

Technical Report

Transportation mode identification and real-time CO₂ emission estimation using smartphones

How CO2GO works

Vincenzo Manzoni^{1,2}, Diego Maniloff¹, Kristian Kloeckl¹, and Carlo Ratti¹

¹SENSEable City Lab, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA

²Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milan, Italy

In a context where personal mobility accounts for about two thirds of the total transportation energy use, assessing an individual's personal contribution to the emissions of a city becomes highly valuable. Prior efforts in this direction have resulted in web-based CO₂ emissions calculators, smartphone-based applications, and wearable sensors that detect a user's transportation modes. Yet, high energy consumption and had-hoc sensors have limited the potential adoption of these methodologies.

In this technical report we outline an approach that could make it possible to assess the individual carbon footprint of an unlimited number of people. Our application can be run on standard smartphones for long periods of time and can operate transparently. Given that we make use of an existing platform (smartphones) that is widely adopted, our method has the potential of unprecedented data collection of mobility patterns.

Our method estimates in real-time the CO₂ emissions using inertial information gathered from mobile phone sensors. In particular, an algorithm automatically classifies the user's transportation mode into eight classes using a decision tree. The algorithm is trained on features computed from the *Fast Fourier Transform* (FFT) coefficients of the total acceleration measured by the mobile phone accelerometer. A working smartphone application for the Android platform has been developed and experimental data have been used to train and validate the proposed method.

1 Introduction

According to the Industrial Energy Analysis, Transportation accounts for one quarter of the world greenhouse gas emissions [3], and personal mobility represents about two thirds of the total transportation energy use [5]. In this context, assessing an individual’s personal contribution to the emissions of a city becomes highly valuable.

Existing applications that measure the individual carbon footprint can be classified into aggregated data, personal diaries, and trip-by-trip applications. Aggregated data applications compute the carbon emissions from weekly, monthly or annual averages. Personal diary applications are based on individual profiles, manually created by users in the form of travel logs. Finally, trip-by-trip data applications work on detailed inputs of single trips. Several of these applications have been implemented as web calculators and require manual input from the user, such as the means of transportation used, the distance covered, etc. Others have been programmed into smartphone devices and have different levels of automation (eg., Carbon Diem¹ and Green Meter²). Selected application are shown in 1, classified by type of input and platform.

Table 1: Applications for the estimation of carbon emissions.

Type of input	Web application	Mobile phone application
Aggregated data	Carbonfund.org Carbon Independent Environmental Protection Agency National Energy Foundation World Resource Institute	Green Calculator
Individual diary	Commute Greener	Carbon Tracker Carbon Tracker Ecorio
Trip by trip data	Carbonfund.org	Carbon Diem Carbon Diem Shop Green SHL CO ₂ Calc

The main drawback of current smartphone implementations is that they use GPS data to detect and identify the user’s transportation mode, which poses the following limitations: a) prolonged use of the GPS is notoriously energy expensive [14, 13, 1]; b) GPS satellite reception suffers from canyoning effects, resulting in areas with weak or completely absent signal [10]. Alternate localization technologies based on Wi-Fi or GSM tower triangulation have an acceptable energy cost, but they lack localization accuracy.

¹Carbon Diem: <http://www.carbondiem.com>

²Green Meter: <http://hunter.pairsite.com/greenmeter/>

A different approach for carbon footprint calculators that run on smartphones is to identify the transportation mode based on the device’s accelerometer. While there has been some research in this direction, most efforts have focused on the deployment of ad-hoc sensors carried by people to identify the transportation mode [8, 6, 2], hence limiting the size of deployment and accessibility.

In this paper we present for the first time an approach that could make it possible to assess the individual carbon footprint of an unlimited number of people. Our application can be run on standard smartphones for long periods of time and can operate transparently. Given that we make use of an existing platform (smartphones) that is widely adopted, our method has the potential of unprecedented data collection of mobility patterns.

Our algorithm uses data from the device’s accelerometer, while GPS data and online map queries are used only sparsely. Furthermore, we allow the cellphone to be normally carried in a user’s pocket, without the need for specific fitting in terms of position and orientation. Finally, the data is made relevant to the user by converting it into CO₂ emissions (as a function of mode of transportation and distance) and burnt calories (health monitoring).

2 Related Work

The commoditization of sensors in mobile phones has increased their availability and provided researchers with opportunities to study large populations at a very low cost. One area of interest which can take advantage of the spreading of these sensors has been *activity inference*, i.e. the ability to tell what activity a person is performing based upon sensor information.

Activity inference has been applied in different areas, such as health monitoring, recommendation systems and study of personal behavior. In this paper, the attention is focused on the transportation mode inference, to support the real time estimation of the carbon footprint of a traveler, using information from mobile phone sensors.

Numerous studies have been carried out using external sensors. [6] propose a method to identify the transportation mode using the Mobile Sensing Platform (MSP)³, which combines an Intel XScale processor with storage capacity, an accelerometer, a GPS receiver, and also sensors for pressure, light, humidity and noise. The algorithms presented in [6] rely on the GPS information to identify the transportation mode.

[8] use machine learning methods for activity recognition. They focus their research on the selection of attribute and machine learning algorithms to maximize the recognition accuracy. However, their goal is to identify activities that can anticipate health problems, such as walking, standing, lying down and falling.

Human-activity detection can be analyzed at different scales. Identification of human activity related to particular geographic areas has been analyzed by [11]. They propose a methodology to identify a person’s daily activity pattern using data from the mobile

³MSP Research Initiative: <http://seattle.intel-research.net/MSP/>.

phone network. A hierarchical model that learns and infers user’s daily movements and the use of different modes of transportation is presented in [7].

3 Transportation Mode Identification

This Section describes the algorithm for the automatic identification of transportation modes. Two stages compose the algorithm. First, the signals acquired from the accelerometers are pre-processed to address variable inter-samples intervals. Then, a supervised machine learning algorithm based on decision trees is applied to features computed by the FFT coefficients of the total acceleration.

3.1 System Design and Data Acquisition

The system proposed in this paper can be implemented on any mobile phone with an accelerometer. A GPS receiver and, optionally, a wireless Internet connection are also required for the computation of the travelled distance.

In our work, the mobile phone used for the data acquisition was a Google Nexus One, running the Android OS version 2.2. This phone has been chosen given its fully-fledged programming capabilities based on the Java programming language. The accelerometer integrated in our mobile device is a BMA150. It measures accelerations within a range of ± 2 g (± 19.61 m/s²), with a sensitivity of 4 mg (0.039 m/s²).

Traces were collected and labeled according to different transportation modality through a custom application developed for such purpose. Figure 1 shows its user interface. Parts (1) and (2) show the real-time data for debugging purposes. Part (3) allows the user to select the transportation modality. Parts (4) and (5) start and stop the logging process. The x , y , and z acceleration, together with GPS data for validation have been collected as fast as the device allows.

Our mobile device samples accelerations with an average sampling rate of 25 Hz and GPS data (absolute position according to WGS-84 datum, accuracy and speed) with a frequency of 1 Hz. Unfortunately, the operating system does not guarantee a fixed sampling frequency. For this reason, certain data preprocessing is performed (see Section 3.2).

Traces for bus, metro, and train have been collected by one person, taking the public transportation along different lines. Car and motorcycle traces have been collected by the same person. The traces for walking and biking have been collected by four people, two males and two females. The data set is composed of several hours of traces for each transportation mode.

3.2 Signal Pre-Processing

The inter-sample interval of the acceleration signal obtained from the mobile phone is not uniform and can vary among mobile phones, due to different computational power and active services. Given that the transportation mode classification algorithm is based on features computed on the FFT coefficients of the total acceleration, and it relies on

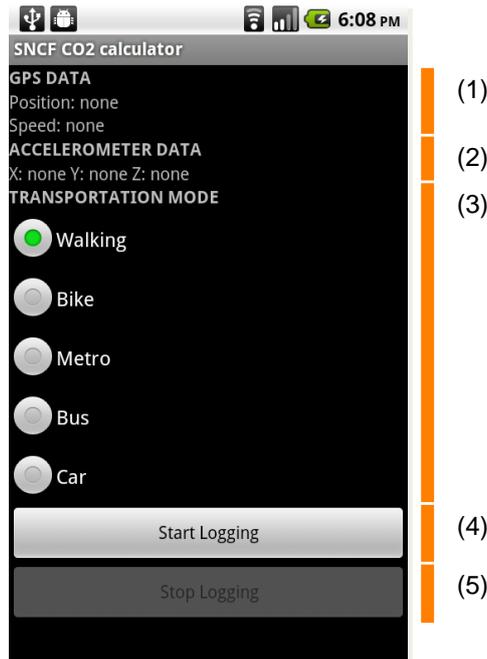


Figure 1: The interface of the application for traces acquisition and their labeling.

samples acquired with a fixed sampling time, the FFT cannot be performed directly, and a signal pre-processing phase is needed. Figure 2 shows a trace 2 seconds long. As it is represented on the bottom chart, the average inter-sample interval is 40.5 ms (which means a sampling frequency of around 25 Hz), but locally it can be slightly faster or slower.

A well established solution in this case is interpolating the samples and to re-sample the signal with a faster sampling frequency. Several interpolation functions can be applied to the signal. Figure 3 shows the comparison between the piecewise linear and the cubic spline interpolation functions.

The two strategies perform similarly when the samples are close enough. The difference of the two strategies is clearly visible when the inter-sample interval is longer. An example is shown in the figure: after 1.5 seconds, the inter-sample interval is of 180 ms, which corresponds to around 5 samples. The cubic spline interpolation is smoother than the piecewise linear one. However, the piecewise linear interpolation is faster to compute and easier to implement on the mobile phone. Moreover, the high-frequencies introduced by the piecewise linear interpolation can be removed by a low-pass filter. The signal is therefore interpolated, re-sampled with a constant sampling frequency of 50 Hz and then filtered with a digital, second-order, low-pass filter with a cut-off frequency of 5 Hz. Given that 25 Hz has been the average slowest sampling frequency experimented, the filter was been designed to have an attenuation of -20 dB in stop band at 12.5 Hz according to the Nyquist theorem (see Figure 4). Moreover, the filter order is chosen as

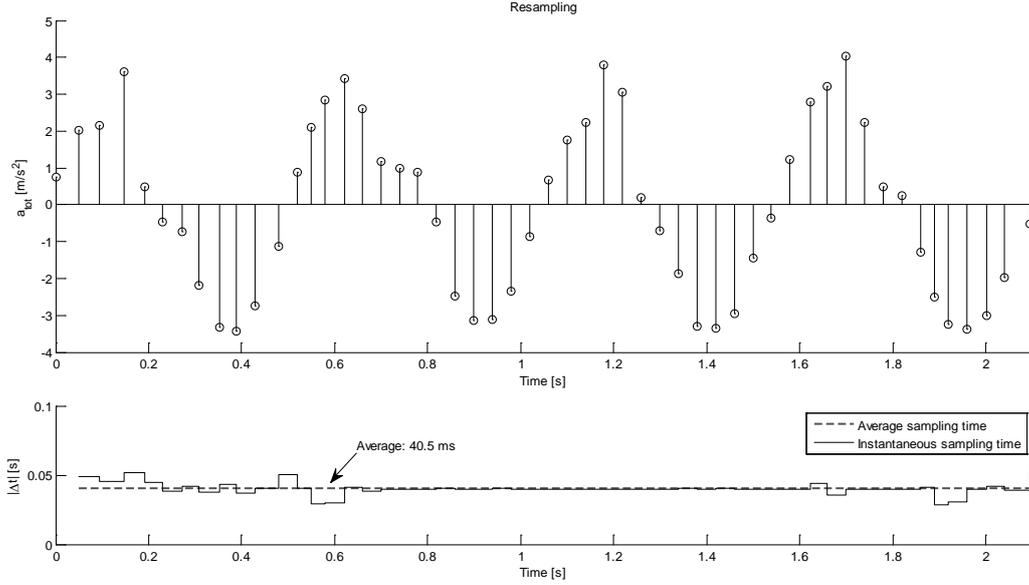


Figure 2: Variability of the inter-samples interval. Top: the acceleration value; bottom: the actual and the average interval between samples.

a compromise between the attenuation rate and the implementation complexity.

Figure 5 shows the block diagram of the pre-processing algorithm, where $a_{\text{phone},x}(t)$ is the acceleration along the x axis read by the phone, $a_{\text{interp},x}(t)$ is the acceleration after the linear piecewise interpolation, f_{res} is the re-sampling frequency, $a_{\text{res},x}(t)$ is the acceleration signal re-sampled, and $\hat{a}_x(t)$ is our estimation. For the sake of simplicity, only the x axis is show, but the algorithm is applied to all three axis.

3.3 Feature Computation

Instead of relying on a fixed (either known or estimated) orientation, we compute the feature set from an orientation-invariant signal. Such signal is the total acceleration, $\hat{a}_{\text{tot}}(t)$ computed as:

$$\hat{a}_{\text{tot}}(t) = \sqrt{\hat{a}_x(t)^2 + \hat{a}_y(t)^2 + \hat{a}_z(t)^2} \quad (1)$$

where $\hat{a}_x(t)^2$, $\hat{a}_y(t)^2$ and $\hat{a}_z(t)^2$ are the accelerations according to the reference system shown in Figure 6, processed with the algorithm shown in Figure5.

Other orientation-invariant signals can be computed, such as the sum of the absolute value of the accelerations. However, the total acceleration has been chosen give its support by previous work [2], and for its clear physical meaning.

The FFT spectrogram of signal $\hat{a}_{\text{tot}}(t)$ exhibits a different signature from one transportation mode to another. Following a line of prior research research, we use the FFT coefficients of this signal as input features to a classification algorithm.

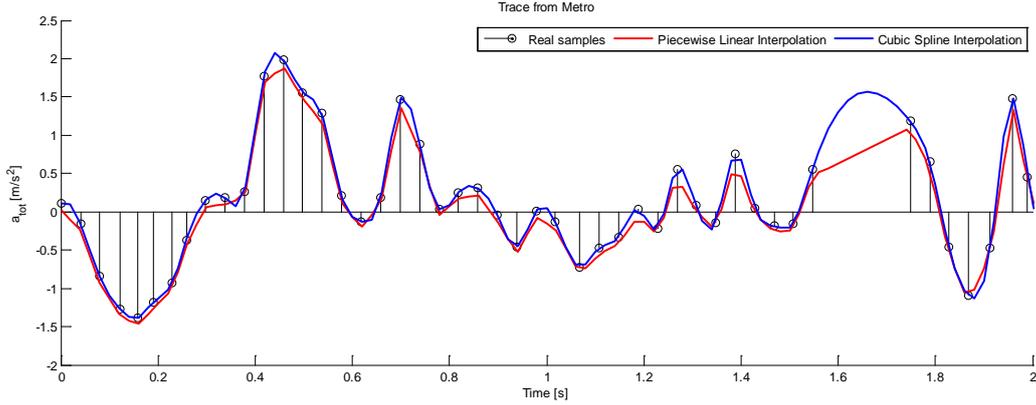


Figure 3: Comparison between the piecewise linear and the cubic spline interpolation functions.

The window size and overlap – i.e. the percentage of overlapping of two consecutive windows – affect the temporal resolution of the spectrogram and therefore the classification accuracy. Figure 7 shows a comparison among spectrograms computed using different window sizes (64, 128 and 256 samples) and windows overlaps (0, 25 and 50%) for the same trace. While small windows and high overlaps generate more instances to train and validate the algorithm and they allow to quickly detect changes, small windows do not catch distinctive peculiarities of transportation systems and high overlaps can overfit the classification algorithm.

The frequency resolution is also important for the classification algorithm. Figure 8 shows a comparison among four spectrograms computed on the same trace (shown on the bottom) with different number of FFT coefficients (32, 16, 8 and 4). Even in this case, a smaller frequency resolution can flag different transportation modes. On the other hand, it makes the algorithm more robust.

The accuracy of the classification strongly depends by these three parameters: window size, windows overlap and number of FFT coefficients. A two-steps optimization process is carried out to compute the parameters values which maximize the classification accuracy. The procedure is described in the next section.

3.4 Classification Algorithm

For our purpose, supervised algorithms are more suitable than un-supervised ones because the training set is labeled with the actual transportation mode. Decision Trees were chosen for two reasons:

1. They usually perform well when the classes are discrete and their number is small. In our case, the transportation modes considered are 8.
2. Previous works state high percentages of accuracy in classifying similar features.

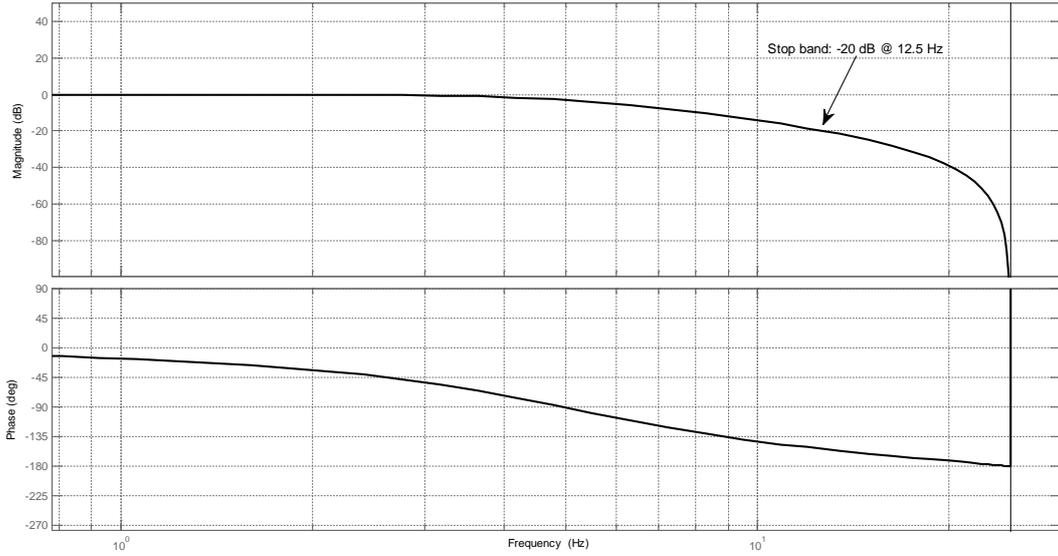


Figure 4: Block diagram of the signal pre-processing algorithm.

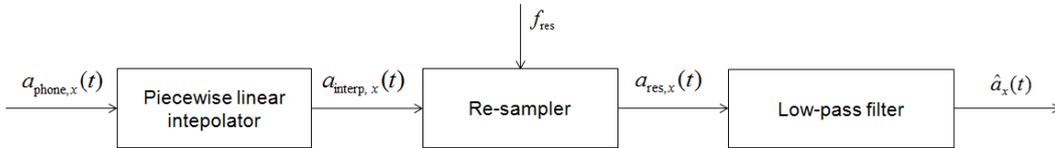


Figure 5: Block diagram of the signal pre-processing algorithm.

3. The classification of an instance is a clear process, mainly the evaluation of a sequence of if/else clauses. Hence, the signal processing can be iteratively optimized for further improving the classification accuracy.

Classification accuracy depends on the features and the parameters of the algorithm. We have optimized these paramteres in two steps:

1. Parameters such as the confidence factor for pruning, the minimum number of instances for leaf, etc., are empirically fixed. Then, several decision trees are built from training sets with different window sizes and window overlaps. All resulting trees are then evaluated with validation sets. For each of them, the classification accuracy is computed.
2. The decision tree with the highest classification accuracy is chosen. Then, the value space of each algorithm's parameter is divided in discrete values, new Decision Tree are trained with all the combinations and the classification accuracy is computed. The one with the highest accuracy is selected.

Figure 9 shows the output of the first phase. The maximum value of classification accuracy is 82.14%, obtained with a window size of 512 samples (10.24 seconds), a

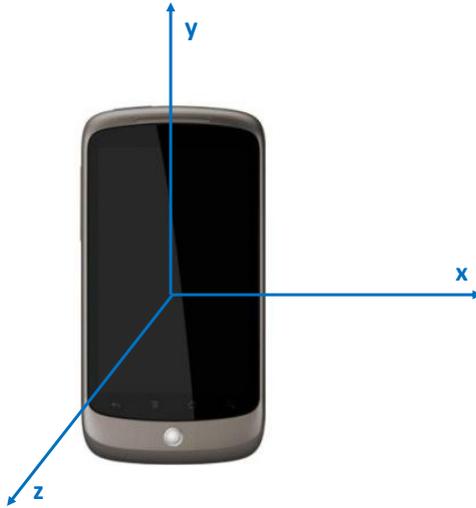


Figure 6: Reference system.

window overlap of 50% (5.12 seconds) and the use of 32 coefficients.

Even though our classification accuracy is slightly lower than that reported by [2], we should note that our data was gathered from a mobile phone’s accelerometer without the use of real-time acquisition systems or several sensors. [15] did use data from mobile phones using only accelerations, but their algorithm is trained on fewer classes (walking, bus, running and a rapid transit).

To summarize, our feature set is composed of:

- 32 FFT coefficients, computed on a window 512 samples long (10.24 seconds), with a windows overlap of 50% (5.12 seconds).
- the signal variance, computed as the sum of the FFT coefficients.

The decision tree algorithm was trained using Weka [4], a well-known environment for knowledge analysis. The tree was generated using the C4.5 algorithm [12], and validated using k -fold cross validation [9], with k equal to 10.

The confusion matrix is represented in Table 2. The head column is the actual class, the head row is the predicted class. The confusion matrix allows us to better understand which transportation modes are identified with higher accuracy, suggesting ways to further improve the classification performances. According to the confusion matrix, walking is identified with the highest accuracy (95%). The reason is that variance associated to walking is much higher than the other transportation modes. In fact, the decision tree classifies it based on such feature. Cars are also very well identified (91.25%). Train and still are the transportation modes identified with less probability than the average 82.14%.

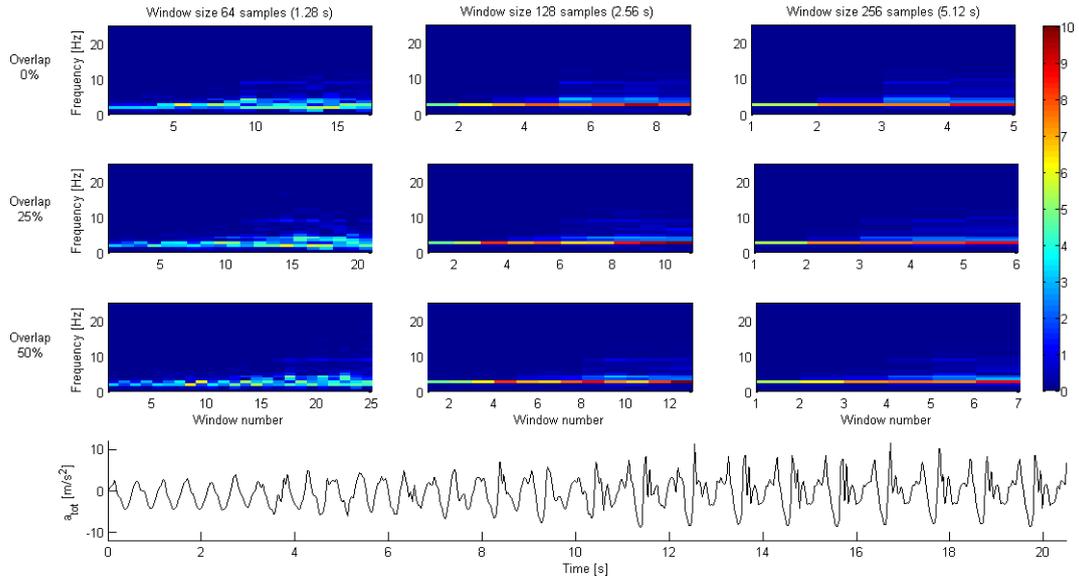


Figure 7: Comparison among spectrograms of different window size and windows overlap for a same walking trace.

4 Concluding Remarks

This report summarizes a method to estimate in real-time the CO_2 emissions using inertial information gathered from mobile phone sensors.

An algorithm identifies transportation modes using decision trees as a supervised classification algorithm. The algorithm's features are the variance and the FFT coefficients of the total acceleration measured by a smartphone's accelerometer. Experimental results show that the classification accuracy of the algorithm is 82.14%.

The method does not rely on a specific orientation of the phone or on data from GPS as previous research works do. However, the GPS and even online maps services can improve the classification accuracy further on. Unfortunately, GPS readings and online queries are energy intensive processes. In any case, the GPS device has to be exploited to compute the traveled distance, a mandatory input for computing the CO_2 emissions.

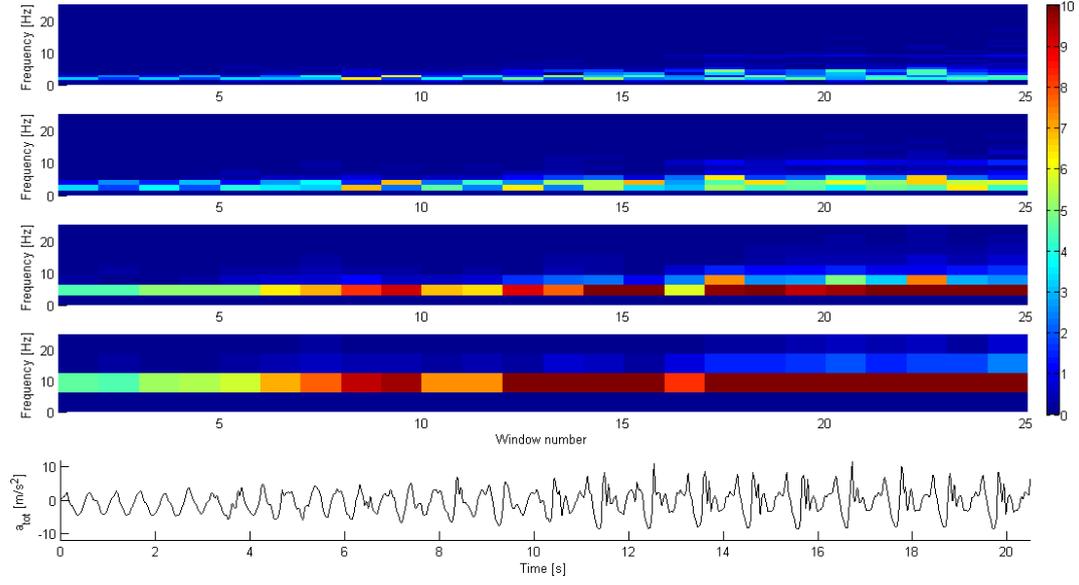


Figure 8: Comparison among spectrograms with different number of coefficients for a same walking trace. Window size: 64 samples (1.28 s), Windows overlap: 0%.

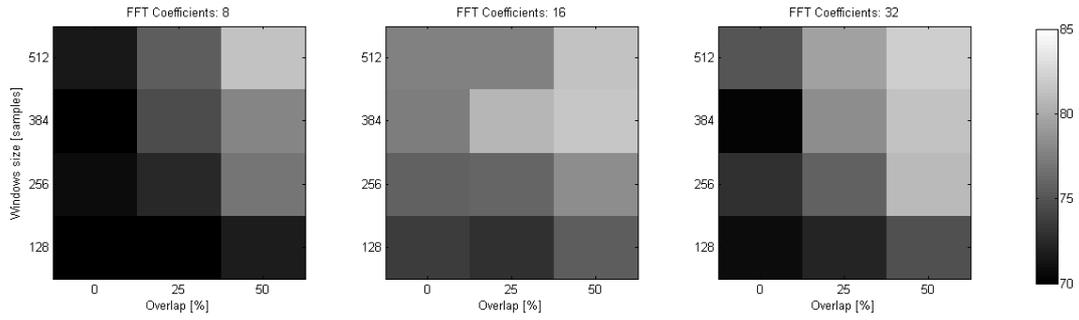


Figure 9: Classification accuracy as function of window size, windows overlap and number of FFT coefficients.

Table 2: Confusion matrix.

Actual	Classified as							
	Bus	Metro	Walk	Bicycle	Train	Car	Still	Motorcycle
Bus	82.50%	8.75%	0.00%	0.00%	7.50%	0.00%	1.25%	0.00%
Metro	5.00%	73.75%	0.00%	2.50%	5.00%	5.00%	7.50%	1.25%
Walk	0.00%	1.25%	95.00%	0.00%	0.00%	0.00%	0.00%	3.75%
Bicycle	0.00%	3.75%	0.00%	88.75%	1.25%	1.25%	1.25%	3.75%
Train	8.75%	3.75%	0.00%	3.75%	77.50%	1.25%	5.00%	0.00%
Car	0.00%	5.00%	0.00%	0.00%	0.00%	91.25%	3.75%	0.00%
Still	12.50%	7.50%	0.00%	1.25%	3.75%	3.75%	70.00%	1.25%
Motorcycle	0.00%	2.50%	1.25%	6.25%	0.00%	2.50%	2.50%	85.00%

References

- [1] W. Ballantyne, G. Turetzky, G. Slimak, and J. Shewfelt. Achieving low energy-per-fix in cell phones. *GPS World*, 17(7):24, 2006. ISSN 1048-5104.
- [2] L. Bao and S.S. Intille. Activity recognition from user-annotated acceleration data. *Pervasive Computing*, pages 1–17, 2004.
- [3] S. de la Rue du Can and L. Price. Sectoral trends in global energy use and greenhouse gas emissions. *Energy Policy*, 36(4):1386–1403, 2008.
- [4] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009. ISSN 1931-0145.
- [5] International Energy Agency (2004). International energy agency, open energy technology bulletin. http://www.iea.org/impagr/cip/archived_bulletins/issue_no23.htm, November 2004.
- [6] J. Lester, P. Hurvitz, R. Chaudhri, C. Hartung, and G. Borriello. MobileSense-Sensing modes of transportation in studies of the built environment. *UrbanSense 2008*, pages 46–50, November 2008.
- [7] L. Liao, D.J. Patterson, D. Fox, and H. Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5-6):311 – 331, 2007. ISSN 0004-3702. doi: DOI: 10.1016/j.artint.2007.01.006. URL <http://www.sciencedirect.com/science/article/B6TYF-4N49VP9-1/2/8f05b8caf7327ceb8762ab5e1b95efc9>.
- [8] M. Luštrek and B. Kaluža. Fall detection and activity recognition with machine learning. *Informatika*, 33(2):205–212, 2009.
- [9] G.J. McLachlan, K.A. Do, and C. Ambrose. *Analyzing microarray gene expression data*. Wiley-IEEE, 2004. ISBN 0471226165.
- [10] B. W. Parkinson. GPS error analysis. *Global Positioning System: Theory and applications.*, 1:469–483, 1996.
- [11] S. Phithakkitnukoon, T. Horanont, G. Di Lorenzo, R. Shibasaki, and C. Ratti. Activity-aware map: Identifying human daily activity pattern using mobile phone data. *Human Behavior Understanding*, pages 14–25, 2010.
- [12] J.R. Quinlan. *C4. 5: programs for machine learning*. Morgan Kaufmann, 1993. ISBN 1558602380.
- [13] D. Raskovic and D. Giessel. Battery-Aware Embedded GPS Receiver Node. In *Mobile and Ubiquitous Systems: Networking & Services, 2007. MobiQuitous 2007. Fourth Annual International Conference on*, pages 1–6. IEEE, 2008.
- [14] F. Simjee and P.H. Chou. Accurate battery lifetime estimation using high-frequency power profile emulation. In *Low Power Electronics and Design, 2005. ISLPED'05. Proceedings of the 2005 International Symposium on*, pages 307–310. IEEE, 2005. ISBN 1595931376.
- [15] Y.C. Yang, T. Toida, and C.M. Hong. Transportations Prediction Using Build-in Triaxial Accelerometer in Cell Phone. 2010.