A Demonstration of a Robust Context Classification System (CCS) and its Context ToolChain (CTC)

Martin Berchtold, Henning Günther and Michael Beigl

Institut für Betriebssysteme und Rechnerverbund

Abstract. This demonstration aims to show the functionality of our tool chain, which supports the robust Context Classification System (CCS) system. The tool chain is developed to collect and annotate sensor data, extract features and to identify the CCS with machine learning techniques. A hands on experience starts by collecting data, visualization and annotation, identification of the CCS, deployment on a commodity phone (e.g. OpenMoko Freerunner [1]) and then test of the CCS through the user. Due to the novel design of our CCS, we are able to detect a high amount of activities (>10 classes with acceleration sensor) and contexts (>5 classes due audio) on the phone in realtime with less than 50% processor load.

1 Overview

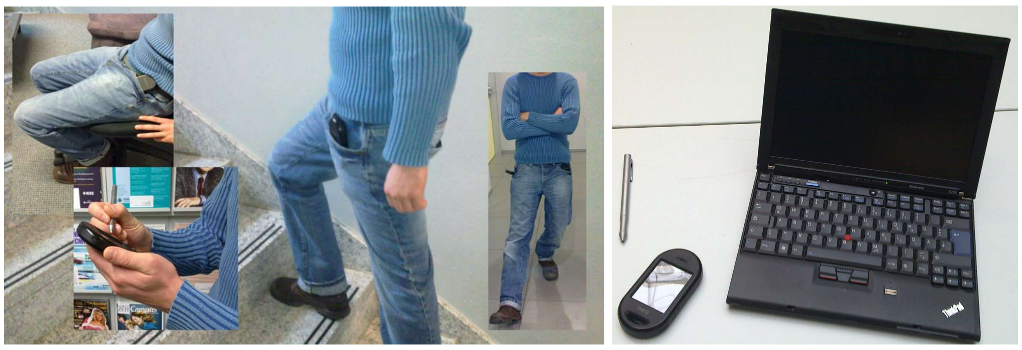

Fig. 1. Example activities recognizable in demonstration and demo equipment.

The purpose of this demonstration is to show the capabilities of our Context-Tool Chain (CTC) [2] and our novel robust Context Classification System (CCS). We will show that using our CTC simplifies the process of collecting and annotating sensory data, the training of classifiers and the deployment of the CCS. Also, the usage of the CCS will demonstrate the capabilities of the classification system, which are high recognition rates for a high amount of classes with a low calculation effort in mean. The CCS also delivers a fuzzy uncertainty with every classification, whereas a filtering upon this demonstrates a further improvement in reliability.

First, we show how data is collected and possibly annotated with the Data Collector Tool (DCT). The data can directly be annotated on the phone with the DCT or, in a second step, be annotated afterwards in the Context Annotator Tool (CAT). The CAT also is responsible for data conversion and visualization. In a next step the features are extracted and the identification of the CCS is done. The identification is done through our novel machine learning algorithm. After identification, the parameters get transmitted to the phone, where a parser interprets them and builds the running CCS. Finally, the CCS can be evaluated through the user, whereas the filtering upon the fuzzy uncertainty can show further improvements. All steps are visualized in figure 2.

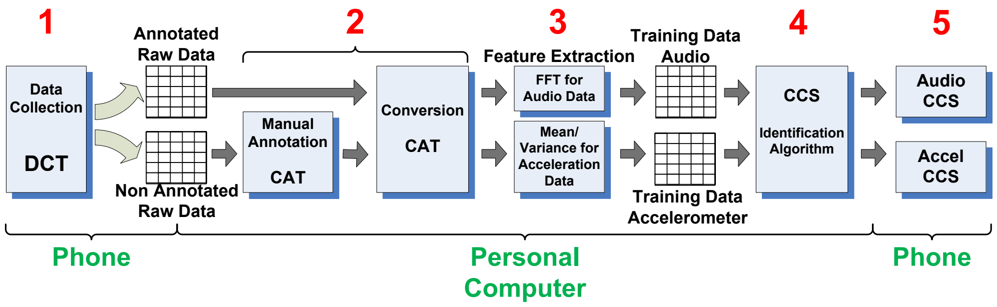

Fig. 2. Training Data Gain using DCT, CAT and Feature Extraction

2 Classification ToolChain

The ToolChain consists of the following tools:

Data Collector Tool: The DCT is a simple application which runs on a mobile phone and collects sensor data. The data is stored using the *Annotation Package Format*, a file format that combines sensor data with annotation data. Annotating the data on-the-fly with context information is also possible. The tool will run on a OpenMoko Freerunner mobile phone for the demonstration.

Context Annotator Tool: The recorded sensor data is then annotated using the CAT. This tool presents the sensor data as graphs to the user, who can then annotate it with context information. Being implemented in python the CAT can run on almost any computer platform. See figure 3 for a screenshot of a running instance.

Context Classifier System (CCS) identification algorithm: The identification algorithm is used to generate suitable classifiers from the annotated sensor data. The algorithm uses a combination of a clustering algorithm, linear regression and genetic algorithm generalization. Details of the algorithm are described in [2].

Context Classification System: The CCS is (for performance reasons) implemented as a native application on the OpenMoko Freerunner. It fetches sensor data and infers the current user-context using the algorithm described in section 3. Context information is either presented to the user (on the phone or remote on a laptop) or e.g. used in an application to switch phone profiles.

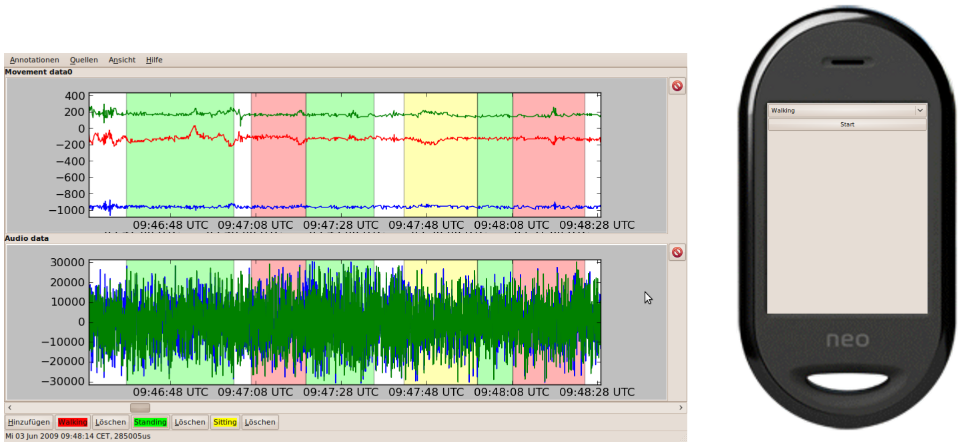
Annotationen Quellen Ansicht Hilfe
Movement data0
Audio data
09:46:48 UTC 09:47:08 UTC 09:47:28 UTC 09:47:48 UTC 09:48:08 UTC 09:48:28 UTC
Hinzufügen Walking Löschen Standing Löschen Sitting Löschen
Mi 03 Jun 2009 09:48:14 CET, 285005us
Walking
Start
neo

Fig. 3. The Context Annotator Tool and a Freerunner running the Context Collector Tool

3 Context Classification System (CCS)

The CCS consists of several steps of processing from a real world value to a tuple of class and reliability. The first step is the sensory, that converts the real world signal into a digital measurement. Secondly, the desired features are extracted from the measurements. In the third step the Recurrent Fuzzy Inference System (RFIS) maps the features onto a classifiable linear set. The linear set is fuzzily classified according to designated fuzzy numbers in the last step. **1. Feature Extraction:** Sensors for classification are e.g. audio and acceleration. The features used for classifying the acceleration measurements are mostly variance and mean values, since they can be calculated with low resource consumption and give good classification results. For audio data the standard extraction method is a "Fast Fourier Transformation (FFT)". Since the dimensionality after the FFT is too high to be used as input for a classifier, mean, variance and frequency centroid is calculated. **2. Recurrent FIS Mapping:** A TSK-FIS [3] is used to map the extracted features onto a set, whose values can be assigned to a class identifier in a separate classification process. The outcome of the mapping at time t is fed back as additional input dimension for the TSK-FIS mapping at $t + 1$. The recurrency not only delivers the desired reliability value, but also stabilizes and improves the mapping accuracy [4]. Our goal is to classify on as many classes as possible. As shown in [5] the abilities of monolithic classifiers are limited, so we use a divide and conquer approach to cope with this problem. Instead of using one classifier to classify on all classes, we use many classifiers each classifying on a small subset of classes. To recognize not only the respective classes, but also the transition between classifiers, each classifier classifies on a complementary class as well. All classifiers are chained in a dynamic queue, where the last classifier classifying on a class different from the complementary class is put first in queue. **3. Fuzzy Classification:** The assignment of mapping result to a class is done fuzzily, so the result is not only a class identifier, but also a membership, representing the reliability of the classification process. Each class is interpreted as a triangular shaped fuzzy number. The mean of the fuzzy number is the identifier itself. The crisp decision (which identifier is the mapping outcome) is carried out based on the highest degree of membership to one of the classes fuzzy number. The overall output of the RFIS mapping and the classification is a tuple $(C_{\mathcal{A}}, \mu_{\mathcal{A}})$ of a class identifier and the membership to it.

Two of the above described CCS are running on the phone, one is classifying the audio, the other one the accelerometer data stream. The CCS for the audio data stream is producing about 16 and the CCS for the accelerometer about 13 classifications per second. A filtering upon the fuzzy uncertainty (μ) reduces the amount of data, but increases reliability sufficiently. The filtering will be part of the demonstration.

4 Summary

In this demonstration the participant can have a hands-on experience of building a context-aware system for a commodity phone. The experience starts by collecting data, whereas the participant does the activities she wants to recognize. An annotation of the data set can be done on the phone or in a separate process with a graphical tool. The machine learning algorithm produces the classification system, which is deployed onto the phone. In an evaluation phase, the participant can experience the accuracy and performance of the built system. A filtering upon a fuzzy uncertainty should show, how the accuracy can be improved.

References

1. : OpenMoko project http://www.openmoko.org.
2. Günther, H., Simrany, F.E., Berchtold, M., Beigl, M.: A tool chain for a lightweight, robust and uncertainty-based context classification system. ARCS Workshop CoSDEO (2010)
3. Tagaki, T., Sugeno, M.: Fuzzy identification of systems and its application to modelling and control. Syst., Man and Cybernetics (1985)
4. Berchtold, M., Beigl, M.: Increased robustness in context detection and reasoning using uncertainty measures - concept and application. Proceedings of the European Conference on Ambient Intelligence (AmI'09) (2009)
5. Berchtold, M., Riedel, T., Beigl, M., Decker, C.: Awarepen - classfication probability and fuzziness in a context aware application. Ubiquitous Intelligence and Computing (2008)

A Demonstration of a Robust Context Classification System (CCS) and its Context ToolChain (CTC) - Demonstration Description

Henning Günther

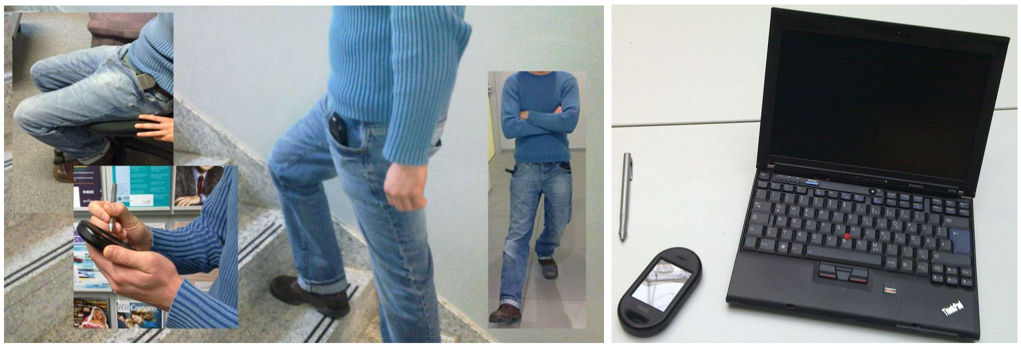

Fig. 1. Example activities recognizable in demonstration and demo equipment.

1 Hardware requirements

The demonstration uses the following hardware components:

- 2-3 OpenMoko Freerunner mobile phones
- 1-2 laptop computers
- Possibly a W-LAN router

The laptop computer is connected via W-LAN (or if wireless is buggy via USB) to both phones. The space the demo requires is as large as a table the laptop(s) are placed on and an additional area of 1m×1m for the user to do the activities.

2 Procedure

1. The user uses a OpenMoko Freerunner phone to collect sensor data. This requires the user to perform certain activities (e.g. walking around, sitting down and/or dancing) or audio contexts (e.g. talking, typing on a keyboard and/or clapping hands). The respective annotation can here already be done on the phone.
2. The collected data is uploaded to the laptop, where its investigated and possibly annotated.
3. A set of classifiers is generated on the laptop through machine learning.
4. The classifiers are uploaded to the phone and the user can try them by performing the same activities used in step 1. The recognized activities and contexts are shown on the phone and remote on a laptop. Possibly a application of automatic phone profiles is triggered.