FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Identificação do modo de transporte através de informação GPS

Vitor João Ferreira Semeano Figueira

DISSERTAÇÃO

Mestrado Integrado em Engenharia Informática e Computação

Orientador: Doutor Daniel Cardoso de Moura

16 de Julho de 2014

Identificação do modo de transporte através de informação GPS

Vitor João Ferreira Semeano Figueira

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Professor Doutor Pedro Alexandre Guimarães Lobo Ferreira Souto (Universidade do Porto, Faculdade de Engenharia, Departamento de Engenharia Informática)

Arguente: Professor Doutor Pedro Henriques Abreu (Universidade de Coimbra, Faculdade de Engenharia, Departamento de Engenharia Informática)

Vogal: Investigador Doutor Daniel Cardoso de Moura (Universidade do Porto, Faculdade de Engenharia, Departamento de Engenharia Eletrotécnica e de Computadores)

16 de Julho de 2014

Resumo

Durante o dia a dia é necessário transitar entre vários sítios importantes para o cidadão comum. A transição envolve um modo de locomoção que permite a deslocação de um local para o outro; por exemplo, de casa para o trabalho, do trabalho para o centro comercial. Assim, deparamo-nos com um conjunto de atividades, cada uma delas caracterizada por um modo de locomoção. A capacidade de recolha de modos de locomoção, permite a criação de conhecimento para diversas aplicações. Esta dissertação procura contribuir para a resolução do problema de inferência de modos de locomoção, em tempo real, em dispositivos móveis, sem que o utilizador tenha de intervir. Espera-se inferir os principais modos de locomoção usados, tais como modo, andar a pé, carro, táxi e autocarro ou até mesmo identificar se o utilizador está em modo estacionário. Através da inferência de modos de locomoção, soluções futuras podem recorrer a este tipo de dados para ajudar o utilizador em decisões do dia a dia, como por exemplo, decidir qual o melhor caminho a tomar tendo em conta o modo de locomoção atual do utilizador.

Sugere-se a análise de várias estratégias capazes de resolver este problema, através de um conjunto finito de classificadores, previamente selecionados, tendo em conta a literatura existente e características do problema. Cada uma das estratégias é composta por várias camadas de processamento. Uma das mais importantes é a criação de características de alto nível e a classificação das instâncias geradas. Nesta dissertação é proposto um conjunto de características, baseado nas acelerações e noutros fatores, como *heading*, que mostraram ter um impacto positivo nos resultados obtidos. As experiências foram realizadas com dois *datasets*, o Geolife, *dataset* público desenvolvido pela Microsoft Research Asia, e o SenseMyCity, desenvolvido nesta dissertação com a aplicação SenseMyCity do Instituto de Telecomunicações.

Foi definido um conjunto de experiências para avaliar as melhores opções disponíveis e escolher as que evidenciassem melhores resultados. As principais experiências foram a análise da seleção de características, para encontrar vantagens da utilização de subconjuntos com um menor número de características, e a análise da melhor estratégia de classificação a usar. Após este trabalho de investigação e experimentação encontrou-se uma solução. A solução final obteve uma exatidão de 80.46% para o *dataset* do SenseMyCity e 63.78% para o *dataset* do Geolife, utilizando o classificador *Random Forest* (RF). Com base nesta solução, foi desenvolvida uma aplicação com o intuito de não interferir com a atuação do utilizador noutras tarefas, e que permite obter um conjunto de inferências em tempo real. A aplicação está pronta para ser integrada como um módulo na aplicação SenseMyCity, para estudo de padrões de mobilidade.

Abstract

Nowadays, the common citizen need to travel between many places. This type of transaction involves a transportation mode that allows travelling from one point to another, such as travelling from home to work or from work to mall centre, among others. Thus, we have a set of activities, where each one of them represents a transportation mode. Collection of transportation modes, allows the creation of knowledge. This dissertations seek to contribute on classifying transportation modes in real time with a smartphone, without the need for user intervention. We hope to infer a set of transportations modes, such as stationary, walk, car, taxi and bus. By inferring transportation modes, we can provide data that can be used for future problems, helping users in their daily routine, like helping the user to get the best route alternative to his destination, based on is current transportation mode.

We proposed an analysis with multiple strategies capable of resolving this problem, using a set of classifiers, previously selected, taking into account the existing literature and features of this problem. Each one of these strategies is composed by a set of layers, such as high level features generation, and classification of these features. Also we proposed a set of features, based on accelerations and other features, like heading, showing a positive contribute to the classification. For these experiences we used 2 datasets, namely Geolife, a public dataset developed by Microsoft Research Asia, and SenseMyCity, developed in this dissertation, using the application SenseMyCity from the Institute of Technologies.

A set of experiences was defined so we can evaluate the best options available with the best results. The most important experiences were the analysis of feature selection, so we can find advantages on removing a set of the worst features from the classification, and analysis of the best classification strategy to use. The final solution achieved a accuracy of 80.46% for the SenseMyCity dataset and 63.78% for the Geolife Dataset, using the classifier Random Forest (RF). With this solution, an application has been developed, aimed for inferring transportations modes, that don't require any user interaction during this process. This application is ready to be integrated as a module to the SenseMyCity application, for study of mobility patterns.

Agradecimentos

Primeiramente tenho a agradecer ao meu orientador, Doutor Daniel Moura, por tudo o que me permitiu atingir, nomeadamente, o conhecimento transmitido, o ambiente de discussão em direção à melhor solução e toda a flexibilidade fornecida para eu próprio desenvolver alternativas e novas ideias. Também fico extremamente grato por toda a sua paciência e esforço dedicado, pois sem o feedback recebido nunca teria conseguido terminar esta dissertação.

Agradeço à minha família pelo apoio que me deu ao longo deste último semestre, especialmente aos meus pais que sempre me apoiaram e se esforçaram para que tudo isto fosse possível. Aos meus amigos e colegas da FUEP só tenho a agradecer pela ajuda quando mais necessitava.

É aos meus pais que dedico todos os resultados alcançados ao longo destes últimos 5 anos, porque eles são a minha fonte de auto motivação e coragem, o ponto de apoio que me permitiu ultrapassar os obstáculos com os quais me deparei. Sem os meus pais não teria chegado até a este patamar. Fico extremamente agradecido por me permitirem continuar a subir na vida.

Vitor Figueira

"Success is not final, failure is not fatal:
it is the courage to continue that counts."

Winston Churchill

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

ADB	Android Debug Bridge
API	Application Programming Interface
ARFF	Attribute-Relation File Format
CFS	Correlation Based Features Selection
DFT	Discrete Fourier Transform
DHMM	Discrete Hidden Markov Model
DT	Decision Tree
FEUP	Faculdade de Engenharia da Universidade do Porto
FFT	Fast Fourier Transform
GIS	Geographical Information System
GPS	Global Positioning System
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
HCR	Heading Change Rate
HDOP	Horizontal Dilution of Precision
HMM	Hidden Markov Model
IBK	K-nearest neighbours classifier
IDE	Integrated Development Environment
IMU	Inertial Measurement Unit
IT	Instituto das Telecomunicações
ITS	Inteligent Transportation Systems
MSB	Multi-Modal Sensor Board
MSP	Mobile Sensing Platform
RF	Random Forests
NN	Neural network
RT	Random Tree
SR	Stop Rate
SVM	Support Vector Machines
WEKA	Waikato Environment for Knowledge Analysis
WWW	*World Wide Web*
VCR	Velocity Change Rate

Capítulo 1

Introdução

1.1 Enquadramento

O cidadão comum tem o seu dia a dia constituído por várias atividades. Cada uma delas é realizada de forma a cumprir um objetivo, que vai conduzir à produção de novas atividades no desenrolar do seu dia a dia. Muitas destas atividades requerem mobilidade de um ponto atual para um ponto destino. A estas atividades podemos atribuir um tipo de transporte, caraterizado por um conjunto de fatores diferenciadores, podendo existir distinção entre estas mobilidades. Várias atividades deste tipo podem constituir situações mais complexas e mais usuais, como ir de casa para o local de trabalho, como ir fazer compras ao centro comercial, entre muitas outras. Uma sequência pode ser observável em cada uma destas atividades, como, por exemplo, ir de casa para o local de trabalho pode exigir ir a pé até à paragem X, apanhar o autocarro Y, sair na paragem Z e andar até ao ponto Q. Este conjunto de atividades é conhecido como transporte multimodal, a capacidade de se deslocar um ponto origem para um ponto destino, através de vários tipos de modos de locomoção.

A necessidade de escolher o melhor caminho para um certo destino pode trazer várias alternativas que contêm diversos pesos em cada mobilidade sugerida. Atualmente, várias soluções comerciais conseguem reunir este tipo de dados e calcular o melhor caminho a tomar, mas só se o utilizador fornecer um ponto origem e um ponto destino ao dispositivo. Caso o utilizador não quisesse introduzir esse tipo de dados numa aplicação móvel, a tarefa seria praticamente impossível de ser realizada com as aplicações atualmente disponíveis.

Felizmente, através do avanço cientifico, vários algoritmos na área de classificação podem ser usados para inferir qualquer tipo de dados. Mas, antes de poder inferir qual o melhor caminho a escolher, é necessário fragmentar essa atividade em várias sub-atividades. Cada uma dessas sub-atividades representa um modo de transporte ou locomoção. Neste contexto é introduzido o problema de inferência de modos de locomoção, a capacidade de inferir os modos de locomoção que um cidadão comum possa ter durante o seu dia a dia, de forma a que os dados inferidos

possam ser usados para inferências de mais alto nível, como, por exemplo, melhorar o caminho entre dois pontos geográficos [LPFK07]. Esta é uma de várias aplicações existentes que podem ser aplicadas usando este tipo de inferências. Exemplo de outro tipo de aplicação, passando pela área de segurança, é a inferência de dados mais alto nível para rastrear eventuais suspeitos.

1.2 Motivação e objetivos

Atualmente, através do avanço tecnológico, vários tipos de tecnologia encontram-se disponíveis e acessíveis ao cidadão comum, não sendo necessário a aquisição de dispositivos adicionais para funções especificas, como, por exemplo, recolha de determinado tipo de dados sensoriais, falar com outras pessoas em tempo real em ambiente de mobilidade, capacidade de jogar em mobilidade, realização de tarefas computacionais, entre muitos outros. Exemplo desse avanço é o *smartphone*, dispositivo móvel com grande conjunto de sensores e funcionalidades a preços acessíveis, conforme as funcionalidades oferecidas. Atualmente, é possível criar qualquer tipo de aplicações para *smartphones*, usando os recursos presentes no dispositivo. Esta agilidade permite a criação de aplicações complexas, que podem ser usadas no contexto de classificação. Uma vez essa possibilidade, podem ser criadas aplicações que tomem decisões sobre certas necessidades existentes, como, por exemplo, necessidades de mobilidade, laborais, pessoais, entre outras. Essas decisões podem substituir atividades que atualmente são exercidas pelo cidadão comum, facilitando as tarefas a ele associadas. A capacidade de inferir modos de transporte é crucial para a simplificação de tarefas, que podem ser feitas sem qualquer interação do cidadão comum.

No entanto, a criação de tais aplicações complexas requer análise cuidada, pois os seus requerimentos computacionais podem ser elevados, prejudicando componentes essenciais do dispositivo móvel, tais como, o tempo de bateria restante e o tempo de resposta do dispositivo móvel. Criação de aplicações que requerem o mínimo de recursos no que diz respeito a bateria e processamento contêm mais valor, trazendo vantagens para o cidadão comum, como um dispositivo móvel com maior tempo de bateria e maior fluidez.

Contudo, a criação de várias aplicações, que fazem tarefas classificativas específicas, pode induzir em confusão e desagrado o cidadão comum. Uma aplicação que consiga combinar vários classificadores de forma a trabalhar em conjunto, acrescenta muito valor na aplicação final, pois simplifica a obtenção de várias funcionalidades numa única aplicação, e diminui a quantidade de recursos necessários no dispositivo móvel. A aplicação SenseMyCity, desenvolvida pela Faculdade de Engenharia da Universidade do Porto (FEUP) e pelo Instituto das Telecomunicações (IT), no âmbito do projeto *Future Cities*, combina várias soluções numa única aplicação que tem como objetivo recolher dados, etiquetar automaticamente estes dados e efetuar classificações de forma a encontrar padrões de mobilidade nos utilizadores, tendo grande potencial na área de *Inteligent Transportation Systems* (ITS). A criação de uma aplicação que permita classificar os modos de locomoção como um serviço, preparado para ser integrado à aplicação SenseMyCity, cria uma aplicação de maior valor, podendo ajudar, em novas formas, os seus utilizadores a tomar decisões no seu dia a dia, bem como permitir à cidade compreender como os seus cidadãos se deslocam.

Uma condição *sine qua non* para construir uma solução com estas características é a definição do que se pretende alcançar, e para isso é necessário ter os **objetivos** bem definidos:

- **Desenvolvimento da solução**: é objetivo desta dissertação desenvolver um método computacional que permita inferir o modo de locomoção a partir de dados recolhidos em tempo real, tentando melhorar a precisão de inferência em relação ao estado da arte;
- **Utilização de *datasets***: é objetivo recorrer a *datasets* públicos e *datasets* fornecidos pelo projeto *Future Cities*, de forma a podermos testar o desempenho da solução com diferentes fontes de dados.
- **Minimização de dados necessários**: é objetivo minimizar a quantidade de dados recebidos pelo sensor GPS de forma a que estes possam ser usados para inferência de modos de locomoção como dados históricos e também sejam compatíveis com a maioria dos dispositivos móveis;
- **Implementação em dispositivo móvel**: finalmente, é objetivo implementar a solução desenvolvida em ambiente móvel, especificamente Android, como um serviço, ficando pronto para uma futura integração na aplicação *SenseMyCity*.

1.3 Metodologias

De forma a poder concretizar estes objetivos, foram recolhidos dados do meio ambiente e das ações do cidadão comum. Após a obtenção destes dados, filtrou-se os não válidos, que foram removidos, e depois os dados *raw* foram convertidos para dados de mais alto nível. Este passo carateriza-se pela extração de novas caraterísticas mais diferenciadoras, que permitem distinguir os vários modos de locomoção. Após a consecução destas caraterísticas alto nível, treinou-se um modelo fornecendo este conjunto de dados ao classificador, para poder testar novos dados adquiridos em tempo real e poder prever com uma boa eficácia o modo de locomoção. Antes de poder devolver um resultado final, os resultados submeteram-se a uma fase de refinamento, tentando adicionar mais robustez à previsão adquirida na primeira fase da classificação. Finalmente, o método foi avaliado com dados que não entraram no conjunto de treino e foram extraídas medidas de desempenho, tais como exatidão, precisão e sensibilidade.

1.4 Estrutura da dissertação

Para além da introdução, esta dissertação contém mais seis capítulos. No capítulo 2, é descrito o estado da arte que descreve as soluções atuais para previsão do modo de locomoção. No capítulo 3, são descritos os métodos usados em todos os componentes da solução sugerida. No capítulo 4 são descritos todos aspetos de implementação, como a *framework* usada para avaliar a solução fragmentada e a implementação da solução na plataforma Android. No capitulo 5 são descritos todos os passos da avaliação sugerida, tanto a nível de características como a nível de classificadores.

No capítulo 6 são descritos todos resultados alcançados das avaliações descritas no capítulo 5, e são também comparados os resultados com a literatura. Finalmente, no capítulo 7 são descritos as conclusões alcançadas, nomeadamente a nível de objetivos completados, dificuldades encontradas e trabalho futuro.

Capítulo 2

Revisão Bibliográfica

2.1 Introdução

A inferência de modos de locomoção é uma área que levantou muito interesse na última década, porque existe grande utilidade na aplicação destas previsões em tempo real ou a longo prazo.

Neste capítulo são descritos os principais trabalhos anteriormente desenvolvidos e que estão relacionados com a inferência de modos de locomoção. Existe uma grande variedade de soluções, variando desde o princípio do processo até ao fim deste, tais como, os sensores de geração de dados, pré-processamento dos dados, de modo a converter dados *raw* em dados alto nível, abordagem classificativa a usar, pós-processamento dos dados, por forma a obter classificações mais refinadas e modos de locomoção a serem inferidos.

A secção 2.2 descreve como os sensores são usados e como contribuem nas soluções anteriores, a secção 2.3 descreve como os dados gerados podem ser convertidos em dados alto nível e qual a vantagem de tal abordagem nas soluções existentes, a secção 2.4 apresenta as diversas soluções de classificação usadas e a secção 2.6 apresenta os modos de locomoção que são inferidos como consequência da fase anterior, a secção 2.5 descreve que tipo de *datasets* são usados e que valor acrescentam nas soluções existentes.

2.2 Sensores

Para prever com exatidão a modalidade de locomoção é necessário que a qualidade, variedade e quantidade de dados fornecida sejam fatores importantes para que tal resultado possa ser inferido. De forma a gerar este tipo de dados é necessário existir vários tipos de ações que variem este tipo de dados. Estes tipos de ações podem ser captados pelos sensores que um indivíduo transporte consigo. Existem diversos tipos de sensores, tais como, sensores capazes de detetar a posição da pessoa em coordenadas geográficas, sensores que detetam a aceleração em três eixos, sensores que detetam o posicionamento de uma parte específica do corpo humano (como, por exemplo, o pé ou

a mão), entre outros menos usuais. Existem diversos tipos de sensores ou conjunto de sensores que constituem uma solução plausível para algumas soluções existentes. Desta forma, podemos categorizar os tipos de soluções sensoriais em três grupos: a) soluções comerciais, b) soluções modificadas, e c) soluções embutidas.

2.2.1 Soluções comerciais

Dispositivos comerciais são dispositivos dedicados para recolha de dados através de vários tipos de sensores, disponibilizados no mercado, mais conhecidos pelo termo *off-the-shelf*[1]. Normalmente são úteis para problemas complexos, que requerem vários tipos de dados específicos, ou requerem processamento computacional muito específico, como, por exemplo, medir a pulsação ou medir o número de passos dados por um indivíduo, entre outros.

Existem várias soluções no ramo dos dispositivos comerciais, como ePulse2, descrito na Figura 2.1, um dispositivo que recolhe informações de pulsação e conta os passos dados através do pedômetro [Tec11], também utilizando um pedômetro integrado em sapatilhas Nike, ou dispositivo próprio criado pela Nike, em conjunção com um iPod, pode obter contagem de passos dados durante qualquer atividade física [Inc14], o Omron oferece dispositivos baseados em pedômetros e acelerómetros, permitindo obter os passos dados e também orientação em três eixos [B.V14]. A comparação entre diversos pedômetros permite analisar a precisão da contagem dos passos de cada dispositivo, podendo concluir qual o que oferece maior precisão nos dados recolhidos [SCB04]. No entanto, utilização de pedômetros para inferência de modos de locomoção nunca foi considerado como uma solução disponível neste grupo de dispositivos, sendo usado para outros fins. Pedômetros podem ser usados na inferência de modos de locomoção, caso as locomoções sejam associadas a atividade física. No caso dos modos de locomoção que não requerem atividades físicas esta solução é inviável, como, por exemplo, inferir andar de carro como um modo de locomoção.

Figura 2.1: Dispositivo comercial que permite a medição da pulsação e contagem de passos [Tec11]

[1] produto comercial, disponível comercialmente; já existente no mercado, lojas, no varejo; produzido em série.

Existem outras soluções comerciais com outros tipos de sensores, como usar um dispositivo GPS diferencial [SH00] em vez dos dispositivos GPS não diferenciais [TWS08]. Um dispositivo GPS diferencial oferece mais precisão na obtenção de todas as componentes de uma coordenada geográfica do que os dispositivos GPS não diferenciais. No entanto, os dispositivos GPS diferenciais são mais caros de adquirir e têm dimensões superiores aos dispositivos GPS não diferenciais. Os dispositivos GPS não diferenciais são mais pequenos e mais baratos. Atualmente, este tipo de sensor já vem integrado na maior parte dos *smartphones* existentes no mercado. Ainda dentro das soluções comerciais, o dispositivo GPS não diferencial é bastante popular, sendo vastamente usado para inferir qualquer tipo de atividade que tenha movimentos associados, tal como utilizar dados GPS para poder criar um histórico dos hábitos de locomoção, podendo dessa forma ajudar a inferência dos modos de locomoção [LPFK07], utilizar dados GIS para aumentar a precisão da inferência de modos de locomoção juntamente com dados GPS [PLFK03], utilizar dados GPS para poder inferir atividades do dia a dia do utilizador [HH13] ou utilizar dados GPS para inferir padrões de transito [Zhe10].

2.2.2 Soluções modificadas

Embora existam várias soluções comerciais, a demanda de problemas complexos pode necessitar de dispositivos mais completos. A criação de dispositivos próprios para recolha de dados específicos torna-se uma solução viável e bastante usada na resolução deste tipo de problemas. Existem várias soluções, tais como a utilização de um Inertial Measurement Unit (IMU), uma unidade de medição de inercia, juntamente com um recetor GPS montado no pé, como descrito na figura 2.2, permitindo recolher dados quanto à sua posição, orientação e velocidade [BGL12], a utilização do SATIRE, um dispositivo composto por um GPS e um acelerómetro, capazes de transmitir dados para um servidor [GAJS06] e a utilização do dispositivo WatchMe, um protótipo experimental em forma de relógio de pulso, contendo sensores como GPS, acelerómetro e um microfone para reconhecimento de voz [MSS04].

Várias soluções baseiam-se na construção de um dispositivo composto por diversos acelerómetros espalhados pelo corpo, onde todos estes sensores estão ligados a um chip construído para o efeito, permitindo recolha de dados rigorosos e a inferência de atividades mais complexas, desde atividades rudimentares, como, por exemplo, sentar-se ou lavar os dentes, até atividades complexas, como, por exemplo, andar de carro ou andar de bicicleta [RM00, KSS03, BI04, PA08].

2.2.3 Soluções em dispositivos móveis

Além de existir uma vasta gama de soluções comerciais ou dinâmicas desenvolvidas propositadamente para recolha de dados de sensores, a sua aquisição pela população torna-se impopular e dispendiosa. Nos últimos anos, o mercado de dispositivos móveis tem avançado a nível tecnológico, permitindo a inserção de vários tipos de sensores embutidos dentro de *smartphones*. Através desta adição tecnológica, deixa de ser necessário utilizar dispositivos avançados na aquisição de

Figura 2.2: Solução sensorial usando IMU e GPS montado no pé de um indivíduo [BGL12]

dados de sensores como GPS, acelerómetro, entre outros. A utilização deste tipo de dados é normalmente fornecida por uma API pública, formalmente documentada pelo fabricante. Nos últimos anos foi observado um crescente interesse em utilizar os sensores embutidos em *smartphones* para inferir modos de locomoção, sendo considerada uma solução bastante importante na recolha de dados. Existem várias soluções em dispositivos móveis, tais como a utilização de dados GPS e dados de acelerómetro recolhidos a partir de um Nokia n95 [RMB+10], a utilização de dados GPS, dados do acelerómetro e dados de GSM, para poder também inferir modos de locomoção dentro de edifícios [WNB12], a utilização de apenas um sensor, sendo este o acelerómetro [WCM10] ou a utilização de apenas um sensor, sendo esse GSM [SVL+06, AM06].

Também é possível a combinação de dispositivos comerciais com dispositivos móveis, podendo combinar as vantagens de ambas as soluções. Existem algumas soluções, tais como a utilização de um dispositivo *multi-modal sensor board* (MSB), constituído por sete sensores diferentes, um dispositivo iMote (*Intel Mote*) que permite ligar o dispositivo MSB a outro dispositivo Bluetooth, um dispositivo Bluetooth para concluir o dispositivo experimental e um dispositivo móvel com sensor Bluetooth integrado para permitir receção de dados [LCB06], a utilização de um *Mobile Sensing Platform* (MSP), contendo 10 sensores diferentes e um dispositivo móvel, recolhendo dados GSM desse dispositivo [FDK+09], a utilização de um dispositivo móvel iPhone, usando o seu acelerómetro para recolher dados em conjunção com um dispositivo Nike + ipod Sport Kit [SLF+08, Inc14] e a utilização de dados GSM e WIFI de um dispositivo móvel em conjunção com um dispositivo com vários sensores, tais como acelerómetros, sensores de pressão, entre outros [WLLB05].

2.3 Características e descritores

Após recolha de dados, é necessário identificar as características que variam os modos de locomoção e permitem a sua identificação correta. Através de dados *raw* é possível convertê-los em dados mais alto nível, algo mais fácil de identificar como sendo uma característica potencial para inferência de modos de locomoção. No entanto, dados *raw* podem ser usados diretamente para inferir modos de locomoção [RM00, MSS04, WLLB05, LPFK07, FDK+09].

O sistema GPS fornece vários tipos de dados durante a sua recolha de dados [TWS08]: posição em coordenadas geográficas, permite obter a posição atual do dispositivo no planeta Terra em latitude, longitude e altitude. O nível de precisão costuma ser elevado para as componentes geográficas de latitude e longitude. No entanto, a precisão para a altitude é muito baixa, devido ao ângulo em que se encontram os satélites em relação ao dispositivo GPS, tornando esta componente inviável para situações reais [TWS08]; velocidade é adquirida após recolha de pelo menos dois pontos geográficos, permitindo calcular a velocidade média. Várias variáveis simples podem ser inferidas destes dados *raw*, como por exemplo aceleração, mediana, ângulo de rotação, entre outras [KSS03, BI04, AM06, SVL+06, HH13].

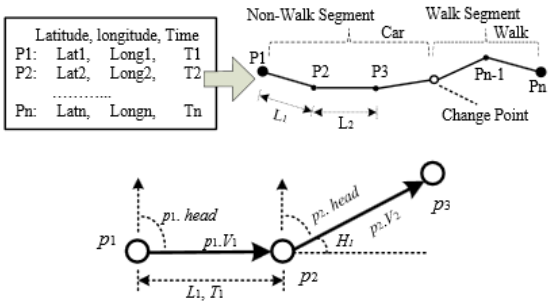

Figura 2.3: Exemplo de segmentação de uma sequência de pontos GPS recolhidos, sugerido em [Zhe10]

Utilização de simples conversões têm sido vastamente usadas, mas conversões mais poderosas e significativas podem ser aplicadas. Podemos encontrar técnicas de segmentação, permitindo análise de um trajeto caracterizado por vários pontos recolhidos, como descrito na figura 2.3. Através desta técnica foi possível extrair várias características importantes em inferir o correto modo de locomoção. Existem várias abordagens usadas com esta técnica, tais como a extração de variáveis como *heading change rate*, *stop rate* e *velocity change rate* de forma a obter características mais robusta para condições de transito [Zhe10, ZQY13] e a extração de variáveis como indicadores de proximidade de pontos geográficos chave no contexto de alguns modos de locomoção (transportes públicos) e indicador de presença de água no ponto geográfico adquirido, em conjunção com *datasets* importantes em fornecer categorização de pontos geográficos [BLvO13].

Várias outras aproximações foram usadas na literatura através de processamento de sinal: a) a utilização de *Fast Fourier Transform* (FTTs), utilizado para converter o espaço de caraterísticas em frequências, extraindo características mais robustas [LCB06, WCM10], b) a utilização de

Discrete Fourier Transform (DFT) para obter coeficientes energéticos da aceleração [RMB+10], e c) a utilização de distribuições de densidade entre uma caraterística e a probabilidade associada a um modo de locomoção [BGL12]. Também foram usados métodos baseados em seleção de caraterísticas (*feature selection*) tal como a utilização de *correlation based features selection* (CFS) para eliminar características redundantes [RMB+10]. Foram usados métodos baseados em estatísticas: a) a utilização de *Kalman filter*, utilizado para prever os próximos pontos através de um modelo linear de forma a obter uma trajetória mais suave [RM00, WNB12], b) a utilização de *Particle filter*, diferenciando do *Kalman filter* pelo modelo não linear a que recorre para resolução do problema [PLFK03], e c) a utilização de *Cohen's kappa coefficient*, utilizado para categorizar os dados de forma mais robusta [BCTH12].

2.4 Inferência modos de transporte

Capacidade de inferir modos de locomoção é um problema muito discutido tendo várias soluções propostas na última década. A necessidade de efetuar esta inferência de modo eficaz tem levado à elevada utilização de algoritmos de classificação, normalmente em duas fases, usando a primeira para a classificação e a segunda para refinamento dos resultados obtidos da classificação.

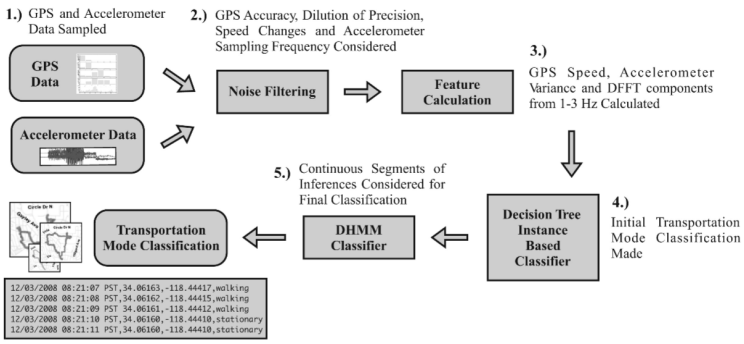

Figura 2.4: Arquitetura do classificador sugerido em [RMB+10].

Existem várias soluções baseadas em modelos de inferência em duas fases, tais como, a utilização de um *decision tree* (DT) C4.5, para instanciar as caraterísticas em classes [RMB+10] (arquitetura descrita na figura 2.4), a utilização de *Random Forests*, um *essemble* constituído por 100 classificadores do tipo *decision tree* [WNB12], sendo estes dois últimos usados em conjunção com um *Discrete Hidden Markov Model* para refinamento da classificação efetuada, a utilização do classificador *decision tree* em conjunção com um algoritmo baseado em grafos, baseado em usar as caraterísticas de *input* e a inferência da *decision tree* para efetuar uma melhor precisão/-classificação do modo de locomoção [Zhe10] e a utilização de *Fuzzy Concepts*, um algoritmo que permite a utilização do conceito de verdade empregue nas caraterísticas fornecidas, em conjunção com um método de segmentação, refinando o processo de inferência [BLvO13].

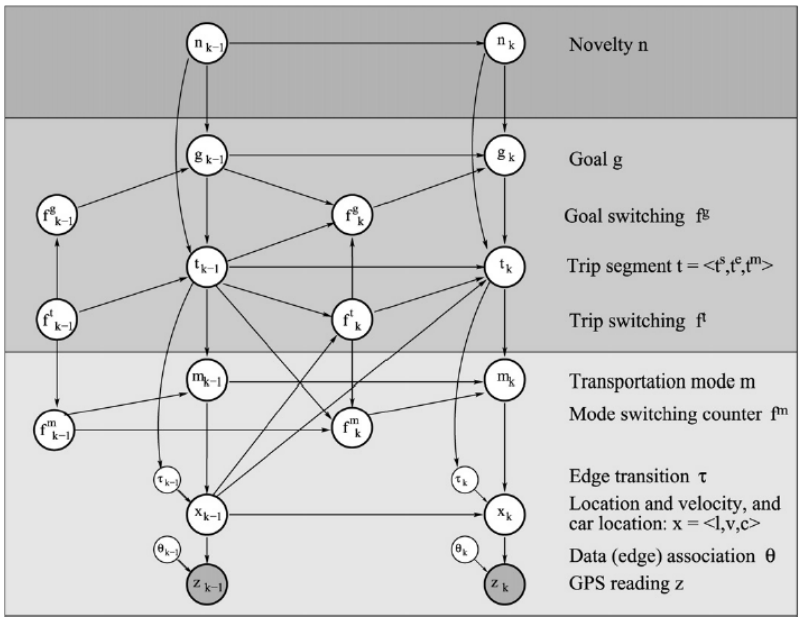

Figura 2.5: Modelo hierárquico de markov, baseado numa rede Bayesiana constituída em várias camadas. Modelo referenciado em [LPFK07].

A maior parte das soluções usa só um algoritmo de classificação, tais como *Naive Bayesian Model*, um simples algoritmo probabilístico baseado no uso de características independentes [PLFK03, KSS03, WLLB05, PA08, SLF+08, BGL12, HH13], DT, baseado na construção de um grafo estilo árvore para classificar caraterísticas em probabilidades [BI04, WCM10], *Hidden Markov Model*, modelo probabilístico com estados escondidos [AM06, GAJS06, LCB06, PA08] ou *Support Vector Machines*, algoritmo de classificação linear que tenta encontrar o hiper-plano que melhor separa os dados de acordo com a classe [GAJS06, SVL+06, WCM10, BCTH12, ZQY13]. Outras soluções também são usadas mas menos frequentes, tais como a utilização de *Hierarchical Markov Model*, sendo construído de várias camadas baseados numa rede bayesiana [LPFK07], como demonstrado na figura 2.5, a utilização de redes neuronais (*Neural Networks*) [RM00] e a utilização de AdaBoost e MultiBoost, algoritmos bastante simples de implementar de forma a construir um essemble de classificadores, onde cada classificador tenta classificar o que o anterior não conseguir [SVL+06].

Autor	Ano	Sensores	Algoritmos de classificação	Precisão de inferência	Modos de Locomoção
Patterson et al. [PLFK03]	2003	GPS	*Naive Bayesian Model*	80%	estacionário; autocarro; carro
Liao et al. [LPFK07]	2007	GPS	*Hierarchical Markov Model*	98%	estacionário; autocarro; carro
Zheng et al. [Zhe10]	2010	GPS	*Decision Trees*	76.2%	andar; carro; bicibleta; autocarro
Reddy et al. [RMB+10]	2010	GPS; Acelerómetro	*Decision Trees* + *hidden markov model*	93.6%	estacionário; andar; correr; bicibleta; carro
Wang et al. [WCM10]	2010	Acelerómetro	*Decision Trees* ou IBK ou SVM	70.73%	estacionário; andar; autocarro; carro; bicibleta; metro
Hummel et al. [HH13]	2012	GPS	Naive Bayesian Model	64.75%	outras atividades
Boldol et al. [BCTH12]	2012	GPS	*Support Vector Machine*	88%	carro; andar; bicibleta; metro; comboio
Biljecki et al. [BLvO13]	2013	GPS; GIS	*Fuzzy Concepts*	91.6%	Todos modos de locomoção
Zhang et al. [ZQY13]	2013	GPS	*Support Vector Machine*	91.004%	carro; andar; bicicleta; autocarro

Tabela 2.1: Tabela das soluções mais relevantes encontradas na literatura.

2.5 *Datasets*

Após a implementação do algoritmo de classificação, é necessário proceder ao treino e à validação do mesmo, de forma a determinar o seu desempenho e a sua precisão em inferir modos de locomoção. Tem sido vasta a utilização de *datasets* caracterizados pelo seu intervalo de tempo, como, por exemplo, um *dataset* correspondente a dez meses de modos de locomoção [Zhe10]. A maior parte das soluções passam por recolher dados de um conjunto de voluntários, que descrevem os seus modos de transporte durante um período definido de tempo [Zhe10, RMB+10, vdSvSdBdH09]; outras aproximações têm sido usadas, como incrementar informação GIS para produção de um *dataset* robusto [BM09, BLvO13].

2.6 Modos de locomoção

Tem sido objetivo de várias soluções inferir os mais variados tipos de atividades, incluindo atividades relacionadas com os transportes. Muitas soluções incluem inferência de vários tipos de atividades, tais como, rastejar, subir ou descer de elevador, subir ou descer escadas [RM00, LCB06, BGL12], ir às compras, ir para o trabalho [HH13], ver televisão, lavar os dentes, trabalhar no computador, ler, relaxar [BI04], escrever, cair [KSS03, GAJS06], entre outras atividades.

Na área dos transportes existe um conjunto variado de atividades que podem ser inferidas. A maior parte destas atividades são fáceis de distinguir através dos dados fornecidos pelos sensores, mas algumas podem induzir em conflito, como distinguir correr de andar de bicicletas, distinguir entre correr e um carro dentro de trânsito [Zhe10, RMB+10, WCM10]. Os modos de locomoção mais usados para inferência são: modo estacionário, usado em praticamente todas as soluções propostas até à data, modo de andar, modo de correr, modo de conduzir um carro e modo de andar de bicicleta [RMB+10]. Outros modos podem ser inferidos, como autocarro, metro, comboio, avião, comboio e barco [BLvO13], como estruturado na figura 2.6. Também existe o problema de distinguir carro de autocarro, sendo usada informação adicional GIS (*Geographical Information System*), adicionando mais informação para um ponto geográfico, tal como a verificação da existência, nas proximidades, de uma paragem de autocarro [PLFK03, LPFK07].

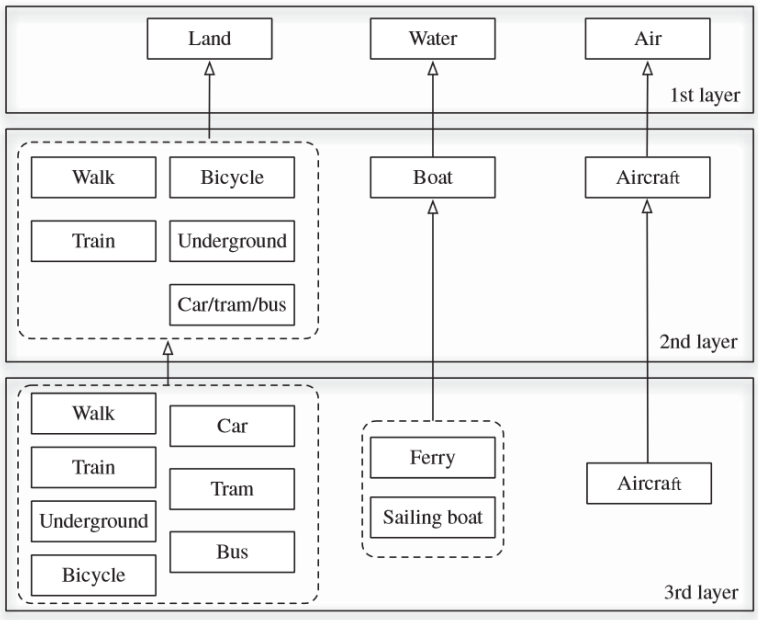

Figura 2.6: Estrutura de modos de locomoção inferidos pela solução descrita em [BLvO13].

2.7 Conclusões

A inferência de modos de locomoção é uma temática complexa e tem tido muito interesse na área de investigação, existindo as mais variadas soluções para resolver este tipo de problemas.

Como se pode deduzir, o universo de sensores de recolha de dados é muito extenso, existindo soluções de vários tipos. No entanto, a utilização de componentes que representam um custo adicional para o dia a dia de um cidadão comum traz inconvenientes e desinteresse à população. Por outro lado, usar sensores existentes num dispositivo móvel para a resolução deste tipo de problemas é um factor muito importante. Várias caraterísticas foram extraídas, podendo observar-se soluções desde a utilização de dados *raw* até caraterísticas de alto nível deduzidas a partir de filtros, como o *kalman filter*. No contexto da inferência de modos de locomoção, o *machine learning* é vastamente utilizado, desde modelos simples de *decision trees* até modelos de duas fases. O grande interesse pela procura de melhores soluções tem levado à criação de *datasets* mais vastos e dedicados ao contexto de inferir modos de locomoção, existindo atualmente vários *datasets*, disponíveis online para uso da comunidade cientifica e académica, evitando, desta forma, a necessidade de criar um *dataset* adicional. Vários tipos de atividades são inferidas, desde atividades simples, como andar, até atividades complexas, como lavar os dentes. Modos de locomoção não foram a exceção, existindo soluções que inferem os modos mais básicos de locomoção, como estacionário e andar, até modos mais complexos, como viajar de autocarro, ou de metro, por exemplo.

Capítulo 3

Método proposto para inferência do modo de locomoção

Neste capítulo são descritos todos os passos efetuados para conseguir inferir modos de locomoção. É descrito o pré-processamento na secção 3.2, o impacto do tamanho da janela na secção 3.3, as características usadas na subsecção 3.4.1, o impacto da seleção de características na subsecção 3.4.2, que classificadores são usados na subsecção 3.5.1 e melhorias usadas na subsecção 3.5.3.

3.1 Explicação geral

A construção de um programa que implica a recolha de dados *raw* do tipo GPS, com o objetivo de inferir o modo de locomoção correto sem interação do utilizador, requer um planeamento detalhado a todos os níveis. Em primeiro lugar, é necessário recolher tais dados *raw* utilizando para isso algum *dataset* existente ou, no caso experimental, um dispositivo móvel que seja dotado de um sensor GPS. O tempo de recolha é importante neste contexto, para se conseguir intervalar os dados recolhidos como sequências temporais. Após a recolha destes dados, selecionamos uma janela ótima, ou seja, quando obtivermos uma sequência de pontos que consiga descrever um modo de locomoção, essa será a quantidade de pontos ótima para podermos descrever tal modo de locomoção. Mas ter apenas um conjunto de pontos que descreva um modo de locomoção não é suficiente. Através de dados *raw* é possível extrair características de alto nível, sendo estas características capazes de diferenciar melhor cada um dos modos de locomoção, do que simplesmente usar dados *raw*. Após a conversão de dados *raw* em características de alto nível, podemos proceder à classificação. Neste processo, é esperado que, através de um conjunto de características de alto nível, o classificador consiga distinguir as diferenças entre ambos modos de transporte e infira o modo de locomoção correto. Após este passo, existe sempre uma incerteza associada na classificação gerada. Desta forma, pode ser necessário adicionar melhorias que consigam prever, tendo

em conta as classificações anteriores, qual o modo de locomoção mais provável de acontecer. A figura 3.1 descreve o fluxo de dados desde os sensores GPS até ao resultado final.

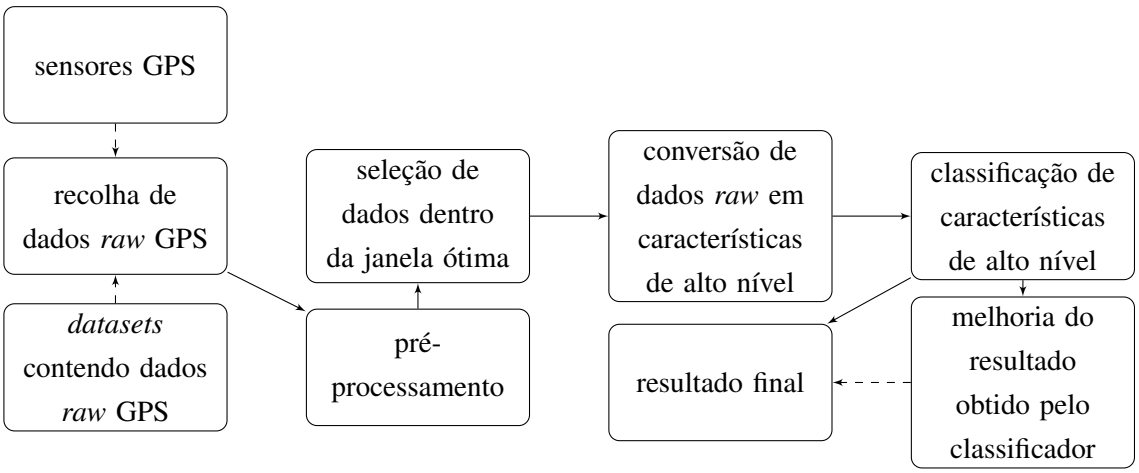

Figura 3.1: Fluxo de dados desde o ponto de receção até ao resultado final.

É de notar que independentemente da fonte dos dados, seja de *datasets* ou dos sensores, os mesmos são sempre gerados a partir de sensores GPS. A diferença é que na figura 3.1 quando é referido que provém de sensores, os dados a serem usados são em tempo real e quando são provenientes de *datasets* são dados anteriormente recolhidos. Também é de notar que a melhoria é opcional, não sendo absolutamente necessária para o contexto da solução final.

3.2 Pré-processamento

Após recolha de dados dos sensores, é necessário filtrar erros existentes entre os dados disponíveis. Existem três tipos de filtros a serem aplicados:

- **Filtragem temporal**: este tipo de filtragem é aplicada a nível sequencial, comparando os *timestamps* de cada coordenada adquirida; se a diferença entre duas coordenadas consecutivas for superior a 6 segundos, a sequência deixa de ser válida se não contiver pontos suficientes para calcular todas as características necessárias até aquele ponto, sendo, de certa forma, dividida caso a quantidade de coordenadas necessárias exista. Esta quantidade aproxima-se entre as 6 e as 9 coordenadas. Este filtro evita que mais de um modo de locomoção seja descrito numa única sequência de pontos que possam ser usados como um único modo de locomoção no processo classificativo;

- **Filtragem da exatidão dos pontos adquiridos**: este tipo de filtragem usa o HDOP para estimar se a coordenada adquirida é suficientemente precisa para ser considerada correta. O HDOP é a diluição de precisão horizontal, uma estimativa utilizada para medir a precisão da coordenada adquirida, recorrendo, para isso, ao número de satélites que o sensor capta, como também as suas respetivas localizações [TWS08]. Estimativas perto de 1 são consideradas ideais e estimativas maiores que 50 são consideravas inválidas, ou seja, são pontos que

não são suficientemente precisos para serem considerados pontos adquiridos com precisão. Desta forma, foi definido um *threshold* de 50, para quando o valor de HDOP for igual ou ultrapassar, o ponto ser descartado do processo sequencial de recolha.

- **Filtragem de acelerações**: este tipo de filtro permite eliminar sequências temporais que contenham uma diferença entre acelerações sequenciais excessivas. Para este filtro decidiu-se aplicar uma diferença de 40 $Km/h/s$, ou seja, se a diferença entre duas acelerações calculadas for maior que 40 km/h a sequência é descartada. Desta forma são descartados os dados que não se situam dentro do que é esperado obter ao calcular características de modos de locomoção.

Através destes filtros é possível aumentar a probabilidade de todas as sequências de coordenadas se aproximarem de apenas um modo de locomoção, e também garantir que as coordenadas captadas pelo sensor GPS correspondem a uma coordenada com uma exatidão aceitável.

3.3 Seleção de janelas temporais

A quantidade de pontos que caracteriza um modo de locomoção é bastante discutível, porque para cada modo de locomoção existe uma grande variação de tempo que os caracterize. De forma a podermos encontrar um tamanho próximo do ótimo, é necessário efetuar uma escolha que caraterize todos os modos de locomoção de uma forma aproximada, porque não existe um único tamanho que consiga caraterizar de uma forma ótima todos modos de locomoção. É preciso ter em atenção que a janela não deverá ser muito grande nem muito pequena. Se a janela for muito pequena, existirão poucas coordenadas que caracterizem o modo de locomoção, incorrendo-se no erro de não existirem características diferenciadores capazes de distinguir todos os modos de locomoção. Se a janela for muito grande, cada conjunto de coordenadas pode ter mais do que um modo de locomoção, introduzindo erros no processo classificativo. Uma outra consequência é o tempo em *stand-by* até obtermos uma inferência, ou seja, quando maior a janela temporal definida, maior será o tempo necessário para obter classificações.

Para procedermos a esta escolha, é necessário testar várias janelas com os dados existentes em todo o processo, e comparar a exatidão alcançada com todo o processo classificativo. Normalmente esta escolha é feita quando todas as partes do sistema estiverem funcionais. Só através dos resultados finais se pode obter a melhor exatidão em diferenciar o tamanho da janela temporal.

Tendo em conta que a melhor janela temporal tem de ser selecionada, um conjunto de janelas foi definido, na escala de coordenadas, ou seja, cada janela temporal é constituída por um conjunto de coordenadas *raw* GPS, existindo um intervalo entre 1 a 6 segundos entre cada duas coordenadas sequenciais. Este conjunto de janelas inclui janelas entre 5 a 20 coordenadas, traduzindo entre 5 a 50 segundos aproximadamente, de forma a podermos testar numa vasta gama de tamanhos.

3.4 Características

Após a obtenção de uma sequência temporal que represente um modo de locomoção, podemos transformar esse conjunto de coordenadas *raw* em características de alto nível que consigam diferenciar os modos de locomoção. É vantajoso esta transformação, porque se não usássemos características mais diferenciadores mas apenas dados *raw*, o processo de classificação poderia não ser capaz de distinguir entre qualquer modo de locomoção.

3.4.1 Características usadas

Através dos dados *raw*, é possível extrair diversas características que ajudem a diferenciar os diferentes modos de locomoção. A principal caraterística que podemos extrair com coordenadas GPS é a velocidade e a partir de diversas velocidades podemos obter a respetiva aceleração. É de ter em atenção que o tempo decorrido entre um ponto e o próximo é de extrema importância para o cálculo correto destas características básicas. A velocidade é calculada como a razão entre a diferença entre duas coordenadas *raw* GPS e o tempo decorrido entre as duas coordenadas.

$$v = \frac{\Delta d}{t} \tag{3.1}$$

Na equação 3.1, a variável Δd representa a diferença entre os dois pontos definidos por coordenadas GPS em quilómetros, e t representa o tempo, em horas, que decorreu durante percurso desde o ponto inicial até ao ponto final. Desta forma, conseguimos obter o resultado em Km/h. É de notar que é necessário calcular a distância entre estas duas coordenadas para obter Δd, usando a fórmula *haversine*, demonstrada posteriormente nesta subsecção. A aceleração é calculada como a razão entre a diferença de duas velocidades sequenciais pelo tempo entre estas duas velocidades.

$$a = \frac{\Delta v}{t} \tag{3.2}$$

Na equação 3.2, a variável Δv representa a diferença entre duas velocidades sequenciais no tempo, em quilómetros por hora, e t representa o tempo decorrido entre as duas velocidades, em segundos. Assim, conseguimos obter o resultado em $km/h/s$.

Para este problema, as seguintes características foram escolhidas, como demonstrado na seguinte lista.

- **Total de distância percorrida**: esta característica refere-se à distância total percorrida entre o ponto inicial e final da sequência temporal, sendo utilizada na literatura [ZCL+10]. Através desta característica podemos obter uma estimativa quanto à relação entre a distância total e o tempo percorrido. Para calcularmos esta característica é necessário determinar, de uma forma sequencial, a distância entre os pontos que constituem a sequência. A soma destas distâncias iguala esta característica. Para obter a distância entre duas coordenadas, foi usada a fórmula *haversine* [Sin84], calculou-se a distância mais curta entre dois pontos na superfície da terra, tendo em conta as latitudes e longitudes dos dois pontos. Através

desta fórmula é possível obter distâncias mais exatas. As constantes φ_1 e φ_2 representam a longitude do ponto 1 e 2 em radianos, respetivamente; a constante $\Delta\varphi$ representa a diferença entre as latitudes dos dois pontos em radianos e $\Delta\lambda$ corresponde à diferença das longitudes entre os dois pontos, em radianos. A equação 3.3 representa o quadrado da metade do comprimento da distância entre os dois pontos.

$$a = \sin^2(\frac{\Delta\varphi}{2}) + \cos(\varphi_1) \times \cos(\varphi_2) \times \sin(\frac{\Delta\lambda}{2}) \tag{3.3}$$

A equação 3.4 representa a distância angular entre os dois pontos, em radianos.

$$c = 2 \times atan2(\sqrt{a}, \sqrt{1-a}) \tag{3.4}$$

A equação 3.5 representa a distância total, entre os dois pontos, tendo em conta a dimensão da Terra (através do raio da Terra, representado pela constante R);

$$d = R \times c, R = 6.371km \tag{3.5}$$

- **Velocidade e aceleração mínima**: esta caraterística escolhe, de entre todas as velocidades e acelerações calculadas dentro da janela temporal, a mais baixa, sendo que a velocidade mínima é a usada anteriormente em [ZCL+10]. É de notar que são duas características diferentes, mas que têm uma forma de calcular igual. Desta forma, podemos ter uma estimativa do quão baixos são os valores que as velocidades e acelerações de um modo de locomoção normalmente podem atingir, sendo possível encontrar aqui certa diferenciação. A equação 3.6 apresenta a fórmula usada para chegar à velocidade mínima.

$$min_v = min(v_1, v_2, v_3, \ldots, v_N), \text{em que } N \text{ representa o tamanho da janela} \tag{3.6}$$

Na equação 3.6, cada velocidade toma o valor v_i, onde v_i corresponde à velocidade i-ésimo ponto da janela, sendo o conjunto total representado por $v_{1\ldots N}$. A equação 3.7 apresenta a fórmula usada para obter a aceleração mínima.

$$min_a = min(a_1, a_2, a_3, \ldots, a_N), \text{em que } N \text{ representa o tamanho da janela} \tag{3.7}$$

Como já demonstrado na equação 3.6, a equação 3.7 segue o mesmo formato, sendo a_i uma das acelerações correspondentes de uma dada janela.

- **Velocidade e aceleração máxima**: esta caraterística escolhe, de entre todas as velocidades e acelerações calculadas dentro da janela temporal, a mais alta, sendo que a velocidade máxima foi usada anteriormente em [ZCL+10]. Desta forma podemos ter uma estimativa do quão alto é o valor que a velocidade e aceleração de um modo de locomoção normalmente

atingem, podendo encontrar-se aqui certa diferenciação. A equação 3.8 apresenta a fórmula usada para chegar à velocidade máxima.

$$max_v = max(v_1, v_2, v_3, \ldots, v_N), \text{em que } N \text{ representa o tamanho da janela} \tag{3.8}$$

Na equação 3.8, cada valor de v_i representa uma velocidade, sendo i pertencente ao intervalo $0 \ldots N$, representativo do tamanho da janela. A equação 3.9 apresenta a fórmula usada para chegar à aceleração máxima.

$$max_a = max(a_1, a_2, a_3, \ldots, a_N), \text{em que } N \text{ representa o tamanho da janela} \tag{3.9}$$

Comparando as equações 3.8 e 3.9, constatamos que têm o mesmo formato, sendo cada valor de a_i representado por uma aceleração do conjunto de acelerações de uma dada janela.

- **Velocidade e aceleração média**: esta característica permite obtermos uma média entre as velocidades e acelerações atingidas para uma dada janela temporal, sendo que a velocidade média é a usada anteriormente em [ZCL+10]; através desta característica, podemos ter uma estimativa dos valores para cada um dos modos de locomoção. A equação 3.10 representa a fórmula usada no cálculo da velocidade média.

$$mean_v = \frac{1}{N} \times \sum_{i=1}^{N} v_i, \text{em que } N \text{ representa o tamanho da janela} \tag{3.10}$$

A equação 3.10 pode ser resumida pelo somatório de todas as velocidades, sendo cada uma representada por v_i, de uma dada sequência temporal, a dividir pelo número total de velocidades existentes, representado por N. A equação 3.11 representa a fórmula usada para a aceleração média.

$$mean_a = \frac{1}{N} \times \sum_{i=1}^{N} a_i, \text{em que } N \text{ representa o tamanho da janela} \tag{3.11}$$

Tal como a equação 3.10, a equação 3.11 segue mesmo formato, usando o conjunto de todas as acelerações, sendo cada uma representada por a_i, em vez do conjunto de todas as velocidades para uma determinada janela temporal.

- **Velocidade e aceleração mediana**: esta característica proposta neste dissertação pretende encontrar o valor central da lista de valores ordenados de velocidade e acelerações, respetivamente; esta característica difere da média por ser menos sensível que a média dos *outliers*, também é indicada para ser usada com conjuntos pequenos de dados, e por ser mais robusta

quando a frequência de coordenadas *raw* GPS apresenta erro. A equação 3.12 representa a fórmula usada no cálculo da velocidade mediana.

$$median_v = \begin{cases} v_c & \text{se número total de velocidades for ímpar} \\ \frac{v_{c1}+v_{c2}}{2} & \text{se número total de velocidades for par} \end{cases} \tag{3.12}$$

A fórmula 3.12 é caracterizada por duas condições. Dada uma sequência temporal, se o número total de velocidades da sequência for par, as duas velocidades centrais, v_{c1} e v_{c2}, têm de ser selecionadas. Em seguida, a média destas velocidades tem de ser calculada para obter a mediana respetiva. Caso o número total de velocidades da sequência seja ímpar, a mediana das velocidades é o número central respetivo, v_c. A equação 3.13 apresenta a fórmula da aceleração mediana.

$$median_a = \begin{cases} a_c & \text{se número total de acelerações for ímpar} \\ \frac{a_{c1}+a_{c2}}{2} & \text{se número total de acelerações for par} \end{cases} \tag{3.13}$$

Tal como a equação 3.12, a equação 3.13 segue o mesmo formato, usando o conjunto das acelerações em vez do conjunto das velocidades para uma dada janela temporal.

- **Desvio padrão amostral das velocidades e acelerações**: esta característica proposta nesta dissertação permite obter a disparidade existente em relação à média das velocidades ou das acelerações dada uma janela temporal; como já discutido anteriormente, existe uma grande diferença entre as velocidades e acelerações adquiridas para os diferentes modos de locomoção. Como a média tenta aproximar as velocidades e acelerações de uma valor médio, através do desvio padrão podemos ter uma noção do quanto estas velocidades e acelerações podem afastar-se do valor adquirido na média; é de notar que se usa o desvio padrão amostral em vez do desvio padrão populacional; pois cada sequência é uma amostra do conjunto de todas as velocidades e acelerações adquiridas que representam dado modo de locomoção; como as sequências são constituídas por poucas coordenadas GPS, isto, de facto, representa uma pequena amostra da população; desta forma concluímos a utilização do desvio padrão amostral em vez do desvio padrão populacional; a equação 3.14 representa a fórmula usada no cálculo do desvio padrão amostral para as velocidades.

$$s_v = \sqrt{\frac{1}{N-1} \times \sum_{i=1}^{N} (v_i - \bar{v})^2} \tag{3.14}$$

Na equação 3.14m s_v representa o desvio padrão amostral das velocidades, N é o número total de velocidades da sequência, v_i representa cada uma das velocidades dentro do intervalo

$1 \ldots N$, e $\bar{v}$ representa a média dos velocidades. Na equação 3.15 representa-se a fórmula usada no cálculo do desvio padrão amostral para as acelerações.

$$s_a = \sqrt{\frac{1}{N-1} \times \sum_{i=1}^{N} (a_i - \bar{a})^2} \tag{3.15}$$

Tal como a equação 3.14 , a equação 3.15 segue o mesmo formato, sendo s_a o desvio padrão amostral das acelerações, a_i o valor de cada uma das acelerações, e $\bar{a}$ a média das acelerações.

- **Diferença de Direção**: esta característica proposta nesta dissertação permite caracterizar o ângulo da direção tomada para um determinado modo de locomoção; através desta característica espera-se diferenciar como cada modo de locomoção se direciona; para obtermos a diferença entre a direção inicial e a final é necessário calcular o *bearing* [Wil14] entre as duas coordenadas que definem a direção inicial e as duas coordenadas que definem a direção final; a equação 3.16 apresenta a fórmula usada para calcular o *bearing* necessário para obter a diferença de direção.

$$\theta = atan2(\sin(\Delta\lambda) \times \cos(\varphi_2), \cos(\varphi_1) \times \sin(\varphi_2) - \sin(\varphi_1) \times \cos(\varphi_2) \times \cos(\Delta\lambda)) \tag{3.16}$$

Na equação 3.16 a variável $\Delta\lambda$ representa a diferença das longitudes entre as duas coordenadas, e as variáveis φ_1 e φ_2 representam a latitude da primeira e segunda coordenada, respetivamente. A variável θ é o *bearing* resultante entre as duas coordenadas a serem usadas.

- ***Stop Rate***: esta característica identifica a frequência de paragens para uma dada sequência. Seguindo a sugestão em [ZCL+10], consideramos útil a utilização desta característica pelo facto de podermos diferenciar pelo número de vezes que um modo de locomoção cessa durante uma sequência. Esta característica usa um *threshold* menor que 2 km/h, tendo em conta possíveis erros de precisão que o GPS possa apresentar, mesmo que o modo seja estacionário. Se para uma determinada velocidade de uma dada sequência for inferior a 2 km/h , é considerado como uma paragem. A equação 3.17 descreve a fórmula usada para o cálculo desta característica.

$$SR = \frac{PS}{\text{Distância percorrida}} \tag{3.17}$$

A equação 3.17 é resumida pela divisão da soma de todos os pontos que apresentem uma velocidade inferior ao *threshold* de 2 km/h, representado por PS a distância total da sequência a ser usada. Desta forma podemos obter uma frequência de quantas paragens tem cada um dos modos de locomoção.

- ***Heading Change Rate***: esta característica identifica a frequência com que a direção muda ultrapassando um *threshold*. Como sugerido em [ZCL+10], considera-se esta característica útil pelo facto de poder diferenciar cada um dos modos de locomoção, sendo muito semelhante à diferença de direção já anteriormente apresentada. O *threshold* definido é de 30º, sendo um ângulo razoável que define um ponto de viragem de um ponto contínuo em linha reta. A equação 3.18 define a fórmula usada para o cálculo do *heading change rate*.

$$HCR = \frac{P_c}{\text{Distância percorrida}} \tag{3.18}$$

  A equação 3.18 é resumida pela divisão da soma de todos os pontos que apresentem uma mudança de direção superior ao *threshold* de 30º, P_c, pela distância total da sequência a ser usada. Desta forma podemos obter a frequência das mudanças de direção para cada um dos modos de locomoção.

- ***Velocity Change Rate***: esta característica representa a frequência com que a velocidade que diferencia duas velocidades consecutivas é superior a um *threshold*. Como sugerido em [ZCL+10], considera-se esta característica útil porque diferencia, para cada modo de locomoção, quantas vezes a velocidade varia passado determinado *threshold*. O *threshold* escolhido é de 20 km/h, porque entre duas velocidades há uma aceleração considerável de mudança de velocidade. A equação 3.19 demonstra a fórmula usada para o cálculo da *velocity change rate*.

$$VCR = \frac{P_v}{\text{Distância percorrida}} \tag{3.19}$$

  A equação 3.19 é resumida pela divisão da soma de todos os pontos que apresentem uma mudança de velocidade superior ao *threshold* de 20 km/h, representado por P_v, pela distância total da sequência a ser usada. Desta forma é possível obter a frequência das mudanças de velocidade de cada um dos modos de locomoção.

3.4.2 Seleção de características

Além de apresentar diversas características capazes de diferenciar as diferenças entre cada modo de locomoção, é necessário analisar se algumas destas características não ajudam de alguma forma a diferenciar modos de locomoção. Tendo isto em conta, é provável que possa existir algumas características que não consigam diferenciar as características presentes num conjunto de dados *raw*. É preciso notar que esta análise é necessária para cada conjunto de dados *raw* existentes. É crucial ter em conta que não podemos assumir resultados iguais para diferentes *datasets*, no que diz respeito a seleção das melhores características e das prejudiciais para inferência de modos de locomoção.

Tendo em conta que existem diversos algoritmos capazes de diferenciar as melhores características das piores, decidiu-se aplicar um algoritmo capaz de apresentar o ganho que certa característica apresenta dado um determinado conjunto de dados para treino. Desta forma, decidiu-se usar um algoritmo robusto denominado Relief-F [KvRv97], um algoritmo que calcula o ganho que uma característica tem dado um conjunto de características e um conjunto de treino. Este algoritmo é uma versão reformulada e mais eficaz do que o algoritmo Relief [KR92]. Isto deve-se ao facto do Relief não conseguir calcular propriamente bem o peso de uma característica dado um conjunto de treino relativamente pequeno, induzindo em erros de atribuição de pesos. Com a introdução do Relief-F, este problema foi corrigido, sendo adaptado para conseguir calcular valores de peso com melhor confiança. Também é preciso notar que este algoritmo não requer utilização de nenhuma heurística, o que facilita consideravelmente a parametrização deste algoritmo aplicado à seleção dos pesos de cada uma das características já mencionadas. Um dos parâmetros mais importantes do Relief-F é o número de vizinhos usados para computar o peso de uma característica. Escolhemos um valor de 10 pelo facto de um valor mais baixo ser passível de apresentar erros ao encontrar o impacto de certa característica no conjunto de treino, e um valor mais alto poder não conseguir encontrar diferenças capazes de caracterizar dada característica pelo seu peso.

Com este algoritmo espera-se conseguir encontrar o conjunto das melhores características através de *ranking*, ou seja, as N melhores características que consigam diferenciar os modos de locomoção.

3.5 Classificadores

Após o cálculo das características de alto nível, é necessário criar o modelo que caracteriza cada um dos modos de locomoção, de forma a que uma instância consiga ser classificada num modo de locomoção. Para construir um modelo deste género, é necessário aplicar um classificador, capaz de encontrar padrões entre as características de alto nível, de modo a que este possa concluir que características define cada um dos modos de locomoção. Para que um algoritmo possa aprender o que caracteriza cada um dos modos de locomoção, é necessário fornecer um conjunto de instâncias de treino já catalogadas, de forma a que o classificador consiga fazer correspondência para um determinado conjunto de características.

3.5.1 Classificadores individuais

A estratégia mais comum de classificação é usar simplesmente um classificador que consiga diferenciar todas as classes a serem inferidas. De entre os muitos classificadores existentes, decidiu-se selecionar um conjunto de algoritmos, maioritariamente usados na literatura existente. Cada um dos algoritmos escolhidos está listado na seguinte lista.

- ***Support Vector Machines* (SVM)**: *Support Vector Machines* [CV95] é um modelo supervisionado, usado para encontrar padrões em dados binários entre duas classes existentes, sendo usado na literatura [WCM10, BCTH12, ZQY13, GAJS06, SVL+06]. De forma a

podermos classificar mais do que duas classes, é necessário usar uma variante de SVM, nomeadamente multiClass SVM. Esta variante permite criar vários problemas binários de classificação, devido à existência de mais de duas classes a serem classificadas. Este classificador tenta encontrar uma divisão entre os dados, de forma a que estes possam ser distinguidos através de um hiperplano que os separa. Este classificador apresenta alguns parâmetros, importantes para o contexto classificativo, que requerem alguma análise. Um dos parâmetros mais importantes é o *Kernel*, que permite transformar as características num espaço de alta-dimensão não linear. Este parâmetro foi introduzido pela necessidade de resolver problemas não lineares. Este parâmetro é importante porque, se bem selecionado, pode trazer resultados perto do ótimo. De entre os *kernels* existentes, escolhemos o linear, o primeiro *kernel* a ser sugerido para SVM, por evitar mais *overfitting* do que um *kernel* não linear e também pelo facto de este problema ter uma complexidade elevada. Outro parâmetro selecionado foi a constante C, variável responsável por penalizar os erros quanto maior este valor for. Esta constante permite controlar *overfitting*, visto estar diretamente relacionado com a quantidade de erros existentes pela separação dos dados. Por isso, decidimos testar com diferentes constantes, nomeadamente 1, 10, 100 e 1000. Através desta experimentação, podemos obter o melhor valor de C para a validação cruzada usada;

- **k-nearest neighbors** [AKA91]: este classificador, usado na literatura [WCM10], é do tipo onde o modelo é constituído pelos próprios dados e a classificação é baseada nos vizinhos, ou seja, agrupando os dados existentes como vizinhos que por sua vez podem caracterizar um conjunto de características pertencentes a uma classe, neste caso, um modo de locomoção. Existem parâmetros que requerem alguma definição antes de usar este classificador. K é o número de vizinhos que serão usados para calcular qual será o melhor vizinho que pode caracterizar um determinado modo de locomoção. Como não sabemos exatamente qual o valor ótimo, decidimos aplicar uma validação cruzada, de forma a podermos encontrar uma constante K ótima. Como pelo menos um dos *datasets* a ser usado é relativamente grande, decidimos usar árvores de pesquisa para a pesquisa das vizinhanças. Esta decisão deve-se ao facto de a quantidade de dados a usar ser consideravelmente grande. Como o *dataset* é sempre o mesmo, podemos usar uma árvore de decisão para proceder a pesquisas mais eficientes e rápidas;

- **Naive Bayes** [JL95]: esta classificador é um dos mais simples conhecidos atualmente, usado na literatura [BGL12, PLFK03, HH13, KSS03, PA08, SLF+08, SVL+06], pertence ao grupo dos algoritmos probabilísticos. Este algoritmo usa a frequência das características que definem um modo de locomoção para as diferenciar dos outros modos de transporte;

- **Neural Network (NN)**: *multilayer perceptron* [Mal86] é um algoritmo de aprendizagem inspirado no cérebro de um animal, usado na literatura [AM06, RM00], constituído por um conjunto de neurónios, em diversas camadas internas interligadas. Cada uma destas ligações tem um peso, que pode se moldar conforme um conjunto de dados de entrada, de forma a poder inferir determinado conjunto de estados, neste caso modos de locomoção. É de notar

que este tipo de classificador pode produzir modelos com *overfitting* em alguns casos, se o *dataset* de treino aplicado ao classificador não estiver suficientemente equilibrado. Caso o número de instâncias de entrada durante a aprendizagem caracterizar melhor um modo de locomoção que os restantes, é de esperar que esse modo de locomoção seja melhor classificado do que os restantes, porque a rede neuronal irá aprender como lidar melhor com esse dado modo de locomoção. Quanto a nível de parâmetros, demos mais destaque ao parâmetro das iterações aplicadas ao algoritmo até que fique adaptado completamente. Se o número de iterações for demasiado grande, a rede neuronal gerada pode ficar com *overfitting* como já tinha sido referido. Desta forma, decidimos aplicar apenas 50 iterações, de forma a evitar problemas futuros de *overfitting*, o que também permitiu que demorasse menos tempo a construir um modelo de NN com os dados disponíveis;

- ***Decision Tree* (DT)** [YS95]: este classificador baseia-se na representação de um grafo em forma de árvore, usado na literatura existente [RMB+10, WCM10, ZCL+10, BI04], sendo cada ramo composto por uma característica diferenciadora, como uma condição, as folhas serão as classes que este classificador deverá inferir. É um algoritmo muito usado atualmente, de fácil aprendizagem, apresenta um bom desempenho de uma forma generalizada na maior parte dos casos onde é aplicado. Também inclui custos de cada decisão, de forma a poder inferir o caminho com o menor custo possível. Para este algoritmos existem diversas variantes que pretendem aumentar o ganho, diminuindo a entropia de cada característica. Para esta dissertação usamos C4.5, capaz de podar nós numa única folha, caso o erro da sub-árvore apresentada por esse nó ser maior do que o erro no nó em si. Também é capaz de encontrar a característica que melhor divide o *dataset*, para um determinado nó;

- ***Random Tree* (RT)**: *Random Tree* [Ho95] é um classificador derivado do DT, selecionando K características aleatoriamente de forma a poder construir a árvore de decisão, de forma a não usar todas as características disponíveis. Esta técnica permite obter uma maior rapidez de construção do modelo classificativo, como também apresenta uma solução, caso o número de características, seja consideravelmente grande, o que pode ser computacionalmente pesado;

- ***Random Forest* (RF)**: *Random Forest* [Ho95] é um *ensemble* de *Random Trees*, sendo um dos classificadores usados na literatura [WNB12]. Este classificador usa K árvores de decisão aleatórias para inferir a classe de uma determinada instância de entrada, usando um sistema de votação para concluir o resultado final da inferência. Usando N características para cada árvore de decisão, é possível geral uma "floresta" de árvores de decisão diferentes entre si, de forma a ter vários resultados que serão selecionados a voto do melhor resultado. A nível de parâmetros, o mais importante é o número das árvores que deverá ser gerado. Tendo em conta que 10 árvores é um número muito reduzido e 1000 poderá produzir modelos relativamente grandes para o ambiente destino, decidimos usar um tamanho de 100, por apresentar uma possível relação entre capacidade de modelação e espaço.

3.5.2 Classificadores em cascata

Outra estratégia que pode ser aplicada é a utilização de múltiplos classificadores, sequencialmente, de forma a que cada um destes classificadores represente uma camada. Intuitivamente é esperado que as primeiras camadas consigam distinguir classes mais generalizadas. Por exemplo, existem vários modos de locomoção, e estes podem ser separados por modos motorizados e modos não motorizados. Desta forma, podemos separar dois grupos totalmente diferentes. A segunda camada corresponde à separação, entre todos os modos de cada grupo já separado na primeira camada, para distinguir cada um dos modos motorizados de locomoção, se considerando que estamos a aplicar a segunda camada ao grupo dos motorizados. Assim, é nos possível separar entre vários modos motorizados, como, por exemplo, entre carro, táxi e autocarro através de uma segunda camada. Podemos aplicar este conceito para diminuir o número de classes que certo classificador tem de classificar, dividindo a classificação em várias camadas, usando vários tipos de classificadores e vários tipos de classes para cada camada.

3.5.3 *Hidden Markov Models* (HMM) para modelar dependência temporal

Muitas vezes, através de uma classificação, não é possível garantir que o resultado adquirido seja totalmente confiável, principalmente quando há dependência temporal entre instâncias e quando essa dependência não é modelada. Desta forma, podemos melhorar os resultados através de uma melhoria, através dos resultados anteriores e do atual resultado obtido. Isto é possível utilizando séries temporais, onde é analisado o espaço temporal dos resultados, como uma sequência, e o objetivo desta análise é permitir que um classificador consiga prever, com confiança, qual o próximo estado mais provável de acontecer, tendo em conta os resultados já adquiridos. Para aplicarmos este conceito, foi usado o HMM ou *Hidden Markov Model* [RJ86], já utilizado mais de uma vez na literatura [RMB+10, LPFK07, WNB12, AM06, GAJS06, LCB06, PA08], um modelo estatístico baseado num processo de Markov, só que os estados constituintes destes modelos não são diretamente visíveis, mas sim à saída do modelo. O conceito passa por criar um modelo constituído por N estados, cada um destes estados representa um grupo de características e, através da história fornecida ao modelo, é analisado o estado final. Cada um destes estados tem como saída final um estado visível, uma das classes que é suposto o modelo prever dada uma sequência temporal de resultados. O parâmetro que mais impacto pode ter é o número de estados internos, é o que pode aumentar a complexidade do modelo construído. Desta forma, decidimos usar 100 estados, porque 10 poderia condensar demasiado a quantidade de características existentes nos percursos usados para classificação, e um número superior de estados internos poderia ser computacionalmente pesado de produzir um modelo de HMM. Poderíamos ter usados mais valores mas pelo fator de tempo não nos foi possível aplicar uma análise mais profunda ao nível de parametrização do HMM.

3.6 Resumo

Através deste capítulo, descrevemos os métodos aplicados nesta dissertação, nomeadamente ao nível dos filtros, concluímos que, as características de alto nível são o que caracteriza cada um dos modos de locomoção a inferir. As características usadas são baseadas nas coordenadas existentes e na sequência temporal que as compõe, sendo possível calcular velocidades, acelerações e todas as derivações, como médias, medianas, máximos, mínimos, desvios padrões e até características mais avançadas como *Heading Change Rate* (HCR), *Velocity Change Rate* (VCR) e *Stop Rate* (SR). Quanto a nível de classificação podemos observar que existe uma vasta gama de algoritmos que podem ser aplicados, tanto que já são usados na literatura como podem ser potencialmente úteis no contexto deste problema. Também foi analisada cada uma das estratégias que podem ser viáveis neste contexto, nomeadamente, a utilização de apenas um classificador que infira todos os modos de locomoção de uma vez, ou então a utilização de classificadores em cascada, divididos em camadas, criando modelos mais simples ao longo destas camadas. A nível de modulação temporal, usamos o HMM de forma a tentar encontrar vantagens em modelar as dependências temporais entre instâncias de uma determinada sequência já previamente classificada.

Capítulo 4

Implementação

Neste capitulo são descritos todos os pontos da implementação realizada, tanto a nível da *framework* experimental como da implementação Android. A *framework* experimental, na secção 4.1, é constituída por diversos componentes, nomeadamente pré-processamento na subsecção 4.1.1, geração das características de alto nível na subsecção 4.1.2, geração dos *datasets* de treino e teste em 4.1.3, processo classificativo na subsecção 4.1.4 e modulação de dependência temporal entre instâncias com *Hidden Markov Model* (HMM), na subsecção 4.1.5. A secção 4.2 refere-se a todos os aspetos da implementação Android, tanto a nível de execução como a nível de classificação.

4.1 *Framework* experimental

Para testar as várias estratégias sugeridas e converter dados *raw* em classificações de modos de locomoção, é necessário ter em conta diversos passos de processamento até obter o resultado pretendido. A linguagem utilizada para o desenvolvimento desta *framework* foi o Java [Ora14], uma linguagem de alto nível suportada na maior parte dos sistemas operativos existentes, incluindo Android. Esta linguagem foi construida usando C/C++ sendo utilizada para contornar certas aspetos complexos da linguagem C, como, por exemplo, alocação de memória. A linguagem Java foi escolhida também porque facilita a conversão do código final para a aplicação experimental em Android. Como o sistema operativo do Android tem uma máquina virtual de Java, é possível usar a maior parte do código desenvolvido nesta *framework* na aplicação Android desenvolvida.

4.1.1 Pré-processamento

O primeiro passo desta *framework* é ter os dados *raw* organizados por sequências temporais que não excedam determinado tempo pré-estabelecido. Para se obter os dados é necessário efetuar uma leitura dos dados existentes, nomeadamente do *dataset* do Geolife e do SenseMyCity. É

preciso ter em conta que os dados *raw* de cada um destes *datasets* estão em formatos diferentes em todos os aspetos, o que requer uma análise individual para cada tipo de *dataset*.

No caso do *dataset* do Geolife, é necessário ter em atenção que os dados não estão armazenados numa base de dados. Neste caso, os dados estão distribuídos por um conjunto de pastas, onde cada pasta corresponde a um conjunto de catalogações adquiridas. No interior de cada uma destas pastas podemos encontrar uma sub-pasta e um ficheiro. Dentro desta sub-pasta encontramos um determinado conjunto de ficheiros, compostos por milhares de coordenadas *raw* GPS. Por norma estão organizadas temporalmente, normalmente apresentando-se com um intervalo de 2 segundos. O ficheiro encontrado no mesmo nível que esta sub-pasta contém um conjunto de intervalos, sendo cada intervalo atribuído a um modo de locomoção. Este formato é descrito e explicado na página oficial onde este *dataset* é distribuído gratuitamente para a comunidade cientifica o poder usar [Zhe10, ZCL+10, ZLWX08, ZLC+08]. Foi feito um módulo em Java que permite a leitura deste formato, a um nível escalável, de forma a que consiga efetuar a leitura de todos os dados, para um modo estruturado em memória. Após a execução deste passo, é necessário limpar todos os intervalos errados, como, por exemplo, intervalos que não estão catalogados ou intervalos que não são intervalados por duas datas (o intervalo não apresente uma data ou duas). Após esta filtragem, é efetuada a junção das coordenadas GPS com o seu respetivo intervalo. Todas as coordenadas que não apresentem um intervalo válido são descartadas, de forma a não enviesar o *dataset*. Nesta fase, já dispomos das coordenadas *raw* GPS no formato pretendido em memória, devidamente catalogadas e devidamente ordenadas pela sua data respetiva, como se pode observar nas instruções já citadas anteriormente.

No casa do *dataset* do SenseMyCity, este *dataset* é constituído por 2 ficheiros csv, previamente extraídos, por ordem temporal crescente. Estes ficheiros foram gerados a partir de uma base de dados em PostgreSQL, situada nos servidores privados do IT, Instituto das telecomunicações em Portugal. Um dos ficheiros representa intervalos catalogados por um modo de locomoção, e o outro contém todas as coordenadas *raw* de GPS recolhidas pelo utilizador durante as sessões de catalogação. É de notar que este dataset foi gerado por uma única pessoa durante aproximadamente um mês, resumindo a um total de aproximadamente dois dias de coordenadas GPS. A conversão destes dados para o formato pretendido em memória é relativamente simples, passando os dados *raw* para memória e efetuando a junção entre intervalos e coordenadas GPS. Como já tinha sido observado no *dataset* do Geolife, todas as coordenadas que não se enquadram num intervalo devidamente catalogado são descartadas, de forma a evitar que os dados fiquem enviesados.

Após estes passos, todos os dados estão prontos para passar à fase seguinte de processamento.

4.1.2 Geração de características de alto nível

Depois da conversão dos *datasets* em dados estruturados de forma a que possam ser usados pelo Java, é necessário convertê-los em características de alto nível. Estas características têm exigências a nível de espaçamento temporal muito críticas. Caso exista uma diferença igual a 6 segundos ou mais, entre duas coordenadas consecutivas, considera-se que não é uma sequência

válida, ou seja, uma sequência que não é capaz de descrever de forma válida um modo de locomoção. O processo de construção de uma instância composta por características de alto nível é um processo iterativo, coordenada a coordenada, analisando o espaçamento temporal a cada coordenadas com a anterior, até atingir a janela pretendida. Caso ocorra a verificação temporal verificada em cima e existirem pontos suficientes que caracterizem um modo de locomoção até aquele ponto, a instância é gerada tendo em conta só aquele conjunto de pontos. No entanto, este comportamento só é aplicado ao *dataset* do SenseMyCity, visto a quantidade de dados ser reduzida e este comportamento ser extremamente raro de acontecer, visto o espaço entre coordenadas ser de um segundo. À medida que cada coordenada é gerada, as velocidades e acelerações são calculadas conforme a disponibilidade dos dados. Duas coordenadas permitem calcular uma velocidade, e três coordenadas permitem calcular duas velocidades que conseguem determinar uma aceleração. Este processo repete-se até chegar ao número de coordenadas definido como janela. Depois de ser atingido o número máximo de coordenadas, todas as características de alto nível são calculadas, tendo em conta as velocidades e acelerações computadas nas iterações anteriores. A geração de cada uma das características é descrita na secção 3.4.1, como também a sua implementação. As definições são descritas exatamente como foram implementadas no código Java. A precisão usada é em Double em vez de Float, devido a problemas de leitura na conversão entre o tipo String e o tipo Float para o *dataset* do Geolife. Isto foi observado durante as validações da *framework* desenvolvida, para demonstrar a sua veracidade. Estas validações são compostas por um conjunto de testes em JUnit [Jt], uma *framework* de testes para Java, usando um conjunto limitado de coordenadas preestabelecidas, de forma sequencial, como pode ser observado na figura 4.1.

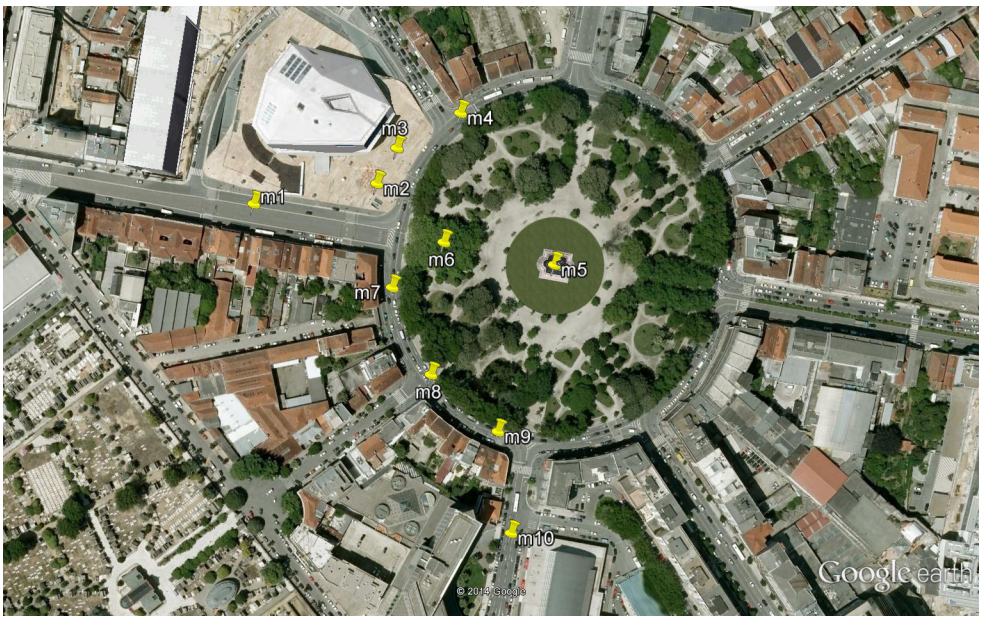

Figura 4.1: Imagem representativa dos pontos criados para testar a geração de características de alto nível, ordenados por ordem crescente.

Através desta sequência de pontos, calculámos os resultados previstos e efetuámos *match* com os resultados computados pelas implementações efetuadas, de forma a garantir que estas características de alto nível estavam a ser bem calculadas. É de notar que entre cada um destes pontos foi dado um determinado tempo de percurso, em segundos, de forma a ser possível o cálculo de velocidades e acelerações da forma correta.

4.1.3 Geração de *datasets*

Depois de confirmada a correta geração das características de alto nível, os dados gerados são guardados no formato *Attribute-Relation File Format* (ARFF) [Wek09], um formato contendo um conjunto de instâncias que partilham um conjunto igual de características. É de notar que para o caso do *dataset* SenseMyCity os dados estão organizados de forma sequencial, ou seja, as instâncias geradas a partir deste *dataset* estão ordenadas por ordem crescente. Esta característica é importante para a experiência com melhoria usando HMM, designadamente a experiência quatro, presente na secção 5.6, porque precisamos de um conjunto de dados sequências como treino para fornecer ao classificador HMM.Para o caso do SenseMyCity, os dados primeiramente são gerados no formato SEQ, um formato usado para representar um conjunto de instâncias sequenciais no tempo, juntamente com a biblioteca JAHMM [Fra06], e só depois são convertidos para o formato ARFF, mantendo a ordem necessária para a experiência 4. Isto deve-se ao facto de existir em viagens com número limitado de coordenadas, e para se poder testar o conjunto de instâncias designadas para teste; é necessário não misturar os dados destas viagens durante todo o processo de teste. Por outras palavras, não podemos fornecer ao classificador HMM uma sequência que seja composta por partes de duas viagens distintas, sendo necessário garantir que cada sequência fornecida ao classificador para teste pertence a uma e única viagem.

Um processo importante é a geração de *datasets* de teste e treino a partir do *dataset* gerado com características de alto nível, como discutido anteriormente. Esta divisão é de aproximadamente 50%, tentando sempre equilibrar o número de instâncias entre cada uma das classes que o compõe. Como é necessário garantir a ordem temporal das instâncias que compõem o *dataset* do SenseMyCity, foi preciso criar processos de separação diferentes para cada *dataset* usado. Mais detalhe sobre este processo está presente no capítulo 5.

De forma a podermos proceder com as experiências propostas neste trabalho, foi necessário criar *datasets* com menos classes a partir dos *datasets* de treino e teste. Isto deve-se ao facto de uma das estratégias ser composta por duas camadas, como descrito secção 5.5, que por sua vez requer *datasets* específicos para as duas camadas. Várias aplicações foram criadas de forma a gerar estes *datasets*, tentando sempre equilibrar as classes constituintes dos *datasets* gerados.

4.1.4 Classificação

Após conclusão deste processo de geração de todos os *datasets*, podemos proceder à criação do modelo usando os *datasets* de treino para o efeito, e os *datasets* de teste para testar o modelo gerado. Para simplificar o processo de implementação e integração com a implementação Android,

decidiu-se usar o Weka [oW14], uma ferramenta construída em Java capaz de criar modelos de classificação para diversos algoritmos de *machine learning*, entre outras funcionalidades. Esta ferramenta também permite guardar os modelos gerados e obter uma matriz de confusão tendo um *dataset* de teste e não só. Desta forma, podemos usar a API deste programa, como uma biblioteca em Java, para usar os recursos disponibilizados na implementação dos classificadores.

Primeiro geramos o modelo através da *Graphical User Interface* (GUI) do Weka, a interface gráfica amiga do utilizador que esta ferramenta oferece e que por sua vez simplifica o processo de criação de modelo. Após a gravação do modelo, e usando a biblioteca em Java do Weka, podemos carregar esse modelo para a memória, fornecendo instâncias para inferir modos de locomoção a partir do mesmo. É de ter em conta que é necessário fornecer um *dataset* num formato reconhecível ao modelo, tal como no formato ARFF já anteriormente mencionado. De forma a que todo o processo da experiência 3 se concretize de modo automatizado, foi necessário fornecer os *datasets* com o conjunto de classes correto, de forma a que a implementação não gerasse qualquer erro de incompatibilidade.

4.1.5 Modulação de dependência temporal entre instâncias com HMM

Após a classificação, o processo de melhoria com HMM é efetuado também em Java, através de uma biblioteca com implementações relacionadas com HMM, nomeadamente JAHMM [Fra06]. Com esta biblioteca é possível criar um modelo, em formato HMM, fornecendo um *dataset* no formato SEQ, capaz de prever, dando uma sequência temporal de dados, o próximo modo de locomoção, neste contexto. Como o tempo é reduzido para implementações complexas de HMM, baseamos os *datasets* de treino e teste no modo de locomoção. Como já tinha sido observado anteriormente, para aplicar este passo, só o poderemos fazer com sequências válidas, neste caso, com o *dataset* do SenseMyCity. Como podemos observar, este classificador é capaz de prever o próximo modo de locomoção, sendo necessário gerar um ficheiro auxiliar, contento de forma sequencial, todas as classificações resultantes usando o *dataset* SenseMyCity.

Analisando em mais detalhe o modelo gerado, convém advertir que a biblioteca não dispõe de uma ferramenta de processamento que carregue um modelo existente num ficheiro para um modelo funcional. Desta forma, foi criado um programa, em Java, que lê um ficheiro HMM, valida-o, e converte-o num modelo funcional com a biblioteca JAHMM. Este programa foi validado comparando os resultados obtidos, usando a ferramenta de conversão e o modelo logo após a geração do modelo pelo JAHMM.

4.2 Implementação em Android

Esta secção engloba, de uma forma geral, como foi procedida a implementação da solução final na plataforma Android, um sistema operativo desenvolvido pela Google, usado atualmente em milhares de *smartphones*. Como a principal linguagem de programação para Android é Java, fica, de certa forma, facilitada a integração de todo o processo nesta plataforma, desde recolha de dados até classificação atual. A maior parte do código desenvolvido nas experiências foi reutilizado na

implementação da aplicação Android, facilitando a integração. Primeiramente, a aplicação reconhece quando o sensor GPS está desligado ou não, permitindo poupar recursos, como coordenadas antigas. Pela documentação disponibilizada aos programadores da plataforma Android [Goo14], podemos concluir que implementando um *Listener* com um tempo fixo para obter dados GPS, não é garantido que essa notificação ocorra. Desta forma, a cada segundo, uma chamada é feita para determinar o último ponto válido obtido pelo sensor GPS. Assim, obtemos pontos de segundo a segundo, evitando o problema de ficar à espera de ser notificado por novos pontos. Esta recolha de segundo a segundo é feita através de um *Timer*, uma classe no Java que permite correr uma tarefa passados X segundos, sendo neste caso 1 segundo, de uma forma eficiente, sem causar grande impacto no processador do dispositivo móvel.

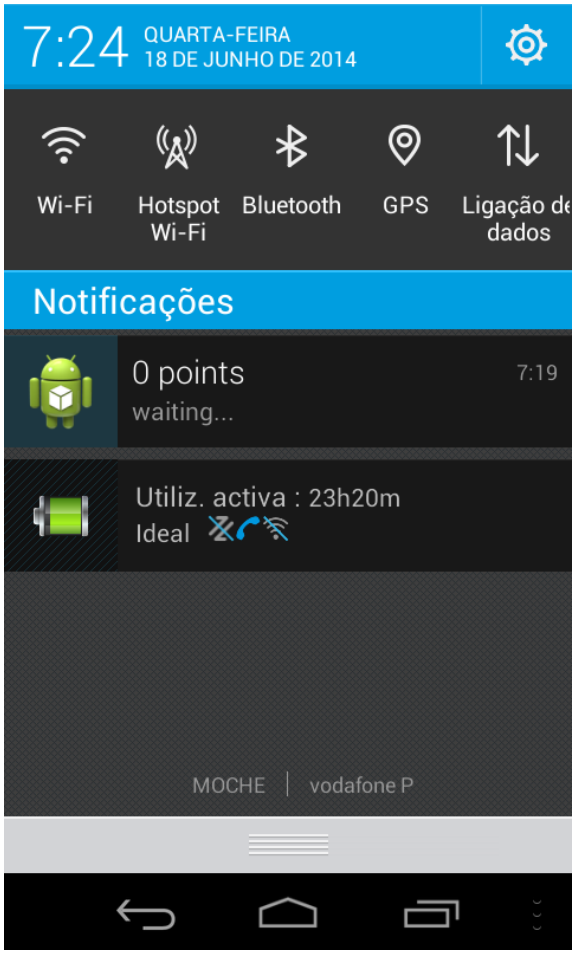

Figura 4.2: Notificação persistente da aplicação Android desenvolvida em modo de espera.

Todas estas coordenadas fornecidas são adicionadas à fila de processamento de pontos, na *Thread* principal do serviço. Cada um destes pontos é analisado ao mesmo nível que o pré-processamento, sendo aplicados filtros já discutidos na secção 3.2. Depois de atingir a janela temporal das 9 coordenadas *raw* GPS, são geradas todas as características de alto nível, encapsuladas numa instância reconhecível pelo modelo, e fornecidas ao classificador, originando uma classificação. A aplicação desenvolvida tem uma interface bastante simples para que se obtenha

algum *feedback*, podendo confirmar se a solução implementada produz resultados. Nas figuras 4.2, 4.3 e 4.4 podemos observar um exemplo desta interface, quando o algoritmo está em espera e durante a inferência de modos de locomoção em tempo real. O processo de geração de características de alto nível é exatamente igual ao já descrito na subsecção 4.1.2, excluindo o facto de serem necessários ficheiros adicionais para melhoria, tendo em conta os resultados obtidos na secção 6.4.

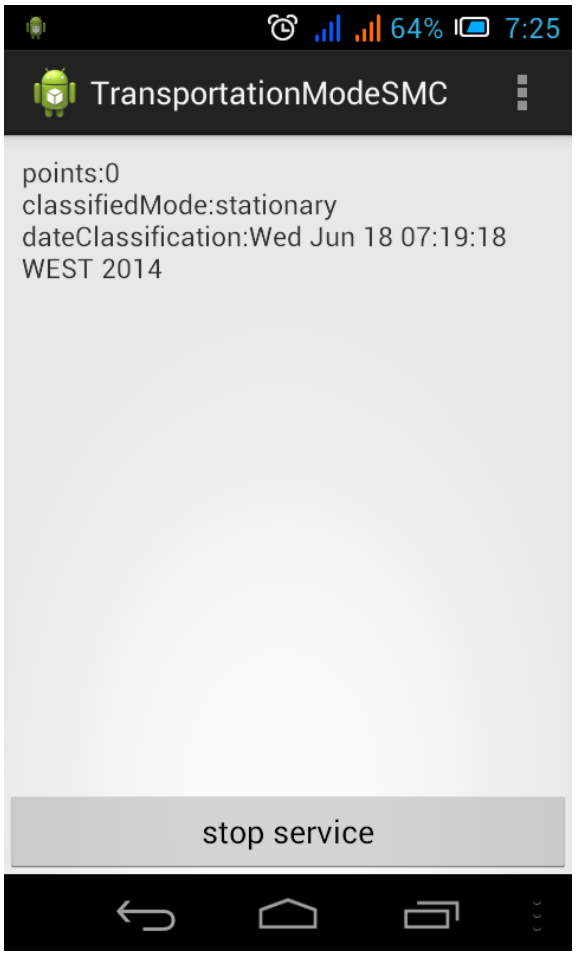

Figura 4.3: interface da atividade da aplicação Android desenvolvida em modo de espera.

É de notar que foi necessária a inclusão de uma biblioteca Weka modificada, que fosse compatível com a plataforma Android. Isto deve-se ao facto da biblioteca oficial desenvolvida do Weka usar bibliotecas nativas do Java que não são suportadas no Java presente nas plataformas Android. Assim, usamos a biblioteca "Weka-for-Android" [Mar11], uma biblioteca adaptada pelo programador RJ Marsan. Como esta adaptação baseia-se numa versão mais antiga do Weka, foi necessário gerar um modelo compatível com esta biblioteca, obtendo-se uma redução menor que 1% em comparação com a biblioteca original. Para gerar este novo modelo foram utilizados os mesmos parâmetros usados na biblioteca original. De forma a simplificar o processamento da aplicação desenvolvida, esta foi criada sob a forma de um serviço em segundo plano, facilitando a recolha de dados e o processamento da mesma. Isto permite que o utilizador não esteja constantemente com a aplicação em primeiro plano e que o processamento não interfira com outras tarefas

executadas pelo utilizador no seu dispositivo Android.

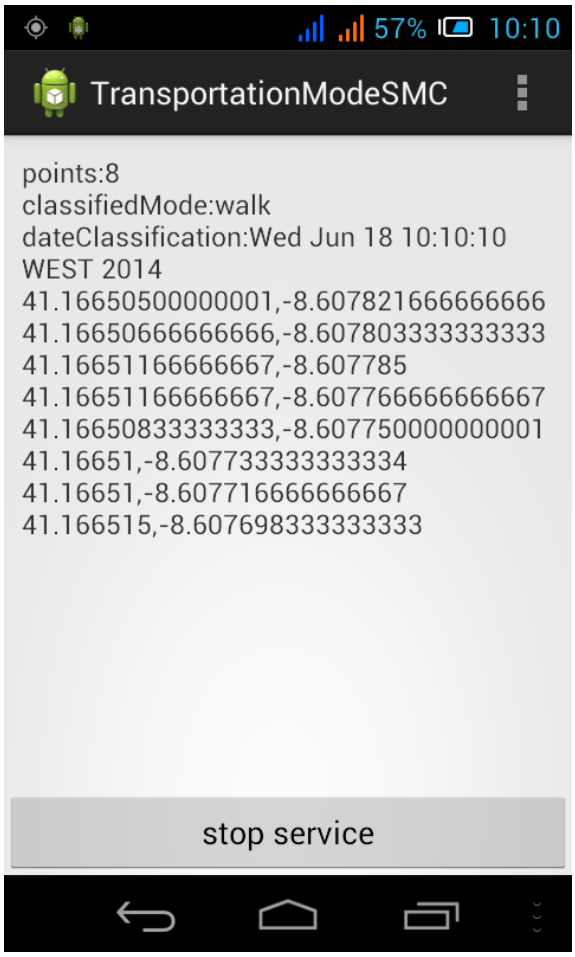

Figura 4.4: Interface da atividade da aplicação Android desenvolvida em modo de inferência de modos de locomoção em tempo real, neste caso no modo *walk*, contendo informações referentes a pontos a serem usados e hora de classificação.

4.3 Resumo

Podemos concluir que foi criado um conjunto de programas capazes de processar os dados *raw* até inferências de modos de locomoção. Cada um destes programas foi desenvolvido tendo em conta a estrutura que compõe cada um dos *datasets* usados, de forma a criar novas estruturas lógicas capazes de gerar novos tipos de dados, como características de alto nível. A partir da flexibilidade presente no Java, foi possível implementar a *framework* completa em Java, usando bibliotecas específicas capazes de resolver problemas classificativos, e também problemas de modulação de dependência temporal. Certas validações tiveram de ser aplicadas de forma a validar os programas desenvolvidos para as experiências descritas no capítulo 5. A implementação Android foi feita através da reutilização do código já desenvolvido para as experiências realizadas e

de uma biblioteca adicional, compatível com a plataforma, evitando grandes problemas de integração. Mesmo com uma versão mais antiga da ferramenta Weka compatível com Android, foi possível obter um decréscimo menor que 1% quanto à exatidão alcançada com a geração de um modelo compatível com a implementação Android.

Capítulo 5

Avaliação da solução

Neste capítulo é descrito que *datasets* são usados na secção 5.1, que medidas são usadas para avaliação das experiências na secção 5.2, e todas as experiências efetuadas a nível procedimental, nomeadamente a experiência sobre a influência do tamanho da janela temporal na secção 5.3, a experiência sobre seleção de características na secção 5.4, a experiência sobre as estratégias de classificação na secção 5.5 e a experiência sobre modulação de dependência temporal das instâncias recorrendo ao classificador HMM na secção 5.6. Também neste capítulo é descrito como é que os *datasets* são divididos em *datasets* de treino e teste, na secção 5.7.

5.1 *Datasets*

De forma a podermos proceder com as experiências, é necessário ter fontes de dados bem explícitas. Para esta dissertação são necessários *datasets* que contenham coordenadas *raw* GPS devidamente catalogadas por um modo de locomoção, para que seja possível criar modelos de classificação capazes de inferir modos de locomoção. Deste modo, decidimos usar dois *datasets*, apresentados na seguinte enumeração.

- ***Dataset* Geolife**: esta *dataset* foi disponibilizado para uso pela Microsoft Research Asia, é baseado num conjunto de coordenadas GPS sendo a maior parte catalogada com modos de locomoção; é de notar que este *dataset* é um subconjunto do *dataset* completo do projeto Geolife [Zhe07], mas só que adicionado o componente de catalogação, que não está incluído no *dataset* completo do projeto Geolife; este *dataset* foi construido com dados recolhidos por 65 utilizadores durante 10 meses, maioritariamente na Ásia.

- ***Dataset* SenseMyCity**: este *dataset* foi construido usando dados catalogados na aplicação SenseMyCity, totalizando um total de aproximadamente 2 dias de dados recolhidos, durante um mês; os dados usados para construir este *dataset* foram recolhidos pelo autor desta

dissertação, nomeadamente nas áreas metropolitanas do Porto, distrito de Santarém e na ilha da Madeira, tudo regiões integrantes de Portugal.

Com estes *datasets* será possível efetuar comparações entre eles, comparando os resultados obtidos com a literatura, usando o *dataset* Geolife, e analisar possíveis melhorias utilizando o *dataset* do SenseMyCity, dado ser possível reproduzir viagens totalmente representadas por vários modos de locomoção.

5.2 Medidas de avaliação

As medidas utilizadas para avaliar o performance do sistema são as tipicamente utilizadas nos trabalhos relacionados: exatidão, precisão e sensibilidade.

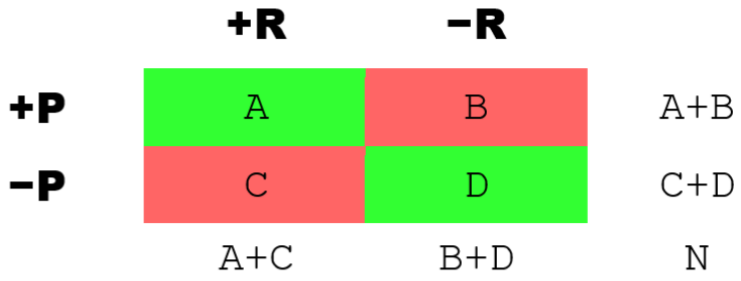

Figura 5.1: Exemplo de uma matriz de confusão binária, onde a cor verde representa classificações corretas e vermelho representa classificações incorretas [Pow07]. A corresponde ao número de instâncias bem classificadas como corretas, B representa o número de instâncias mal classificadas como corretas, C corresponde ao número de instâncias mal classificadas como corretas, D corresponde ao número de instâncias bem classificadas como incorretas, $+P$ corresponde ao número de instâncias classificadas como corretas, $-P$ corresponde ao número de instâncias classificadas como incorretas, $+R$ corresponde ao número de instâncias corretas, $-R$ corresponde ao número de instâncias incorretas, e N corresponde à exatidão resultante da tabela de confusão.

- **Exatidão [Pow07]**: esta medida permite determinar a percentagem de instâncias que foram corretamente classificadas no conjunto de todas as classificações. A equação 5.1 mostra como é calculada esta medida, tendo em conta as tabelas presentes na figura 5.1.

$$N = \frac{A+D}{A+B+C+D} \tag{5.1}$$

  Analisando a equação 5.1, e tendo em conta as variáveis das tabelas presentes na figura 5.1, N representa a exatidão, $A+D$ representa a soma de todas instâncias bem classificadas e $A+B+C+D$ representa conjunto de todas as instâncias classificadas;

- **Precisão [Pow07]**: esta medida devolve a percentagem de instâncias classificadas sem erros, de entre todas as instâncias classificadas para uma determinada classe. Desta forma,

podemos analisar se, para uma determinada classe, o classificador apresenta muitos erros. A equação 5.2 mostra como esta medida é calculada para a classe positivos, tendo como base as tabelas presentes na figura 5.1.

$$P_p = \frac{A}{A+B} \tag{5.2}$$

Analisando a equação 5.2, e recorrendo às variáveis das tabelas presentes na figura 5.1, P_p representa a precisão para a classe positivos, A corresponde ao número de instâncias positivas bem classificadas, e $A+B$ representa o conjunto de instâncias classificadas como positivas;

- **Sensibilidade [Pow07]**: esta medida devolve a percentagem de instâncias classificadas corretamente, de entre todas as instâncias constituintes de uma determinada classe. Desta forma podemos saber se o classificador consegue identificar maior parte das instâncias correspondentes à sua verdadeira classe. A equação 5.3 mostra a fórmula para calcular a sensibilidade para a classe positivos, presente nas tabelas da figura 5.1.

$$S_p = \frac{A}{A+C} \tag{5.3}$$

Analisando a equação 5.3, e recorrendo às variáveis das tabelas presentes na figura 5.1, S_p corresponde à sensibilidade da classe positiva, A corresponde ao número de instâncias positivas bem classificadas, e $A+C$ corresponde ao conjunto de instâncias pertencentes à classe positiva.

Através destas medidas, conseguimos comparar resultados a nível de exatidão e também conseguimos analisar o desempenho geral para cada classe, tendo em conta a precisão e sensibilidade das classes classificadas.

5.3 Experiência 1: Influência do tamanho da janela temporal

A geração de características é feita através de um conjunto finito de coordenadas *raw* GPS, sendo diretamente dependente do número de coordenadas usadas para cada instância de características de alto nível gerada. Como o tamanho da janela temporal é fixo para as soluções propostas, é necessário concluir, para cada um dos *datasets* usados, qual a melhor janela temporal a utilizar. Usamos o termo janela temporal porque os pontos não podem exceder um *threshold* de 6 segundos entre duas coordenadas, permitindo ter uma estimativa temporal para cada janela definida. Decidimos usar janelas baseadas na quantidade de coordenadas *raw* GPS em vez de janelas baseadas em tempo, pelo facto de nem sempre podermos garantir um tamanho fixo de tempo Isto deve-se ao facto de o tempo que separa cada uma das coordenadas *raw* GPS não ser fixo no *dataset* do Geolife. Se seguíssemos esta abordagem, teríamos de descartar pontos desnecessariamente, que são válidos para este contexto. Usando a abordagem com coordenadas *raw* GPS, podemos

também criar intervalos de tempo aproximados, que traduz de certa forma as janelas temporais. Primeiramente, decidimos dar um conjunto de janelas temporais entre as 5 coordenadas e as 20 coordenadas *raw* GPS, nomeadamente 5, 8, 9, 10, 11, 15 e 20 coordenadas. É de notar que aplicamos maior número de janelas temporais entre 8 e 10 por começar a ter mais do que uma aceleração possível de ser calculada. Considerados que abaixo de 8 o número de acelerações que podiam ser calculadas não fosse suficiente, e um número maior que 11 já poderia induzir noutros tipos de erros, como adição de outros modos de locomoção. No caso do valor de exatidão alcançado ser um extremo do intervalo de janelas temporais sugeridas, é adicionado um novo conjunto de janelas temporais até atingir um pico estabilizado.

De forma a procedermos com estas classificações, foi usado um subconjunto dos classificadores que apresentam melhores resultados na literatura, tendo como destaque o trabalho que criou o *dataset* Geolife para catalogação de modos de locomoção [ZCL+10, ZQY13]. Desta forma, este subconjunto de classificadores é constituído pelos classificadores mais usados na literatura, nomeadamente *Support Vector Machines* (SVM) e *Decision Tree* (DT), e também pelo *Random Forest* (RF), pelo facto de este apresentar, de uma forma geral, melhores resultados que o DT, por ser um *ensemble* de um conjunto finito de DT. É de notar que esta experiência é efetuada pelo programa Weka.

Outro fator muito importante é usar o mesmo conjunto de instâncias produzidas em todas as janelas temporais. Isto é uma tarefa complicada, porque necessita que todas as coordenadas *raw* GPS que constituíram o conjunto de instâncias seja também igual . De forma a podermos manter o maior número possível de coordenadas constantemente iguais, temos de analisar qual a classe que apresenta o menor número de instâncias, e fixar esse como o número total de instâncias que cada classe pode ter para cada *dataset* gerado, dado o tamanho de uma janela temporal. Decidimos gerar os *datasets* com estas restrições devido a dois problemas, nomeadamente, usando o maior tamanho de janela temporal para calcular o tamanho de cada classe, para todos os outros tamanhos, induzindo em *datasets* com poucas instâncias, e usando o tamanho de janela temporal mais pequeno, produzindo classes incompletas, conforme se aumente o tamanho da janela temporal.

Quando esta experiência estiver completa, a melhor janela temporal para cada *dataset* será o tamanho fixo utilizado para as restantes experiências, inclusive na solução final.

5.4 Experiência 2: Seleção de características

Além de existir um conjunto finito de características sugeridas na literatura e no âmbito desta dissertação, é necessário avalia-las através de classificadores que consigam calcular o ganho de informação que estas características possam ter, mediante os *datasets* disponíveis, de forma a podermos reunir as N melhores características e testar se é de facto vantajoso utilizar unicamente as N melhores para determinado *dataset*. Esta experiência é composta por duas fases: a utilização de um classificador de seleção de características capaz de atribuir um *ranking* das melhores por ordem decrescente de *rank*, e obtenção da melhor exatidão, para cada *dataset*, recorrendo unicamente às N melhores características apontadas pela fase anterior.

A primeira fase consiste em utilizar um classificador de seleção de características, sendo escolhido o Relief-F [KvRv97], como já explicado na secção 3.3, de forma a ser possível construir uma listagem por ordem decrescente de *ranking*. Os *datasets* usados para esta fase são os de treino para cada um dos tipos de *datasets* utilizados, nomeadamente Geolife e SenseMyCity. Tendo concluído o processo de classificação de todas as características para cada um dos *datasets*, procedemos para a segunda fase. Esta fase define um conjunto das N melhores características classificadas pelo Relief-F. Como só existe um conjunto de 15 características, decidimos filtrar cada um dos *datasets* de forma a só conterem as 5 e 10 melhores, em adição ao *dataset* completo. Desta forma, temos três testes a efetuar para cada um dos *datasets*. Para esta fase será usado o conjunto de classificadores já previamente definidos na secção 3.5, porque é importante analisar para todo o conjunto de classificadores qual a melhor exatidão obtida, com ou sem seleção de características.

Após a conclusão desta experiência é esperado poder concluir-se, perante os *datasets* disponíveis, se é vantajoso a utilização de seleção de características de forma a obter melhores resultados de exatidão no processo de inferência de modos de locomoção.

5.5 Experiência 3: Estratégias de classificação

Após decisão acerca do tamanho da janela temporal e efetuada uma análise se é vantajoso ou não, mediante os *datasets* disponíveis, usar seleção de características, é necessário proceder à construção do modelo que apresenta melhores resultados a nível de exatidão. Esta experiência consiste em testar todas as camadas constituintes de cada estratégia sugerida, para concluir qual o melhor classificador a usar em cada caso, comparando os *datasets* de forma a podermos analisar os melhores resultados e a literatura existente. Em todas as fases de teste desta experiência, usamos todos os classificadores.

Esta experiência envolve análise de duas estratégias: uma com duas camadas, sendo a primeira dedicada a separar o modo de locomoção andar dos modos de locomoção motorizados, e a segunda camada dedicada a diferenciar os distintos modos de locomoção, e outra estratégia com uma única camada, classificando todas as classes existentes numa única camada. A primeira estratégia foi sugerida nesta dissertação por ser possível separar melhor as classes, diminuindo a complexidade de cada camada, e aumentando a complexidade da estratégia, convertendo numa solução com duas camadas em vez de uma. A segunda estratégia é a que maioritariamente é utilizada mais frequentemente na literatura existente.

Para a estratégia com duas camadas é necessário, em primeiro lugar, selecionar o classificador que melhor se adequa a cada uma delas. Desta forma, dois conjuntos de *datasets*, devidamente repartidos em treino e teste, foram usados para avaliar a primeira e a segunda camadas, para encontrar o classificador com melhor exatidão. É de notar que para obter os *datasets*, para cada camada para treino e teste, foi necessário converter e filtrar os dados através de programas construídos neste tema de dissertação, como referido na subsecção 4.1.3. Após a seleção dos classificadores que apresentaram melhor exatidão, analisamos ao detalhe, numa tabela de confusão, as medidas

precisão, sensibilidade e exatidão de cada uma das fases, nomeadamente primeira e a segunda camadas, tendo em conta que a primeira é perfeita e analisamos ainda todo o conjunto das camadas, considerando os erros provenientes da primeira. Desta forma será exequível a análise de possíveis problemas encontrados, que possam justificar os resultados obtidos.

Para a estratégia com uma camada, é necessário proceder à escolha do classificador que apresenta uma melhor exatidão. Posteriormente, foi analisada a respetiva matriz de confusão, examinando as medidas precisão, sensibilidade e exatidão. Finalmente, a exatidão de cada uma das estratégias, para cada um dos *datasets* usados, foi comparada, analisando as medidas de cada estratégia e de cada matriz de confusão respetiva, de forma a concluir qual a melhor estratégia e porque razão a outra estratégia não se mostra vantajosa.

Após conclusão desta fase de seleção de estratégia, uma comparação entre os resultados obtidos é efetuada, de forma a comparar o que pode ou não ter acontecido, caso existam grandes alterações de resultados obtidos entre os dois *datasets*. Foram também analisadas que estratégias são as melhores para ambos os *datasets* e classificadores. Finalmente foi efetuada uma comparação com a literatura existente, recorrendo apenas aos artigos que usem unicamente sensores GPS para o ato de inferência de modos de transporte. Desta forma, podemos situar os resultados obtidos, tanto para o *dataset* Geolife como para o *dataset* SenseMyCity.

Com esta experiência obtivemos uma análise completa sobre: os melhores classificadores a usar, a melhor estratégia a utilizar e o modo como os resultados obtidos se localizam na literatura existente, tendo em conta os dados usados para esta experiência.

5.6 Experiência 4: Modulação da dependência temporal entre instâncias com HMM

Após a realização de todas as avaliações referentes à classificação, foi analisado se seria vantajoso aplicar um processo classificativo adicional, capaz de melhorar o resultado obtido, usando os resultados já alcançados anteriormente para obter uma previsão de qual será a próxima classificação mais provável. Esta experiência baseia-se na criação de um HMM, capaz de prever a próxima classe, para ser testado, comparando os resultados alcançados com os melhores resultados obtidos na secção 5.5; sendo possível analisar se é vantajoso ou não a utilização de uma melhoria, tendo em conta o HMM utilizado e os dados a serem usados.

O classificador a utilizar é o HMM, um classificador de séries temporais, capaz de prever os próximos estados, fornecendo uma sequência de dados que antecede a previsão. Antes de prever é necessário treinar, o que é esperado ser efetuado nesta experiência. Este classificador só irá usar modos de locomoção para efetuar tanto treino como teste. Os dados a serem testados nesta experiência são gerados durante a experiência da secção 5.5, conforme explicado na subsecção 4.1.5.

Com esta experiência conseguimos analisar se é vantajoso utilizar este classificador como uma melhoria adicional e, caso não o seja, conseguir concluir qual o motivo.

5.7 Divisão dos *datasets*

Uma passo muito importante antes de poder proceder com as experiências é como é que os dados estarão organizados, a nível de treino e teste. Para esta dissertação os *datasets* foram divididos em dois subconjuntos, um para treino e outro para teste, cada um com 50% das instâncias. Decidimos proceder com esta divisão por ser relativamente simples de aplicar, além do facto de na literatura relacionada com o *dataset* Geolife não ser referido avaliações com validação cruzada, mas sim uma divisão no *dataset* Geolife, não especificando a divisão [ZCL+10, ZQY13].

Além de efetuar uma divisão de 50%, é necessário tentar equilibrar as classes constituintes de cada *dataset*, de forma a que o classificador não tenda para classes mais bem representadas que outras. Desta forma, é tido em conta que o *dataset* completo que será usado para divisão já se encontra equilibrado a nível de classes, permitindo no ato da divisão equilibrar cada um dos *datasets* resultantes com um número igual de classes. No entanto, a experiência da secção 5.5 requer construção de *datasets* mais refinados, como criar um *dataset* de treino e teste com duas classes, nomeadamente andar e motorizado. Também esta mesma experiência requer a criação de um *dataset* de treino e teste contendo todas as classes motorizadas. No caso do *dataset* do SenseMyCity, é necessário incluir a classe estacionário, por consequência da experiência da secção 5.6, que requer sequências temporais, incluindo múltiplos modos de locomoção. Por causa da adição desta classe, existe ligeiro desequilibro ao tentar reequilibrar as classes na geração dos *datasets* de treino e teste.

5.8 Resumo

Neste capítulo foi definido um conjunto de experiências que visa avaliar a solução proposta e analisar os resultados obtidos, com os dados disponíveis para o efeito. Estes dados são *datasets* públicos, fornecidos pela Microsoft Research Asia, do projecto Geolife, e também *datasets* privados, neste caso, fornecidos pela aplicação SenseMyCity. Desta forma foi possível comparar diferentes resultados e comparar com a literatura existente os resultados alcançados. Tendo os *datasets* definidos, a primeira experiência permite testar qual é a melhor janela temporal a usar, a segunda experiência permite concluir se é vantajoso ou não a utilização de seleção de características para as próximas experiências, a terceira permite concluir quais são os melhores classificadores a aplicar em cada estratégia, determinando também qual a melhor estratégia a aplicar; podendo estes resultados ser comparados entre os dois *datasets*, como também entre a literatura existente; finalmente, a última experiência permite analisar se é vantajoso ou não a utilizar uma melhoria com HMM, recorrendo para o efeito aos resultados obtidos da terceira experiência.

Capítulo 6

Resultados

Neste capítulo são apresentados todos os resultados alcançados com as experiências descritas no capítulo 5. Primeiramente, na secção 6.1, são descritos os resultados da influência do tamanho da janela temporal, na secção 6.2 descrevem-se os resultados da seleção de características, na secção 6.3 apresentam-se os resultados para as estratégias de classificação usadas e finalmente, na secção 6.4, são expostos os resultados com modulação de dependência temporal entre instâncias com *Hidden Markov Models* (HMM).

6.1 Experiência 1: Influência do tamanho da janela temporal

A geração de características de alto nível é feita a partir de um conjunto de pontos consecutivos (janela temporal), de tamanho pré-definido. De forma a analisar o efeito do tamanho da janela temporal na performance de classificação do modo de locomoção, é necessário testar os diferentes possíveis tamanhos que esta janela temporal pode tomar. Assim, podemos observar nas figuras 6.1 e 6.2 as diferentes janelas temporais analisadas e os respetivos resultados para um conjunto pré-selecionado de classificadores que trazem melhor relevância para este contexto. Para estas experiências foram utilizados os classificadores *Support Vector Machines* (SVM) e *Decision Tree* (DT), por serem usados em vários artigos na literatura e foi preferido o *Random Forest* (RF) pelo facto de apresentar sempre melhores resultados, de uma forma generalizada, em relação aos resultados gerados pelo DT.

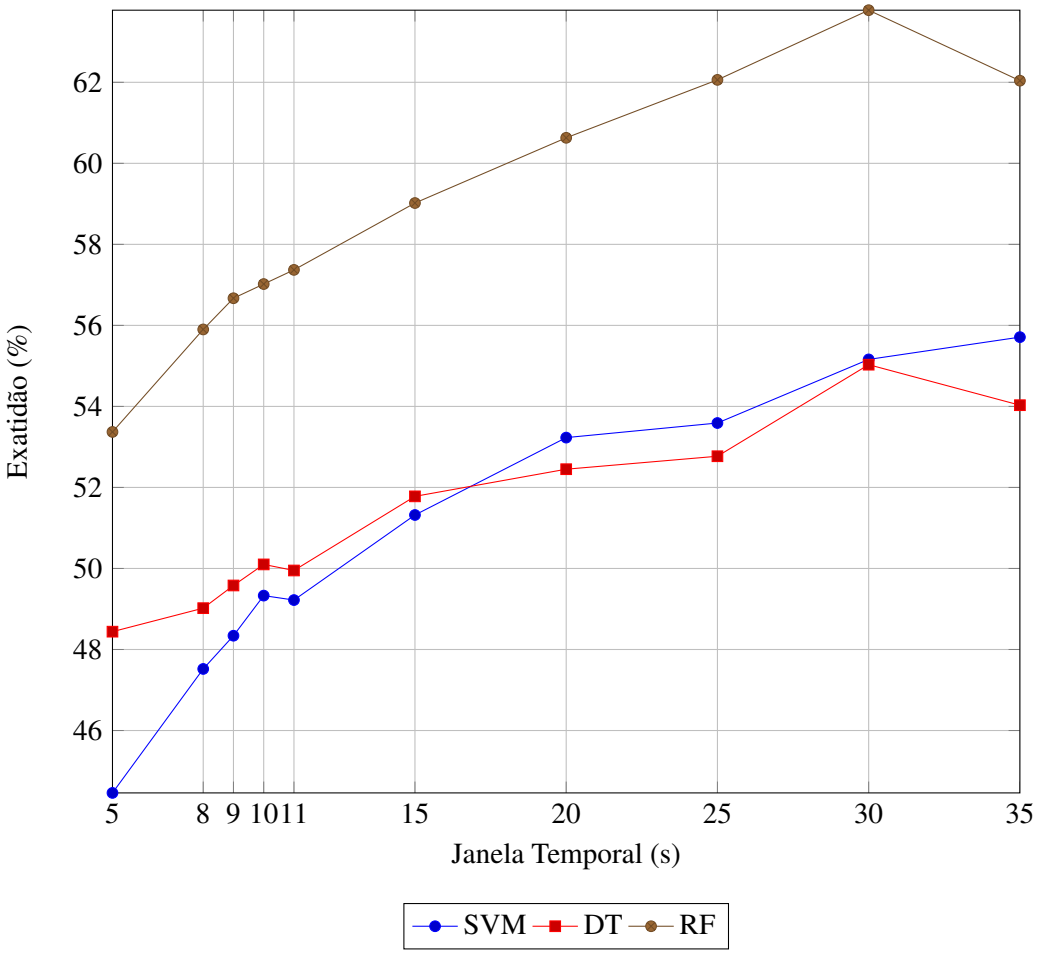

Figura 6.1: Gráfico representando a evolução da exatidão demonstrada na variação do tamanho da janela temporal para o *dataset* Geolife.

Através da análise dos resultados visíveis no gráfico da figura 6.1, concluiu-se que a melhor janela se situa nos trinta pontos para cada cálculo de caraterísticas de alto nível. O melhor resultado usando SVM está na janela de trinta e cinco pontos, com uma exatidão de aproximadamente 55.71%. O melhor resultado usando DT e RF situa-se na janela dos trinta pontos, com 55.03% e 63.78%, respetivamente. De entre os três classificadores, o que mais se destaca é claramente o RF, apresentando o melhor resultado de exatidão.

De forma a usar sempre os mesmos conjuntos de pontos no *dataset* de treino e teste, foi necessário limitar o número de pontos a serem utilizados para cada classe, a partir da classe que apresenta o menor número de pontos disponíveis. É de notar que esta limitação é necessária para garantir a utilização do maior número de dados equivalentes nesta experiência.

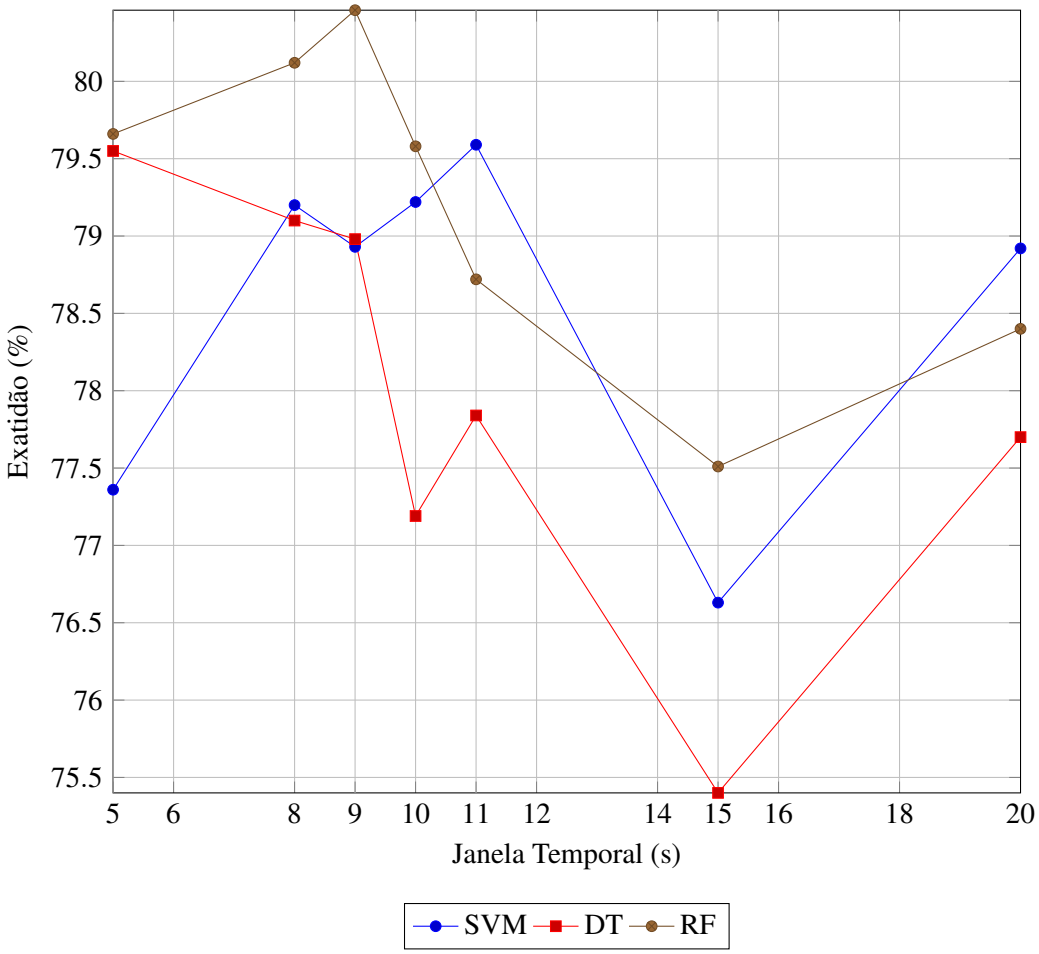

Figura 6.2: Gráfico representando a evolução da exatidão demonstrada na variação do tamanho da janela temporal para o *dataset* SenseMyCity.

Pela análise dos resultados presentes no gráfico da figura 6.2, pode-se concluir que a melhor janela para o *dataset* SenseMyCity está entre os cinco e os quinze pontos. O melhor resultado para o SVM situa-se na janela dos quinze pontos, com uma exatidão de 79.63%, o melhor resultado para o DT está na janela dos cinco pontos, com uma exatidão de 79.55% e, finalmente, o melhor resultado para RF situa-se na janela dos nove pontos, com uma exatidão de 80.46%. Assim, a melhor janela a usar é constituída de nove pontos.

6.2 Experiência 2: Seleção de características

A existência de várias características de alto nível requer a análise destas, de forma a se poder determinar se a remoção de algumas características traz uma vantagem significativa. Usando o Relief-F [KvRv97], um algoritmo que calcula a relevância de cada característica no que diz respeito à sua capacidade de inferência, obtivemos uma lista das características ordenada por relevância de acordo com o critério do Relief-F para cada um dos *datasets* usados. É de notar que para esta experiência foram usados exclusivamente *datasets* de treino.

Para o *dataset* do Geolife, foi gerada a tabela 6.1 por ordem decrescente de importância.

Posição	Característica	Pontuação
1º	diferença da direção	0.01
2º	média da velocidade	0.009
3º	mediana da velocidade	0.009
4º	desvio padrão amostral da velocidade	0.009
5º	velocidade mínima	0.009
6º	velocidade máxima	0.007
7º	aceleração máxima	0.006
8º	desvio padrão amostral da aceleração	0.006
9º	mediana da aceleração	0.005
10º	aceleração mínima	0.005
11º	média da aceleração	0.005
12º	HCR	0.004
13º	VCR	0.001
14º	distância máxima	0.001
15º	SR	0.001

Tabela 6.1: Lista de características ordenada por ordem decrescente de relevância para inferência de locomoção no *dataset* Geolife.

Para podermos concluir se é vantajoso aplicar a seleção de caraterísticas, foi efetuada uma experiência com as 5, 10 e 15 melhores características, segundo o critério do Relief-F. Estes resultados podem ser observados na tabela 6.2, que contém a utilização de todos classificadores previamente selecionados e todos os conjuntos de características referidos com os seus respetivos resultados de exatidão.

Num. Caract.[1] / Classificador	5	10	15
SVM	49.81%	54.94%	**55.16%**
IBK	45.64%	**48.76%**	48.64%
Bayes	47.98%	**50.75%**	46.63%
NN	52.70%	**57.60%**	55.63%
DT	53.05%	53.66%	**55.63%**
RT	45.77%	48.84%	**50.48%**
RF	55.22%	61.38%	**63.78%**

Tabela 6.2: Resultados de seleção das características para o *dataset* Geolife.

[1] Número das melhores características usadas.

Através da análise da tabela 6.2, pode-se concluir que, de uma forma generalizada, quanto menor o número de características a serem usadas neste contexto, menor é a exatidão encontrada com os classificadores de maior desempenho, designadamente, SVM, DT e RF. A melhor combinação de classificador com janela de características é com o RF e com 15 características, com 63.78%. Como o melhor resultado se encontra na janela onde nenhuma caraterística é removida, torna-se evidente a vantagem em não usar seleção de caraterísticas.

Para o *dataset* do SenseMyCity, a tabela 6.3 foi gerada também pelo Relief-F, com os mesmos parâmetros, e os resultados foram ordenados por ordem decrescente de relevância.

Posição	Característica	Pontuação
1º	distância máxima	0.139
2º	média da velocidade	0.127
3º	mediana da velocidade	0.123
4º	velocidade mínima	0.11
5º	velocidade máxima	0.038
6º	desvio padrão amostral da velocidade	0.036
7º	diferença da direção	0.025
8º	desvio padrão amostral da aceleração	0.024
9º	aceleração máxima	0.021
10º	HCR	0.019
11º	aceleração mínima	0.02
12º	média da aceleração	0.016
13º	mediana da aceleração	0.01
14º	SR	0.007
15º	VCR	0.007

Tabela 6.3: Lista de características ordenada por ordem decrescente de relevância para inferência de locomoção no *dataset* SenseMyCity.

Seguindo os mesmos passos do *dataset* anteriormente analisado, procedeu-se à utilização da seleção das melhores características por conjuntos de 5, 10 e 15 características, de forma a se poder concluir se a seleção de características é vantajosa a um nível significativo. Na tabela 6.4 é possível analisar os resultados de todas as combinações de janelas com classificadores.

Num. Caract.[1] / Classificador	5	10	15
SVM	78.26%	**79.36%**	78.93%
IBK	75.97%	75.04%	**75.97%**
Bayes	**75.55%**	75.38%	74.03%
NN	78.34%	**78.85%**	77.83%
DT	78.60%	78.93%	**79.10%**
RT	72.93%	74.87%	**75.55%**
RF	77.58%	79.36%	**80.46%**

Tabela 6.4: Resultados da seleção de características para o *dataset* SenseMyCity.

Com a análise da tabela 6.4, pode concluir-se que, de forma generalizada, na janela 15 existe maior exatidão que nas janelas 5 e 10. A combinação que tem a melhor exatidão é RF, com uma janela de 15 características, ou seja, sem seleção de características, com uma precisão de 80.46%. Tendo em conta o resultado obtido, não é vantajoso, neste caso, usar seleção de características.

Comparando os resultados para os dois *datasets* analisados, podemos notar que existe uma ordenação diferente para cada *dataset*. Para o *dataset* do Geolife, podemos notar que os resultados são mais baixos, em comparação com o *dataset* do SenseMyCity. Isto deve-se ao facto de as coordenadas GPS provenientes do Geolife não conseguirem diferenciar com tanta exatidão as diferentes características. Isto comprova-se observando os resultados do SenseMyCity, com valores superiores para maior parte das características. Assim pode concluir-se que à partida é esperado que o *dataset* do SenseMyCity consiga diferenciar melhor os modos de locomoção do que o *dataset* do Geolife. Ao nível das características mais relevantes para inferência de modos de locomoção, verifica-se que existe um agrupamento com as características das velocidades que se apresentam como as características mais relevantes, tanto para o *dataset* Geolife como para o SenseMyCity. As características com pior relevância que se podem observar são as sugeridas em [ZCL+10], nomeadamente HCR, VCR e SR. Tanto para o *dataset* do Geolife como para o SenseMyCity, estas características apresentam-se como as menos relevantes. É de notar que as características distância máxima e diferença de direção invertem a relevância, sendo a distância máxima o que mais se destaca no *dataset* SenseMyCity e a diferença de direção o que mais se destaca para o *dataset* do Geolife. No entanto, no *dataset* do SenseMyCity a característica diferença de direção apresenta uma relevância maior, em comparação com a relevância da distância máxima para o *dataset* do Geolife, que, por sua vez, é a característica com a pior relevância. Uma possível explicação para as diferenças observadas entre a distância máxima e a diferença de direção pode estar na precisão das coordenadas GPS adquiridas para o *dataset* Geolife. Visto existir imprecisão nas coordenadas GPS do *dataset* Geolife, espera-se que exista uma diferença de direção maior, devido ao erro de aquisição de coordenadas GPS, do que o erro existente nas coordenadas GPS recolhidas no SenseMyCity. Também erros de catalogação podem levar a uma mistura de outros modos de locomoção com um único modo de locomoção.

Também através desta análise, podemos concluir que é vantajoso a utilização das características sugeridas nesta dissertação. De modo a confirmar esta vantagem, foi realizada uma experiência adicional em que são removidas as características propostas nesta dissertação.

Dataset	Removendo Características propostas	Com todas Características
Geolife	60.99%	**63.78%**
SenseMyCity	79.10%	**80.46%**

Tabela 6.5: Comparação dos resultados obtidos entre usar todas as características e removendo todas as características propostas nesta dissertação, usando para isso o classificador que apresentou melhores resultados para ambos *datasets*, (RF).

Como podemos observar na tabela 6.5, os resultados obtidos usando todas as características são superiores em comparação aos obtidos não usando as características sugeridas nesta dissertação. Desta forma, as características propostas permitem incrementar os resultados atingidos, demonstrando vantagens neste contexto.

6.3 Experiência 3: Estratégias de classificação

De forma a podermos determinar qual a melhor estratégia de classificação a aplicar para cada um dos *datasets*, realizam-se vários testes para obter resultados de precisão, sensibilidade e exatidão, os quais foram comparados. Apresentamos duas soluções, uma constituída por duas camadas, e outra solução constituída por uma única camada. Com o objetivo de facilitar a leitura dos resultados alcançados, esta secção está dividida em duas partes, a subsecção 6.3.1 e a 6.3.2, que correspondem à análise dos resultados do *dataset* Geolife e do *dataset* SenseMyCity, respetivamente.

6.3.1 *Dataset* Geolife

6.3.1.1 Solução com duas camadas

Para a solução composta por duas camadas, é necessário fazer análise a cada camada quanto à escolha do melhor classificador a usar em cada uma. A tabela 6.6 mostra os resultados para o *dataset* Geolife.

Camada	SVM	IBK	Bayes	NN	DT	RT	RF
1º	78.62%	75.89%	73.45%	79.37%	80.28%	77.26%	**82.51%**
2º	56.11%	52.37%	41.38%	59.33%	58.50%	55.51%	**65.42%**

Tabela 6.6: Resultados da primeira e segunda camadas (considerando que a primeira camada é perfeita) para o *dataset* Geolife.

Com base na análise dos resultados para a primeira camada, podemos concluir que o resultado com melhor exatidão refere-se ao RF, com 82.51%. Quanto à segunda camada, o resultado com

melhor exatidão refere-se também ao RF, com 65.42%. Uma análise mais detalhada dos melhores resultados pode ser visualizada nas tabelas 6.7, 6.8 e 6.9.

Condição / Teste	Andar	Motorizado	Precisão
Andar	**1557**	1034	60.1%
Motorizado	177	**4154**	**95.9%**
Sensibilidade	**89.8%**	80.1%	82.5%

Tabela 6.7: Matriz de confusão do classificador com melhor resultado para a camada 1 do *dataset* Geolife.

Como se pode observar na tabela 6.7, a classe com melhor precisão é Motorizado, com 95.9%. No entanto, é esta classe a que apresenta uma pior sensibilidade com 19.9% de instâncias mal classificadas. A classe com melhor sensibilidade é a Andar, com 89.8%. No entanto, é a classe Andar que apresenta a pior precisão, com 39.9% de instâncias que são classificadas como Andar mas pertencentes à classe Motorizado. Os maus resultados devem-se ao desequilíbrio existente entre a quantidade de instâncias da classe Motorizado e a quantidade de instâncias da classe Andar; na classe Motorizada a quantidade é maior que na classe Andar. Este desequilíbrio podia ser corrigido removendo instâncias da classe Motorizado, ou adicionando instâncias artificiais na classe Andar, ou ainda dar maior peso aos erros durante a classificação da classe Andar. A falta de tempo foi o motivo pelo qual não se aplicou uma correção para o desequilíbrio presente. No entanto, este desequilíbrio só se aplica para o conjunto de dados de treino, pelo facto de que no conjunto total de instâncias existem três classes motorizadas, nomeadamente, Carro, Táxi e Autocarro. Como os *datasets* são constituídos por um número de instâncias equilibrado para cada umas das classes (Andar, Carro, Táxi e Autocarro), espera-se que o número de instâncias da classe Motorizado seja triplo do número de instâncias da classe Andar. Também podemos observar que as instâncias bem classificadas como Andar aproximam-se do número de instâncias bem classificado para a classe Motorizado. Isto deve-se maioritariamente à existência de características semelhantes entre a classe Andar e Motorizado, quando a classe Motorizado apresenta velocidades equivalentes à classe Andar.

Condição / Teste	Carro	Táxi	Autocarro	Precisão
Carro	**1181**	320	226	**68.4%**
Táxi	223	**949**	254	66.5%
Autocarro	358	413	**1264**	62.1%
Sensibilidade	67.0%	56.4%	**72.5%**	65.4%

Tabela 6.8: Matriz de confusão do classificador com melhor exatidão para a camada 2 do *dataset* Geolife, assumindo que a camada 1 é perfeita.

Como pode ser observado na tabela 6.8, a classe com maior precisão é a classe Carro, com uma precisão de 68.40%. A classe com maior sensibilidade é a classe Autocarro, com 72.5%, no entanto, é a classe com a pior precisão, apresentando 37.9% de instâncias classificadas como Autocarro mas pertencentes a outras classes, distribuindo-se com 17.6% pertencendo à classe Carro e 20.3% pertencendo à classe Táxi. A classe com pior sensibilidade é a classe Táxi com 43.6% de instâncias mal classificadas. Estes valores razoavelmente elevados devem-se à dificuldade existente em diferenciar os vários modos de locomoção motorizados, nomeadamente carro, táxi e autocarro. No entanto, para cada classe, o número total de instâncias bem classificadas é sempre superior à soma das classes mal classificadas. Uma análise mais profunda indica-nos que ambas as classe Carro e Táxi fazem uma confusão semelhante com a classe Autocarro. Mas, a classe que confunde mais a classe Autocarro é Táxi, e o mesmo se verifica quando a classe Autocarro confunde a classe Táxi, sendo esta classe a que gera mais confusão nas classificações para Autocarro. Isto é justificável pelo facto de ambos estes modos de locomoção circularem em ambientes urbanos na maior parte do tempo e estarem sujeitos a um número de paragens muitas vezes semelhantes entre eles. Quanto à classe Carro, a classe que mais se confunde é a classe Autocarro, com 20.3% de instâncias mal classificadas pertencentes à classe Carro.

Condição / Teste	Andar	Carro	Táxi	Autocarro	Precisão
Andar	**1557**	228	261	545	60.09%
Carro	43	**1093**	310	195	66.61%
Táxi	34	206	**865**	181	**67.26%**
Autocarro	100	235	246	**823**	58.62%
Sensibilidade	**89.79%**	62.03%	51.43%	47.19%	62.67%

Tabela 6.9: Matriz de confusão para o conjunto da camada 1 com a camada 2 do *dataset* Geolife, considerando os erros da camada 1.

Analisando a tabela 6.9, a classe com maior precisão é Táxi, com 67.26%, e a classe com maior sensibilidade é Andar, com 89.79%. A classe com pior precisão e sensibilidade é Autocarro com 41.38% e 48.57%, respetivamente. Isto deve-se ao facto de as características que definem o modo de locomoção Autocarro serem muito semelhantes às da classe Andar, tendo esta classificado 545 instâncias pertencentes à classe Autocarro. Também para a classe Andar, a classe que apresenta um maior número de instâncias mal classificadas pela classe Andar é precisamente Autocarro. Desta forma, podemos concluir que a classe Andar apresenta dificuldades em conseguir diferenciar-se da classe Autocarro, pelo facto de as classe Andar e Autocarro serem semelhantes, já em que ambas param muitas vezes, e normalmente a classe Autocarro anda devagar, com velocidades semelhantes à classe Andar num ambiente urbano. Por outro lado, a classe Andar foi a que apresentou o menor número de instâncias mal classificadas pela classe Autocarro, com 100 instâncias classificadas como Autocarro pertencentes a Andar. Para as classes motorizadas, a classe com maior precisão é Táxi, com 67.26%, e a classe com melhor sensibilidade é Carro, com

62.03%. O pior resultado, tanto para precisão como para sensibilidade, continua a ser o mesmo, o da classe Autocarro, para o conjunto de classes motorizadas. Como já tinha sido observado, existe uma grande dificuldade em diferenciar as várias classes motorizadas, e tendo em conta as semelhanças entre a classe Andar e Autocarro, é lógico que o que apresenta piores resultados é a classe Autocarro.

Pode observar-se que existe uma diminuição de exatidão em comparação com a camada 1, devido à semelhança presente entre as três classes motorizadas, nomeadamente Carro, Táxi e Autocarro. Ao analisar o desempenho da camada 2, considerando que esta é perfeita e considerando a propagação de erros da camada 1, pode concluir-se que não existe uma grande diferença a nível de exatidão, existindo um decréscimo de aproximadamente 3%. Os resultados mostram que mesmo com propagação dos erros gerados pela classificação da camada 1, a camada 2 consegue manter um desempenho semelhante ao atingido na camada 2 com propagação de erros.

6.3.1.2 Solução com uma camada

A análise de uma solução só com uma camada fica mais simplificada pelo facto de classificar de uma só vez todas as classes existentes. Estes resultados estão indicados na tabela 6.10.

SVM	IBK	Bayes	NN	DT	RT	RF
55.16%	48.64%	46.63%	55.63%	55.03%	50.48%	**63.78%**

Tabela 6.10: Resultados para seleção do melhor classificador da solução com uma única camada para o *dataset* Geolife.

Analisando a tabela 6.10, podemos concluir que o melhor resultado para o *dataset* Geolife é o classificador RF, com exatidão de 63.78%. A tabela 6.11 apresenta a respetiva matriz de confusão para o classificador RF, o que permite analisar mais detalhadamente os melhores resultados para este *dataset*.

Condição / Teste	Andar	Carro	Táxi	Autocarro	Precisão
Andar	**1470**	155	215	467	63.7%
Carro	62	**1133**	298	197	**67.0%**
Táxi	51	224	**923**	191	66.5%
Autocarro	151	250	246	**889**	57.9%
Sensibilidade	**84.8%**	63.4%	54.9%	51.0%	63.78%

Tabela 6.11: Matriz de confusão do melhor resultado para a solução de uma única camada do *dataset* Geolife.

Observando a tabela 6.11, conclui-se que a classe com melhor precisão é Carro com 67.0%, e a que apresenta pior precisão é a classe Autocarro, com 42.1% de instâncias classificadas como Autocarro pertencentes a outras classes. A classe que apresenta melhor sensibilidade é a classe

Andar, com 84.8%, e a classe com pior sensibilidade é Autocarro com 49.0% de instâncias mal classificadas. Tanto para precisão como para sensibilidade, a classe Autocarro é o pior resultado, devido à dificuldade existente e às semelhanças entre outras classes, como já tinha sido observado anteriormente. De entre as três classes motorizadas, a que apresenta uma melhor sensibilidade é Carro, com 63.4%, é também a que tem a melhor precisão com 47.0%. Os piores resultados já analisados aplicam-se aos piores resultados para o grupo das classes motorizadas. Comparando os resultados da classe Andar com as classes motorizadas, é de notar que a primeira camada consegue separar classes motorizadas da classe Andar com uma maior sensibilidade, como já tinha sido observado, daí a classe Andar apresentar a maior sensibilidade do que as demais classes. A ordem de diferença da sensibilidade da classe Andar em relação às classes motorizadas situa-se aproximadamente nos 25%. O resultado mais baixo de sensibilidade para motorizados deve-se ao facto de as características que os diferenciam serem muito semelhantes entre as classes motorizadas. No entanto, a precisão está equilibrada para todas as classes, existindo uma percentagem de instâncias mal classificadas semelhantes entre a classe Andar e as classes motorizadas.

6.3.1.3 Escolha da solução

Pela comparação entre as tabelas 6.9 e 6.11 e através das parametrizações aplicadas, concluiu-se que a melhor estratégia a usar para o *dataset* do Geolife é a utilização de uma única camada. A exatidão final demonstrada na figura 6.9 tem o valor de 63.78%. Para a estratégia com duas camadas é notável a sensibilidade existente na classe Andar em comparação com a sensibilidade da mesma classe na tabela 6.11, sendo esta menor. Isto era esperado, pois, para a estratégia com duas camadas, a primeira é dedicada a separar a classe Andar do conjunto de classes motorizada. Desta forma podemos obter um maior conjunto de instâncias classificadas para a classe Andar se pretendermos usar a estratégia das duas camadas em vez da estratégia com uma única camada. Como o que queremos é a melhor combinação de inferência entre os modos selecionados para classificação, não nos podemos basear nos valores de uma classe. Em comparação com os valores de sensibilidade das classes motorizadas entre as duas tabelas, podemos observar que as sensibilidades da estratégia com uma única camada são ligeiramente superiores às sensibilidades apresentadas para a estratégia com duas camadas. Quanto a precisão, é de notar que os papéis invertem-se. Para a classe Andar obtemos valores superiores de precisão para a estratégia com uma única camada e para as classes motorizadas obtemos valores de precisão ligeiramente superiores para a estratégia com duas camadas. Esta inversão de valores permite que ambos os valores de exatidão se aproximem, podendo mais facilmente chegar-se a um veredicto final. Tendo em conta os resultados apresentados, podem observar-se agora os valores de exatidão para cada uma das tabelas. Para a estratégica com uma única camada obtivemos uma exatidão de 63.78%, enquanto que para a solução com duas camadas obtivemos 62.67%. Desta forma, podemos concluir não ser vantajoso a utilização da estratégia com duas camadas, pelo facto de ter uma estrutura mais complexa do que somente ser composta por uma única camada e de não permitir obter níveis mais altos de exatidão.

6.3.2 *Dataset* SenseMyCity

6.3.2.1 Solução com duas camadas

Para a solução composta por duas camadas, é necessário fazer a análise a cada camada quanto à escolha do melhor classificador a usar em cada uma. A tabela 6.12 mostra os resultados para o *dataset* SenseMyCity.

Camada	SVM	IBK	Bayes	NN	DT	RT	RF
1º	85.87%	86.38%	81.64%	**87.90%**	84.26%	85.36%	86.46%
2º	76.64%	69.20%	74.63%	78.37%	**79.07%**	71.63%	77.51%

Tabela 6.12: Resultados da primeira e segunda camadas, considerando que a primeira camada é perfeita, para o *dataset* SenseMyCity.

A partir da tabela 6.12 podemos observar os vários resultados obtidos tanto para a primeira como para a segunda camada, considerando que a primeira camada é perfeita, entre os diversos classificadores selecionados para o SenseMyCity. Para a primeira camada, podemos concluir que o classificador com os melhores resultados é o NN, com uma exatidão de 87.90%. Para a segunda camada, podemos concluir que o classificador com melhores resultados é o DT, com uma exatidão de 79.07%. Desta forma, a solução com duas camadas será composta pelo NN na primeira camada e pelo DT na segunda camada. Os melhores resultados estão referidos nas tabelas 6.13 e 6.14 para a primeira e segunda camada, respetivamente. É de notar que uma das classes a ser avaliada para o dataset do SenseMyCity é Estacionário, pelo facto de esta ser necessária para as melhorias com HMM, como é analisado na secção 6.4. Também é de salientar que a classe Táxi do grupo das classes motorizadas não existe para o *dataset* do SenseMCity por não existirem dados catalogados pelo SenseMyCity com a classe Táxi.

Condição / Teste	Estacionário	Andar	Motorizado	Precisão
Estacionário	440	3	3	98.7%
Andar	0	**111**	87	56.1%
Motorizado	0	50	**488**	**90.7%**
Sensibilidade	**100.0%**	67.7%	84.4%	87.9%

Tabela 6.13: Matriz de confusão do melhor resultado obtido para a primeira camada do *dataset* SenseMyCity.

Através dos resultados presentes na tabela 6.13, e excluindo a classe Estacionário desta análise por apresentar valor muito próximos de 100.0%, podemos concluir que a classe com melhor precisão e melhor sensibilidade é a classe Motorizado, com valores de 90.7% e 84.4%, respetivamente. Desta forma, a classe com pior precisão e pior sensibilidade é Andar, com 43.9% de instâncias mal classificadas e 32.3% de instâncias classificadas como Estacionário ou Motorizado

mas que pertencem à classe Andar. Isto pode ser devido a um desequilíbrio presente na classe de teste, existindo um menor número de instâncias da classe Andar em comparação ao maior número de instâncias da classe Motorizado. Como já observado para o *dataset* do Geolife no subcapítulo 6.3.1, a falta de tempo é uma dos principais motivos da ausência de uma correção que contornasse este desequilíbrio.

Teste \ Condição	Carro	Autocarro	Precisão
Carro	**133**	34	**79.64%**
Autocarro	87	**324**	78.83%
Sensibilidade	60.45%	**90.50%**	79.07%

Tabela 6.14: Matriz de confusão do melhor resultado para a camada 2 do SenseMyCity considerando que a camada 1 é perfeita.

Analisando os resultados da tabela 6.14, a classe com melhor sensibilidade é o Autocarro, com 90.50%, e a classe que apresenta uma melhor precisão é a classe Carro, com 79.64%. Desta forma, a classe com pior precisão é Autocarro, com 21.17% de instâncias mal classificadas. Isto maioritariamente deve-se ao facto de em ambas as classes motorizadas existirem diversas características semelhantes que dificultam a classificação de cada modo de locomoção. Como pode ser observado, as precisões são muito semelhantes, diferenciando-se por uma percentagem aproximadamente inferior a 1%. A classe com pior sensibilidade é Carro com 39.55%. Isto deve-se basicamente à grande quantidade de características semelhantes entre as duas classes motorizadas. É frequente as velocidades destes dois modos de locomoção se assemelharem quando ambos estão a ser produzidos em ambientes urbanos, como em grandes cidades. Também é de notar que existe um ligeiro desequilíbrio entre estas classes, sendo a classe Autocarro a que contém mais instâncias em relação ao número de instâncias da classe Carro. Desta forma são justificáveis os resultados apresentados na tabela 6.14.

Teste \ Condição	Estacionário	Andar	Carro	Autocarro	Precisão
Estacionário	**440**	3	0	3	98.65%
Andar	0	**111**	22	65	56.06%
Carro	0	1	**133**	34	**79.17%**
Autocarro	0	49	65	**256**	69.19%
Sensibilidade	100.00%	67.68%	60.45%	**71.51%**	79.53%

Tabela 6.15: Matriz de confusão para o conjunto da camada 1 com a camada 2 do *dataset* SenseMyCity, considerando os erros da camada 1.

Através dos resultados presentes na tabela 6.15 e excluindo a classe Estacionário desta análise

por causa dos seus elevados valores de precisão e exatidão, podemos concluir que a classe com melhor precisão é Carro com 79.17% e a classe com melhor sensibilidade é Autocarro. A classe com pior precisão é Andar, com 43.94% de instâncias classificadas como Andar pertencentes a outras classes. Isto deve-se a um ligeiro desequilíbrio existente entre a classe Andar e as classes motorizadas relativamente ao número de instâncias que constituem cada uma destas classes. Como pode ser observado, esta precisão, em comparação com as classes motorizadas, apresenta uma diferença de aproximadamente 15%. Desta forma, conclui-se que a camada 1 não consegue diferenciar com exatidão a classe Andar das classes motorizadas. No entanto, a segunda camada apresenta precisão superior para ambas as classes motorizadas, nomeadamente Carro e Autocarro. A classe com pior sensibilidade é Carro, com 39.55% de instâncias mal classificadas. Devido às várias características semelhantes entre a classe Autocarro, é de esperar que a classe Autocarro classifique uma quantidade de instâncias pertencente à classe Carro. De entre todas as classes que classificam instâncias pertencentes a outras classes, a classe Autocarro é a que classifica maior número de instâncias que são pertencentes à classe Autocarro, exatamente 65 instâncias são classificadas como Autocarro pertencentes à classe Carro. Considerando só as classes motorizadas, podemos observar que as de melhor precisão e melhor sensibilidade continuam a ser as mesmas já anteriormente referidas. A pior sensibilidade também se mantém. Só a precisão é exceção; agora atribuímos à classe Autocarro a pior precisão obtida entre as classes motorizadas. Isto deve-se ao facto de existir uma grande dificuldade em distinguir a classe Autocarro da classe Andar, pois as velocidades ou até o comportamento dentro de áreas urbanas, como cidades, se assemelham bastante entre diversos modos de locomoção motorizados.

Analisando a primeira camada com a segunda, podemos concluir que existe um decréscimo da precisão obtida, devido à dificuldade apresentada em diferenciar as classes motorizadas. Comparando as tabelas 6.14 e 6.15, podemos concluir que não existe grande diferença de exatidão, tendo em conta propagação de erros e não tendo em conta a propagação de erros. O decréscimo compreendido entre a exatidão destas tabelas é aproximadamente de 1%. Note-se que a tabela 6.15 apresenta uma exatidão a um nível global, considerando a propagação de erros, e a tabela 6.14 mostra a exatidão da camada 2, tendo em conta que a primeira camada é perfeita. Tendo isto, podemos concluir que o desempenho da camada 2 com propagação de erros consegue manter uma exatidão aceitável em comparação à exatidão alcançada com 79.07% na tabela 6.14.

6.3.2.2 Solução com uma camada

A análise de uma solução só com uma camada fica mais simplificada pelo facto de classificar de uma vez só todas as classes existentes. Estes resultados estão visíveis na tabela 6.16.

SVM	IBK	Bayes	NN	DT	RT	RF
78.93%	75.97%	74.03%	77.83%	79.10%	75.55%	**80.46%**

Tabela 6.16: Resultados para seleção do melhor classificador da solução com uma única camada para o *dataset* do SenseMyCity.

Através do estudo da tabela 6.16, podemos concluir que o classificador com melhor resultado é o RF, com 80.46%. Também podemos observar que a variação entre os diversos classificadores não é grande, quando o pior resultado tem o valor de 74.03%. De forma a se obter uma análise mais profunda do melhor resultado, a tabela 6.17 apresenta a matriz de confusão do melhor resultado obtido usando o classificador RF.

Condição / Teste	Estacionário	Andar	Carro	Autocarro	Precisão
Estacionário	**440**	1	0	0	**99.8%**
Andar	0	**112**	20	49	61.9%
Carro	0	6	**163**	73	67.4%
Autocarro	0	45	37	**236**	74.2%
Sensibilidade	**100.0%**	68.3%	74.1%	65.9%	80.46%

Tabela 6.17: Matriz de confusão do melhor resultado para a solução de uma única camada do *dataset* SenseMyCity.

Analisando os resultados da tabela 6.17, verifica-se que a classe com maior precisão é a classe Autocarro com 74.2% e a classe com maior sensibilidade é a classe Carro com 74.1%. A classe com pior precisão é Andar com 38.1% de instâncias mal classificadas. Isto deve-se a um ligeiro desequilíbrio entre o número de instâncias da classe Andar e o número de instâncias das classes motorizadas. Também deve-se ao facto de existirem em algumas características muito semelhantes entre estas classes, tais como, alguns comportamentos presentes em ambiente urbanos, onde a velocidade muitas vezes é muito baixa e a quantidade de paragens converge para um número muito semelhante entre todos os tipos de modos de locomoção analisados nestas experiências. A classe com pior sensibilidade é Autocarro com 34.1% de instâncias classificadas como Autocarro, mas que pertence a outras classes, nomeadamente Andar e Carro. Isto deve-se, como já tinha sido observado, às diversas características que os diferenciam serem, por vezes, muito semelhantes, dependendo do comportamento ou do ambiente onde se encontram. Como, por exemplo, Autocarro e Carro podem se confundir em ambientes urbanos, com trânsito, onde a velocidade é condicionada por limites e por sinalização vertical ou luminosa, produzindo padrões de locomoção semelhantes entre todas as classes motorizadas com pelo menos quatro rodas, visto não termos evidências de que o mesmo se pode afirmar com classes motorizadas de duas rodas. Analisando as classes motorizadas, ambas apresentam resultados capazes de justificar alguma dificuldade em se diferenciar, mas de uma forma generalizada ambas são distinguíveis ao ponto do resultado gerado ser considerado válido. Note-se que este *dataset* não conta com um número elevado de instâncias como o *dataset* do Geolife tem.

6.3.2.3 Seleção da melhor solução

Pelos resultados presentes nas tabelas 6.15 e 6.17, conclui-se que a estratégia com o melhor resultado é a que usa só uma única camada, com uma exatidão de 80.46%. Comparando cada uma das classes, podemos concluir que a estratégia usando duas camadas não é mais vantajosa nestas condições, em comparação com a estratégia com uma camada. Tanto em precisão como em sensibilidade, a estratégia com duas camadas apresenta piores resultados, em comparação com a precisão e sensibilidade da estratégia com uma única camada, tendo esta apresentado em precisão 68.3% em comparação com 67.88% e uma sensibilidade de 61.9% em comparação com 56.06%, respetivamente. Quanto às classes motorizadas para a estratégia com duas camadas, podemos observar que ambas as classes motorizadas têm um desempenho semelhante, sendo a classe Carro a que obtém uma melhor precisão e a classe Autocarro uma melhor sensibilidade. Para a estratégia com uma única camada, verifica-se que ambas as classes também apresentam um desempenho semelhante, sendo a classe com melhor precisão Autocarro e a classe com melhor sensibilidade Carro. No entanto, quem obtém melhor sensibilidade entre as duas estratégias em média é a estratégia com uma única camada, com um valor aproximado de 70%, em comparação com um valor aproximado de 66% para a estratégia com duas camadas. Em relação à sensibilidade entre as classes motorizadas, podemos concluir que a melhor estratégia é usando duas camadas, com uma média de sensibilidade de 74% em comparação com 71%, obtidos usando a estratégia com uma única camada. Analisando estes resultados, podemos concluir que tanto para Andar como para as classes motorizadas, a estratégia com uma única camada apresenta um melhor desempenho do que a estratégia com duas camadas. Desta forma, não se mostra vantajoso usar a estratégia com duas camadas para a maior parte das situações existentes, usando estas parametrizações e *datasets*. Também a estratégia com uma única camada apresenta um nível de complexidade mais baixo quando comparada com uma estratégia com duas camadas, sendo claramente vantajoso usar a estratégia com uma única camada.

6.3.3 Comparação entre *datasets*

Comparando em detalhe o *dataset* do Geolife com o SenseMyCity, e tendo em conta as melhores estratégias concluídas nos subcapítulos 6.3.1 e 6.3.2, podemos observar que no *dataset* do Geolife existe uma maior dificuldade em distinguir as classes motorizadas do que a classe Andar. As sensibilidades para as classes motorizadas aproximam-se umas das outras, com sensibilidades muito próximas de 50%, induzindo em dificuldade distinguir estes classes. No caso de SenseMyCity, esta confusão não é tão visível como no *dataset* anterior. Isto também deve-se ao facto de não existirem instâncias da classe Táxi contendo características muito semelhantes às outras classes motorizadas. No entanto, é de notar que, tanto para o *dataset* Geolife como para o *dataset* SenseMyCity, pode ser observado que a classe Autocarro tem certa dificuldade em se distinguir da classe Andar. Mesmo sendo Andar a classe que se distingue mais no *dataset* do Geolife, é a classe que apresenta mais classes classificadas como Autocarro pertencentes a Andar. O mesmo pode ser observado para o *dataset* do SenseMyCity. As classes que mais se distinguem são as classes Carro

e Andar, para o *dataset* do SenseMyCity e o *dataset* do Geolife, respetivamente. Esta diferença entre os *datasets* é causada pela ausência da classe Táxi no *dataset* do SenseMyCity, não existindo uma classe de classificação difícil comparável à classe Carro. Caso contrário, era muito provável que a classe que mais se distinguisse para ambos os *datasets* fosse a classe Andar.

A nível de estratégias, podemos concluir que é mais vantajoso usar a classificação com uma única camada do que duas camadas, para ambos os *datasets*. Esta conclusão foi alcançada tendo em conta os resultados obtidos a partir da análise de cada *dataset*. Esta tendência deve-se ao facto de ser difícil distinguir as classes motorizadas e também à existência de um ligeiro desequilíbrio no número de instâncias usadas nas avaliações.

6.3.4 Comparação com a literatura

Com a finalidade de comparar os melhores resultados obtidos com ambos os *datasets* com a literatura, a tabela 6.18 apresenta uma comparação dos resultados presentes na literatura com os resultados obtidos. É de notar que todos os artigos presentes nesta tabela só utilizam dados provenientes de recetores GPS, não usando qualquer outro tipo de sensores.

Posição	Ano	Autor	*Dataset*	Validação	Classes usadas	Exatidão
1º	2012	Adel Bolbol [BCTH12]	*dataset* próprio	única divisão não explícita	carro, andar, autocarro, bicicleta, metro e comboio	88%
2º	2013	Lei Zhang [ZQY13]	Geolife Completo	única divisão não explícita	carro, andar, autocarro e bicicleta	83.71%
3º	**2014**	**Vitor Figueira**	**SenseMyCity**	**divisão em 50%**	**carro, andar, autocarro e táxi**	**80.46%**
4º	2010	Yu Zheng [ZCL+10]	Geolife Completo	única divisão não explícita	carro, andar, autocarro e bicicleta	75.6%
5º	**2014**	**Vitor Figueira**	**Geolife**	**divisão em 50%**	**carro, andar, autocarro e táxi**	**63.78%**

Tabela 6.18: Comparações entre os melhores resultados obtidos e os resultados existentes na literatura, por ordem decrescente de exatidão adquirida.

Analisando a tabela 6.18, podemos enquadrar as soluções desenvolvidas na literatura. Neste top 5, o *dataset* SenseMyCity posiciona-se na terceira posição dos melhores e o *dataset* Geolife situa-se na última posição. Esta diferença deve-se à existência de erros no *dataset* Geolife na catalogação dos modos de locomoção para um indeterminado conjunto de coordenadas GPS. Um exemplo notável destes erros de catalogação foi a localização de mais de um conjunto de pontos, catalogados com o modo de locomoção Andar, e estes, ao serem analisados em mais detalhe num mapa global, encontram-se em pelo oceano. Como é óbvio de observar, este modo de locomoção

não é possível, em condições normais, de ser produzido no meio do oceano sem qualquer tipo de suporte especial que permita produzir tal modo de locomoção. Um outro motivo pelos melhores resultados alcançados em [ZCL+10, ZQY13], deve-se à estratégia usada, que se baseia na segmentação das sequências por outras mais pequenas, conseguindo fragmentar uma sequência com vários modos de locomoção. Também um outro motivo para estes resultados, é a utilização de uma divisão diferente da aplicada nesta dissertação. Segundo [ZCL+10], usam uma divisão de 70% para treino e 30% para teste, ao contrário do que foi usado nesta dissertação, 50%. O melhor classificado deve-se ao facto de usar um *dataset* próprio, não se baseando no *dataset* do Geolife, não sendo comparável com estes.

6.4 Experiência 4: Modulação de dependência temporal entre instâncias com HMM

De forma a melhorar os resultados já obtidos, a aplicação de melhorias pode trazer vantagens ao nível da exatidão alcançada ao longo do tempo. Usando o HMM [RJ86] na sua forma mais simplificada, um algoritmo de séries temporais, podemos classificar uma série de resultados de modo a inferir qual o próximo estado mais provável. Aplicando esta melhoria a um conjunto de séries já constituídas no *dataset* SenseMyCity, a tabela 6.19 apresenta os resultados alcançados em comparação com os já obtidos.

Solução sugerida	Sem HMM	Com HMM
Solução com várias camadas	**79.07%**	77.74%
Solução com uma camada	**80.46%**	79.10%

Tabela 6.19: Comparações entre os melhores resultados obtidos e os resultados obtidos da modulação de dependência temporal entre instâncias com HMM.

Através dos resultados visíveis na tabela 6.19, podemos concluir que as soluções que não usam esta melhoria de HMM têm os melhores resultados. Não existe melhoria visível, devido aos resultados com HMM serem ligeiramente inferiores aos obtidos sem HMM. Logo, não se torna vantajoso a utilização deste HMM como melhoria para as soluções sugeridas. Isto deve-se principalmente ao facto de ter sido usada uma implementação básica do algoritmo HMM, por falta de tempo para analisar uma melhoria robusta a aplicar neste problema classificativo. Também só foram usadas as classificações propriamente ditas para proceder com o treino deste HMM, o que diminuí consideravelmente o conhecimento de predição do HMM aplicado.

6.5 Resumo

Neste capítulo capítulo concluímos e analisamos diversos resultados ao nível de precisão, sensibilidade e exatidão. Para seleção da janela, houve intervalos distintos para os diferentes *datasets* usados, chegando à conclusão que a melhor janela para o *dataset* do Geolife é a dos 30 pontos e

a melhor para o *dataset* do SenseMyCity é a janela dos 9 pontos. Quanto à seleção de caraterísticas, concluímos, para ambos os *datasets*, que não houve vantagem em usar tal filtragem, dado que o uso de todas as características no processo de inferência determinou resultados elevados. Quando às estratégias utilizadas para inferir modos de locomoção, obtivemos diversos resultados para ambos os *datasets*, usando um conjunto de classificadores já previamente selecionados tanto para a abordagem de uma única camada para a abordagem de duas camadas. A abordagem que apresentou melhores resultados foi a de uma camada, tanto para o *dataset* do Geolife como para SenseMyCity, usando esta para comparar os resultados obtidos com a literatura existente. Quanto à modulação de dependência temporal entre instâncias com o HMM sugerido, tivemos resultados muito semelhantes que não justificaram o uso de tal melhoria com este HMM, tendo os resultados com melhorias aplicadas ligeiramente mais baixos.

Capítulo 7

Conclusões e trabalho futuro

Este capítulo concluí o tema desta dissertação; está dividido em duas secções: a 7.1, na qual se apresenta o resumo e análise dos objetivos alcançados, e a 7.2 onde se faz um reflexão sobre os aspetos a melhorar futuramente.

7.1 Satisfação dos objetivos

A inferência de modos de locomoção mostrou-se uma tarefa complexa, sendo constituída por várias fases complicadas para atingir valores próximos dos ótimos. Para a construção de uma solução capaz de inferir modos de locomoção foram necessárias várias fases de análise e reflexão, sendo diversas vezes penoso conseguir os resultados pretendidos.

A solução obtida evidenciou resultados diferentes entre os *datasets* do Geolife e *dataset* do SenseMyCity, tendo o SenseMyCity alcançado resultados mais elevados, ultrapassando os alcançados na literatura [ZCL+10], mas em comparação com os resultados determinados pelo *dataset* Geolife não conseguimos essa proeza, pelo facto de existirem catalogações que não correspondiam com a trajetória geográfica adequada, pela inexistência de um equilíbrio ótimo entre cada uma das classes usadas no processo de classificação em algumas das fases experimentais, e também pelo facto de usarmos a classe táxi em vez de bicicleta, como usado na literatura [ZCL+10], que pode ser mais fácil de diferenciar do que distinguir táxi das restantes classes usadas, especialmente as motorizadas.

Através das experiências realizadas e da análise dos resultados obtidos, conseguiu-se otimizar parâmetros de todas as fases constituintes da solução, permitindo assim construir a solução final na plataforma Android, como um serviço. Este serviço por sua vez consegue recolher dados e inferir modos de locomoção em tempo real, desde que o sensor GPS interno do dispositivo esteja a fornecer coordenadas *raw* GPS ao serviço. Também alguma otimização é feita a nível de processamento, de forma a não influenciar na interação do utilizador com outras aplicações no mesmo dispositivo. No entanto, não foi possível integrar esta implementação na aplicação

SenseMyCity devido ao SenseMyCity estar em processo de reformulação para integração de pulgins. É importante notar que o tempo de experimentação foi extenso, de forma a garantir um nível de confiança aceitável em todas as experiências efetuadas, bem como a sua validade para o contexto desta dissertação.

Era sugerida a análise de um conjunto finito de classificadores, com o objetivo de encontrar o que evidenciasse melhor um desempenho superior, em ambos os *datasets*. Esta análise foi efetuada a vários níveis, tanto a nível de seleção da janela temporal, como na seleção do melhor classificador para inferência de modos de locomoção. Foi proposto, nesta dissertação, um conjunto de características relacionadas com aceleração, desvio padrão amostral das velocidades e acelerações, e diferença de direção; através dos resultados da seleção de características concluiu-se que estas características apresentam um incremento positivo na classificação, demonstrando-se de relevo para o contexto da inferência de modos de locomoção.

A nível de estratégias, propusemos uma nova abordagem, constituída por duas camadas; a primeira dedicada a classificar duas classes, nomeadamente andar e classes motorizadas, e a segunda classe dedicada a classificar entre os diversos modos de locomoção motorizados. O objetivo desta abordagem era fragmentar a classificação, de forma a se poder separar os modos de locomoção motorizados dos não motorizados. Os resultados obtidos mostraram que a estratégia constituída por duas camadas apresenta valores relativamente inferiores quando comparada com a estratégia com uma única camada. Desta forma, concluiu-se que a estratégia proposta não se mostrou vantajosa ao ponto de ser usada como alternativa à estratégia com uma única camada.

Após implementação na plataforma Android, pudemos restringir o número de dados *input* recebidos pelo sensor GPS, permitindo uma utilização reduzida de memória, quando simultaneamente conseguimos inferir modos de locomoção em tempo real. Esta solução mostra como é possível obter conhecimento sobre modos de locomoção em tempo real, na maior parte dos dispositivos móveis, sem que o utilizador tenha de interagir com a aplicação ou esta classificação interfira com a ação do utilizador sobre o seu dispositivo móvel.

7.2 Trabalho futuro

Após realização de todas as experiências e implementada a solução final, podemos concluir que existem aspetos a aperfeiçoar num trabalho futuro. Na estratégia de duas camadas, podemos observar certo desequilibro entre as classes, o que pode ter incorrido em alguns erros. Um trabalho futuro seria garantir o equilíbrio entre todas as classes de forma a aumentar a confiança do *dataset* utilizado para a classificação.

A nível de características, é sempre útil a criação de novas características, que possam de certa forma distinguir melhor cada um dos modos de locomoção, permitindo construir modelos de inferência mais robustos.

Também existe um grande problema nos *datasets*, na catalogação de um determinado conjunto de pontos, no que diz respeito à sua exatidão. Conforme se refere nesta dissertação, o *dataset* do Geolife apresenta catalogações que não são corretas, tanto a nível geográfico como a nível de

características implícitas. Tal facto induz em características incorretas e valores inesperados para uma determinada catalogação. Assim, um trabalho futuro será assegurar a catalogação adequada dos *datasets* a utilizar, de modo a minimizar a quantidade de dados incorretos.

No que diz respeito à melhoria com dependência temporal entre instâncias, concluímos que não achamos vantagem em aplica-la no contexto da solução final. Isto basicamente deveu-se à básica implementação usada para prever os próximos modos de locomoção, visto na literatura existente, ser apresentada uma melhoria ao usar este tipo de classificadores. Um trabalho futuro seria aperfeiçoar esta implementação, de forma a que fosse robusta o suficiente para conseguir melhorar a solução final aqui presente.

Outro trabalho futuro para a solução final, mas de carácter opcional, será integrar a implementação na aplicação SenseMyCity, como um pulg-in. Isto é atingível desde que exista uma API capaz de suportar cada um destes pulg-ins como um serviço em Android com uma determinada designação reconhecível pela aplicação SenseMyCity.

Referências

[AKA91] David W Aha, Dennis Kibler e Marc K Albert. Instance-Based Learning Algorithms. *Mach. Learn.*, 6(1):37–66, 1991. URL: http://dx.doi.org/10.1023/A:1022689900470, doi:10.1023/A:1022689900470.

[AM06] Ian Anderson e Henk Muller. Practical Activity Recognition using GSM Data. Technical Report CSTR-06-016, Department of Computer Science, University of Bristol, 2006. URL: http://www.cs.bris.ac.uk/Publications/Papers/2000563.pdf.

[BCTH12] A Bolbol, T Cheng, I Tsapakis e J Haworth. Inferring hybrid transportation modes from sparse GPS data using a moving window SVM classification. *Computers, Environment and Urban Systems*, 36(6):526–537, 2012. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-84869864790&partnerID=40&md5=28e97f2a51b1724ba0f2c20ef476426e.

[BGL12] J B Bancroft, D Garrett e G Lachapelle. Activity and environment classification using foot mounted navigation sensors. In *2012 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2012 - Conference Proceedings*, 2012. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-84874237297&partnerID=40&md5=f50c226b7b89750fad180e9473bfb5f2.

[BI04] L Bao e S S Intille. Activity recognition from user-annotated acceleration data, 2004. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-35048842427&partnerID=40&md5=6c844ec02021e454647efeadc51f0e5d.

[BLvO13] F Biljecki, H Ledoux e P van Oosterom. Transportation mode-based segmentation and classification of movement trajectories. *International Journal of Geographical Information Science*, 27(2):385–407, 2013. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-84874411735&partnerID=40&md5=7f5c767e6ab8d0e68211ef416caa3add.

[BM09] Wendy Bohte e Kees Maat. Deriving and validating trip purposes and travel modes for multi-day GPS-based travel surveys: A large-scale application in the Netherlands. *Transportation Research Part C: Emerging Technologies*, 17(3):285–297, June 2009. URL: http://www.sciencedirect.com/science/article/pii/S0968090X08000909, doi:http://dx.doi.org/10.1016/j.trc.2008.11.004.

[B.V14] Healthcare Europe B.V. Activity Monitoring |Omron HealthCare, 2014. URL: http://www.omron-healthcare.com/eu/en/our-products/activity-monitoring.

[CV95] Corinna Cortes e Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995. URL: http://dx.doi.org/10.1023/A:1022627411411, doi:10.1023/A:1022627411411.

[FDK+09] Jon Froehlich, Tawanna Dillahunt, Predrag Klasnja, Jennifer Mankoff, Sunny Consolvo, Beverly Harrison e James A L. UbiGreen: Investigating a Mobile Tool for Tracking and Supporting Green Transportation Habits, 2009.

[Fra06] Jean-Marc François. Jahmm - Hidden Markov Model (HMM), 2006. URL: http://www.run.montefiore.ulg.ac.be/~francois/software/jahmm/.

[GAJS06] R K Ganti, T F Abdelzaher, P Jayachandran e J A Stankovic. SATIRE: A software architecture for smart AtTIRE. In *MobiSys 2006 - Fourth International Conference on Mobile Systems, Applications and Services*, volume 2006, pages 110–123, 2006. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-33748982636&partnerID=40&md5=e3cad028539cb5467ff228452ed85495.

[Goo14] Google Inc. Android Developers, 2014. URL: http://developer.android.com/index.html.

[HH13] K A Hummel e A Hess. Movement activity estimation and forwarding effects for opportunistic networking based on urban mobility traces. *Wireless Communications and Mobile Computing*, 13(3):343–360, 2013. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-84873425493&partnerID=40&md5=b6071290ca6a99b8923b25ecbb590479.

[Ho95] Tin Kam Ho. Random decision forests, 1995. doi:10.1109/ICDAR.1995.598994.

[Inc14] Apple Inc. Apple - Corra ou treine com Nike + iPod., 2014. URL: http://www.apple.com/pt/ipod/nike/.

[JL95] George H John e Pat Langley. Estimating Continuous Distributions in Bayesian Classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pages 338–345, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. URL: http://dl.acm.org/citation.cfm?id=2074158.2074196.

[Jt] Junit-team. JUnit. URL: http://junit.org/.

[KR92] Kenji Kira e Larry A Rendell. A Practical Approach to Feature Selection. In *Proceedings of the Ninth International Workshop on Machine Learning*, ML92, pages 249–256, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc. URL: http://dl.acm.org/citation.cfm?id=141975.142034.

[KSS03] N Kern, B Schiele e A Schmidt. Multi-sensor activity context detection for wearable computing, 2003. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-0242592241&partnerID=40&md5=77faca1c4c79b0dfb1e0b3121278a0b1.

[KvRv97] Igor Kononenko, Edvard Šimec e Marko Robnik-Šikonja. Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF. *Applied Intelligence*, 7(1):39–55, 1997. URL: http://dx.doi.org/10.1023/A:1008280620621, doi:10.1023/A:1008280620621.

[LCB06] Jonathan Lester, Tanzeem Choudhury e Gaetano Borriello. A Practical Approach to Recognizing Physical Activities. In *In Proc. of Pervasive*, pages 1–16, 2006.

[LPFK07] Lin Liao, Donald J Patterson, Dieter Fox e Henry Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5–6):311–331, April 2007. URL: http://www.sciencedirect.com/science/article/pii/S0004370207000380, doi:http://dx.doi.org/10.1016/j.artint.2007.01.006.

[Mal86] C Malsburg. Frank Rosenblatt: Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. In Günther Palm e Ad Aertsen, editors, *Brain Theory SE - 20*, pages 245–248. Springer Berlin Heidelberg, 1986. URL: http://dx.doi.org/10.1007/978-3-642-70911-1_20, doi:10.1007/978-3-642-70911-1_20.

[Mar11] RJ Marsan. rjmarsan/Weka-for-Android, 2011. URL: https://github.com/rjmarsan/Weka-for-Android.

[MSS04] N Marmasse, C Schmandt e D Spectre. WatchMe: Communication and awareness between members of a closely-knit group, 2004. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-35048843219&partnerID=40&md5=111139bd8d7137c7d9ed283a1e5012b6.

[Ora14] Oracle. java.com: Java + You, 2014. URL: http://www.java.com/.

[oW14] Machine Learning Group at the University of Waikato. Weka 3 - Data Mining with Open Source Machine Learning Software in Java, 2014. URL: http://www.cs.waikato.ac.nz/ml/weka/.

[PA08] Nam Pham e Tarek Abdelzaher. Robust Dynamic Human Activity Recognition Based on Relative Energy Allocation. In SotirisE. Nikoletseas, BogdanS. Chlebus, DavidB. Johnson e Bhaskar Krishnamachari, editors, *Distributed Computing in Sensor Systems SE - 39*, volume 5067 of *Lecture Notes in Computer Science*, pages 525–530. Springer Berlin Heidelberg, 2008. URL: http://dx.doi.org/10.1007/978-3-540-69170-9_39, doi:10.1007/978-3-540-69170-9_39.

[PLFK03] D J Patterson, L Liao, D Fox e H Kautz. Inferring high-level behavior from low-level sensors, 2003. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-0142156648&partnerID=40&md5=71f516c3265c1e65026c8d41f775323b.

[Pow07] David M W Powers. Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. Technical Report SIE-07-001, School of Informatics and Engineering, Flinders University, Adelaide, Australia, 2007.

[RJ86] L R Rabiner e B H Juang. An introduction to hidden Markov models. *IEEE ASSp Magazine*, 1986.

[RM00] Cliff Randell e Henk Muller. Context awareness by analysing accelerometer data. *International Symposium on Wearable Computers, Digest of Papers*, (Los Alamitos, CA, United States):175–176, 2000. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-0034506339&partnerID=40&md5=fa4bb930c09deb5384197353ee8ef062.

[RMB+10] S Reddy, M Mun, J Burke, D Estrin, M Hansen e M Srivastava. Using mobile phones to determine transportation modes. *ACM Transactions on Sensor Networks*, 6(2), 2010. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-77749264950&partnerID=40&md5=421d8691ce33e05dcdda11bed8e56ddb.

[SCB04] P L Schneider, S E Crouter e D R Bassett Jr. Pedometer Measures of Free-Living Physical Activity: Comparison of 13 Models. *Medicine and Science in Sports and Exercise*, 36(2):331–335, 2004. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-0842346947&partnerID=40&md5=a83e8ebe1418248a666c2be91dd736ff.

[SH00] Y Schutz e R Herren. Assessment of speed of human locomotion using a differential satellite global positioning system. *Medicine and Science in Sports and Exercise*, 32(3):642–646, 2000. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-0034103180&partnerID=40&md5=856aa8876129b0430fe780088112c819.

[Sin84] R.˜W. Sinnott. Virtues of the Haversine. *skytel*, 68:158, 1984.

[SLF+08] T Saponas, J Lester, Jon Froehlich, J Fogarty e J Landay. ilearn on the iphone: Real-time human activity classification on commodity mobile phones. *University of Washington CSE Tech Report UW-CSE-08-04-02*, 2008.

[SVL+06] T Sohn, A Varshavsky, A LaMarca, M Y Chen, T Choudhury, I Smith, S Consolvo, J Hightower, W G Griswold e E De Lara. Mobility detection using everyday GSM traces, 2006. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-33750288403&partnerID=40&md5=87bbe284ee7d9b2d9c943774ae9f4629.

[Tec11] Impact Sports Technologies. Impact Sports Technologies - ePulse2, 2011. URL: http://www.impactsports.com/epulseii.html.

[TWS08] A D Townshend, C J Worringham e I B Stewart. Assessment of speed and position during human locomotion using nondifferential GPS. *Medicine and Science in Sports and Exercise*, 40(1):124–132, 2008. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-38049061120&partnerID=40&md5=34855bc1ba1cdccd702950b6b8a0a244.

[vdSvSdBdH09] Stefan van der Spek, Jeroen van Schaick, Peter de Bois e Remco de Haan. Sensing Human Activity: GPS Tracking. *Sensors (Basel, Switzerland)*, 9(4):3033–55, January 2009. URL: http://www.mdpi.com/1424-8220/9/4/3033, doi:10.3390/s90403033.

[WCM10] S Wang, C Chen e J Ma. Accelerometer based transportation mode recognition on mobile phones. In *APWCS 2010 - 2010 Asia-Pacific Conference on Wearable Computing Systems*, pages 44–46, 2010. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-77954410768&partnerID=40&md5=0f4c62c899fa1677207c096728ce9069.

[Wek09] Weka. weka - ARFF, 2009. URL: http://weka.wikispaces.com/ARFF.

[Wil14] Ed Williams. Aviation Formulary V1.46, 2014. URL: http://williams.best.vwh.net/avform.htm.

[WLLB05] E Welbourne, J Lester, A Lamarca e G Borriello. Mobile context inference using low-cost sensors. In Strang T. e Lindenhoff-Popien C., editors, *Lecture Notes in Computer Science*, volume 3479 of *First International Workshop on Location- and Context-Awareness, LoCA 2005*, pages 254–263, Department of Computer Science and Engineering, University of Washington, Box 352350, Seattle, WA 98195, United States, 2005. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-24944580132&partnerID=40&md5=50cbf762f0a9c24f2f668550fd1a64b4.

[WNB12] P Widhalm, P Nitsche e N Brandie. Transport mode detection with realistic Smartphone sensor data. In *Proceedings - International Conference on Pattern Recognition*, pages 573–576, 2012. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-84874577943&partnerID=40&md5=9e25c4a1c933966a51d04141ba678fd6.

[YS95] Yufei Yuan e Michael J Shaw. Induction of fuzzy decision trees. *Fuzzy Sets and Systems*, 69(2):125–139, January 1995. URL: http://www.sciencedirect.com/science/article/pii/016501149400229Z, doi:http://dx.doi.org/10.1016/0165-0114(94)00229-Z.

[ZCL+10] Yu Zheng, Yukun Chen, Quannan Li, Xing Xie e Wei-Ying Ma. Understanding Transportation Modes Based on GPS Data for Web Applications. *ACM Trans. Web*, 4(1):1:1—-1:36, 2010. URL: http://doi.acm.org/10.1145/1658373.1658374, doi:10.1145/1658373.1658374.

[Zhe07] Yu; Xing Lie Zheng. GeoLife: Building social networks using human location history, 2007. URL: http://research.microsoft.com/en-us/projects/geolife/.

[Zhe10] Yu Zheng. GPS Trajectories with transportation mode labels, 2010. URL: research.microsoft.com/apps/pubs/?id=141896.

[ZLC+08] Y Zheng, Q Li, Y Chen, X Xie e W.-Y. Ma. Understanding mobility based on GPS data. In *UbiComp 2008 - Proceedings of the 10th International Conference on Ubiquitous Computing*, pages 312–321, 2008. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-59249083364&partnerID=40&md5=99b57f5f9628f4f48a0429f259a9debb.

[ZLWX08] Yu Zheng, Like Liu, Longhao Wang e Xing Xie. Learning Transportation Mode from Raw Gps Data for Geographic Applications on the Web. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 247–256, New York, NY, USA, 2008. ACM. URL: http://doi.acm.org/10.1145/1367497.1367532, doi:10.1145/1367497.1367532.

[ZQY13] L Zhang, M Qiang e G Yang. Mobility transportation mode detection based on trajectory segment. *Journal of Computational Information Systems*, 9(8):3279–3286, 2013. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-84877050615&partnerID=40&md5=01126ab44e94cdfd6ac216bba039e4b9.