Voice & Auditory Interaction

Janne Pylkkönen janne@speechly.com
Ari Nykänen ari@speechly.com

Feb 22nd 2020

Speechly

About us

Speechly develops a real-time voice interface API for web and mobile.

- A venture funded start-up founded in 2016
- Currently 11 employees based in Helsinki (tech) and Chicago (sales).
- Customers mostly based in the US
- Backed e.g. by Cherry Ventures, TQ Ventures, and (of course!) Business Finland

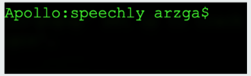
	Input methods	Output methods
VISION	Text	Text
	GUI	GUI
AUDITION	Speech	Speech

Input methods

Output methods

VISION

Input methods	Output methods
Text	Text
GUI	GUI



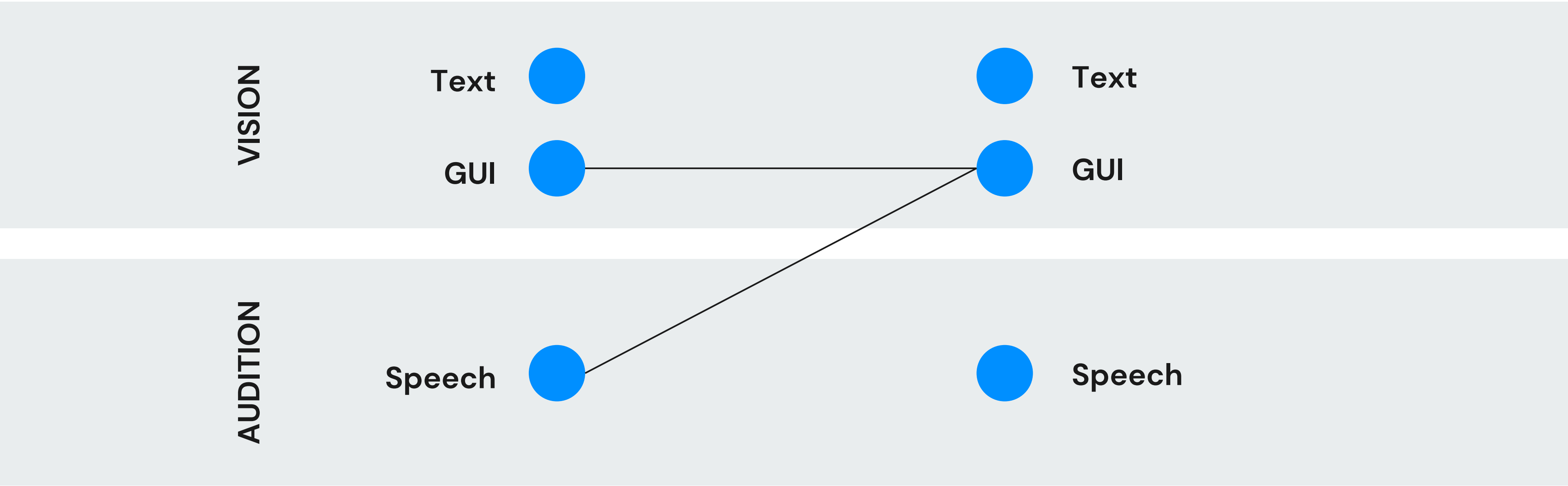
AUDITION

Input methods	Output methods
Speech	Speech

Speechly

Input methods

Output methods



Speechly

Speechly’s approach

- User speech is processed in real-time using automatic speech recognition (ASR) and streaming natural language understanding (NLU)
- Speech input (words, intents and entities from the NLU) can be used to control any aspect of an app alongside GUI input
- Any output method can be used
    - GUI (or a VR/AR UI) is assumed to be the default option and there are ready-made GUI input components for toggling listening on/off and GUI showing real-time transcript of user speech.
- Using speech for input and GUI for output enables uninterrupted input while providing real-time feedback to the user.

Working Memory: Limitations

- WM is composed of 2 ”systems" that maintain and process information:

VISUAL-SPATIAL TASKS
MOTOR TASKS

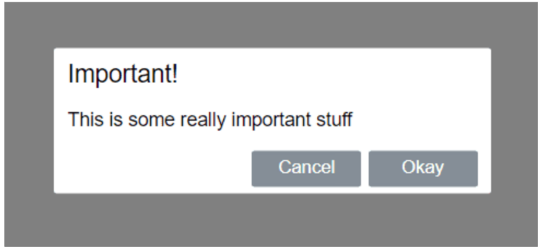
PHONOLOGICAL TASKS
(LANGUAGE IN ALL FORMATS)

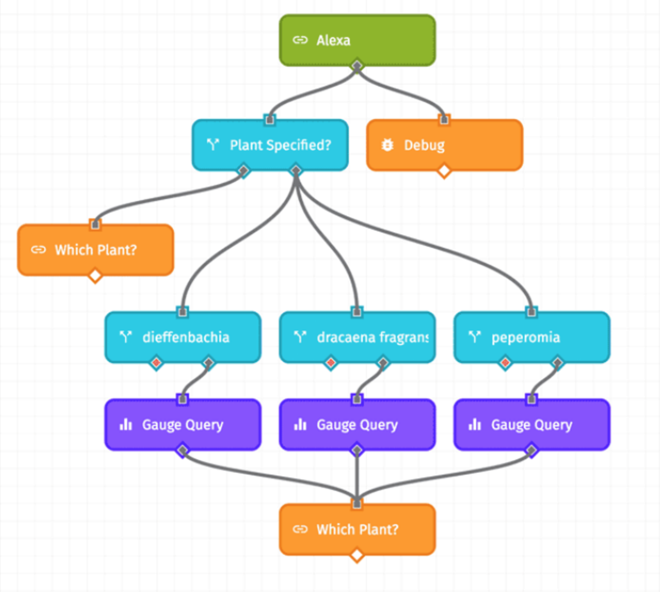
Speechly

Celia Hodent: The Gamer’s Brain: How Neuroscience and UX Can Impact Video Game Design (2017).

Direct speech input vs conversational voice UI

- Direct speech input can be contrasted against conversational designs in two ways:
  - Traditionally also the output channel is speech, so the user and the system take turns listening and speaking
  - In conversational designs the input is more often modal; available input options change depending on the input system state and this needs to be communicated to the user.

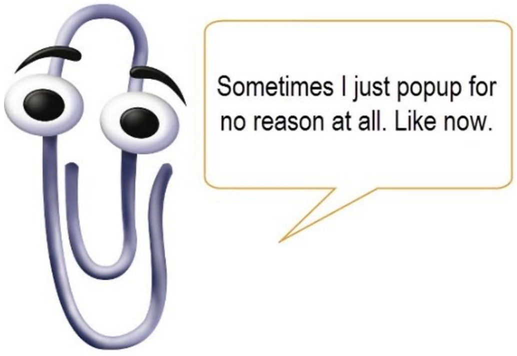




Conversational voice experience for checking moisture of an office plant

Feedback without AI personas

- When using direct speech input the use of an AI persona becomes optional or may even distract the user.
- Yet the app still needs to cope with information entered in fragments and ensure that the user can tell the system state.
- For additional feedback, some kind of simple hint or notification system usually suffices.



Speechly

User jobs well-suited for speech input

- Selection from a large set of known options
  - Repeated tasks like adding groceries to a shopping list/cart
  - Issuing commands in a professional application with 100s of options
- Keyboardless use
  - Enhancing mobile UIs
  - VR
- **Thoughts?**

Challenging jobs for speech input

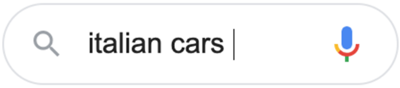
- Arbitrary names of people and places (ASR challenge)
- Dealing with strong accents (ASR challenge)
- Distinguishing between very similar expressions (ASR / NLU challenge)
- **Any experiences?**

Recovering from misunderstandings

- Eventually the speech input engine will make a mistake
- The key for a good voice UX is how **quickly** you can detect a mistake and how **easily** you can recover from that
- To allow user to detect the mistake early, you can…
  - Display the speech-to-text transcript in real-time
  - Show and highlight any changes in the app state in real-time
- To recover from the mistakes you can enable…
  - Repeating the incorrect information
  - Clearing/undoing the incorrectly interpreted input
  - Providing information about supported phrases

Reach of manipulation (input)

Widget

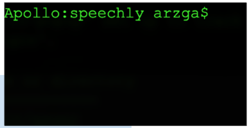


Widget group

DIRECT VISIBILITY OF EFFECT

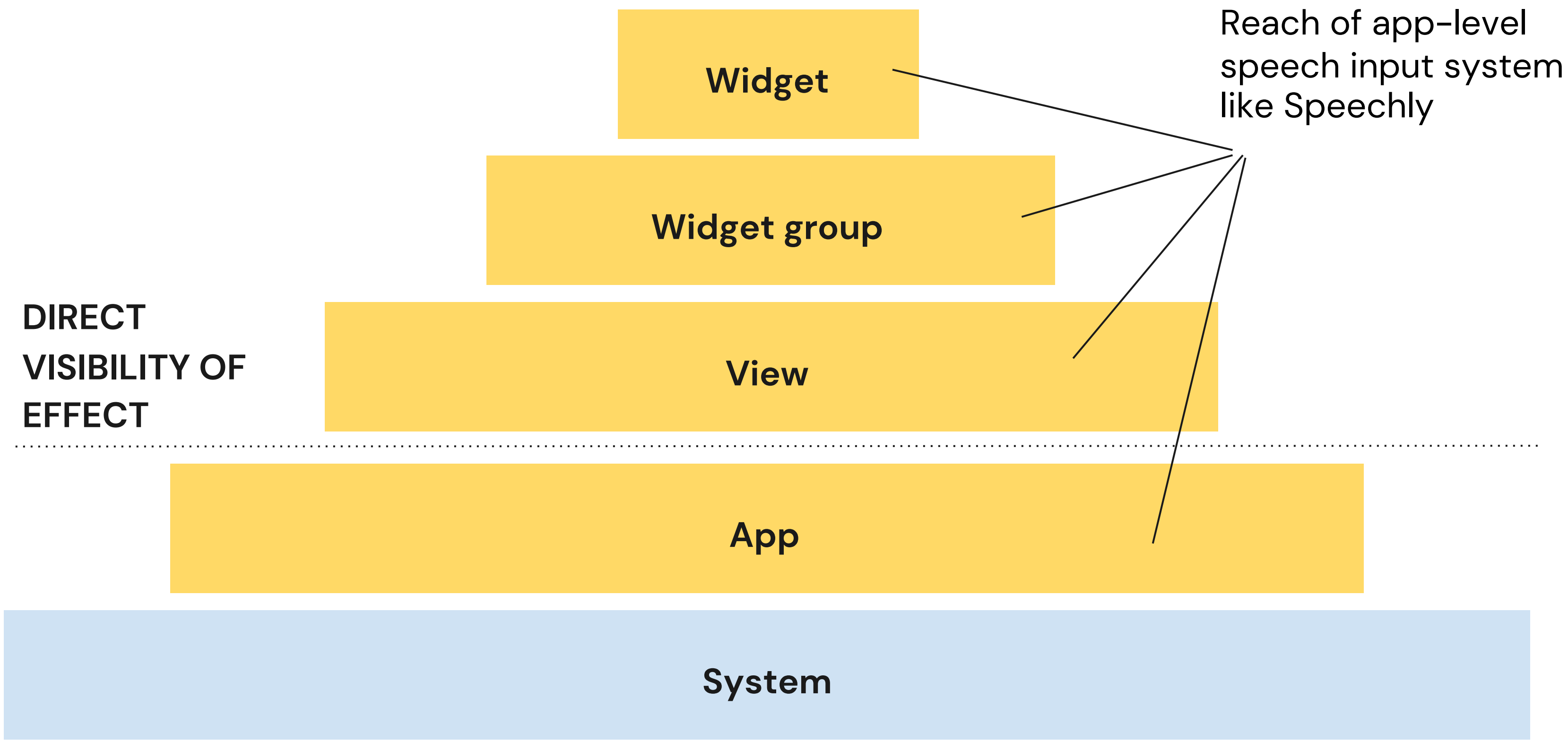
View

App



System

Reach of manipulation (input)



Reach of manipulation (input)

Widget reach

Widget group reach

Designing for direct speech input

1. Discover the user jobs that you want to enable in your app
2. Learn about spoken expressions users would naturally use for each job
3. Tag and generalize the expressions for the natural language understanding (NLU) system
4. Ensure it's possible to map intents and keywords from the NLU system to app state changes. Enable providing partial information.
5. Provide visual cues about the supported expressions in the GUI

User phrases for searching for flight options

“Book a flight from Miami to Helsinki for tomorrow.”

“One-way flight from Stockholm to London for 2 passengers.”

“To London.”

Tag the expressions for the natural language understanding (NLU) system

“Book a flight from **Miami** to **Helsinki** for **tomorrow**.”

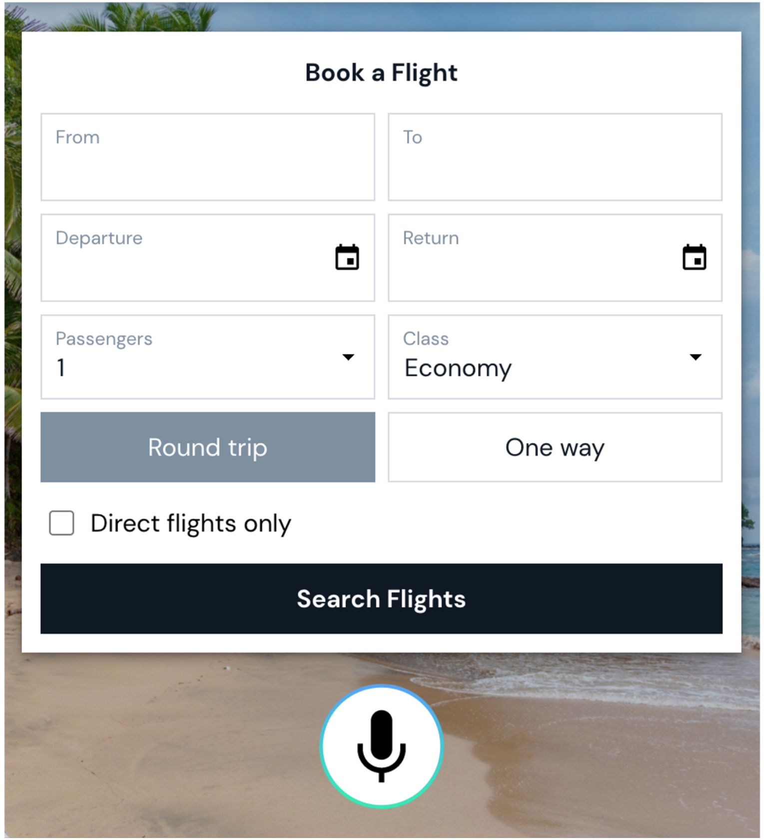
“One-way flight from **Stockholm** to **London** for **2** passengers.”

“To **London**.”

Generalize the spoken expressions for the NLU system

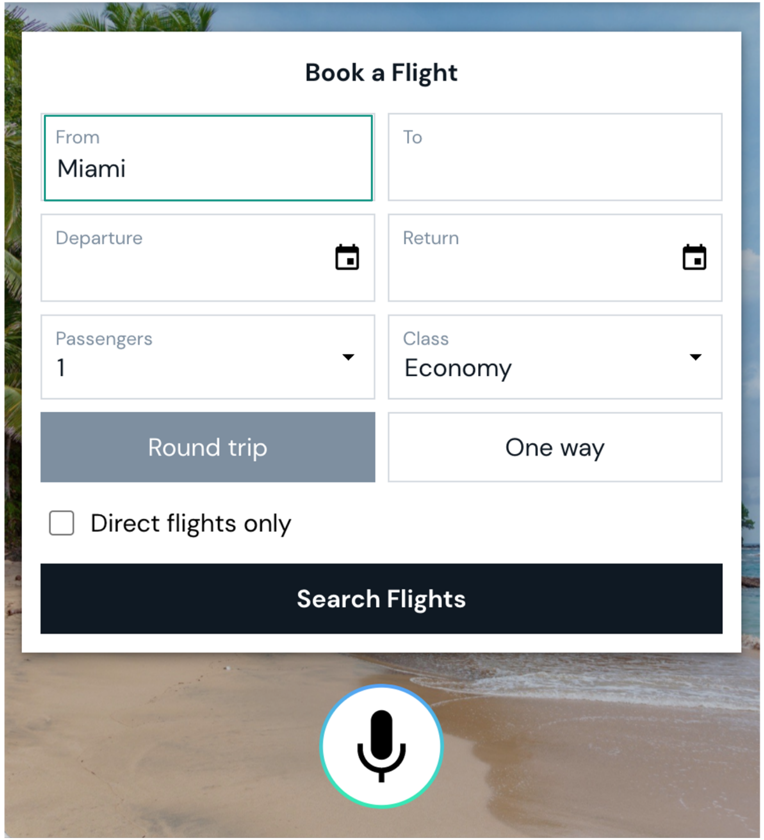
"Book a flight
from [**Miami** | **London** | **Helsinki**]
to [**Miami** | **London** | **Helsinki**]
for **$DATE**."

“Book a flight



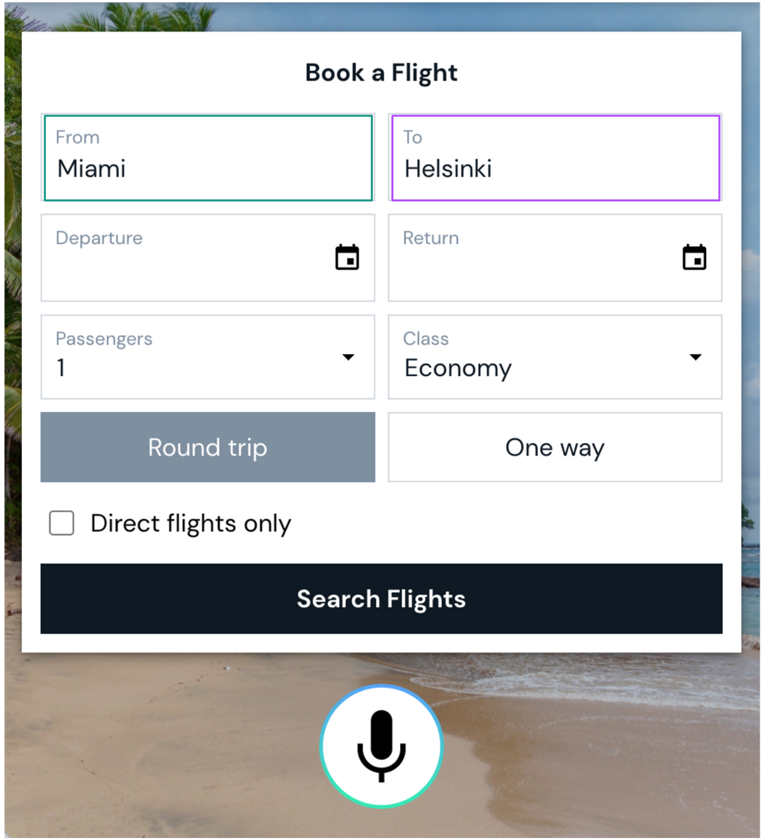
Speechly

“Book a flight
from Miami



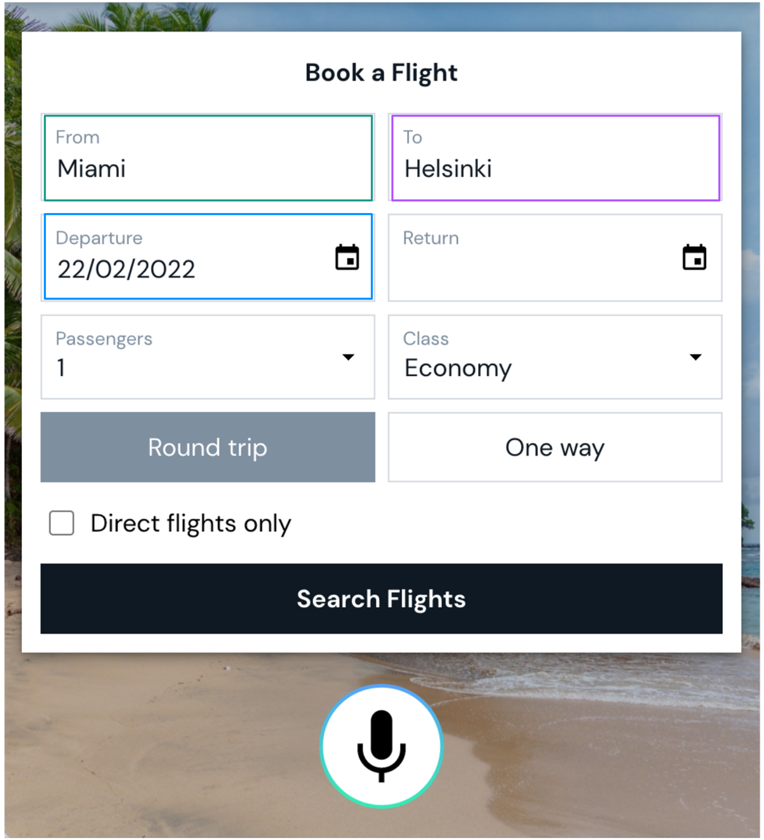
Speechly

“Book a flight
from Miami
to Helsinki



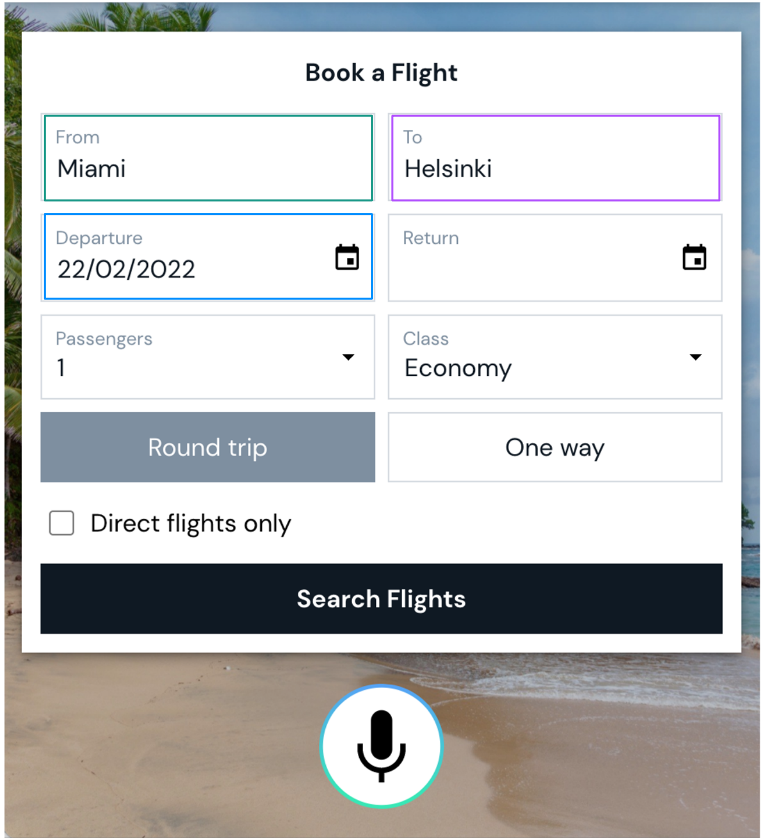
Speechly

“Book a flight
from Miami
to Helsinki
for tomorrow.”



Speechly

"Book a flight
from Miami
to Helsinki
for tomorrow."



{intent: "book",
 words: ["BOOK", "A", …],
 entities: [
  {type: "from",
   value: "MIAMI"},
  {type: "to",
   value: "HELSINKI"},
  {type: "departure",
   value: "22/02/2022"}
 ]
}

Speechly

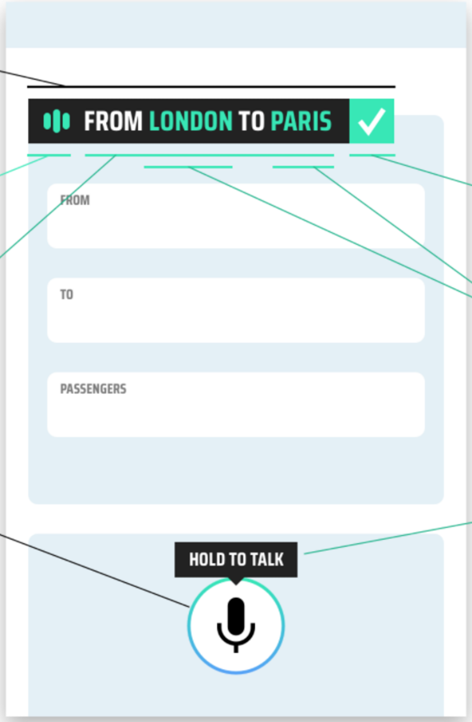
Speechly's Voice UI Design System

Big Transcript
An overlay component that provides feedback about the voice input

VU meter
Indicates voice activity

Transcript
Shows all spoken words in text

Push-To-Talk Button
A component for toggling listening on and off. Floating close to edge of the viewport or positioned near relevant input fields.



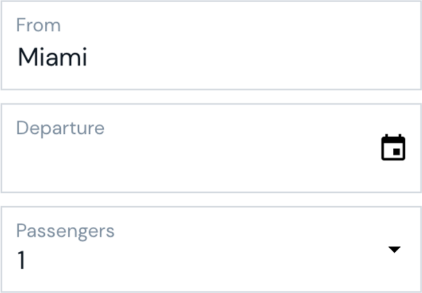
Confirmation checkmark
The app has responded to the voice command

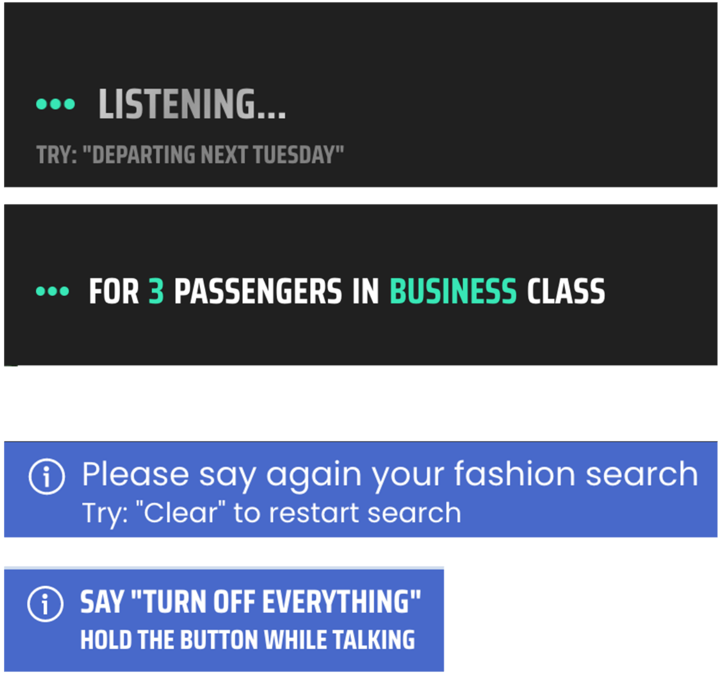
Entities
Keywords extracted from the transcript that the app uses as parameters for voice actions

Intro Callout
Provides a basic usage hint

Speechly

Learning and feedback systems





Next up

Speechly's speech technology
Demo apps
Demo of NLU setup
CodePen fiddling

Speechly

Speechly

Just Say the Word! from Doppio Games

Speech control of VR/AR applications / Zoan

Experiments

- “Memory” of last selected device, room and verb (in smart home)
- Apply alterations to last item (in pizza ordering)
- Undo (in pizza ordering)
- Speech-controlled Pac Man

Thanks!

Janne Pylkkönen janne@speechly.com
Ari Nykänen ari@speechly.com

github.com/speechly

Speechly

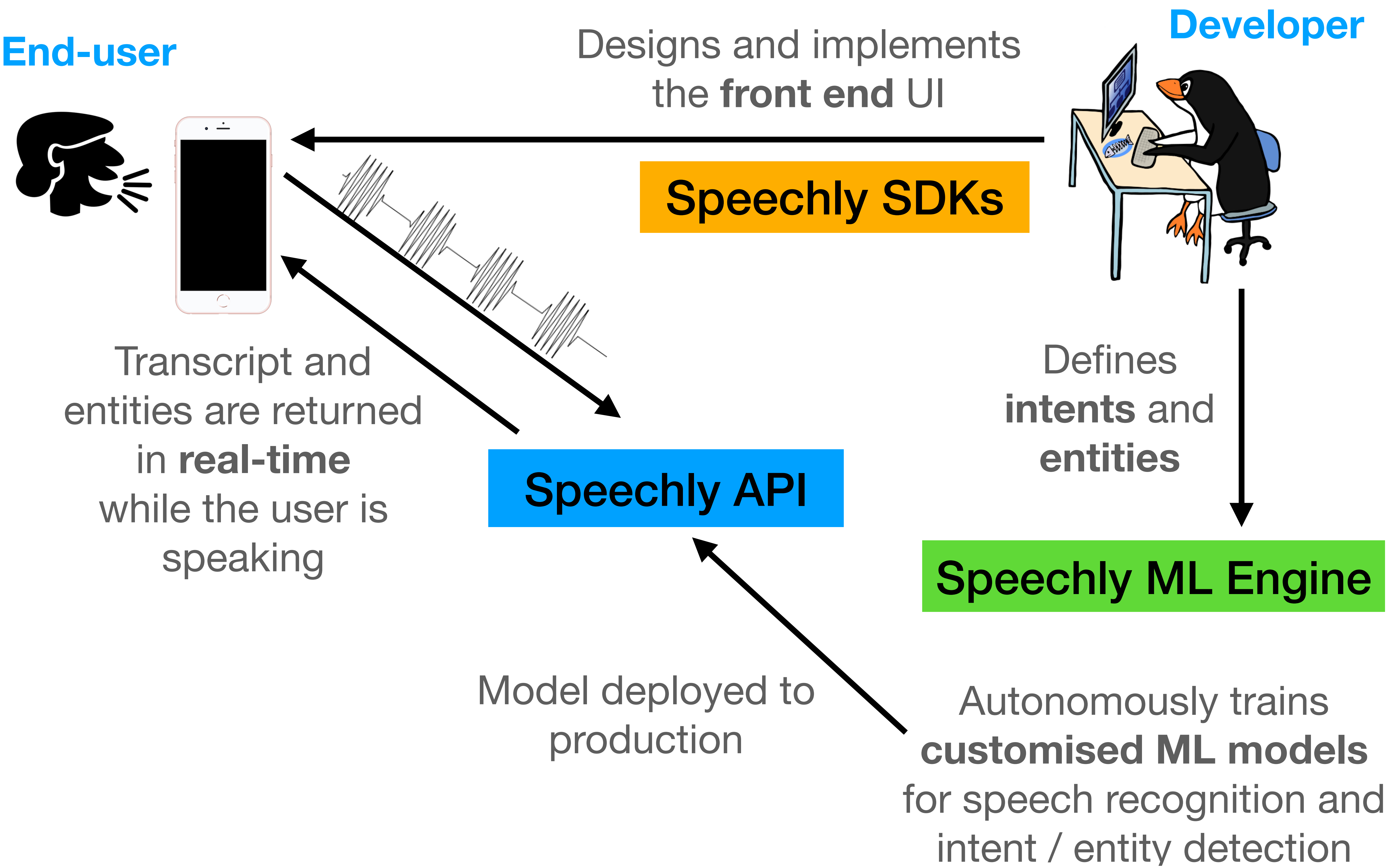
Speechly ASR+NLU Technology in a Nutshell

Janne Pylkkönen
Feb 22nd 2022

Speechly

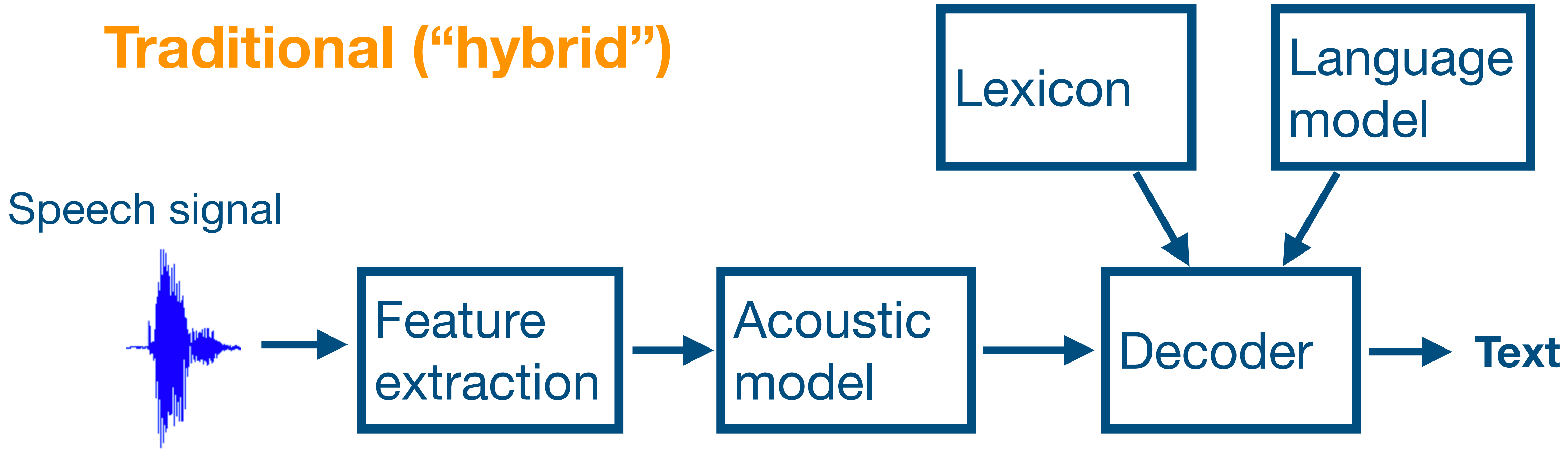
Definitions

- Utterance: Something a user says, typically a spoken sentence or a command
- Transcript: Written representation of the speech
- Intent: The task the user wants to achieve. For example, in utterance *”Turn off the living room lights”* the intent could be defined as ”turn_off”
- Entities: “Parameters” of the intent. In the above example, we can identify ”living room” and “lights” as entities.


End-user
Designs and implements the **front end** UI
Developer
Speechly SDKs
Transcript and entities are returned in **real-time** while the user is speaking
Speechly API
Defines **intents** and **entities**
Speechly ML Engine
Model deployed to production
Autonomously trains **customised ML models** for speech recognition and intent / entity detection

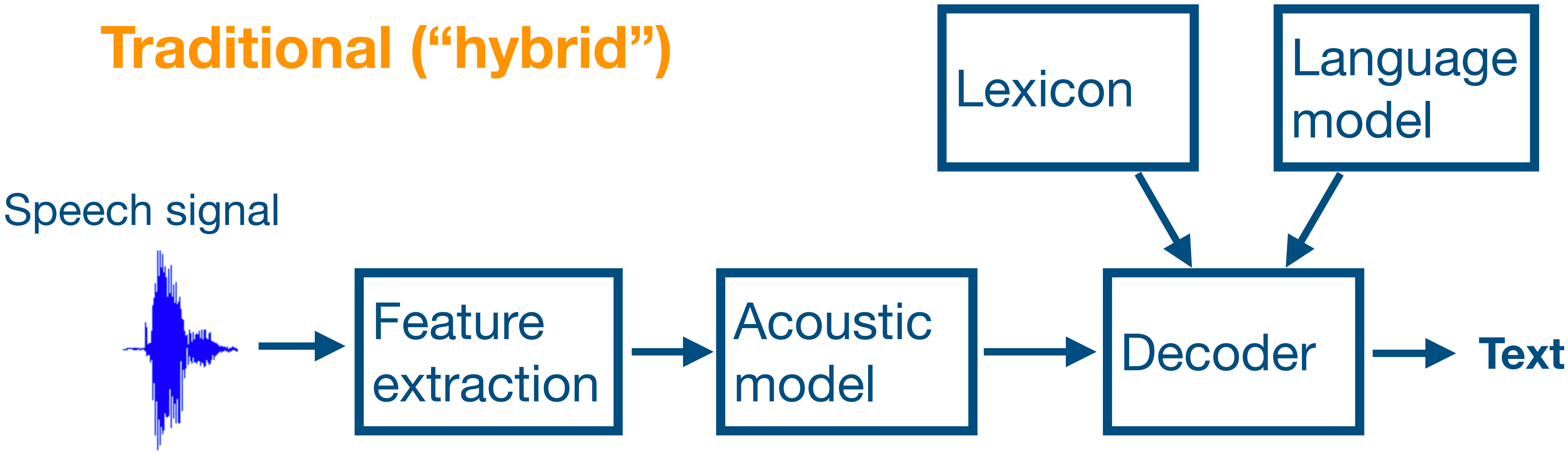
Components of a Speech Recognizer

Traditional (“hybrid”)

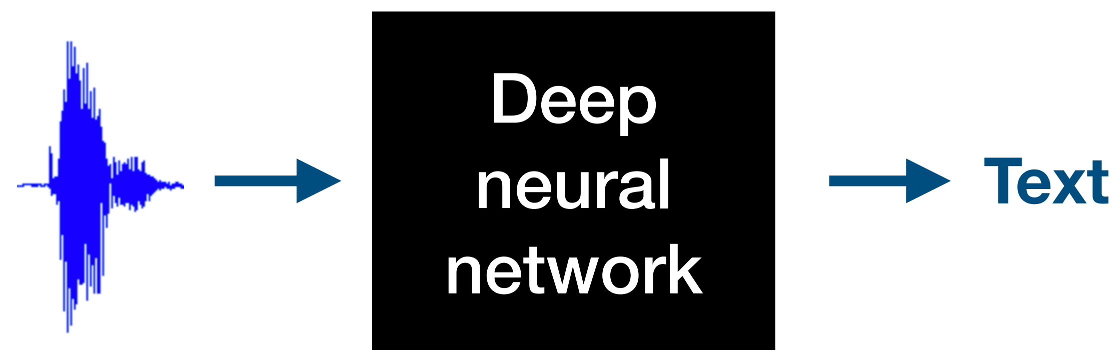


Components of a Speech Recognizer

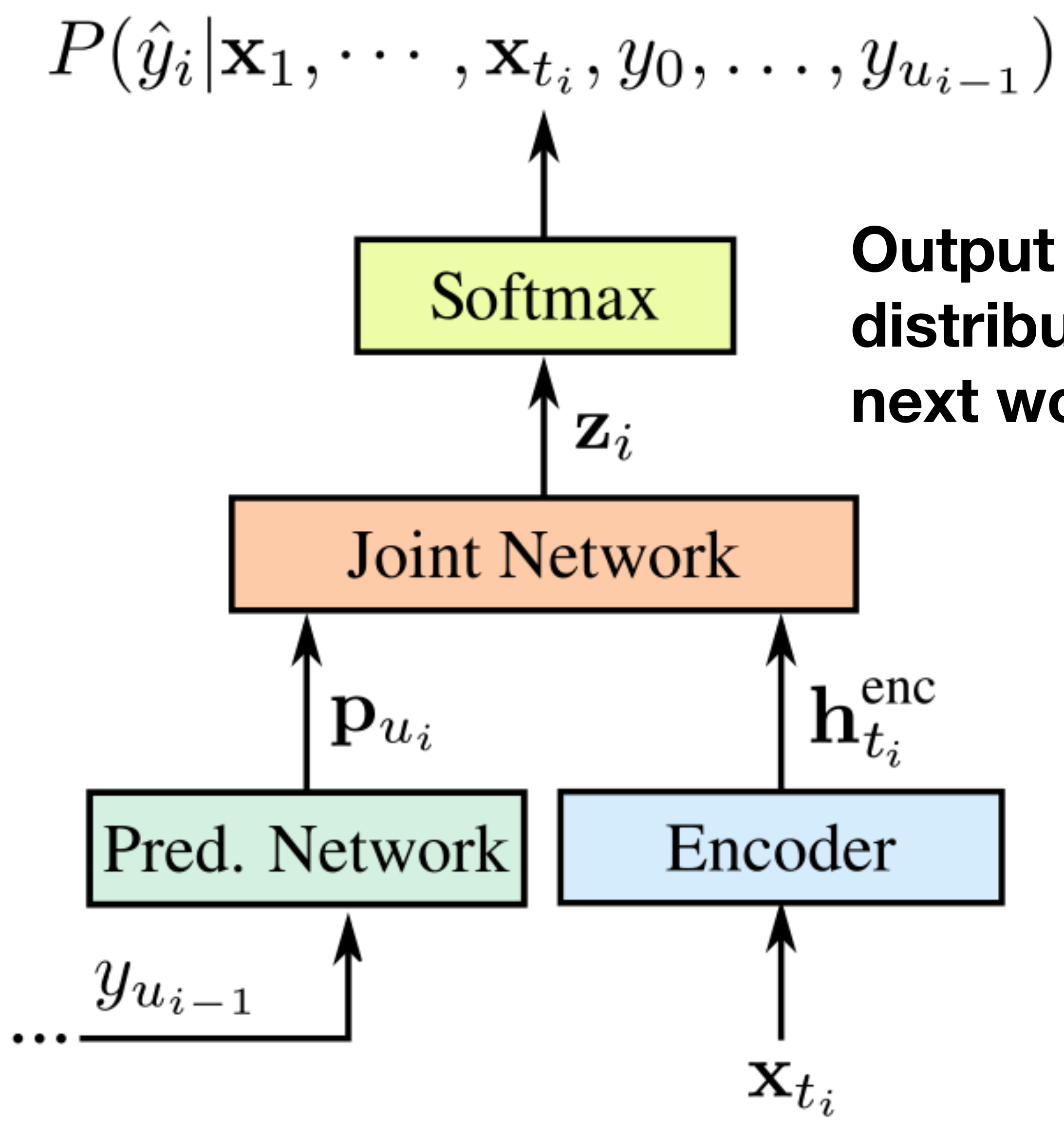
Traditional ("hybrid")



Modern ("end-to-end")



End-to-end ASR: RNN-Transducer



Output is a probability distribution over the next word/token

Prediction network resembles a language model: Takes previous words as inputs

Encoder takes acoustic features as input

- RNN-T uses only the “left” context to predict the next symbol, therefore it is suitable for streaming applications.
- Trained with large amounts (e.g. 10000h+) of matched speech and transcripts

Customizing the ASR Model

- Typically end-to-end models require matched speech and transcripts for training and also for adaptation/fine-tuning
- If only textual data is available for adaptation, one could use a TTS system to produce matched speech and use that for adaptation
- At Speechly, we have developed our own adaptation method, to quickly customise the RNN-T model based on text-only data

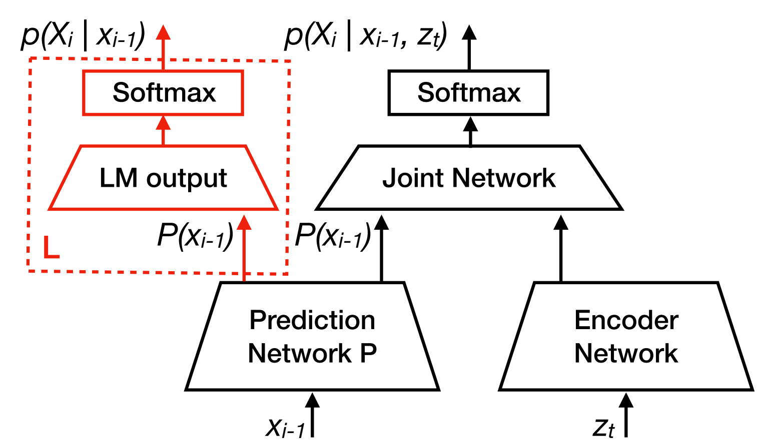
Fast Text-Only Domain Adaptation of RNN-Transducer Prediction Network

Janne Pylkkönen[1], *Antti Ukkonen*[1,2], *Juho Kilpikoski*[1], *Samu Tamminen*[1], *Hannes Heikinheimo*[1]

[1]Speechly, Finland
[2]Department of Computer Science, University of Helsinki, Finland
`firstname@speechly.com`

Abstract

Adaption of end-to-end speech recognition systems to new tasks is known to be challenging. A number of solutions have been proposed which apply external language models with various fusion methods, possibly with a combination of two-pass decoding. Also TTS systems have been used to generate adaptation data for the end-to-end models. In this paper we show that RNN-transducer models can be effectively adapted to new domains using only small amounts of textual data. By taking advantage of model's inherent structure, where the prediction network is interpreted as a language model, we can apply fast



Speech recognition tasks

- Typical automatic speech recognition (ASR) tasks:
  - Keyword detection
  - Command-and-control
  - Search by speech
  - Dictation
  - Conversational interaction

Easier ↓ **Harder**

- Speech characteristics relating to the recognition task:
  - Isolated words vs. continuous speech
  - Speaker dependent vs. independent
  - Vocabulary size
  - Read speech, planned speech, conversational speech
  - Non-standard speech (accented, child speech, speech disorders)
  - Environmental noise
  - Distance to the microphone: close-talk, near-field, far-field

From Transcripts to Understanding

- To refine the ASR outputs to something more usable, we need NLU models (=Natural Language Understanding)
- Intent detection (intent classification) is a text classification problem
- Entity detection (entity extraction/recognition) is a sequence tagging problem. As such, it is more difficult than pure text classification.
- Entity and intent detection are typically done with separate models, but there are e.g. transformer-based models which perform both tasks at once. Model training requires annotated transcripts.
- NLU models are typically relatively small compared to the ASR model. This is possible as they utilise word embeddings to map the symbols to a meaning-bearing vector space.
- Additional challenge in Voice UIs: To appear responsive, entities (and intents) should be extracted in a streaming manner, while the user is speaking