências

- ACTIVITY: Android API Reference. 2017.  
<https://developer.android.com/reference/android/app/Activity.html>. Acessado em: 20/01/2017. 39
- ALBINO, V.; BERARDI, U.; DANGELICO, R. M. Smart cities: Definitions, dimensions, performance, and initiatives. *Journal of Urban Technology*, v. 22, n. 1, p. 3–21, 2015. Disponível em: <<http://dx.doi.org/10.1080/10630732.2014.942092>>. 11, 12, 13, 14
- ANASTASI, G. et al. Urban and social sensing for sustainable mobility in smart cities. In: IEEE. *Sustainable Internet and ICT for Sustainability (SustainIT)*, 2013. [S.l.], 2013. p. 1–4. 17
- ANTONIOU, A. et al. Using future internet infrastructure and smartphones for mobility trace acquisition and social interactions monitoring. In: SPRINGER. *The Future Internet Assembly*. [S.l.], 2012. p. 117–129. 22
- APPLE iTunes. 2017. <https://www.apple.com/itunes/>. Acessado em: 20/01/2017. 27
- BAIER, P.; DURR, F.; ROTHERMEL, K. Psense: Reducing energy consumption in public sensing systems. In: IEEE. *2012 IEEE 26th International Conference on Advanced Information Networking and Applications*. [S.l.], 2012. p. 136–143. 22
- BECKER, B. G. Visualizing decision table classifiers. In: IEEE. *Information Visualization, 1998. Proceedings. IEEE Symposium on*. [S.l.], 1998. p. 102–105. 7, 34, 35
- BHATTACHARYA, S. et al. Robust and energy-efficient trajectory tracking for mobile devices. *IEEE Transactions on Mobile Computing*, v. 14, n. 2, p. 430–443, Feb 2015. ISSN 1536-1233. 23
- BLANCO, R. et al. Repeatable and reliable search system evaluation using crowdsourcing. In: ACM. *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. [S.l.], 2011. p. 923–932. 24
- BRABHAM, D. C. Crowdsourcing as a model for problem solving an introduction and cases. *Convergence: the international journal of research into new media technologies*, Sage publications, v. 14, n. 1, p. 75–90, 2008. 24
- BRABHAM, D. C. Crowdsourcing the public participation process for planning projects. *Planning Theory*, v. 8, n. 3, p. 242–262, 2009. Disponível em: <<http://plt.sagepub.com/content/8/3/242.abstract>>. 24
- BURKE, J. A. et al. Participatory sensing. *Center for Embedded Network Sensing*, 2006. 20
- CABERO, J. M. et al. Reliability of bluetooth-based connectivity traces for the characterization of human interaction. *Ad Hoc Networks*, Elsevier, v. 24, p. 135–146, 2015. 22

- CALABRESE, F.; FERRARI, L.; BLONDEL, V. D. Urban sensing using mobile phone network data: A survey of research. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 47, n. 2, p. 25:1–25:20, nov. 2014. ISSN 0360-0300. Disponível em: <<http://doi.acm.org/10.1145/2655691>>. 9
- CAMPBELL, A. T. et al. People-centric urban sensing. In: ACM. *Proceedings of the 2nd annual international workshop on Wireless internet*. [S.l.], 2006. p. 18. 22
- CARAGLIU, A.; BO, C. D.; NIJKAMP, P. Smart cities in europe. *Journal of urban technology*, Taylor & Francis, v. 18, n. 2, p. 65–82, 2011. 12
- CHATZIMILIOUDIS, G. et al. Crowdsourcing with smartphones. *IEEE Internet Computing*, IEEE, v. 16, n. 5, p. 36–44, 2012. 25, 28
- CHON, Y. et al. Automatically characterizing places with opportunistic crowdsensing using smartphones. In: ACM. *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. [S.l.], 2012. p. 481–490. 22
- CIDADES Sustentáveis. 2017. <https://cidadesustentaveljornalismo.wordpress.com/>. Acessado em: 17/01/2017. 8, 14
- COLLOBERT, R.; BENGIO, S. Links between perceptrons, mlps and svms. In: ACM. *Proceedings of the twenty-first international conference on Machine learning*. [S.l.], 2004. p. 23. 33
- CROCKFORD, D. The application/json media type for javascript object notation (json). 2006. 40
- DOCUMENTAÇÃO FusedLocationAPI Android. 2017. <https://developers.google.com/android/reference/com/google/android/gms/location>. Acessado em: 15/01/2017. 38
- DORAN, D.; GOKHALE, S.; DAGNINO, A. Human sensing for smart cities. In: ACM. *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. [S.l.], 2013. p. 1323–1330. 17
- ESTELLÉS-AROLAS, E.; GONZÁLEZ-LADRÓN-DE-GUEVARA, F. Towards an integrated crowdsourcing definition. *Journal of Information science*, Sage Publications, v. 38, n. 2, p. 189–200, 2012. 24, 28
- FANG, S.-H. et al. Transportation modes classification using sensors on smartphones. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 16, n. 8, p. 1324, 2016. 8, 34
- FOURSQUARE. 2016. <https://foursquare.com/>. Acessado em 07/2016. 24
- FRANK, R. et al. Energy-efficient rate-adaptive passive traffic sensing using smartphones. In: IEEE. *Ad Hoc Networking Workshop (MED-HOC-NET), 2013 12th Annual Mediterranean*. [S.l.], 2013. p. 67–74. 22
- GANTI, R. K.; YE, F.; LEI, H. Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine*, Citeseer, v. 49, n. 11, p. 32–39, 2011. 21
- GAURA, E. I. et al. Edge mining the internet of things. *IEEE Sensors Journal*, IEEE, v. 13, n. 10, p. 3816–3825, 2013. 17



- GIFFINGER, R.; PICHLER-MILANOVIĆ, N. *Smart cities: Ranking of European medium-sized cities*. [S.l.]: Centre of Regional Science, Vienna University of Technology, 2007. 14, 27, 52
- GOOGLE JSON: A Java serialization/deserialization library that can convert Java Objects into JSON and back. 2017. <https://github.com/google/gson>. Acessado em: 20/01/2017. 40
- GOOGLE Play. 2017. <https://play.google.com/store>. Acessado em: 20/01/2017. 27
- GOOGLE Scholar. 2017. <https://scholar.google.com.br/>. Acessado em: 20/01/2017. 27
- GOOGLE Search. 2017. <https://www.google.com/>. Acessado em: 15/01/2017. 27
- HALL, M. et al. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, ACM, v. 11, n. 1, p. 10–18, 2009. 38, 40
- HANCKE, G. P.; JR, G. P. H. et al. The role of advanced sensing in smart cities. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 13, n. 1, p. 393–425, 2012. 15
- HARRISON, C. et al. Foundations for smarter cities. *IBM Journal of Research and Development*, IBM, v. 54, n. 4, p. 1–16, 2010. 12
- HE, J. et al. Towards mobile sensor-aware crowdsourcing: Architecture, opportunities and challenges. In: SPRINGER. *International Conference on Database Systems for Advanced Applications*. [S.l.], 2014. p. 403–412. 25
- HETMANK, L. Components and functions of crowdsourcing systems-a systematic literature review. *Wirtschaftsinformatik*, v. 4, 2013. 25
- HOSEINI-TABATABAEI, S. A.; GLUHAK, A.; TAFAZOLLI, R. A survey on smartphone-based systems for opportunistic user context recognition. *ACM Computing Surveys (CSUR)*, ACM, v. 45, n. 3, p. 27, 2013. 22
- HOSSAIN, M.; KAURANEN, I. Crowdsourcing: a comprehensive literature review. *Strategic Outsourcing: An International Journal*, v. 8, n. 1, p. 2–22, 2015. Disponível em: <<http://dx.doi.org/10.1108/SO-12-2014-0029>>. 25
- HOWE, J. The rise of crowdsourcing. *Wired magazine*, v. 14, n. 6, p. 1–4, 2006. 24
- HU, S. et al. Smartroad: Smartphone-based crowd sensing for traffic regulator detection and identification. *ACM Transactions on Sensor Networks (TOSN)*, ACM, v. 11, n. 4, p. 55, 2015. 21
- HUNT, J. Gang of four design patterns. In: *Scala Design Patterns*. [S.l.]: Springer, 2013. p. 135–136. 39
- KANHERE, S. S. Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces. In: IEEE. *2011 IEEE 12th International Conference on Mobile Data Management*. [S.l.], 2011. v. 2, p. 3–6. 24
- LANE, N. D. et al. Piggyback crowdsensing (pcs): energy efficient crowdsourcing of mobile sensor data by exploiting smartphone app opportunities. In: ACM. *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*. [S.l.], 2013. p. 7. 25

- LANE, N. D. et al. A survey of mobile phone sensing. *IEEE Communications Magazine*, v. 48, n. 9, p. 140–150, Sept 2010. ISSN 0163-6804. 9, 18, 19, 20, 28
- LARI, Z. A.; GOLROO, A. 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*. [S.l.: s.n.], 2015. p. 11–15. 36
- LARMAN, C. *Applying UML and Patterns: An Introduction to Object Oriented Analysis and Design and Iterative Development*. [S.l.]: Pearson Education India, 2012. 8, 42
- MAN, Y.; NGAI, E. C.-H.; LIU, Y. A glimpse of energy-efficient location-triggered mobile application design and implementation in data collection scenarios. In: *IEEE. Mobile Ad-hoc and Sensor Networks (MSN), 2012 Eighth International Conference on*. [S.l.], 2012. p. 108–115. 22
- MYLONAS, G.; THEODORIDIS, E.; MUÑOZ, L. Integrating smartphones into the smartantander infrastructure. *IEEE Internet Computing*, IEEE, v. 19, n. 2, p. 48–56, 2015. 21
- PEARL, J. Bayesian networks. *Department of Statistics, UCLA*, 2011. 8, 35
- PEREIRA, F. de O. Mobile geogames-novas interfaces para o uso de dispositivos moveis na coleta de dados de mobilidade urbana em pequenas areas. 21
- PERERA, C. et al. Sensing as a service model for smart cities supported by internet of things. *Transactions on Emerging Telecommunications Technologies*, Wiley Online Library, v. 25, n. 1, p. 81–93, 2014. 16
- PIDCOCK, S. et al. Notisense: An urban sensing notification system to improve bystander privacy. In: *Proceedings of the 2nd International Workshop on Sensing Applications on Mobile Phones (PhoneSense)*. [S.l.: s.n.], 2011. p. 1–5. 22
- PLATT, J. et al. Sequential minimal optimization: A fast algorithm for training support vector machines. technical report msr-tr-98-14, Microsoft Research, 1998. 33
- POWERS, D. M. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. Bioinfo Publications, 2011. 48
- PRIYANTHA, B.; LYMBEROPOULOS, D.; LIU, J. Littlerock: Enabling energy-efficient continuous sensing on mobile phones. *IEEE Pervasive Computing*, IEEE, v. 10, n. 2, p. 12–15, 2011. 22
- QUINN, A. J.; BEDERSON, B. B. Human computation: a survey and taxonomy of a growing field. In: *ACM. Proceedings of the SIGCHI conference on human factors in computing systems*. [S.l.], 2011. p. 1403–1412. 24, 28
- QUINTELLA, C.; CAMPOS, C.; ANDRADE, L. Como inferir o meio de transporte das pessoas em cenários urbanos a partir de smartphones. In: *SBC. Simpósio Brasileiro de Computação Ubíqua e Pervasiva (SBCUP 2016)*. [S.l.], 2016. 8, 37
- QUINTELLA, C. A. de M. S. *Aplicação de aprendizado de máquina para inferência de modo de transporte em traces de smartphones*. 2013. 10, 36, 37, 38, 46, 48

- ROLIM, C. O. et al. Urbosenti: Uma arquitetura ubíqua orientada a serviços para o sensoriamento urbano. 17
- SAGL, G.; RESCH, B.; BLASCHKE, T. Contextual sensing: Integrating contextual information with human and technical geo-sensor information for smart cities. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 15, n. 7, p. 17013–17035, 2015. 17
- SCHILDT, S. et al. User study on the feasibility of incentive systems for smartphone-based dtms in smart cities. In: ACM. *Proceedings of the 2014 ACM international workshop on Wireless and mobile technologies for smart cities*. [S.l.], 2014. p. 67–76. 17
- SERIVCES: Android API Reference. 2017.  
<https://developer.android.com/guide/components/services.html>. Acessado em: 20/01/2017. 39
- SIERRA, K.; BATES, B.; SIEBEN, J. *SCJP Sun Certified Programmer for Java 5 Study Guide (Exam 310-055)*. [S.l.]: McGraw-Hill, Inc., 2005. 39
- SILVA, T. H.; LOUREIRO, A. A. Computação urbana: Técnicas para o estudo de sociedades com redes de sensoriamento participativo. 21
- SILVA, T. H. et al. Redes de sensoriamento participativo: Desafios e oportunidades. 21
- SWENDEMAN, D. et al. Reliability and validity of daily self-monitoring by smartphone application for health-related quality-of-life, antiretroviral adherence, substance use, and sexual behaviors among people living with hiv. *AIDS and Behavior*, Springer, v. 19, n. 2, p. 330–340, 2015. 21
- THINK Listen Learn. 2017. <http://thinklistenlearn.com/page/5/>. Acessado em: 15/0/2017. 8, 23
- TOTAL Commander File Manager. 2017.  
<https://play.google.com/store/apps/details?id=com.ghisler.android.TotalCommander>. Acessado em: 14/01/2017. 41
- VANOLO, A. Smartmentality: The smart city as disciplinary strategy. *Urban Studies*, Sage Publications, p. 0042098013494427, 2013. 17
- WAZE - Free Community-based Mapping, Traffic& Navigation App. 2016.  
<https://www.waze.com/>. Acessado em 07/2016. 24
- WEKA Wiki: Use WEKA in your Java code. 2017.  
<https://weka.wikispaces.com/Use+WEKA+in+your+Java+code>. Acessado em: 20/01/2017. 38
- WIKIPEDIA: The Free Encyclopedia. 2017. <https://www.wikipedia.org/>. Acessado em: 20/01/2017. 24
- WU, L.; YANG, B.; JING, P. Travel mode detection based on gps raw data collected by smartphones: a systematic review of the existing methodologies. *Information*, Multidisciplinary Digital Publishing Institute, v. 7, n. 4, p. 67, 2016. 36
- WU, X. et al. Top 10 algorithms in data mining. *Knowledge and information systems*, Springer, v. 14, n. 1, p. 1–37, 2008. 33

- XIAO, G.; ZHICAI, J.; JINGXIN, G. Inferring trip ends from gps data based on smartphones in shanghai. In: *Transportation Research Board 94th Annual Meeting*. [S.l.: s.n.], 2015. 36
- YANG, D. et al. Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing. In: *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*. New York, NY, USA: ACM, 2012. (Mobicom '12), p. 173–184. ISBN 978-1-4503-1159-5. Disponível em: <<http://doi.acm.org/10.1145/2348543.2348567>>. 25
- YANG, F.; YAO, Z.; JIN, P. J. 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*, Transportation Research Board of the National Academies, n. 2526, p. 90–98, 2015. 35
- YUEN, M.-C.; KING, I.; LEUNG, K.-S. A survey of crowdsourcing systems. In: IEEE. *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third International Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*. [S.l.], 2011. p. 766–773. 24
- ZHAO, L. et al. Privacy sensitive resource access monitoring for android systems. In: IEEE. *2015 24th International Conference on Computer Communication and Networks (ICCCN)*. [S.l.], 2015. p. 1–6. 23
- ZHAO, Y.; ZHU, Q. Evaluation on crowdsourcing research: Current status and future direction. *Information Systems Frontiers*, v. 16, n. 3, p. 417–434, 2014. ISSN 1572-9419. Disponível em: <<http://dx.doi.org/10.1007/s10796-012-9350-4>>. 24, 28

# APÊNDICE A – Classe de Inferência de Modo de Transporte - Classificador.java

```

package unirio.citytracks.utils.inferencia;

import android.content.*;
import android.support.annotation.*;

import java.util.*;

import unirio.citytracks.modelo.entidade.*;
import weka.classifiers.bayes.*;
import weka.classifiers.functions.*;
import weka.classifiers.rules.*;
import weka.core.*;
import weka.core.converters.*;

public class Classificador {

    private final String TRAIN_REDE_NEURAL_FILENAME = "chunksTreinamentoRedeNeural.arff";
    private final String TRAIN_SMO_FILENAME = "chunksTreinamentoSMO.arff";
    private final String TRAIN_MOTORIZADO_FILENAME = "chunksTreinamentoMotorizado.arff";
    private final String TRAIN_NAO_MOTORIZADO_FILENAME = "chunksTreinamentoNaoMotorizado.arff";

    private MultilayerPerceptron multilayerPerceptron;
    private SMO smo;
    private DecisionTable decisionTableMotorizado;
    private DecisionTable decisionTableNaoMotorizado;
    private BayesNet bayesNet;
    private Instances trainRedeNeural;
    private Instances trainSMO;
    private Instances trainMotorizado;
    private Instances trainNaoMotorizado;

    public Classificador() {
        multilayerPerceptron = new MultilayerPerceptron();
        smo = new SMO();
        decisionTableMotorizado = new DecisionTable();
        decisionTableNaoMotorizado = new DecisionTable();
        bayesNet = new BayesNet();
    }

    public Chunk classificar(Chunk chunk) throws Exception {

        chunk.setModoDeTransporteRedeNeural(ModosDeTransporte.valueOf(classificarPorRedeNeural(chunk)));
        chunk.setModoDeTransporteSMO(TiposDeModosDeTransporte.valueOf(classificarPorSMO(chunk)));

        if(chunk.getModoDeTransporteSMO().toString().equals(TiposDeModosDeTransporte.MOTORIZADO.
            toString())){
            chunk.setModosDeTransporteMotorizadosDecisionTable(ModosDeTransporteMotorizados.
                valueOf(classificarPorDecisionTableMotorizado(chunk)));
        } else if (chunk.getModoDeTransporteSMO().toString().equals(TiposDeModosDeTransporte.
            NAO_MOTORIZADO.toString())){

```

```

chunk.setModosDeTransporteNaoMotorizadosBayesNet(ModosDeTransporteNaoMotorizados.
valueOf(classificarPorBayesNetNaoMotorizado(chunk)));
chunk.setModosDeTransporteNaoMotorizadosDecisionTable(ModosDeTransporteNaoMotorizados.
valueOf(classificarPorDecisionTableNaoMotorizado(chunk)));
}

return chunk;

}

public String classificarPorRedeNeural(Chunk chunk) throws Exception {

String[] classes = new String[ModosDeTransporte.values().length];

int i = 0;
for (ModosDeTransporte m : ModosDeTransporte.values()) {
classes[i++] = m.toString();
}

Instances data = getInstances(chunk, classes);

double clsLabel = multilayerPerceptron.classifyInstance(data.firstInstance());

return trainRedeNeural.classAttribute().value((int) clsLabel);

}

public String classificarPorSMO(Chunk chunk) throws Exception {

String[] classes = new String[TiposDeModosDeTransporte.values().length];

int i = 0;
for (TiposDeModosDeTransporte m : TiposDeModosDeTransporte.values()) {
classes[i++] = m.toString();
}

Instances data = getInstances(chunk, classes);

double clsLabel = smo.classifyInstance(data.firstInstance());

return trainSMO.classAttribute().value((int) clsLabel);

}

public String classificarPorDecisionTableMotorizado(Chunk chunk) throws Exception {

String[] classes = new String[ModosDeTransporteMotorizados.values().length];

int i = 0;
for (ModosDeTransporteMotorizados m : ModosDeTransporteMotorizados.values()) {
classes[i++] = m.toString();
}

Instances data = getInstances(chunk, classes);

double clsLabel = decisionTableMotorizado.classifyInstance(data.firstInstance());

return trainMotorizado.classAttribute().value((int) clsLabel);

}

```

```

public String classificarPorDecisionTableNaoMotorizado(Chunk chunk) throws Exception {

String[] classes = new String[ModosDeTransporteNaoMotorizados.values().length];

int i = 0;
for (ModosDeTransporteNaoMotorizados m : ModosDeTransporteNaoMotorizados.values()) {
classes[i++] = m.toString();
}

Instances data = getInstances(chunk, classes);

double clsLabel = decisionTableNaoMotorizado.classifyInstance(data.firstInstance());

return trainNaoMotorizado.classAttribute().value((int) clsLabel);

}

public String classificarPorBayesNetNaoMotorizado(Chunk chunk) throws Exception {

String[] classes = new String[ModosDeTransporteNaoMotorizados.values().length];

int i = 0;
for (ModosDeTransporteNaoMotorizados m : ModosDeTransporteNaoMotorizados.values()) {
classes[i++] = m.toString();
}

Instances data = getInstances(chunk, classes);

double clsLabel = bayesNet.classifyInstance(data.firstInstance());

return trainNaoMotorizado.classAttribute().value((int) clsLabel);

}

@NonNull
public Instances getInstances(Chunk chunk, String[] classes) {
ArrayList<Attribute> attributeList = new ArrayList<Attribute>(2);

Attribute velocidadeMaxima = new Attribute("velocidade_maxima");
Attribute aceleracaoMaxima = new Attribute("aceleracao_maxima");
Attribute numeroDeMudancasDeDirecao = new Attribute("mudancas_de_direcao");

ArrayList<String> classVal = new ArrayList<String>();
for (String classe : classes) {
classVal.add(classe);
}

Attribute modoDeTransporte = new Attribute("modo_de_transporte", classVal);

attributeList.add(velocidadeMaxima);
attributeList.add(aceleracaoMaxima);
attributeList.add(numeroDeMudancasDeDirecao);
attributeList.add(modoDeTransporte);

Instances data = new Instances("chunks", attributeList, 0);

Instance predictInstance = new DenseInstance(data.numAttributes());
predictInstance.setValue(velocidadeMaxima, chunk.getVelocidadeMaxima());
predictInstance.setValue(aceleracaoMaxima, chunk.getAceleracaoMaxima());

```

```

predictInstance.setValue(numeroDeMudancasDeDirecao, chunk.getNumeroDeMudancasDeDirecao());

data.add(predictInstance);
data.setClassIndex(data.numAttributes()-1);
return data;
}

public void treinarAlgoritmos(Context contexto) throws Exception {

    treinarRedeNeural(contexto);

    treinarSMO(contexto);

    treinarMotorizado(contexto);

    treinarNaoMotorizado(contexto);

}

public void treinarNaoMotorizado(Context contexto) throws Exception {

    ConverterUtils.DataSource source = new ConverterUtils.DataSource(contexto.getAssets().
    open(TRAIN_NAO_MOTORIZADO_FILENAME));
    trainNaoMotorizado = source.getDataSet();
    trainNaoMotorizado.setClassIndex(trainNaoMotorizado.numAttributes() - 1);

    //treinar decision table
    decisionTableNaoMotorizado.buildClassifier(trainNaoMotorizado);

    //treinar bayes net
    bayesNet.buildClassifier(trainNaoMotorizado);
}

public void treinarMotorizado(Context contexto) throws Exception {
    ConverterUtils.DataSource source = new ConverterUtils.DataSource(contexto.getAssets().
    open(TRAIN_MOTORIZADO_FILENAME));
    trainMotorizado = source.getDataSet();
    trainMotorizado.setClassIndex(trainMotorizado.numAttributes() - 1);

    //treinar decision table
    decisionTableMotorizado.buildClassifier(trainMotorizado);
}

public void treinarSMO(Context contexto) throws Exception {
    ConverterUtils.DataSource source = new ConverterUtils.DataSource(contexto.getAssets().
    open(TRAIN_SMO_FILENAME));
    trainSMO = source.getDataSet();
    trainSMO.setClassIndex(trainSMO.numAttributes() - 1);

    //treinar smo
    smo.buildClassifier(trainSMO);
}

public void treinarRedeNeural(Context contexto) throws Exception {
    ConverterUtils.DataSource source = new ConverterUtils.DataSource(contexto.getAssets().
    open(TRAIN_REDE_NEURAL_FILENAME));
    trainRedeNeural = source.getDataSet();
    trainRedeNeural.setClassIndex(trainRedeNeural.numAttributes() - 1);

    //treinar multilayer perceptron

```



---

```
multilayerPerceptron = new MultilayerPerceptron();
multilayerPerceptron.setLearningRate(0.1);
multilayerPerceptron.setMomentum(0.2);
multilayerPerceptron.setTrainingTime(2000);
multilayerPerceptron.setHiddenLayers("3");
multilayerPerceptron.buildClassifier(trainRedeNeural);
}

}
```