JOHANNES KEPLER
UNIVERSITÄT LINZ
INFORMATIK

Kontextvorhersage: Ein weiterer Schritt in Richtung "intelligenter" Systeme

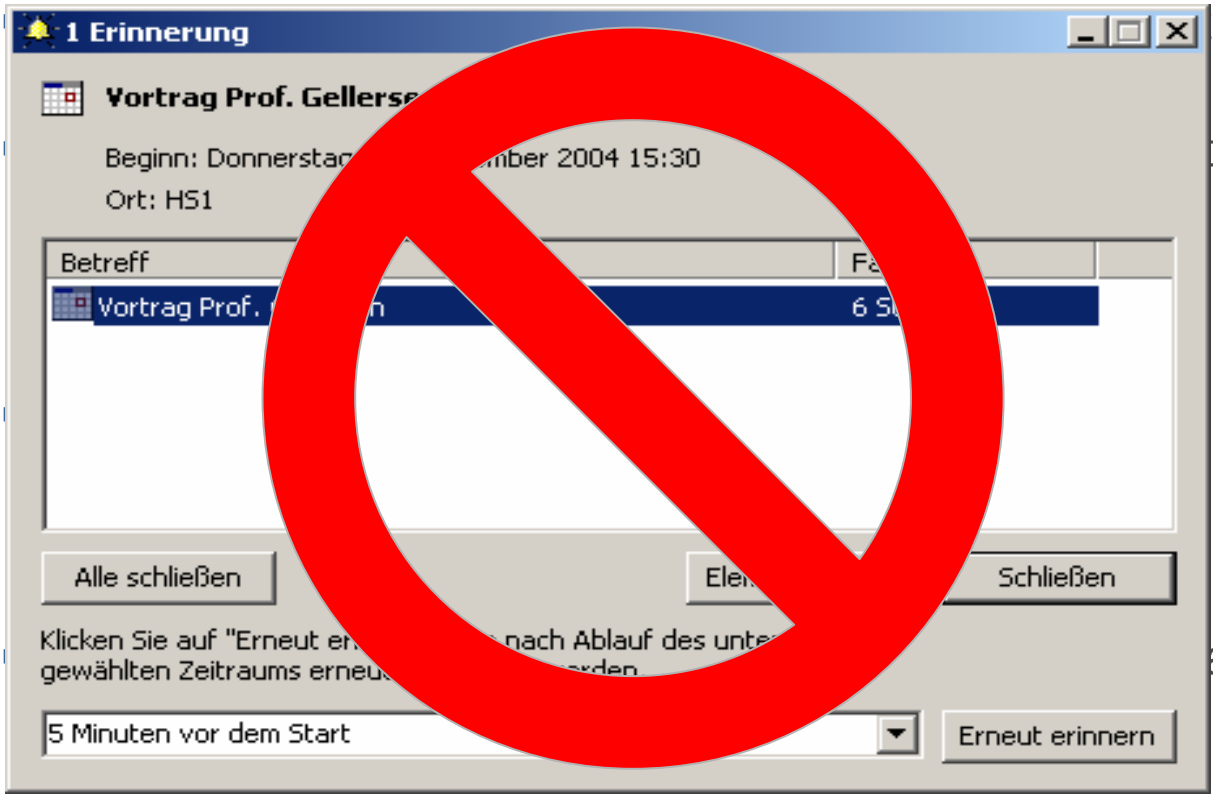
1. Juni 2005, Darmstadt

Gastvortrag

Rene Mayrhofer
Institut für Pervasive Computing
Johannes Kepler Universität Linz, Austria
rene@soft.uni-linz.ac.at

Problemfeld

Was ist *Pervasive Computing*?



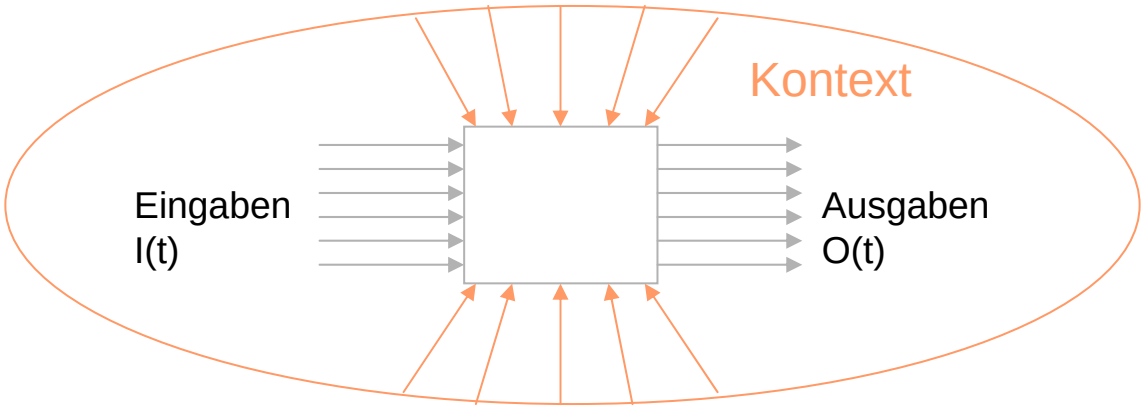
Was ist *Context Awareness*?

Context Awareness

Viele Definitionen für Kontext in diesem Sinne, z.B.:

- *"Three important aspects of context are: where your 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]

Kontext ist alles ***außer*** den expliziten Ein- und Ausgaben [LS 2000].



Problembeschreibung:

Wie kann ein Gerät seinen (oder seines Benutzers) ***aktuellen Kontext*** *ableiten und* ***zukünftigen Kontext*** *ausgehend von aufgezeichneten Kontexthistorien vorhersagen?*

[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)

Kontext hat verschiedene Aspekte, z.B.:

- Zeit
- Ort
- physikalisch (Temperatur, Luftfeuchtigkeit, etc.)
- sozial (unter Kollegen / Familienmitgliedern etc.)

⇒ Daher vernünftig, mehrere einfache Sensoren zur Erfassung dieser Aspekte zu verwenden (siehe Gellersen et.al.)

Proaktivität

Proaktives vs. reaktives Verhalten eines (Computer-) Systems

- Bestimmt, ob das System lediglich auf Änderungen in seiner Umgebung reagiert, oder ob es in Voraussicht handeln kann
- Wichtige Eigenschaft von Software-Agenten
- Definition von Proaktivität basierend auf Systemzuständen [MRF 2003a]

- Der interne Zustand einer (Moore) Zustandsmaschine ist abhängig vom letzten Zustand und den aktuellen Eingaben: $q_t = \delta(q_{t-1}, a_{t-1})$

- Reaktives System: Ausgabe anhängig vom aktuellen Zustand

$$b_t = \lambda(q_t)$$

- Proaktives System: Ausgabe zusätzlich von vorhergesagten zukünftigen Zuständen abgängig

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

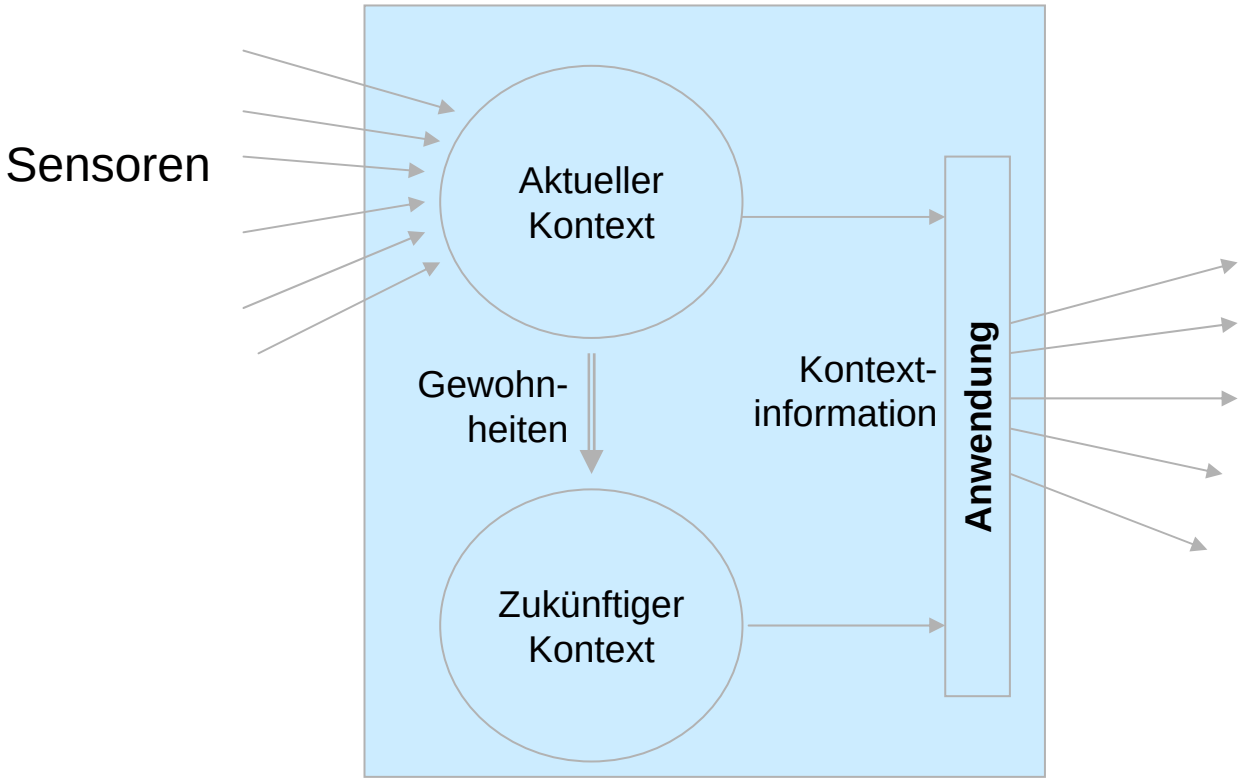
[MRF 2003a] R. Mayrhofer, H. Radi, A. Ferscha: „Recognizing and Predicting Context by Learning From User Behavior“, Proceedings of MoMM2003, OCG, September 2003

Verwandte Arbeiten

- TEA
- Smart-Its
- Universität Helsinki
- Context Toolkit
- Robotics
- CIS
- Neural Network House
- Aware Home
- MavHome
- ContextCube
- Portolano
- …

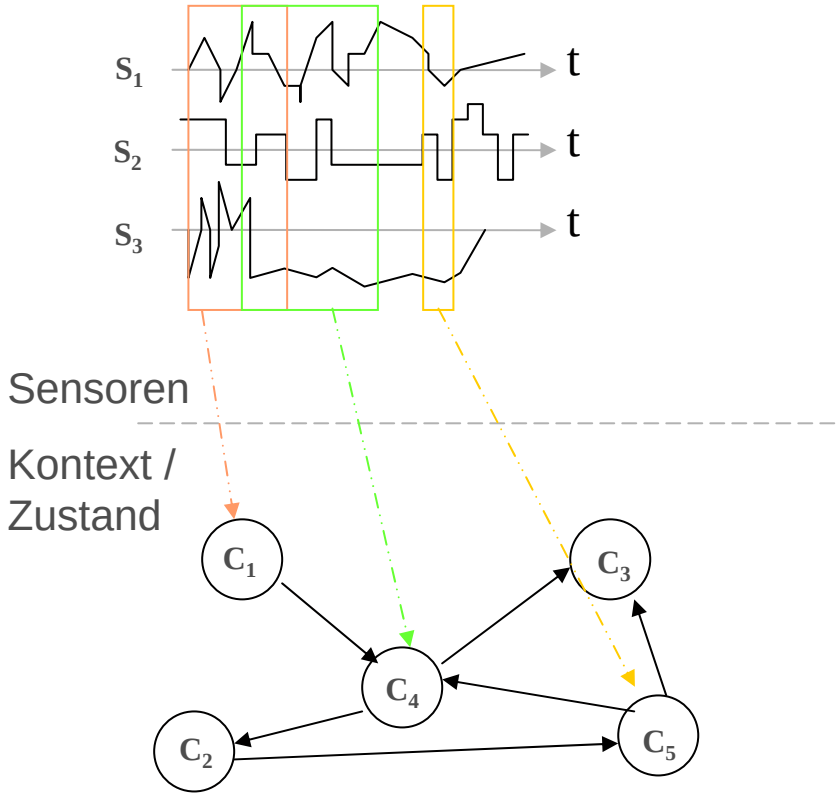
Ansatz zur Kontextvorhersage

- Erkennen des aktuellen Kontext aus mehreren Sensoren (z.B. [Schmidt 2002])
- Vorhersage zukünftiger Kontexte durch Lernen von Benutzerverhalten
- Möglichst **ohne** Verwendung von Domänen-spezifischem Wissen



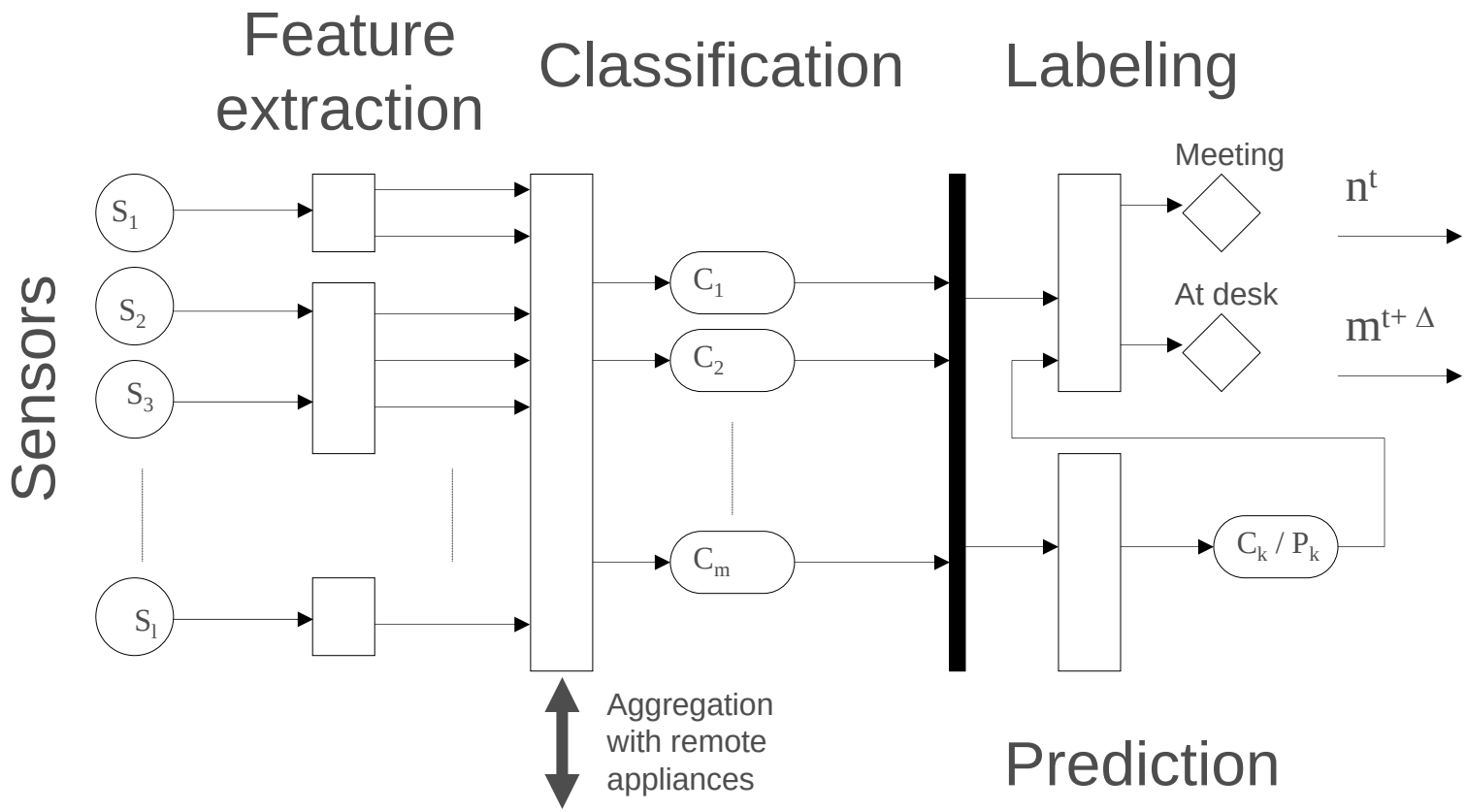
[Schmidt 2002] A. Schmidt: "Ubiquitous Computing – Computing in Context", PhD Thesis, Lancaster Univ.

Konzept



- Sensoren liefern Zeitreihen
- Spezielle Muster in Eingabeströmen können als Zustände einer (teilweise beobachtbaren, aber nicht steuerbaren) Zustandsmaschine interpretiert werden
- Zustände werden als Benutzer- bzw. Gerät-Kontexte aufgefasst
- Vorhersage der Zustandstrajektorie durch Extrapolation der Historie in die Zukunft

Architektur



$<s_1, s_2, ..., s_l>_t \rightarrow <f_1, f_2, ..., f_n>_t \rightarrow <c_1, c_2, ..., c_m>_t \rightarrow <c_1, c_2, ..., c_m>_{t+\Delta}$

Input vector (Sensor vector) — Feature vector — Class vector — Future class vector

[MRF 2003a] R. Mayrhofer, H. Radi, A. Ferscha: „Recognizing and Predicting Context by Learning From User Behavior“, Proceedings of MoMM2003, OCG, September 2003

Schritt 1: *Sensors*

- Sensoren liefern Zeitreihen
- Entweder zu fixen Zeitpunkten abgetastet oder Ereignisse
- Beispiele für aktuell verfügbare „Sensoren“ zur Erfassung des aktuellen Benutzerkontext:
  - Zeit
  - Anwendung
  - Helligkeit
  - Mikrofon
  - Bluetooth
  - WLAN
  - In Dockingstation
- Zusätzliche Sensoren können abgefragt werden:
  - GPS
  - GSM
  - Kompass
  - Beschleunigungssensoren
  - Neigungssensoren
  - Temperatursensoren
  - Drucksensoren

- Weitergabe von Sensordaten an andere Geräte möglich

Schritt 2: *Feature Extraction*

- Wandelt rohe Sensordaten in aussagekräftigere Werte (*Merkmale*)
- Verwendet Wissen über die Anwendungsdomäne
- Aus einem Sensordatenstrom können mehrere Features gewonnen werden

⇒ **Hochdimensionaler Feature-Raum**

- Unterschiedliche Typen von Features:
  - Numerische (kontinuierliche): z.B. Helligkeitssensor, Pulssensor
  - Numerische (diskrete): z.B. Anzahl von Access Points in Reichweite
  - Ordinale: z.B. Wochentag
  - Nominale: z.B. aktuelle WLAN-SSID im Infrastruktur-Modus, Liste von über WLAN/BT erreichbaren Geräten in unmittelbarer Umgebung

- Aber: nur 2 Operationen für jede Feature-Dimension benötigt
  - Distanzmetrik
  - Adaptionsoperator

[MRF 2003b] R. Mayrhofer, H. Radi, A. Ferscha: „Feature Extraction in Wireless Personal and Local Area Networks“, Proceedings of 6th IFIP MWCN 2003, World Scientific, October 2003

Schritt 3: *Classification*

- Klassifiziert Features und findet häufig vorkommende Muster (sog. Cluster) in den Eingabedaten ⇒ ohne Benutzerinteraktion möglich, unaufdringlich
- Verschiedene Arten von Klassifikationsalgorithmen
  - Typ (partitionierend / hierarchisch)
  - „Weiche“ / „harte“ Klassifikation
  - Überwacht / unüberwacht
- Voraussetzungen für geeignete Algorithmen zur Kontexterkennung:
  - Online bzw. inkrementell
  - Adaptiv
  - Dynamische Anzahl von Klassen und dynamische Struktur
  - Finden von Clustern in Unterräumen
  - „Weiche“ Klassifikation
  - Robustheit gegen Rauschen
  - Geringer Ressourcenbedarf
  - Einfachheit
  - Interpretierbarkeit der Klassen / Wahrung des Datenschutzes

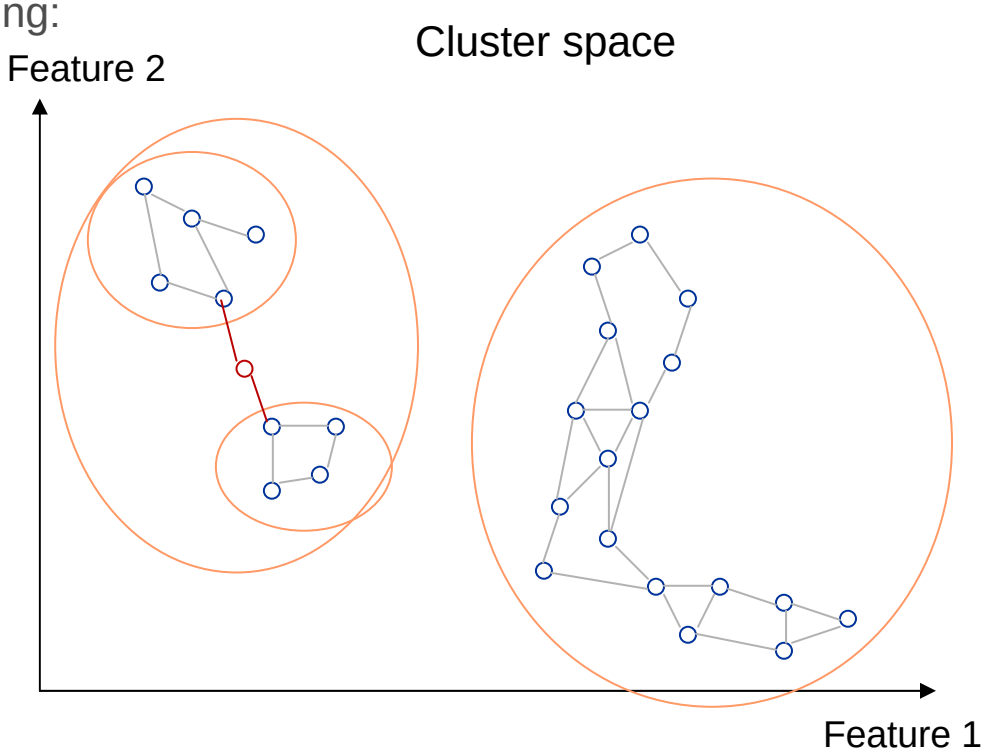
Klassifikationsalgorithmen

Algorithmus	Topologie / Anzahl Klassen	Online / inkrementell	Adaptiv	Hart / Weich
K-Means	fix	ja	ja	hart
FCM	fix	ja	nein	weich
Neural Gas	fix	ja	nein	teilweise
SOM	fix	nein	nein	weich
ART	variabel	ja	nein	weich
IDBSCAN	variabel	inkrementell, aber nicht online	nein	weich
SNN	fix oder variabel	ja	nein	weich
Growing Neural Gas [Fri 2003]	variabel	ja	ja	weich

[Fri 2003] B. Fritzke: „A Growing Neural Gas Network learns Topologies“, Advances in Neural Information Processing Systems (7), MIT Press

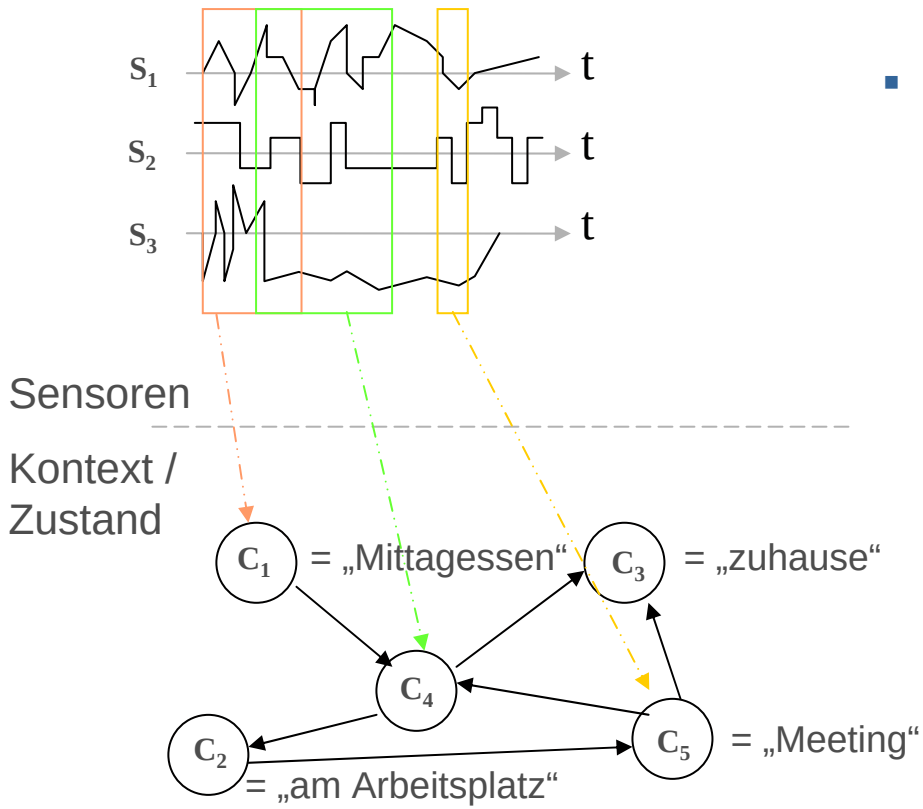
Variation: Lifelong Growing Neural Gas (LLGNG)

- Modifikation von LLGNG für lebenslanges Lernen [Ham 2001]
- Unterschied: verwendet lokale Kriterien für jeden Cluster, um unbeschränktes Einfügen neuer Cluster zu verhindern
- Erweiterungen von LLGNG zur besseren Kontexterkennung:
  - Verwendung eines heterogenen Eingaberaumes
  - Ausnutzung der internen Topologie, um Cluster mit beliebigen Formen zu erlauben
    ⇒ Meta-Cluster als zusätzliche Abstraktion



[Ham 2001] F.H. Hamker: „Life-long Learning Cell Structures – continuously learning without Catastrophic Interference", Neural Networks (14)

Schritt 4: *Labeling*



- 1:{0,1} Zuordnung von (Meta-) Clustern zu Kontext-Namen
- Derzeit 2 Möglichkeiten:
  - Bei stabilen (Meta-) Clustern von Schritt 1: direkte Zuordnung (Meta-) Cluster-ID zu Name
  - Bei durch Adaption/Lernen zu stark veränderlichen (Meta-) Clustern: zusätzlicher, sehr einfacher Clustering-Schritt über Clusterdaten [Lae 2001]

[Lae 2001] K. van Laerhoven and S. Lowette: “Real-time analysis of data from many sensors with neural networks”, Proceedings of ISWC, IEEE Press, October 2001

Schritt 5: *Prediction*

- Erkannte Kontexte können als „Zustände" einer Zustandsmaschine verstanden werden
- Beobachtung der entstehenden Zustandstrajektorie erlaubt Vorhersage
- Aspekte von Zeitreihenvorhersage:
  - Periodische Muster (z.B. Wochenenden, Meetings)
  - Sequentielle Muster (z.B. Reisevorbereitungen, Zubereitung von Speisen)
  - Trends (sich ändernde Gewohnheiten)

- Voraussetzungen für geeignete Algorithmen zur Kontextvorhersage:
  - Unüberwachte Modellbildung
  - Online
  - Inkrementelle Modellerweiterung
  - Konfidenzabschätzung
  - Automatisches (implizites) Feedback
  - Manuelles (explizites) Feedback
  - Langzeit- vs. Kurzzeitvorhersage

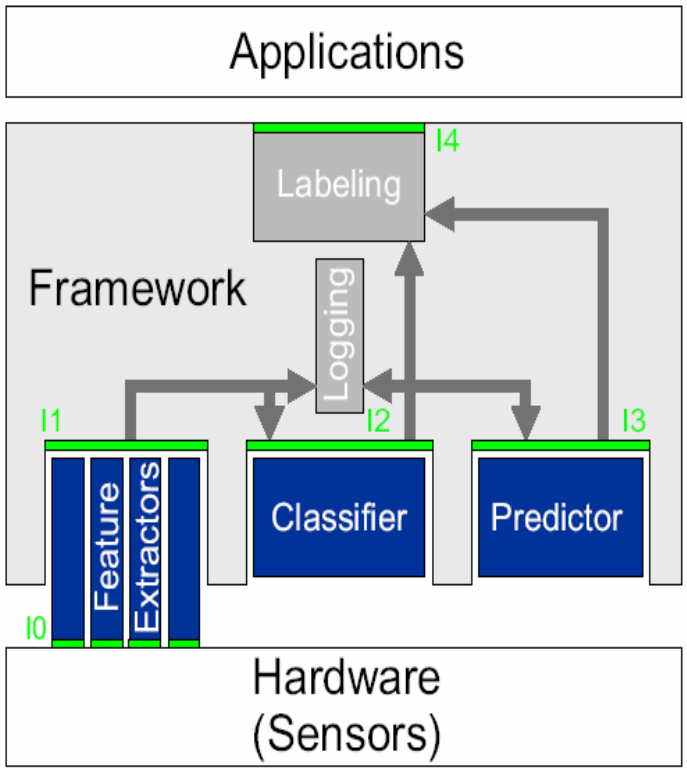
Möglichkeiten zur Kontextvorhersage

- Basierend auf Trajektorie von Kontexten, d.h. Kontext-Klassen
- Vorteil gegenüber unabhängiger Vorhersage von Feature-Dimensionen: gemeinsame Betrachtung aller Aspekte von Kontext
- Prinzipiell 2 Optionen:
  - Jede Dimension des Kontext-Klassenvektors getrennt als kontinuierliche Zeitreihe
    ⇒ Abhängigkeiten zwischen verschiedenen Kontexten nicht berücksichtigt, aber erlaubt beliebige Überlappungen
  - Aggregation aller Kontexte in eine einzelne, kategorische Zeitreihe
    AAACCBCCAAADDDDDDDEEEEEEBBAAAA……..
    ⇒ berücksichtigt explizit die Zusammenhänge zwischen verschiedenen Kontexten, da diese als sich gegenseitig ausschließend betrachtet werden

⇒ Flaches vs. hierarchisches/überlappendes Kontextmodell

- Viele bekannte Methoden zur Zeitreihenvorhersage, aus denen für spezielle Anwendungen ausgewählt werden kann

Implementierung als Software Framework

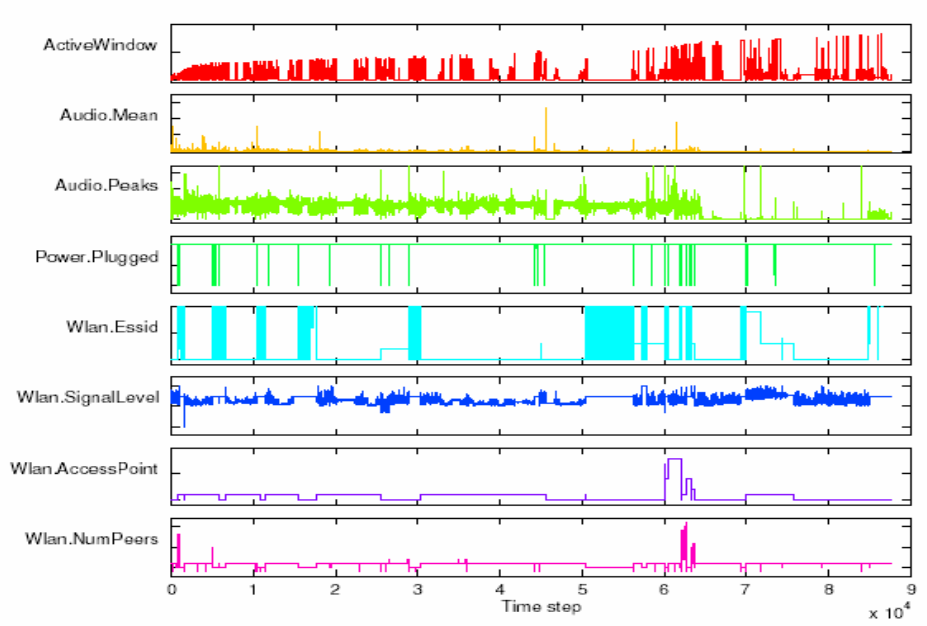


Cross-Plattform:

- Derzeit für Win32, Windows CE (>=3.0), Linux IA32 und ARM sowie Symbian OS
- Basierend auf dynamisch ladbaren Modulen:
  - Feature Extraktoren (= Sensordatenerfassung + Feature Extraktion)
  - Klassifikationsmethoden
  - Vorhersagemethoden
- Benennung von Kontexten über externe Applikation über netzwerktransparente Protokolle
  $\Rightarrow$ Trennung von HCI und automatischer Kontexterkennung
- Speziell im Hinblick auf Geräte mit geringen Ressourcen entwickelt

Evaluierung mit Realdaten

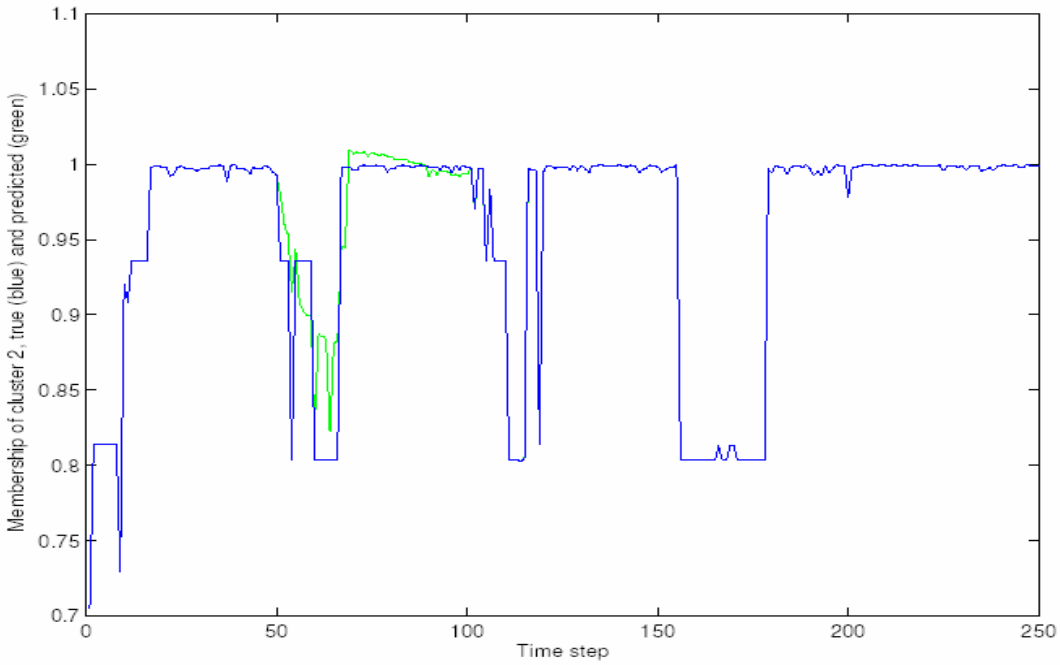
- Aufgezeichnete Echtdaten
- Aufzeichnung vollständig im Hintergrund ⇒ Unaufdringlichkeit gewahrt
- Zeitraum: ca. 2 Monate
- 28 Dimensionen im Feature-Raum
- ~ 90000 Datenpunkte



Klassifikation:

- Evaluierung von K-Means, SOM und LLGNG
  - K-Means: kleinster Fehler mit 6 Clustern: 0,7451
  - SOM: kleinster Fehler mit 71x21 (=1491) Neuronen in Ausgabeschicht: 0,5659
  - Erweiterter LLGNG: 9 Meta-Cluster aus 109 Clustern: 0,0069
- LLGNG erzielt kleineren Fehler mit weniger Clustern als SOM
- Erweiterte Variante bietet zusätzlich Konzept der Meta-Cluster

Evaluierung mit Realdaten (2)



Algorithmus	Fehler
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

Vorhersage:

- Derzeit beste Ergebnisse mit ARMA und saisonaler Korrektur
- HMM, SVM etc. mit diesen Daten weniger geeignet

Wissenschaftlicher Beitrag

Entwicklung einer Architektur zur Kontextvorhersage:

- Vorhersage von Kontexten auf hoher Ebene anstelle einzelner Aspekte durch Interpretation von Kontexten als Zustände und Beobachtung der daraus entstehenden Trajektorie
- Flexible Einsatzmöglichkeiten durch möglichst einfache, klar definierte Schnittstellen zwischen Einzelschritten
- Klassifikation, Benennung und Vorhersage explizit als eigenständige, austauschbare Teile
- Umgang mit heterogenen Eingabedaten

Kriterien / Methoden zur Klassifikation und Vorhersage

Implementierung der Architektur in Form eines offenen Software-Frameworks:

- Direkte Verwendung verschiedenster Sensortypen ohne interne Kodierung von nicht-numerischen Sensordaten
- Erweiterung von LLGNG um das Konzept der Meta-Cluster sowie Optimierungen zur Verbesserung der Laufzeit
- Unterstützung verschiedener Plattformen und Entwicklung im Hinblick auf Geräte mit geringen Ressourcen

“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

Danke für Ihre Aufmerksamkeit!