



# Context prediction based on learning user habits: A next step towards "smart" systems



31. January 2006, Helsinki

## **Guest lecture 1**

Rene Mayrhofer  
Computing Department  
Lancaster University, UK  
[rene@comp.lancs.ac.uk](mailto:rene@comp.lancs.ac.uk)

# Problem area

Introduction

What is *Pervasive Computing*?

Approach

Architecture

Implementation

Results

Contribution

What is

What is

- 1 Erinnerung
- Vortrag Prof. Gellersen
  - Beginn: Donnerstag, 11. September 2004 15:30
  - Ort: HS1

Betreff	Farbe
Vortrag Prof. Gellersen	6 S

Alle schließen    Element löschen    Schließen

Klicken Sie auf "Erneut erinnern" nach Ablauf des untergewählten Zeitraums erneut zu werden.

5 Minuten vor dem Start    Erneut erinnern

objects

context  
the

ation in

# Context Awareness

## Introduction

Many definitions in this sense, e.g.:

- *“Three important aspects of context are: where you are, who you are with, and what resources are nearby [...]. Context encompasses more than just the user’s location, because other things of interest are also mobile and changing.”* [SAW 1994]
- *“any information that can be used to characterize the situation of an entity, where an entity can be a person, place or a physical or computational object”* [Dey 1999]
- A working model for context [Schmidt 2002]

## Approach

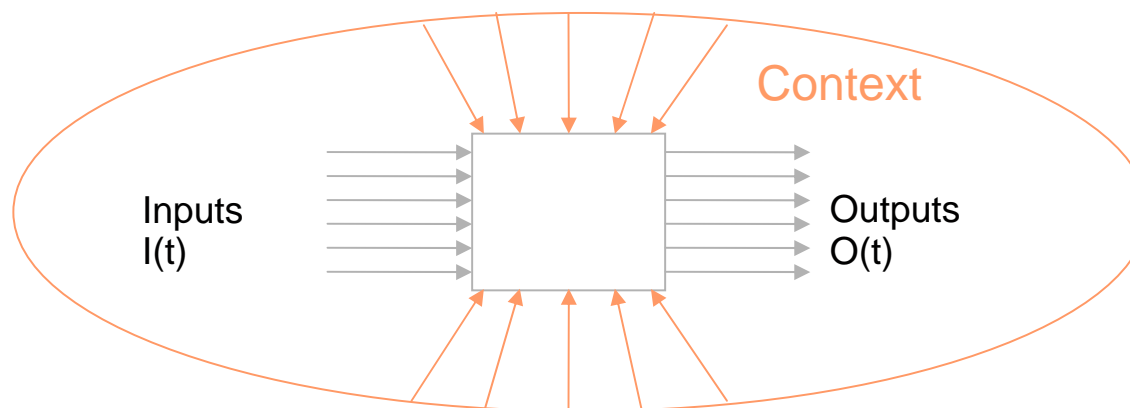
## Architecture

Context is everything **except** the explicit in- and outputs [LS 2000].

## Implementation

## Results

## Contribution



## Problem description of this research:

How can the **current context** of a device or its user be determined autonomously from sensor information and the **future context** be predicted from recorded context histories?

[SAW 1994] B.N. Schilit, N. Adams, R. Want: „Context-aware computing applications“

[Schmidt 2002] A. Schmidt: “Ubiquitous Computing – Computing in Context”, PhD Thesis, Lancaster Univ.

[Dey 1999] A. Dey, G.D. Abowd, D.Salber: „A Context-based infrastructure for smart environments“

[LS 2000] H. Lieberman, T. Selker: „Out of context: Computer systems that adapt to, and learn from, context“



## Context Awareness (2)

### Introduction

Context has a vast multitude of different aspects, e.g.

- time
- location
- physical (temperature, humidity, etc.)
- social (with colleagues / family etc.)

### Approach

### Architecture

⇒ It seems sensible to use multiple small sensors instead of a single, but more powerful one (cf, Gellersen et.al.)

### Implementation

### Results

### Contribution





# Proactivity

## Introduction

Proactive vs. reactive behaviour of a (computer-) systems

- Determines the capabilities of a system to merely react to changes in the environment or to act in advance
- Important feature of **software agents**
- Here: definition of proactivity based on system states [MRF 2003a]

## Approach

## Architecture

## Implementation

## Results

## Contribution

- The current internal state of a (Moore) state machine depends on the last system state and its current inputs:

$$q_t = \delta(q_{t-1}, a_{t-1})$$

- **Reactive system:** determine the output depending on the current state

$$b_t = \lambda(q_t)$$

- **Proactive system:** additional dependency on **predicted** future states

$$b_t = \lambda\left\langle q_t, \bar{q}_{t+1}, \bar{q}_{t+2}, \dots, \bar{q}_{t+m} \right\rangle$$



## Related work

### Introduction

- TEA

- Smart-Its

### Approach

- University of Helsinki

- Context Toolkit

### Architecture

- Robotics

- CIS

### Implementation

- Neural Network House

- Aware Home

- MavHome

### Results

- ContextCube

- Portolano

### Contribution

- University of Washington

- ...



# Basic approach to context prediction

Introduction

Approach

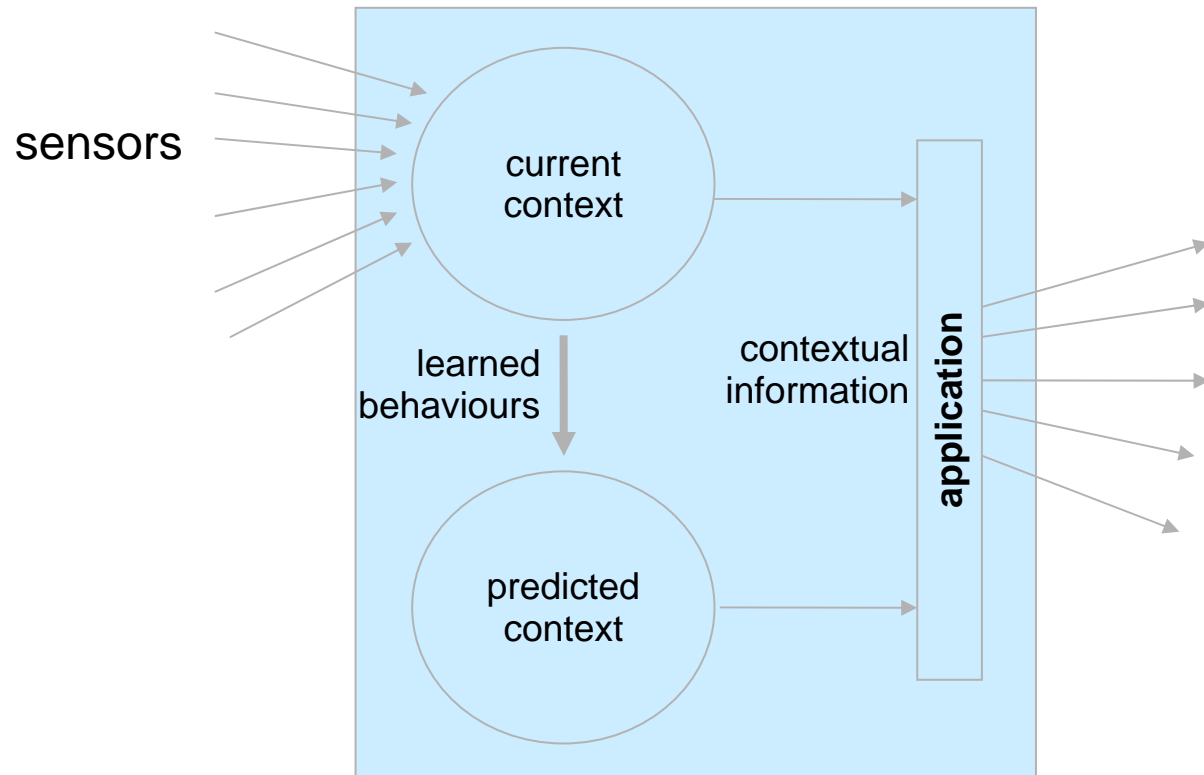
- Recognize the current context based on multiple sensors (e.g. [Schmidt 2002])
- Predict future context by learning user behaviour
- ... as far as possible, without domain specific knowledge

Architecture

Implementation

Results

Contribution





# Concept

Introduction

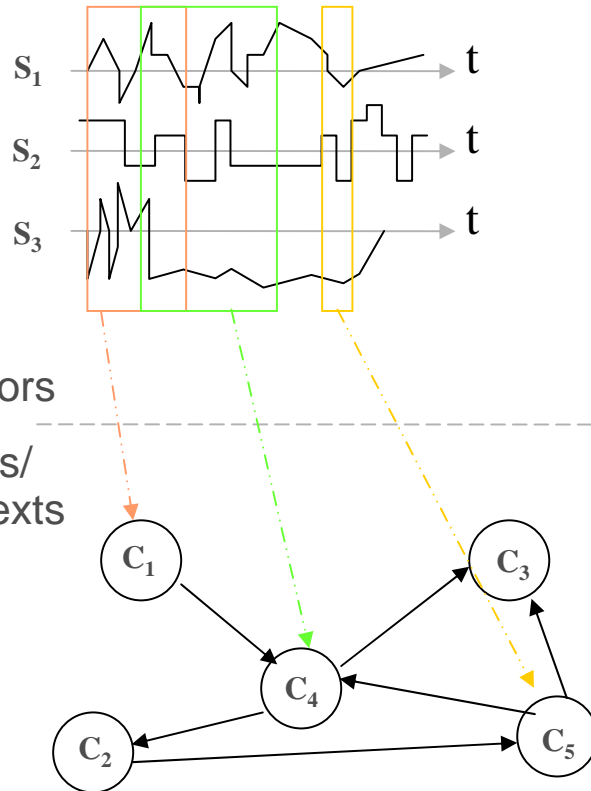
Approach

Architecture

Implementation

Results

Contribution



- Sensors yield time series
- Particular patterns in the input streams can be interpreted as the states of a (observable but not controllable) state machine
- These states are understood as the device (or user) contexts
- Then it becomes possible to predict future contexts by extrapolating the state trajectory into the future

# Architecture

Introduction

Approach

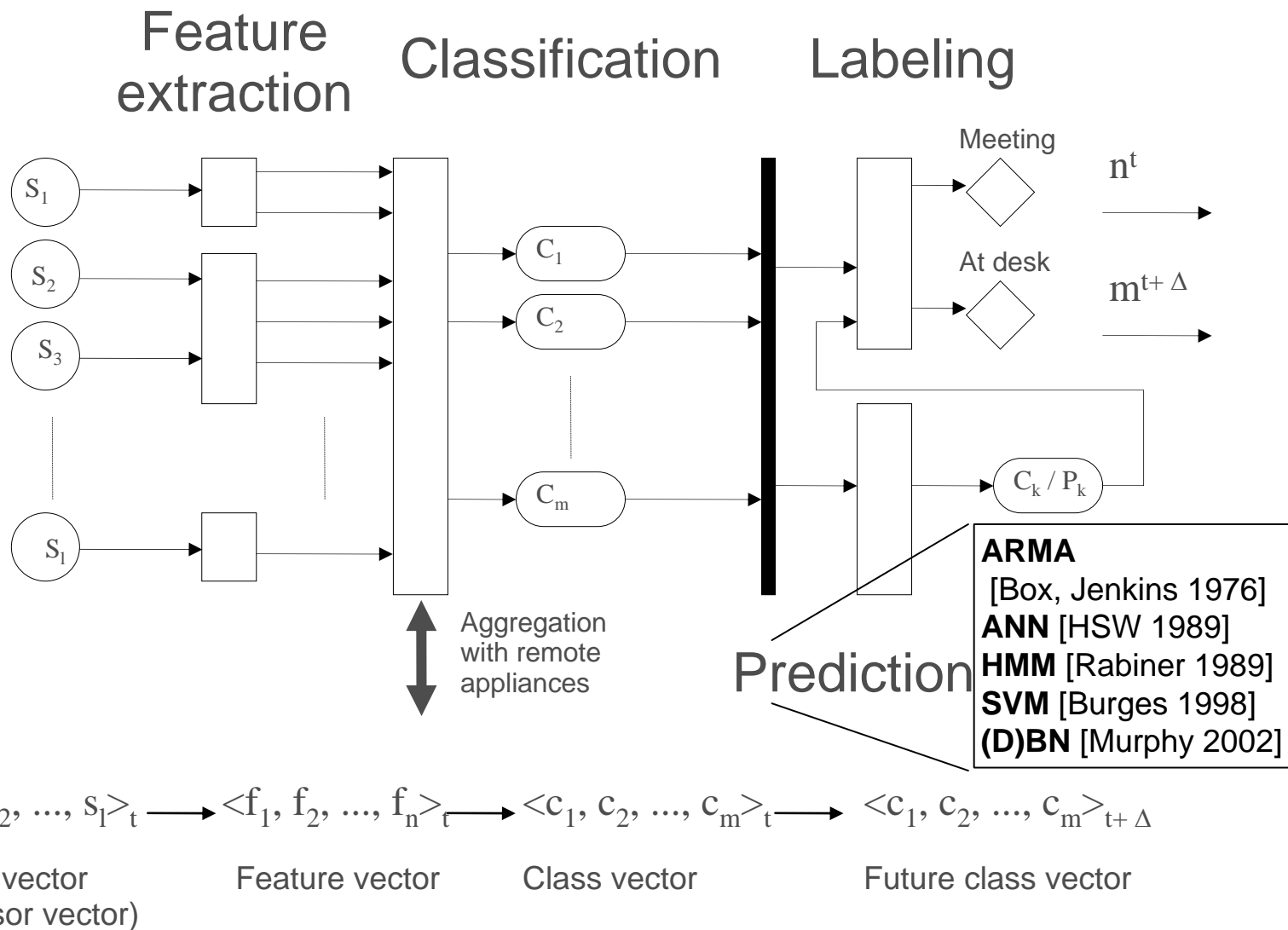
Architecture

Implementation

Results

Contribution

Sensors



## Step 1: *Sensor data acquisition*

### Introduction

- Sensors yield time series
- Sampled either at regular intervals or based on events
- Examples for currently available “sensors” that can help to determine the current context on a typical mobile off-the-shelf device:

### Approach

### Architecture

- time
- application being used
- brightness
- microphone
- Bluetooth
- WLAN
- docked/undocked

### Implementation

### Results

- Additional sensors can be connected easily:

- GPS
- GSM
- compass
- accelerometers
- tilt sensors
- temperature sensors
- pressure sensors
- ....

### Contribution

- **Sharing of sensor values between nearby devices**





## Step 2: Feature extraction

Introduction

- Transforms raw sensor values into more meaningful features
- Applying domain-specific knowledge

Approach

- Multiple features can be generated from a single sensor data stream (and vice versa)

Architecture

⇒ High dimensional feature space

Implementation

- Different types of features:
  - Numerical (continuous): e.g. brightness, heart rate, microphone
  - Numerical (discrete): e.g. number of access points in range
  - Ordinal: e.g. day of week
  - Nominal: e.g. current WLAN SSID, list of WLAN/BT devices in range

Results

Contribution

- Notice: only two operations are necessary for each feature dimension
  - similarity measure (distance metric)

- adaptation op

$$\alpha(f, g, a) : \alpha(f, g, a) := \begin{cases} f & \text{if } a \leq 0.5 \\ g & \text{if } a > 0.5 \end{cases}$$



## Step 3: Classification

Introduction

- Classifies features and recognizes common patterns (clusters) in the input data  $\Rightarrow$  possible without user interaction, **unobtrusive**

Approach

- Different types of classification algorithms

- Type (**partitioning** / hierarchical)
- „**soft**“ / „hard“ classification
- supervised / **unsupervised**

**Architecture**

Implementation

- Requirements on algorithms for context recognition:

- Online / incremental
- Adaptive
- Dynamic number of classes and dynamic structure
- Finding clusters in sub spaces
- “Soft” classification
- Robustness against noise
- Low resource consumption
- Simplicity
- Interpretability of classes / protection of user privacy

Results

Contribution



# Classification algorithms

Introduction

Approach

Architecture

Implementation

Results

Contribution

Algorithm	Topology / number of classes	Online / incremental	Adaptive	Hard / soft
K-Means	fixed	yes	yes	hard
FCM	fixed	yes	no	soft
Neural Gas	fixed	yes	no	partially
SOM	fixed	no	no	soft
ART	variable	yes	no	soft
IDBSCAN	variable	incremental, but not online	no	soft
SNN	fixed or variable	yes	no	soft
Growing Neural Gas [Fri 2003]	variable	yes	yes	soft



# Variation: Lifelong Growing Neural Gas (LLGNG)

Introduction

- Modification of GNG for life-long learning [Ham 2001]
- Difference: applies local criteria for each cluster to prevent the unbounded insertion of new clusters (instead of a global limit on the number of clusters)

Approach

Architecture

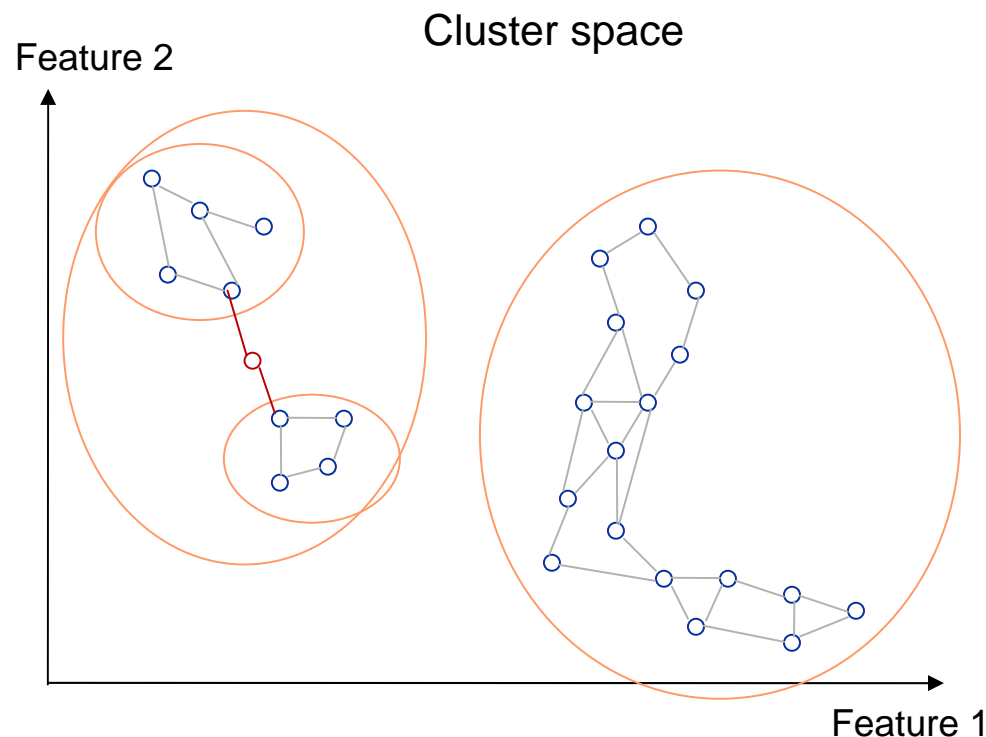
- Extensions to LLGNG for context recognition:

- Usage of a heterogeneous feature space
- Utilising the internal topology to allow arbitrarily shaped clusters  
⇒ **Meta clusters** as additional level of abstraction

Implementation

Results

Contribution





## Step 4: Labeling

Introduction

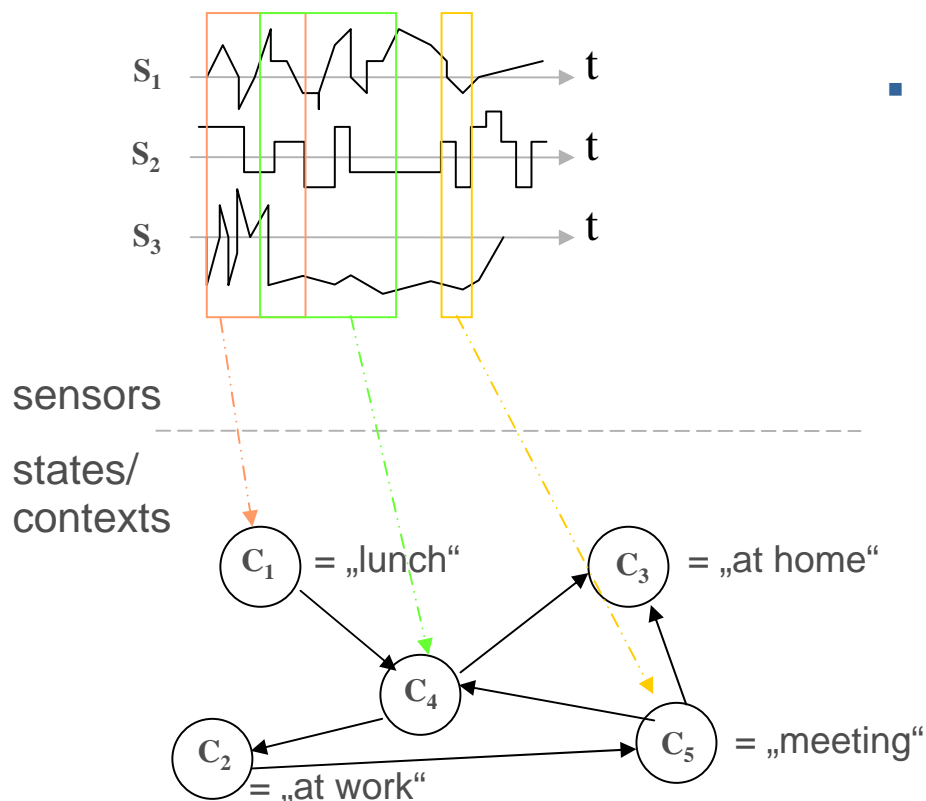
Approach

Architecture

Implementation

Results

Contribution



- $1:\{0,1\}$  mapping of (meta) clusters to context labels
- basically two options:
  - for stable (meta) clusters from classification step: direct mapping of (meta-) cluster ID to labels
  - for significantly varying (meta) clusters (by learning/adaptation): additional, simple clustering step [Lae 2001]



## Step 5: Prediction

Introduction

- Recognized contexts can be interpreted as „states” of a state machine
- Monitoring the state trajectory allows to extrapolate and thus to predict it

Approach

- Important aspects of time series prediction:
  - periodical patterns (e.g. week ends, regular meetings)
  - sequential patterns (e.g. travel preparations, preparing a meal)
  - trends (changing behaviours)

Architecture

Implementation

- Requirements on algorithms for context prediction:
  - Unsupervised model estimation
  - Online
  - Incremental model growth
  - Confidence estimation
  - Automatic (implicit) feedback
  - Manual (explicit) feedback
  - Long term vs. short term prediction

Results

Contribution



# Options for context prediction

Introduction

- Based upon the trajectory of context classes
- Advantage over the independent prediction of different feature values: consideration of all aspects of context that are recognized by the available sensors

Approach

Architecture

- Two options:
  - Predict each dimension of the class vector as **continuous time series**  
⇒ does not consider relationships between context classes, but allows for overlapped contexts
  - Aggregate all classes to a single, **categorical time series** *by using the most probably context class at each time step:*  
*AAACCBCCAAADDDDDDDDEEEEEEBBAAAAA.....*  
⇒ implicitly considers relationships between context classes, because they are considered as mutually exclusive

Implementation

Results

Contribution

⇒ **Flat vs. hierarchical/overlapping context model**

- Many known methods for time series prediction to select from for a specific application
- Selection is necessary, because the requirements/features of the specific methods are too diverse (there is no “best” method for time series prediction yet)



# Implementation as Software Framework

Introduction

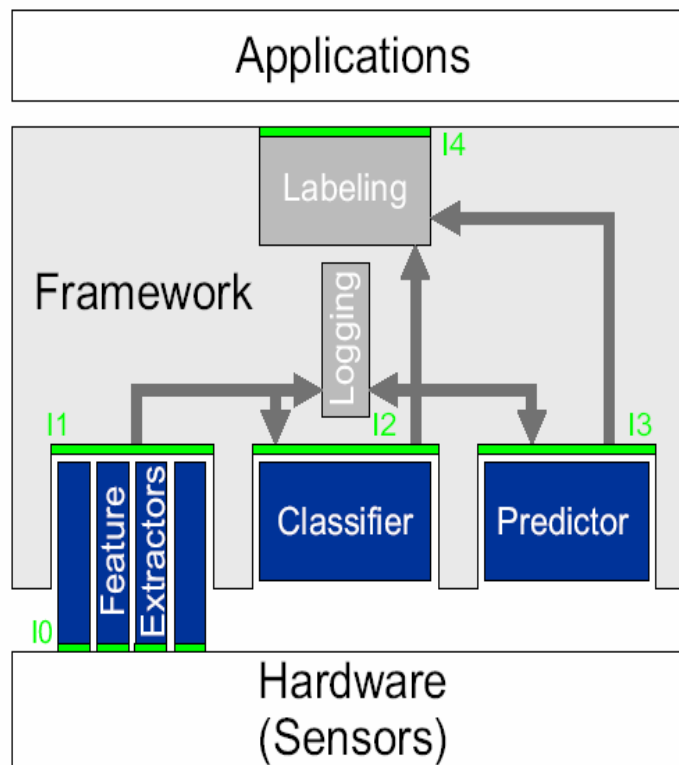
Approach

Architecture

Implementation

Results

Contribution



Cross-platform:

- Currently for Win32, Windows CE (>=3.0), Linux IA32 and ARM and (partially) Symbian OS
- Based on modules that are loadable at run-time:
  - Feature extractors (= sensor data acquisition + feature extraction)
  - Classifiers
  - Predictors
- Labeling is realized by providing network transparent interfaces (SOAP)
  - ⇒ splitting HCI issues from context recognition/prediction issues
- Designed for resource limited devices



# Initial evaluation with real-world data

Introduction

- Recorded with a standard laptop

Approach

- Recording completely in the background  $\Rightarrow$  unobtrusive operation

Architecture

- Time frame: ca. 2 months

Implementation

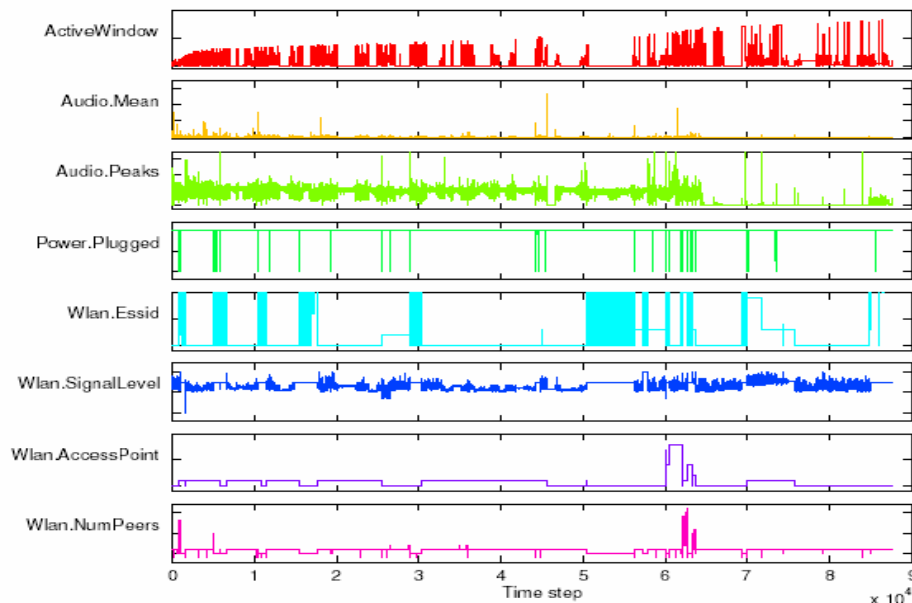
- 28 feature dimensions
- $\sim$  90000 data points

Results

Contribution

## Classification:

- Evaluation of K-means, SOM and LLGNG
  - K-means: smallest error with 6 clusters: 0,7451
  - SOM: smallest error with 71x21 (=1491) neurons in output layer: 0,5659
  - Extended LLGNG: 9 meta clusters with 109 clusters: 0,0069
- LLGNG produces a smaller error with less clusters than SOM
- Extended variant offers the additional concept of meta cluster





## Evaluation with real-world data (2)

Introduction

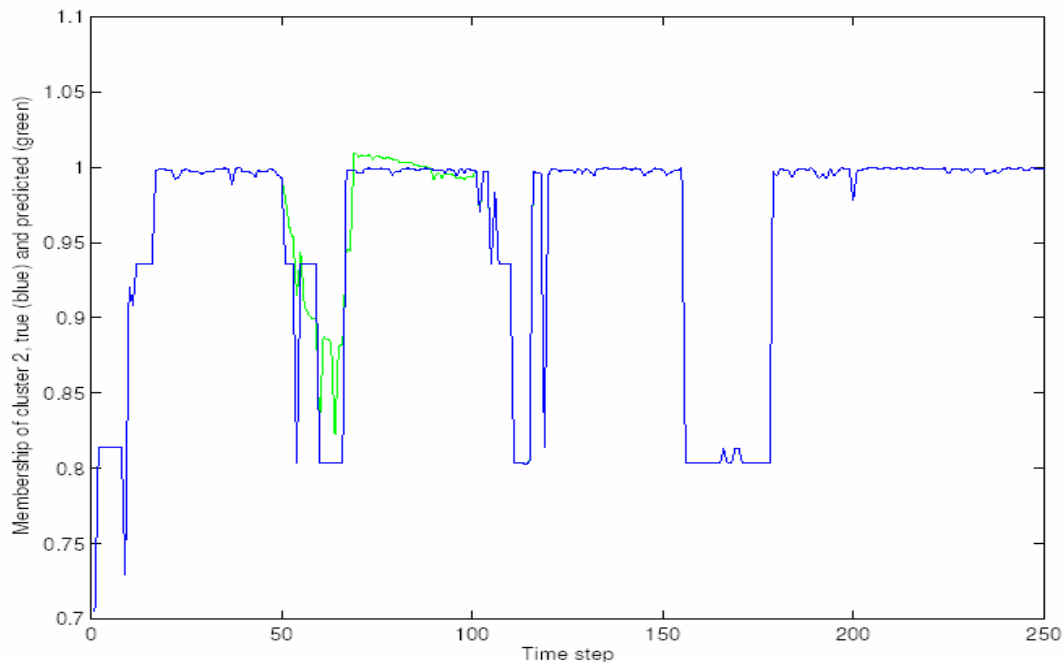
Approach

Architecture

Implementation

Results

Contribution



Algorithm	Error
ARMA	0,04
MLP	0,62
SVR	0,24
Central tendency	0,46
ALZ	0,46
ALZ + Duration	0,44
HMM	0,46
SVM	0,46

### Prediction:

- Currently best results with ARMA and seasonal correction in pre-processing
- HMM, SVM etc. less suitable (for this data set, no general statements should be deducted)



# Contribution of this project

Introduction

Development of an **architecture for context prediction**:

- Predicting context on a high level instead of predicting individual aspects of context. This is achieved by interpreting contexts as states and monitoring the emerging state trajectory.
- Flexibility due to simple, clearly defined interfaces between the 5 steps
- Classification, labeling, and prediction are exchangeable plug-ins
- Able to deal with heterogeneous input data in a unified manner

Approach

Architecture

Implementation

Requirements / methods for context classification and prediction

Results

Implementation of the architecture in terms of an open software framework:

- Direct use of different sensor types without re-coding of non-numerical data
- Extension of LLGNG by the concept of meta clusters and various optimisations
- Supporting different platforms
- Developed for resource limited devices

**Contribution**



# Open Issues and Outlook

## Introduction

- Wide open: exploring the area of **applications** that employ context prediction and user studies

⇒ new research issues expected

## Approach

- Gather new data sets **with** ground truth  
⇒ should allow better quantitative comparison of algorithms

## Architecture

- User interface for **labeling**

## Implementation

- Evaluate new prediction algorithms (e.g. [EAE 2004]), especially for online recognition of **periodicities**

## Results

- Support for **event** based features in the architecture
- Optional use of **domain-specific knowledge** for recognition and prediction, if available

## Contribution

- Implementation of additional classification and prediction algorithms within the framework
- Porting the framework to new platforms, e.g. Familiar or Montavista-based mobile phones (sensor and UI support)



## Summary

- Upcoming applications of computer systems that go beyond the usual desktop demand new ways of user interaction.
- Recognizing current and predicting future context opens manifold possibilities for more intuitive and “smart” user interaction.
- This research project proposes a modular architecture for context prediction based on 5 steps. it allows an automatic recognition and prediction of high-level context from low-level, simple sensor data and focuses on unobtrusive operation. The proposed steps are:
  - Sensor data acquisition
  - Feature extraction
  - Classification
  - Labeling
  - Prediction
- This architecture has been developed for online, life-long learning with continuous adaptation to new situations and explicitly considers low resource requirements and protecting user privacy.
- To directly use heterogeneous feature values without internal conversation, only two operations need to be implemented for each feature.
- This is only the beginning, many issues are still open...



*“It is hard to predict, especially the future.”*



Niels Bohr

Winner of the 1922 Nobel Prize in Physics



***“If we knew what it was we were doing, it would not be called research, would it?”***



Albert Einstein

Winner of the 1921 Nobel Prize in Physics



***Thank you for your attention!***

