Smart-Home Steuerung mit einem hausinternen Sprachassistenzsystem

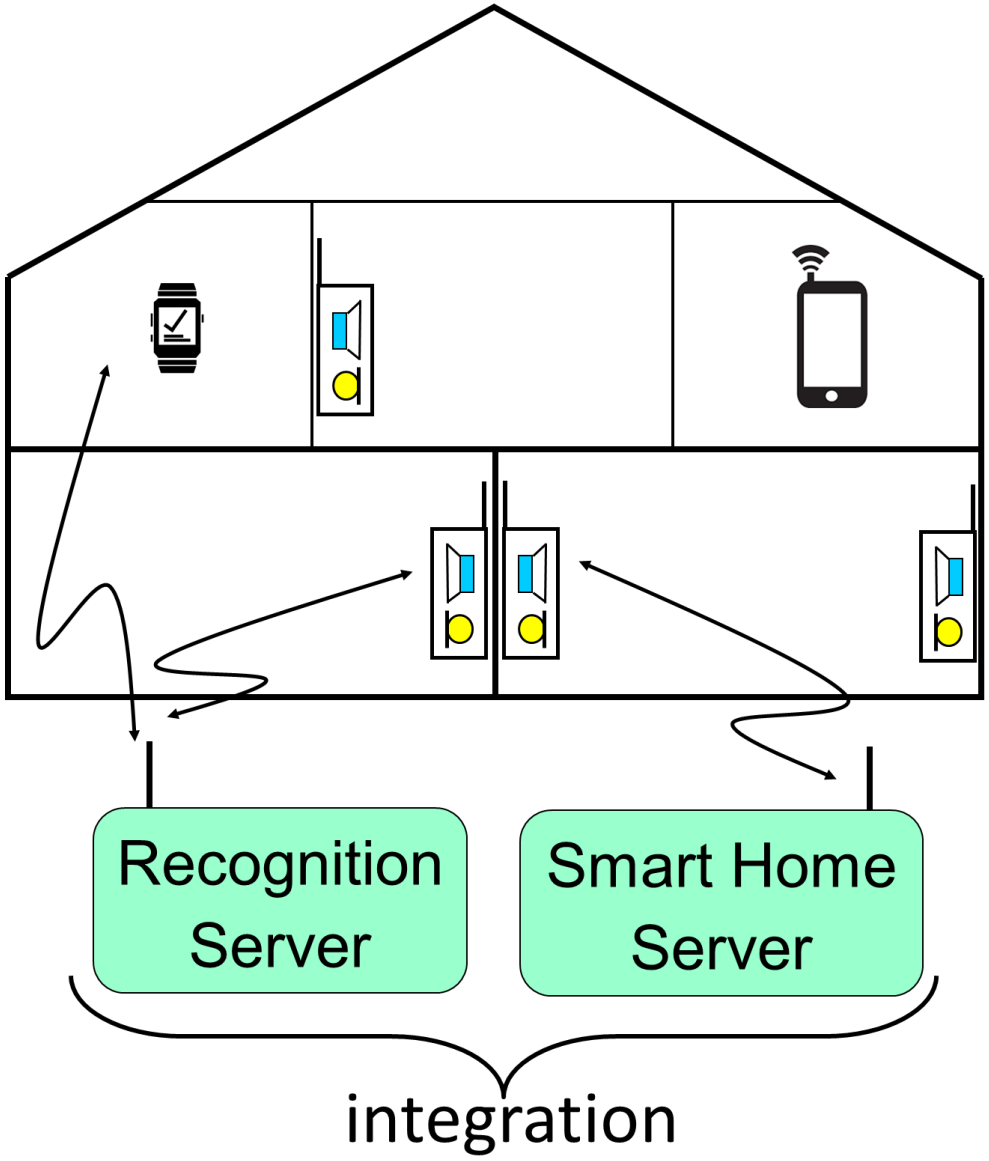
Hans-Günter Hirsch, Alexander Micheel, Jan Stähler, Michael Gref, Janik Göbel
hans-guenter.hirsch@hs-niederrhein.de

iPattern - Institut für Mustererkennung, Hochschule Niederrhein
März 2020

- Projekt zur Smart-Home Sprachsteuerung
- Clients zur Spracherfassung/-wiedergabe und Dialoggestaltung
- Erkennungs-Server
- Dialoggestaltung

Hochschule Niederrhein
University of Applied Sciences
iPattern
Institut für Mustererkennung
Institute for Pattern Recognition

Smart-Home Sprachsteuerung



Ziele des Projekts:

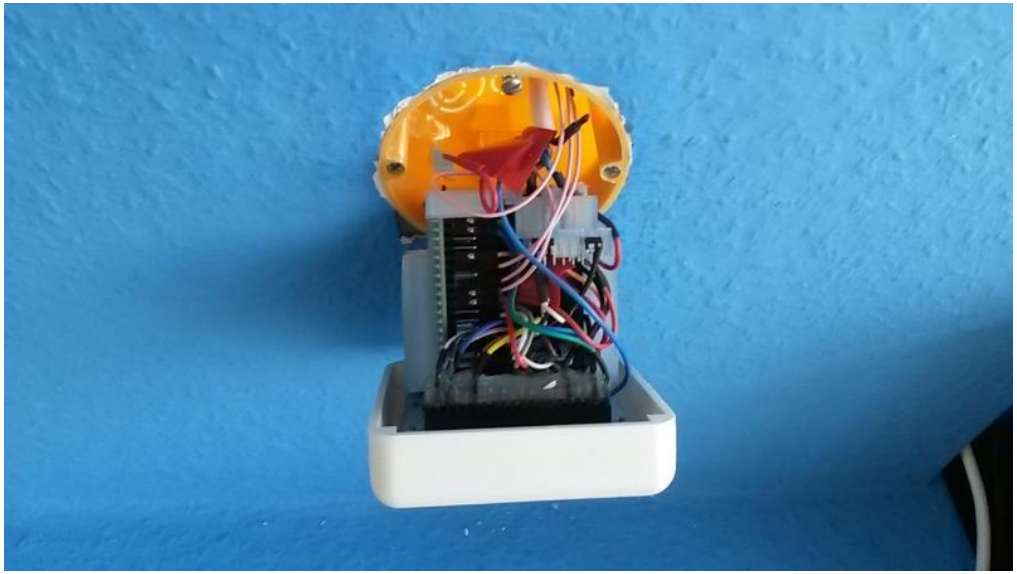
- Entwicklung kleiner kompakter kostengünstiger Einheiten (Clients) zur Spracheingabe/-erkennung
- Verwendung eines hausinternen Erkennungs-Servers (data privacy)
- Erste Untersuchung zur parallelen Verwendung mehrerer Clients in einem Raum (Problematik eines Sensornetzwerks)

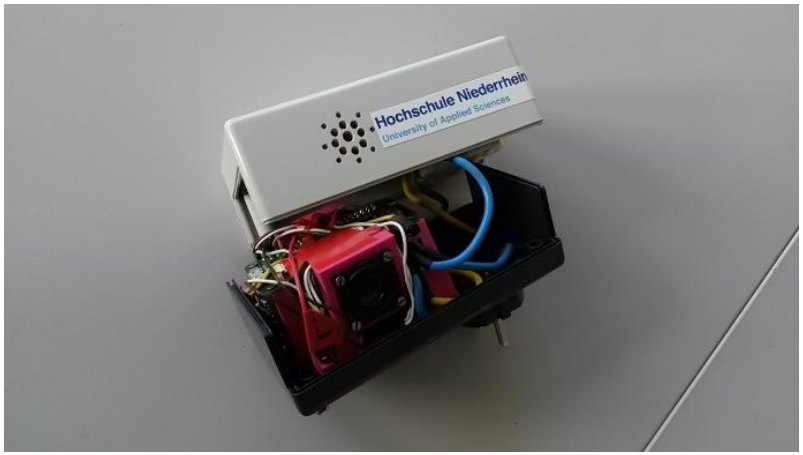
Anforderungen an Client

- klein und kompakt, z.B. Integration in Schalterdose
- geringer Energiebedarf ➔ begrenzte Rechenleistung
- Kontrolle des Dialogs mit dem Benutzer ➔ keine Notwendigkeit eines zusätzlichen Dialog-Servers, Möglichkeit einer raumabhängigen Dialoggestaltung
- Direktes Aussenden von Steuerbefehlen an Smart-Home Server (z.B. Homematic) oder an zu steuernde Geräte
- Schlüsselworterkennung oder/und andere Sensorik zur Aktivierung
- kostengünstig

Clients

Integrierte Devices

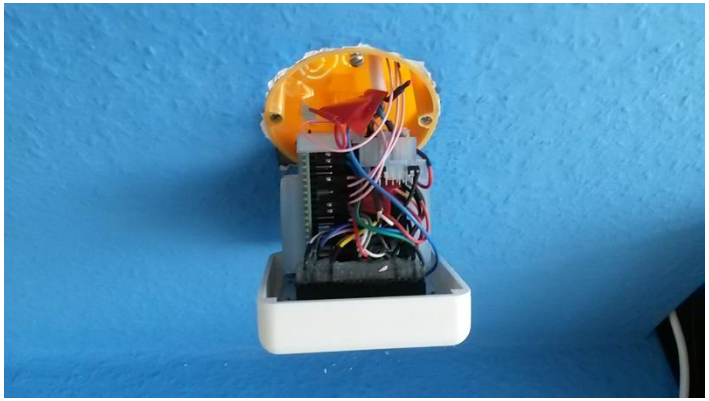


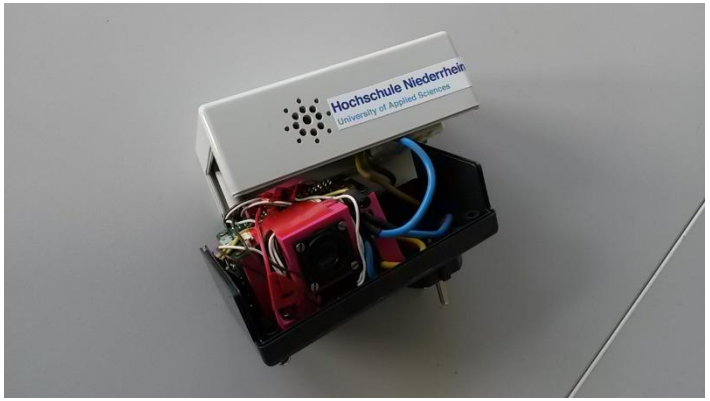


Clients

Hardware der integrierten Devices

- Pi-Zero
- Mikrofon-Boards mit 2 oder 4 Mikrofonen (Seeed Studio ReSpeaker)
- Endverstärker & Lautsprecher
- Netzteil und 220V Schaltrelais



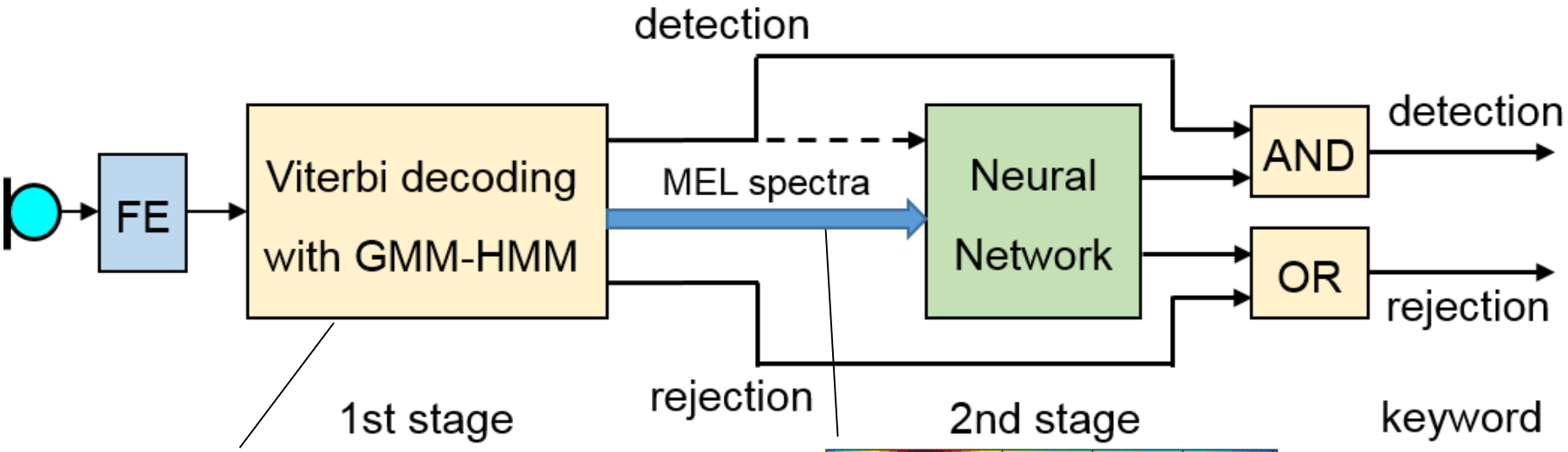


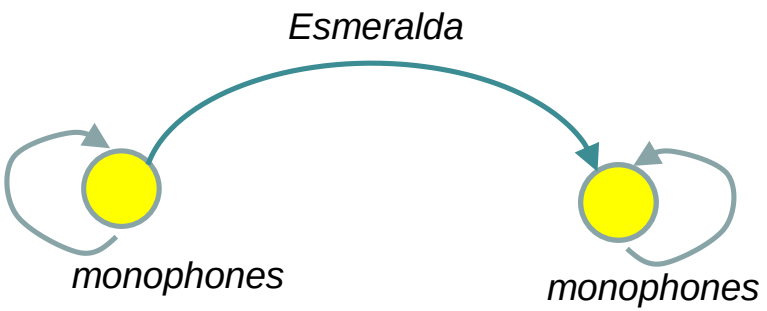
Software

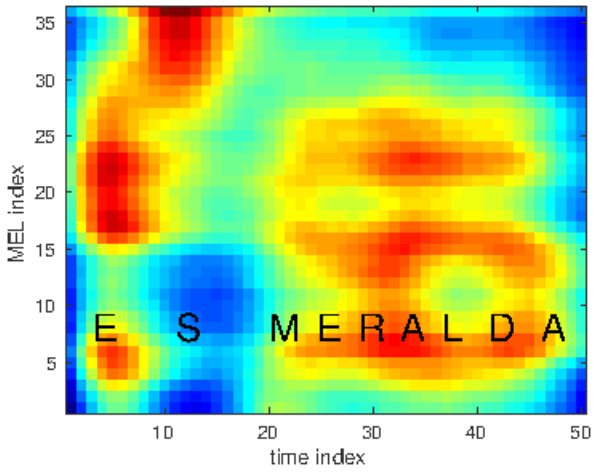
- Dialog Controller in Form einer Finite-State-Machine
- Schlüsselwort-Erkennung

Schlüsselwort „Esmeralda“ Erkennung

Zweistufiger Ansatz mit 1. GMM-HMM Erkennung des Schlüsselworts (inkl. garbage Modellen) und 2. Neuronales Netz zur Verifikation des MEL Spektrums







Anforderungen an Erkennungs-Server

- Einhaltung der Privatsphäre ➔ Sprachsignal verlässt das Haus nicht!
- kostengünstig ➔ beschränkte Rechenressourcen

Raspberry Pi mit GMM-HMM

Nvidia Jetson mit DNN-HMM

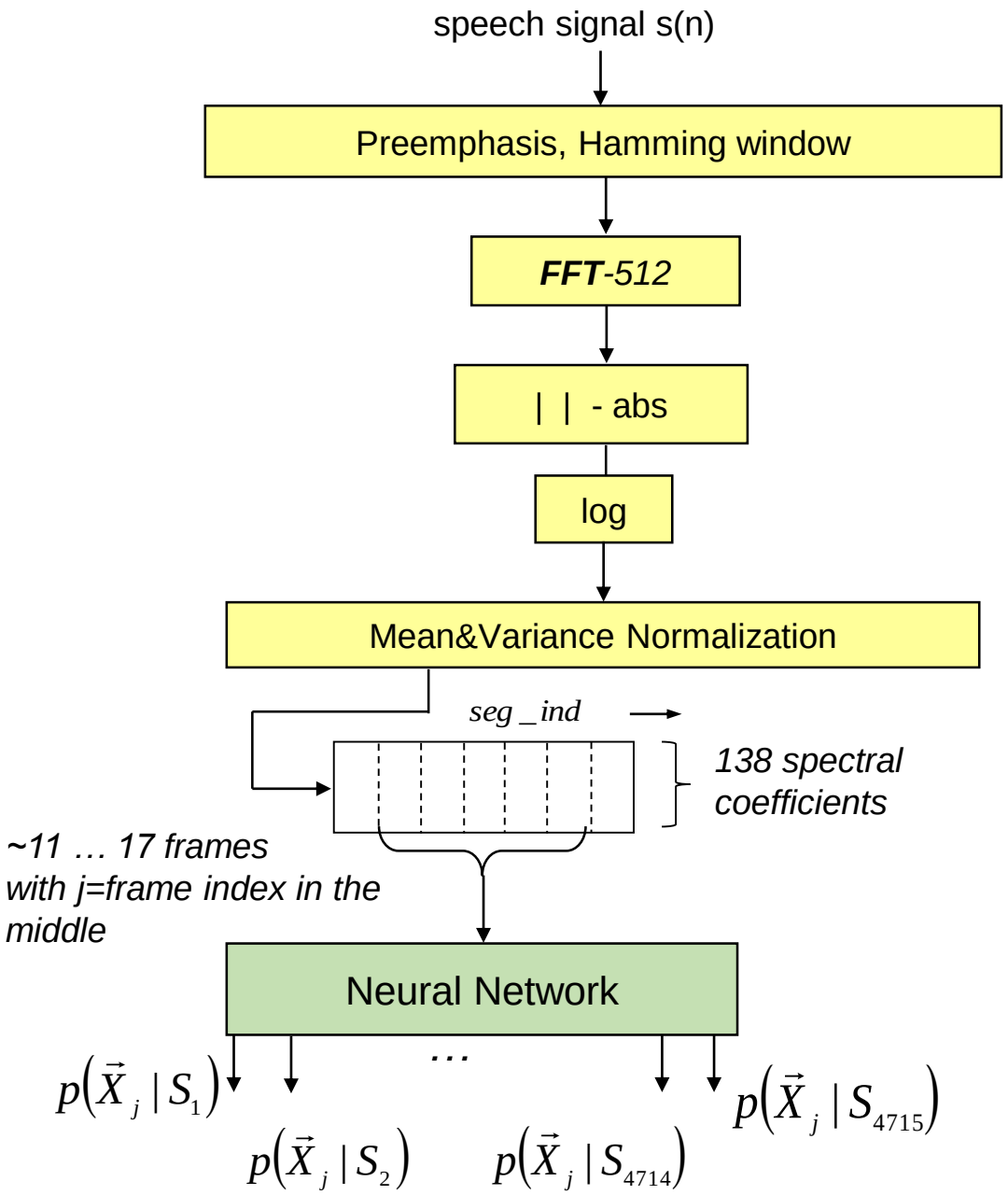
Dell Optiplex 3060

Kaldi Erkenner

GMM-HMM auf PI

- Sprachsignal (8/16 kHz) gelangt per IP vom Client zum Server
- Extraktion der MFCCs, Deltas und Delta-Deltas (39 akustische Merkmale)
- Viterbi Erkenner mit HMMs (wort- oder triphonbasiert)
- Triphon HMMs werden mit den Werkzeugen von HTK mit Hilfe von mehreren Hundert Stunden Sprache trainiert, z.B. RVG, open source TU Darmstadt, …)
- Phonolex Lexikon
- Verwendung der Grammatik, die der Client im entsprechenden Dialogzustand sendet
- Parallele Prozesse zur Sprachanalyse, Erkennung, …
- Optional ist Pruning durch die Festlegung der Anzahl von HMM Zuständen mit höchster Wahrscheinlichkeit möglich.

DNN-HMM auf NVidia Jetson



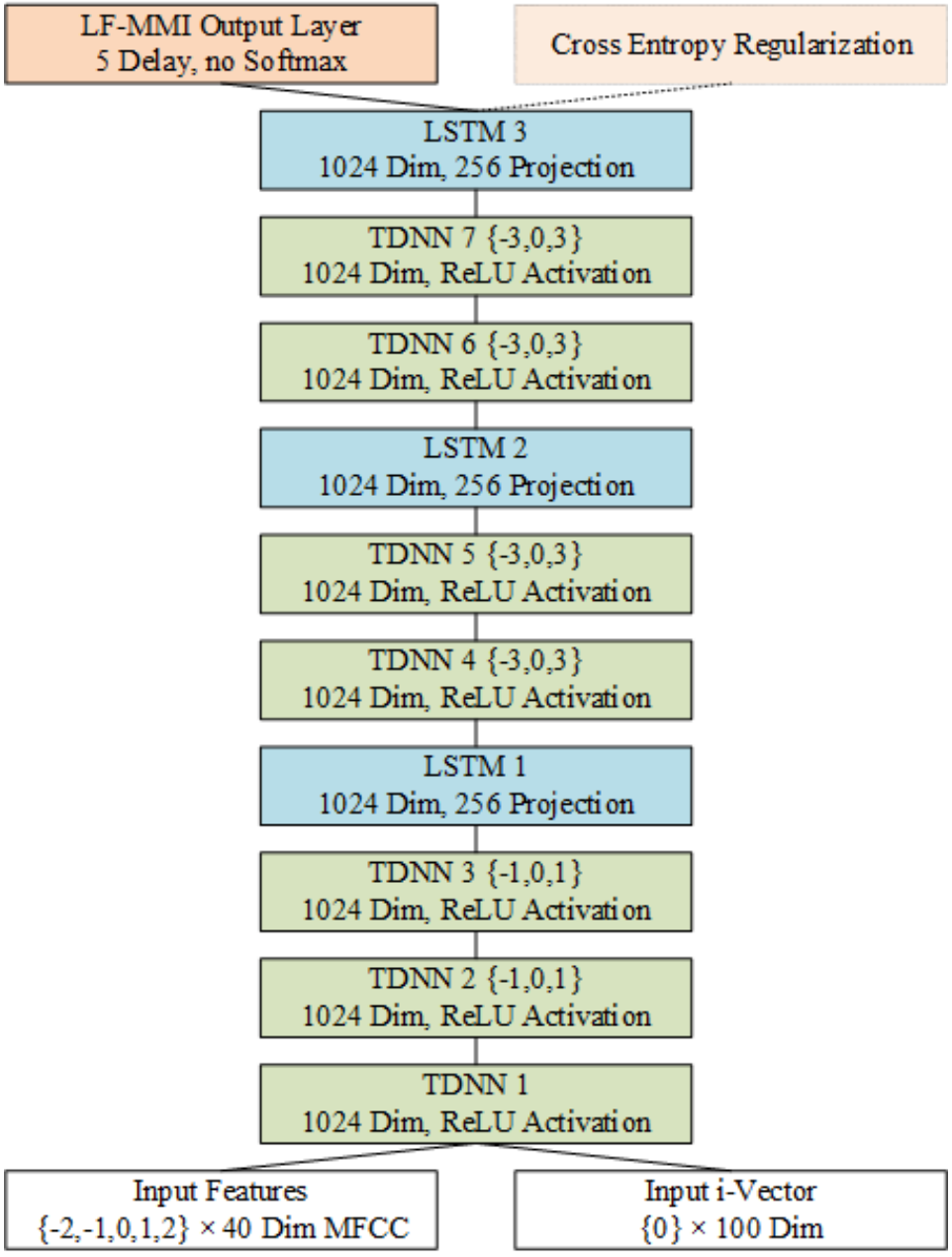
Ein neuronales Netz wird anstelle des GMM zur Bestimmung der Emissionswahrscheinlichkeiten eingesetzt. Eingangswerte des Netzes sind Folgen mit ~11 Spektren. Jedes Spektrum beinhaltet 138 Koeffizienten, die aus 257 DFT Koeffizienten bestimmt werden.

Eingang: 11 mal 138 = 1518 Werte

Am Ausgang werden die Emissionswahrscheinlichkeiten aller „tied“ Triphon Zustände (4715 Werte) bestimmt.

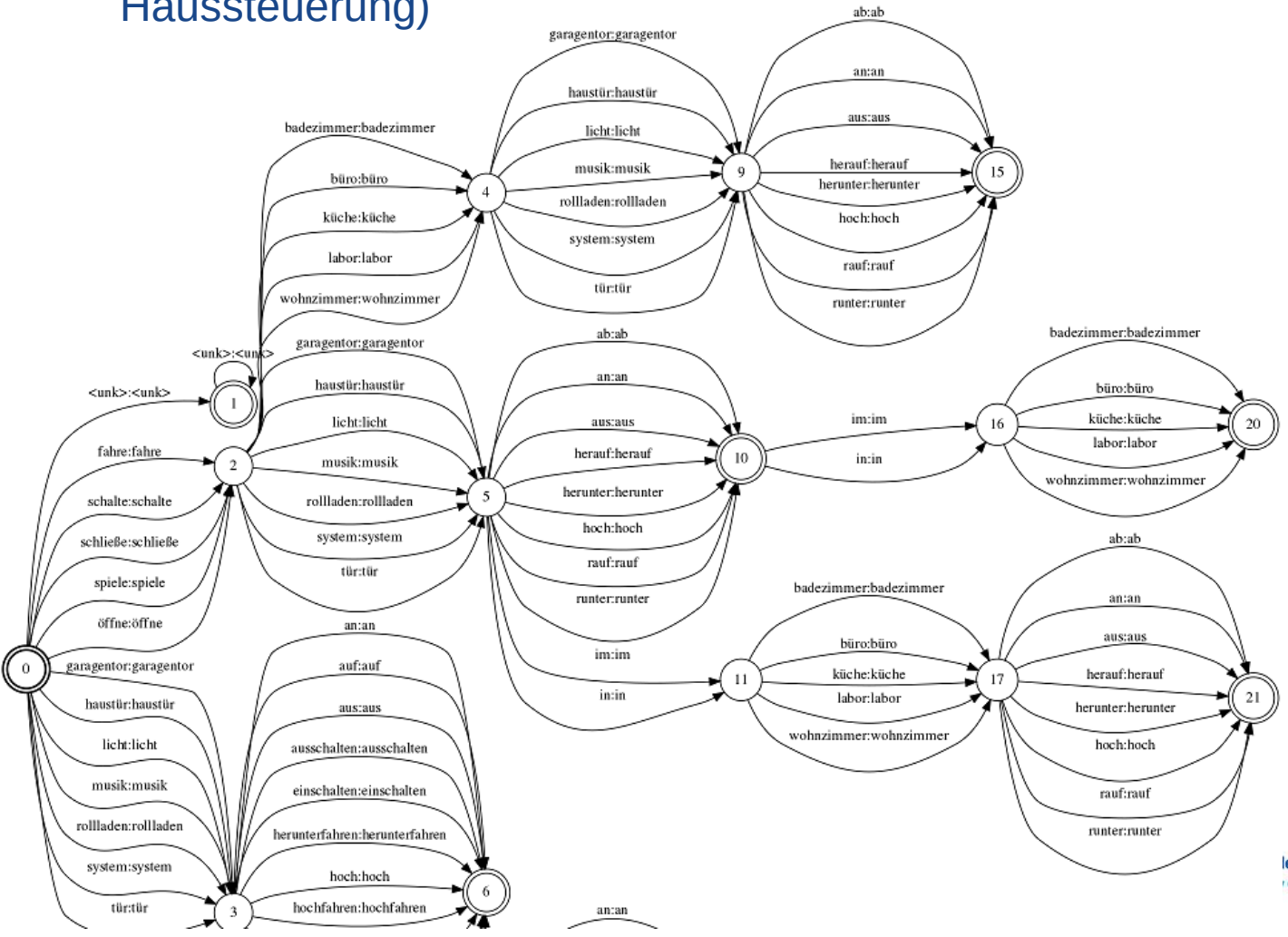
Kaldi basierter Erkenner

- Auf dem Framework GStreamer basierende Echtzeit Version des Kaldi Toolkits (https://github.com/alumae/kaldi-gstreamer-server)
- Echtzeitfähige Netzwerktopologie mit LSTMs und TDNNs
- Input: 5 Frames mit 40 MFCCs & iVector
- Lattice-Free Maximum Mutual Information („Chain“) Modell
- Training mit mehreren Hundert Stunden deutscher Sprache(z.B. RVG, TuDa, Podcasts Forschergeist, M-Ailabs, …) inkl. data augmentation (speed perturbation, Störgeräusche, Hall)



Kaldi basierter Erkenner

- Language Model, trainiert mit allen deutschen Wikipedia Artikeln (~2 Mill. Lexikoneinträge)
- Feste Grammatik zur Erkennung von Kommandosequenzen (z.B. zur Haussteuerung)



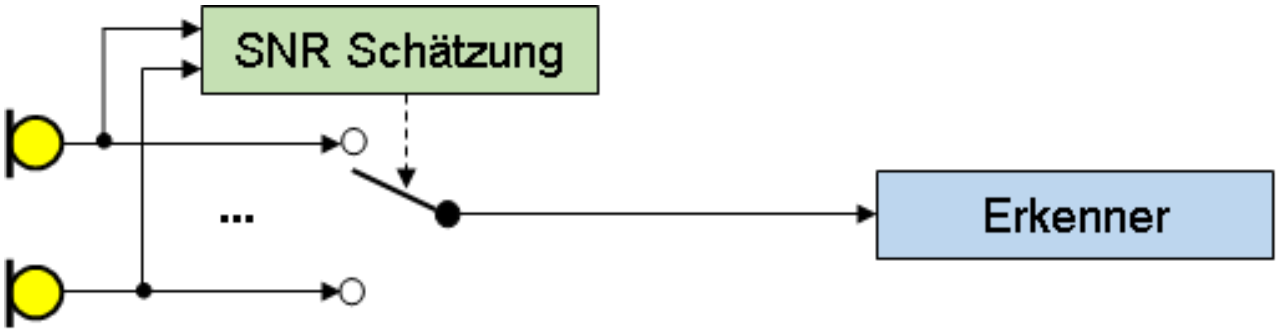
Dialoggestaltung

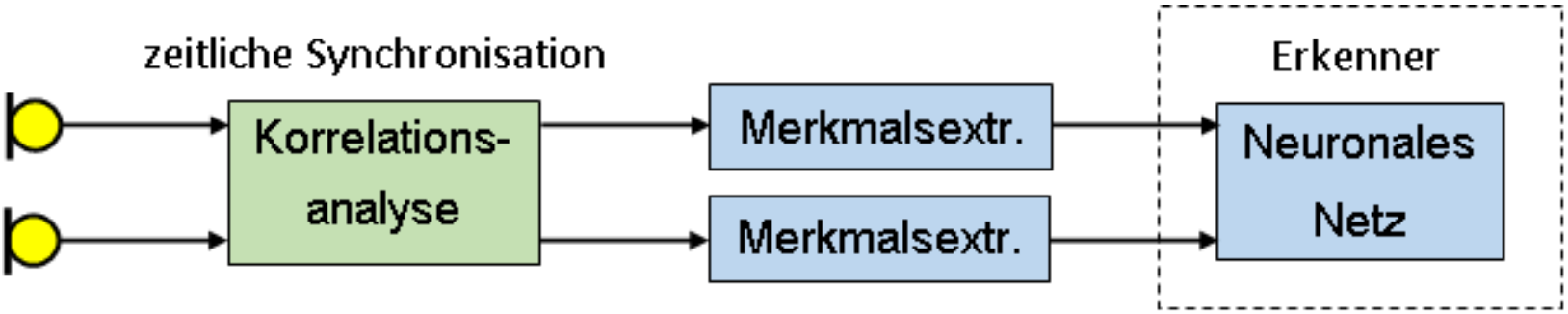
- Softwaremodul auf dem Client zur Dialoggestaltung
- Textbeschreibung des Dialogs als finite state Modell
- Zur Erkennung wird in Abhängigkeit des Dialogzustands eine zustandsabhängige Grammatik zum Server übertragen
- Sprachsignal wird per IP zum Server übertragen
- Erkennungsergebnis erhält der Client vom Server in Textform
- Sprachwiedergabe von Files auf dem Client

- Dialogsoftware kann flexibel um benötigte Schnittstellen zur Steuerung erweitert werden, z.B. Protokoll zur Bedienung eines Homematic Servers, Modbus, CAN, …
- Einfache GUIs können für Clients mit Display generiert werden.

Mehrere Clients in einem Raum

- Erste Ansätze zur Verarbeitung mehrerer in einem Raum erfasster Mikrofonsignale
- Signale werden mit NTP Zeitstempel zur groben zeitlichen Synchronisation versehen





Demo

- Integration von Client und Server auf einem PI
- Lineares Array mit 4 Mikrofonen

Zusammenfassung

- Vorstellung eines Client-Server basierten Sprachassistenzsystems
- Client übernimmt die Gestaltung und Kontrolle des Dialogs sowie die direkte Ansteuerung externer Geräte
- Server ist als kostengünstiges System ausgelegt, so dass die Sprache nicht nach „außen“ transportiert werden muss und die Privatsphäre gewahrt bleibt.