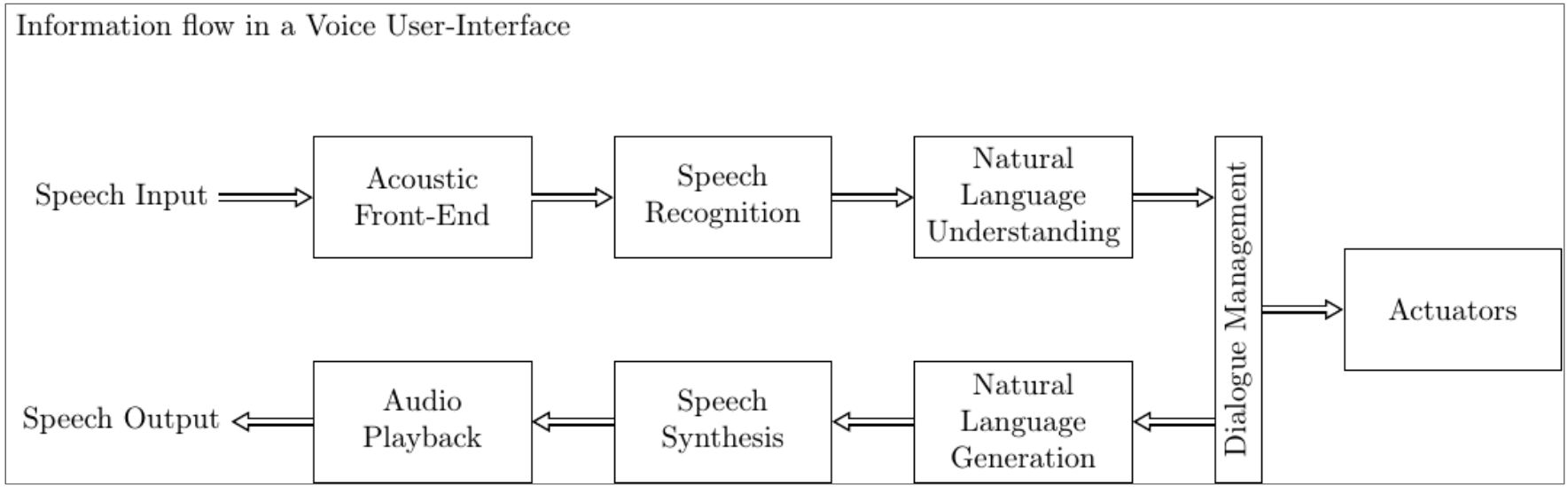
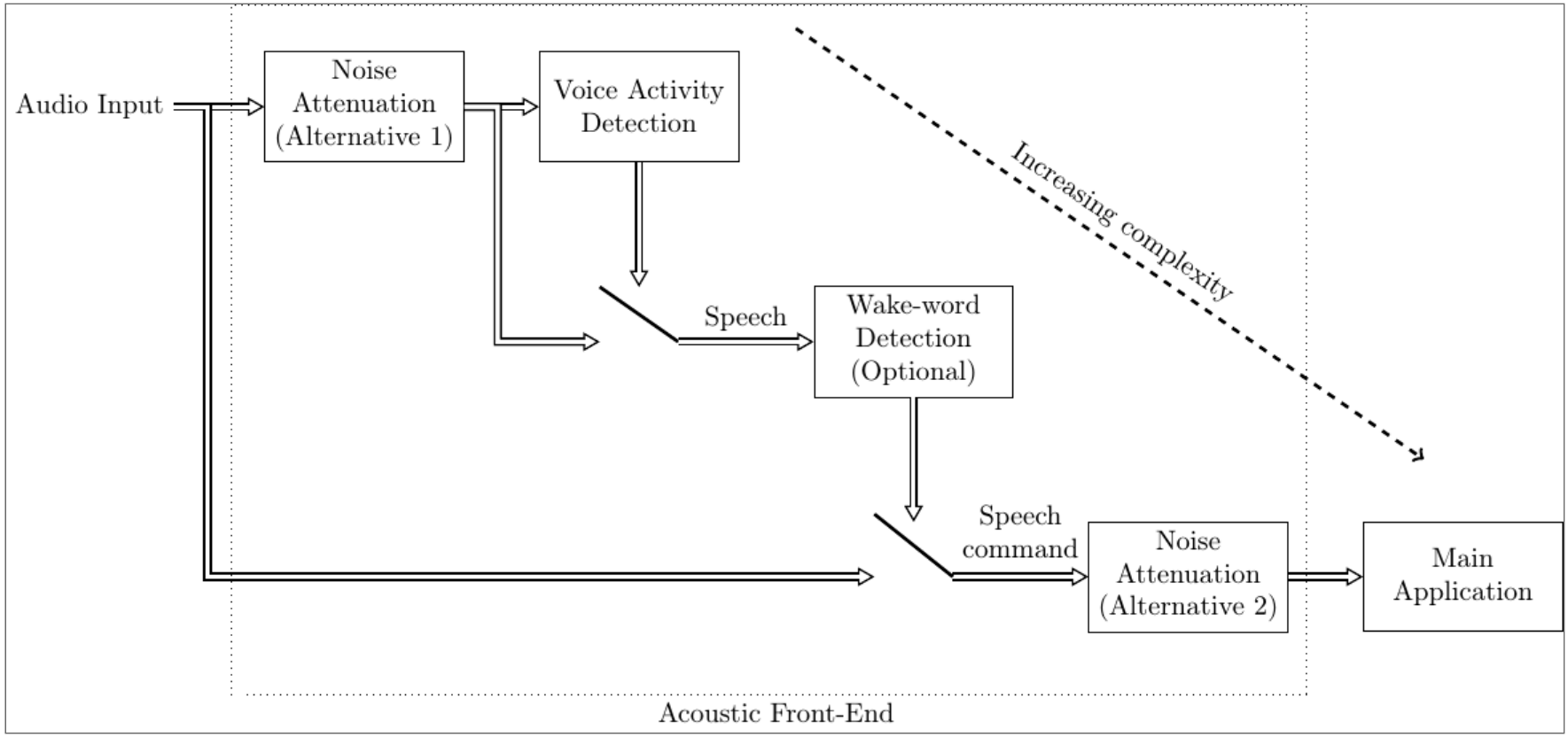
Speech interface architecture

Archiecture overview



Front-end / Preprocessing



- *Noise attenuation* (speech enhancement) removes background noise, echo and reverberation.
- *Voice activity detection* (VAD) determines whether signal is speech or not. If signal is not speech, then we do not need any further processing.
- *Wake-word detection* (or keyword spotting) is triggered by *the keyword* like "Siri" or "Alexa" and activates the main functionalities of the interface.

- Later modules are resources intensive (CPU, transmission, data) and activated only when we know that the signal is speech directed to the device.
- Voice activity detection is relatively cheap to run and can therefore be active all the time. Wake-word detection is also not too expensive.
- Wake-word detection is also a *nuisance-filter* in that it makes sure device is activated only when user chooses to use it.

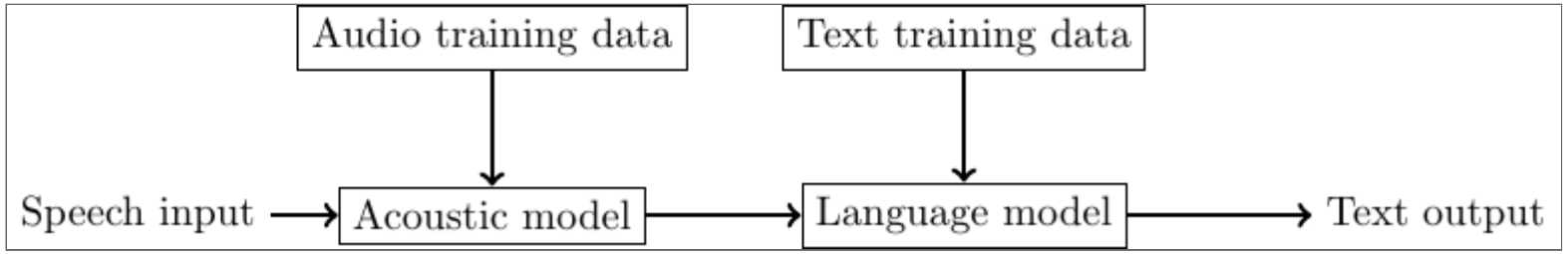
Speech recognition

Formally, speech recognition is defined as the task of finding the *most likely* sequence of words given an observation. That is, when we pick up a speech signal, we look at all possible sequences of words, and choose the most probable sentence.

	Which one is most likely?
Hypothesis 1	How to recognize speech?
Hypothesis 2	How to wreck a nice beach?
Hypothesis 3	Lau to repo nice peach?

Likelihoods are useful on both word and phoneme level.

Signal flow


Audio training data
Text training data
Speech input
Acoustic model
Language model
Text output

Phoneme models = Acoustic model

- $P(A)$ How often do a phoneme A appear in the target language?
- $P(B \mid A)$ How likely is the observation B when A is the phoneme?
- $P(B)$ How likely is the observation B?
- Then likelihood that observation B is phoneme A is

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$ (Bayes' theorem).

- The acoustic model provides (is) the probabilities $P(A)$, $P(B)$ and $P(B \mid A)$.
- Then check $P(A \mid B)$ for every phoneme and choose the most likely.

Diphones and Triphones

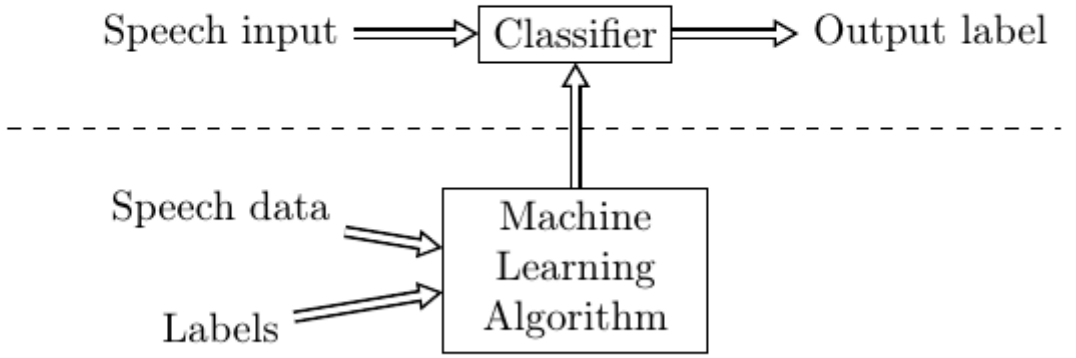
- Phones (phonemes) are very low-level information.
    - Does not take coarticulation into account.
- More information available if we look at combinations of phones.
    - Diphones - two phones
    - Triphones - three phones
- Apply Bayes' theorem as before
- Much more work to check all diphones or even triphones.
    - With N phones we have N^2 diphones and N^3 triphones.
- But, languages do not use all di/triphone combinations.

Phoneme classifier

- Analyze sound for different properties = features
    - For example, energy in different frequency bands.
- Feed features to machine learning model.
- Machine learning model outputs probability, $P(B \mid A)$.

Machine Learning Approach for Information Extraction

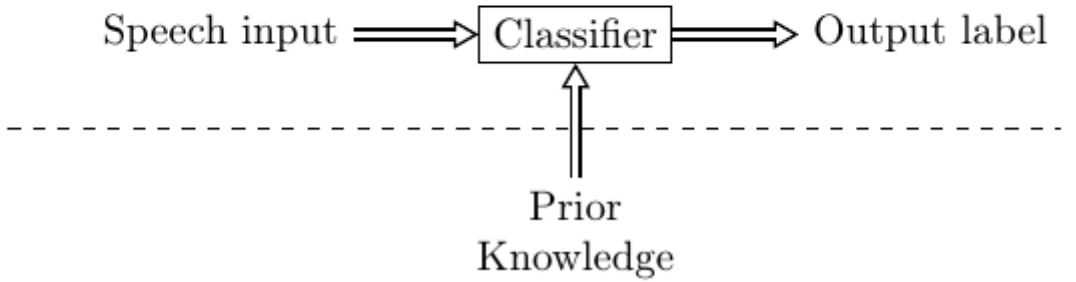
Application (on-line)



Training (off-line)

Signal Processing Approach for Information Extraction

Application (on-line)



Language model

- List of probabilities for combinations of words.

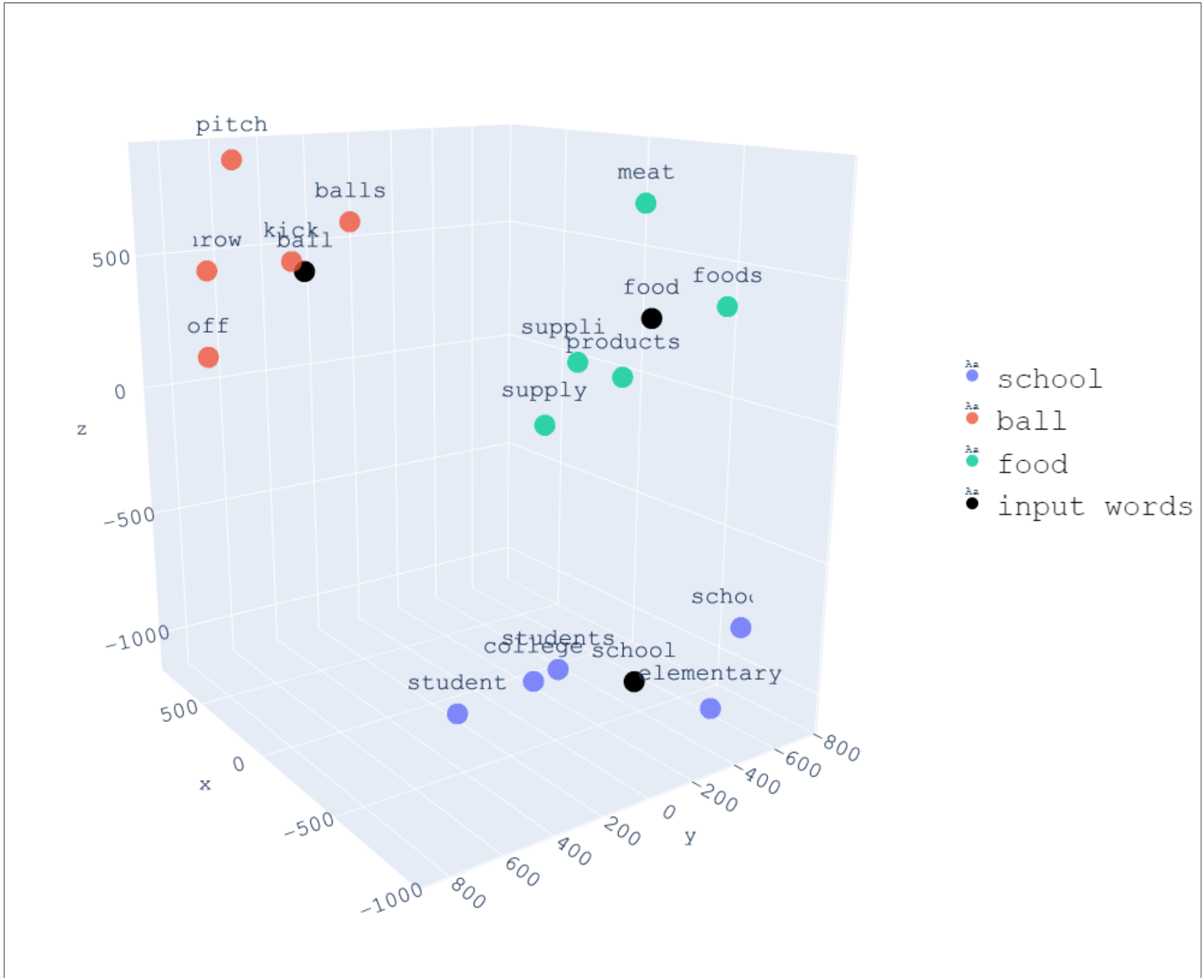
Word1	Word2	Probability
eat	bike	0
eat	sandwich	0.091
eat	out	0.07
to	lunch	0.001

...

- NB: This list becomes very large.
    - Native speakers often use a vocabulary of $\sim$ 20.000 words.
    - Advanced non-native speakers use perhaps 5.000 words.

Embeddings

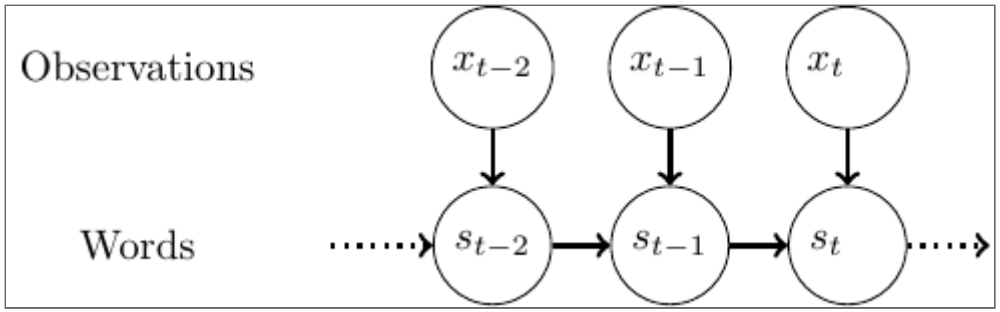
- Similar words should have similar probabilities in language model.
- How do we measure similarity between words? *Man, Woman, Dude, Bike*
    1. Words are of different length $\Rightarrow$ direct comparison is difficult.
    2. Similar words can "look" very different, like "Man", "Guy" and "Dude".
    3. To add a word to a list, we'd need to define relationship to all previous words.
- Generate a mapping = list = *embedding* from words to *vectors*
    - Similar words should be near each other.
    - Learn embedding by training from large text database.
    - Similar words have similar relationships (probabilities).


pitch
balls
meat
kick
ball
row
500
food
foods
off
suppli
products
0
supply
z
school
ball
food
input words
-500
-1000
scho
students
college
school
student
elementary
500
0
x
-500
-1000
800
600
400
200
0
y
-200
-400
-600
-800

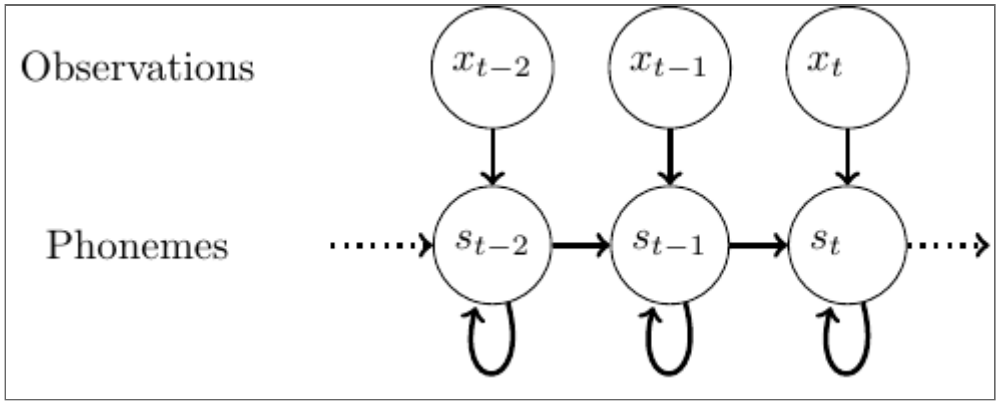
Sequence modelling: HMM

- Problem: We know likelihoods of words and likelihoods of transitions between words. How do we find the most likely sentence?
- Hidden Markov models (HMMs) are models of transition probabilities of sequences.
- The words in the sequence are assumed "hidden" (i.e. unknown).
    - We can still assign values to likelihoods of words.
    - Combine with transition likelihoods and some sentences start to stand out as more likely.

- HMM for word-to-word transitions



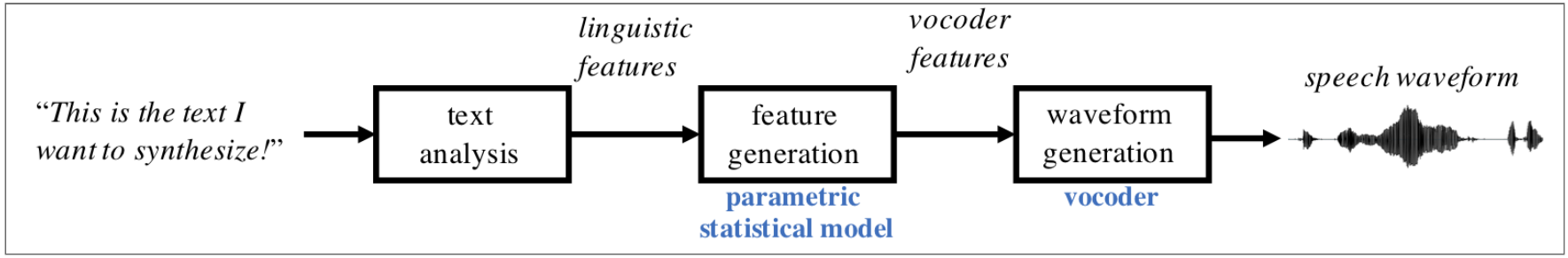
- HMM for phoneme-to-phoneme transitions



Speech synthesis

- Text-to-speech = the opposite of speech recognition
- However, text does not contain non-text information such as intonation
  $\Rightarrow$ Have to guess/create intonation (feature generation).
- Creating intelligble speech is straightforward. Natural-sounding speech is hard.

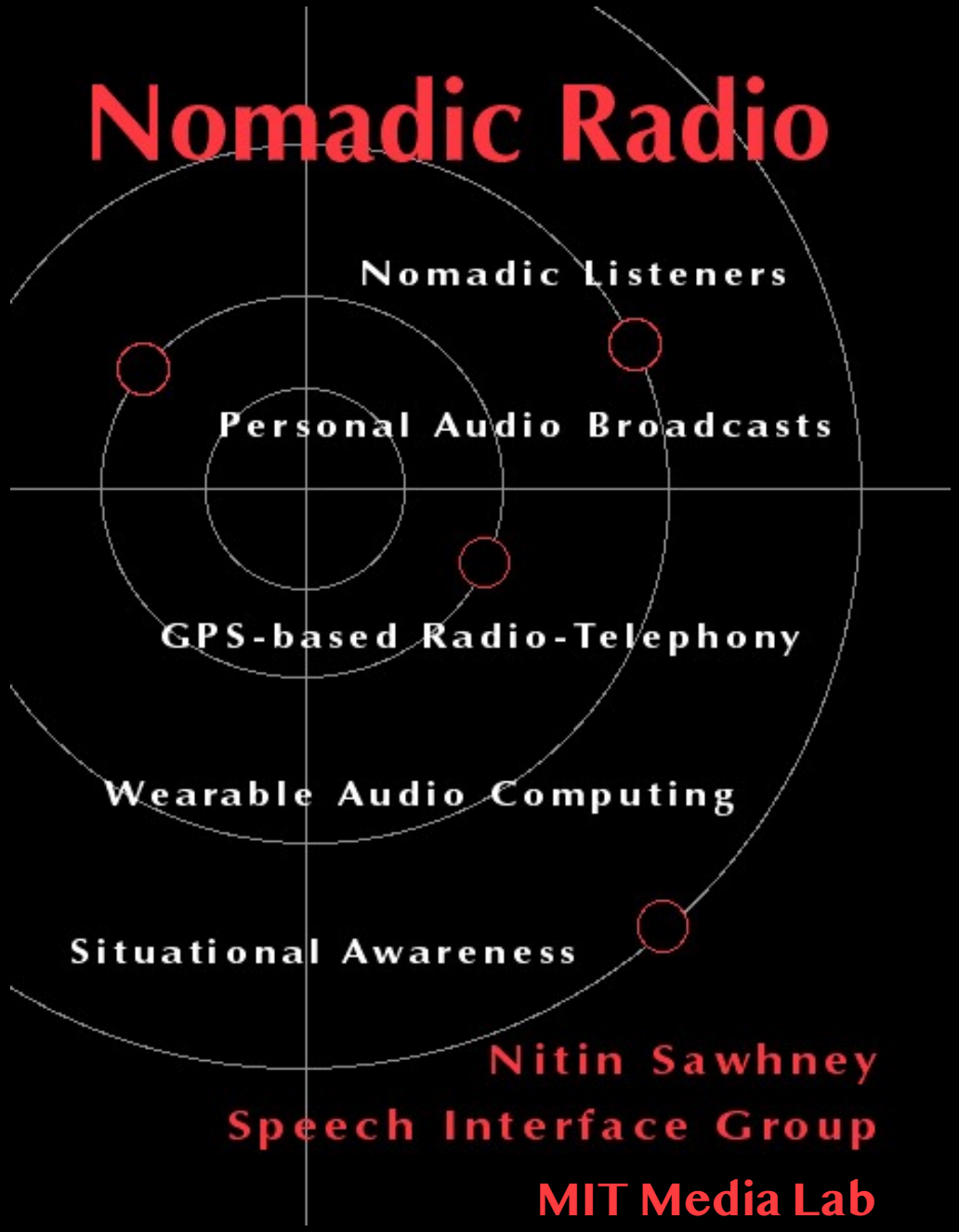
Statistical parametric speech synthesis:



Current trends

- Machine learning (ML) methods have developed rapidly and speech applications are among its important drivers.
- One direction in ML is end-to-end models, which replaces whole structure with a single, big ML model.
    - More data in, (presumably) a better model out.
- Bigger-data trend is counter-balanced by the difficult of obtaining data for many smaller languages.
    - Increasingly contributions to smaller languages (e.g. transfer-learning = transferring model from one language to another).

Nomadic Radio

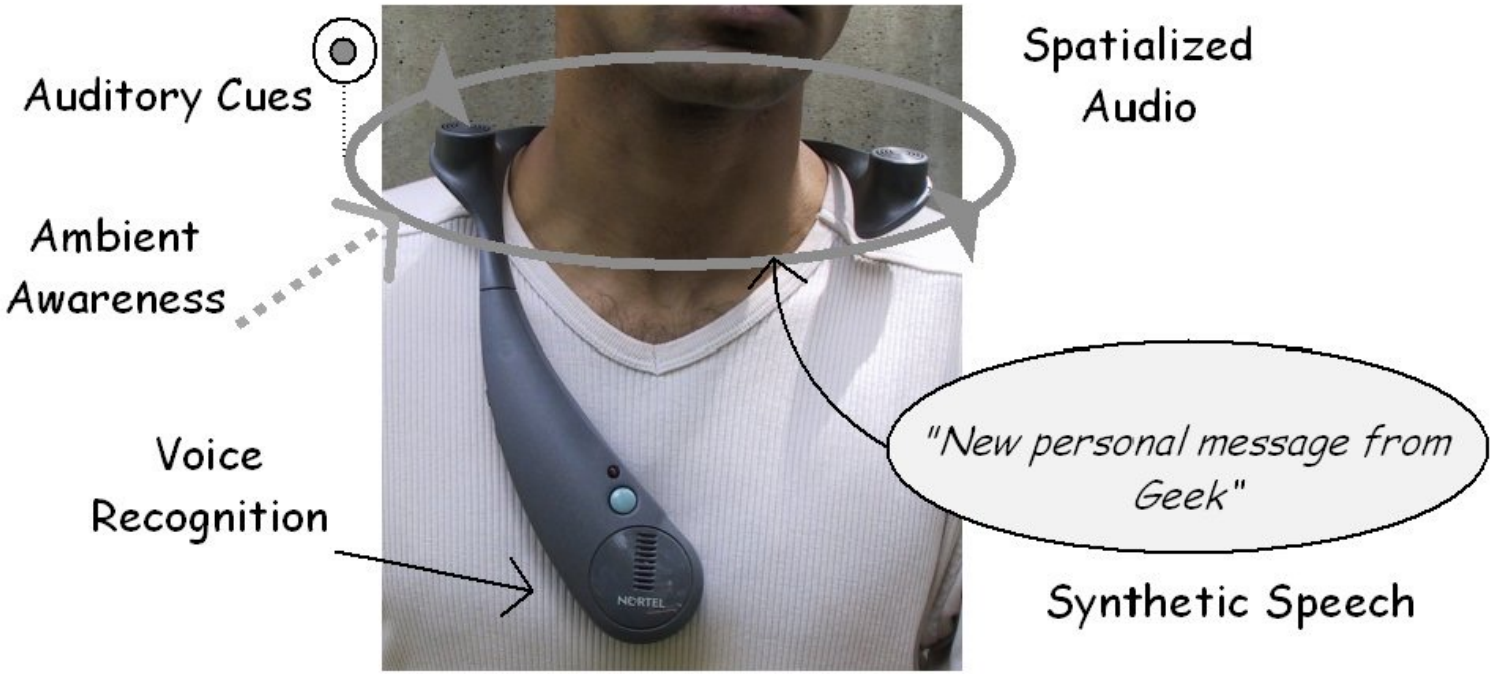


Nitin Sawhney
Speech Interface Group
MIT Media Lab

Wearable Computing Project

MIT Media Lab, mid-1990's

Nomadic Radio: Speaking and Listening on the Run



Nomadic Radio

Wearable Audio Computing

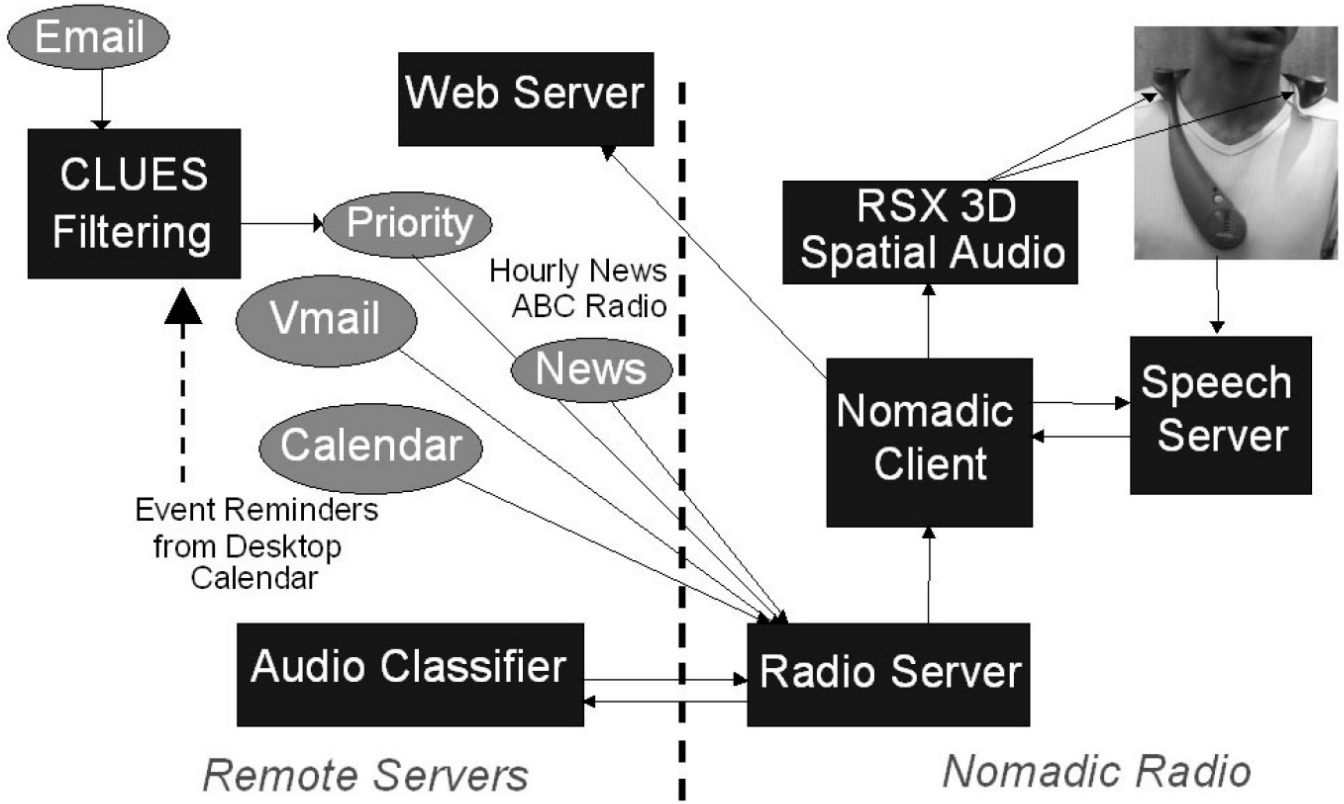
The Digital Future in our Kitchen?

Nitin Sawhney and Chris Schmandt

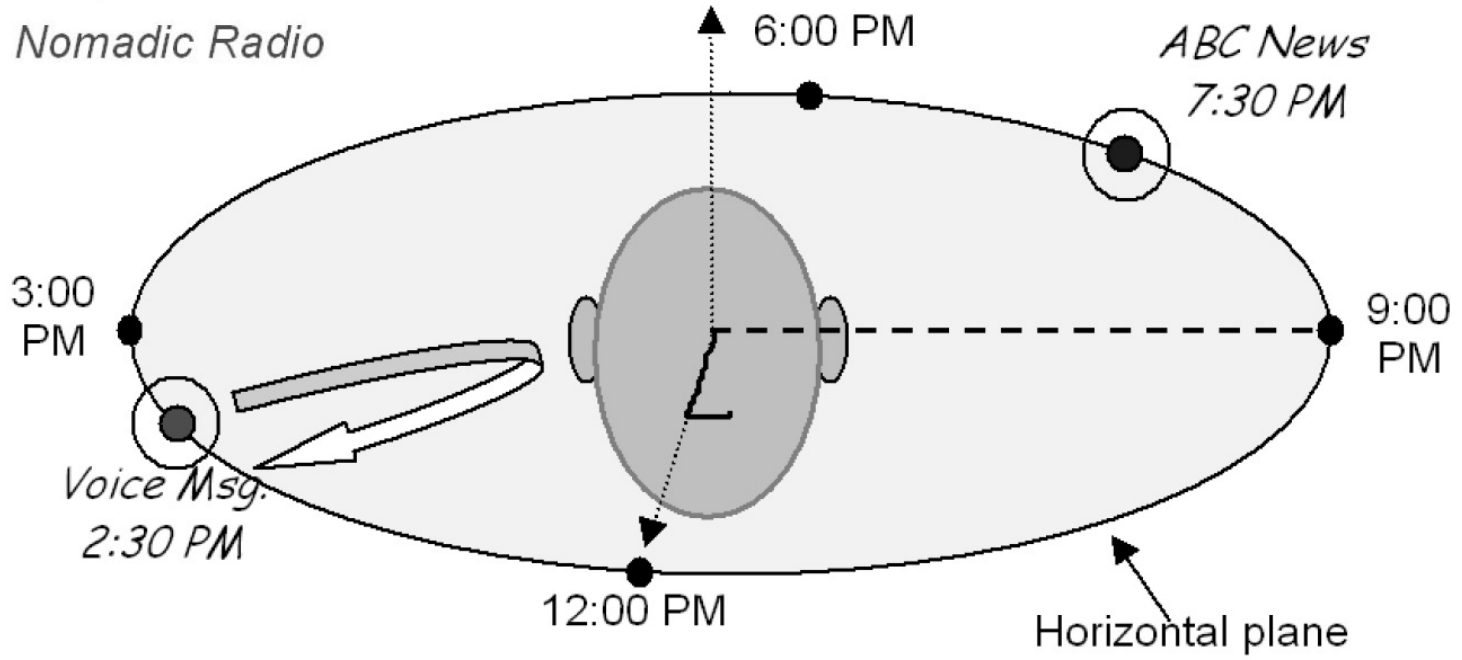
Speech Interface Group

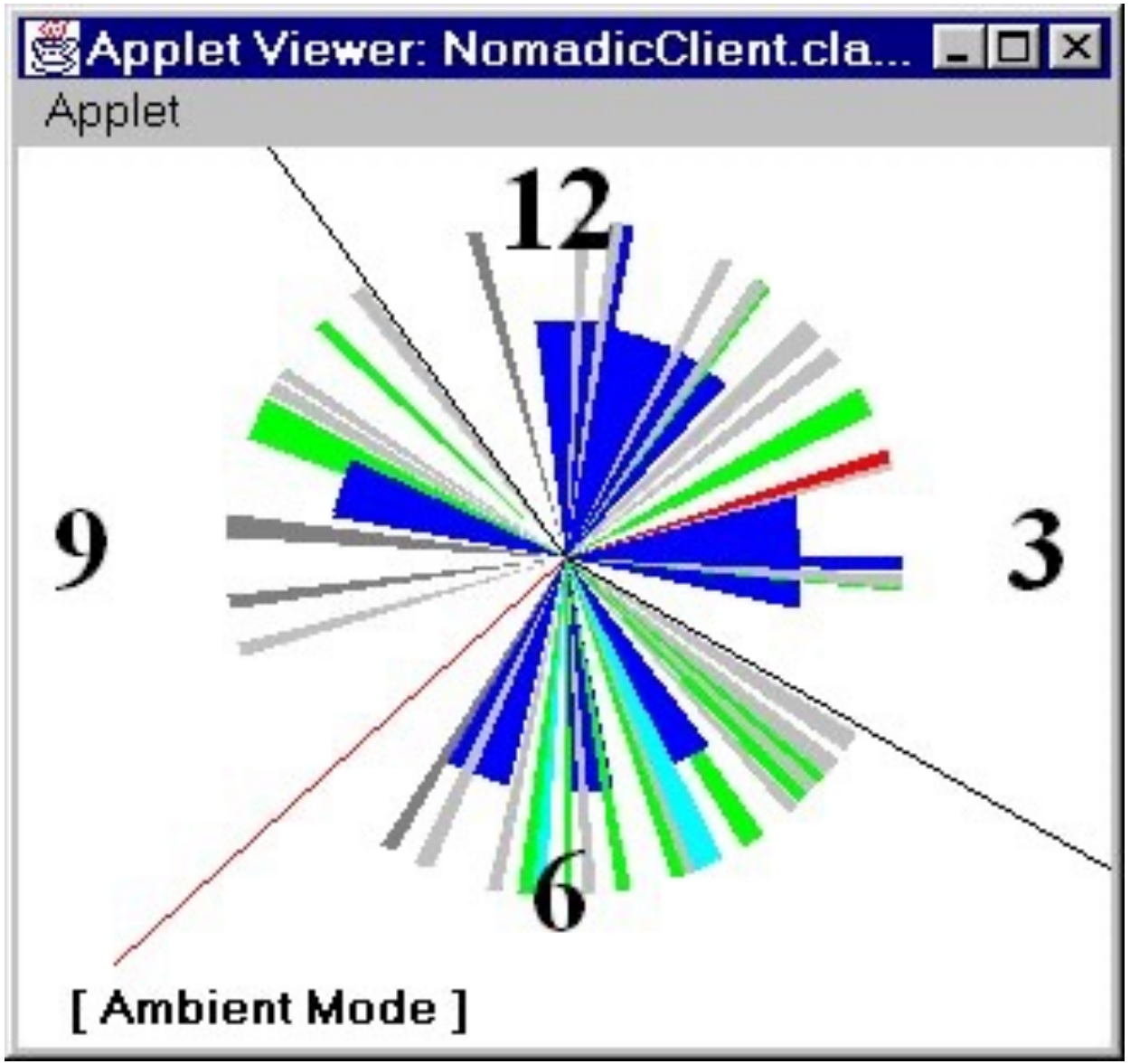
MIT Media Lab

Nomadic Radio in the Wild: Designing a Rugged Form Factor

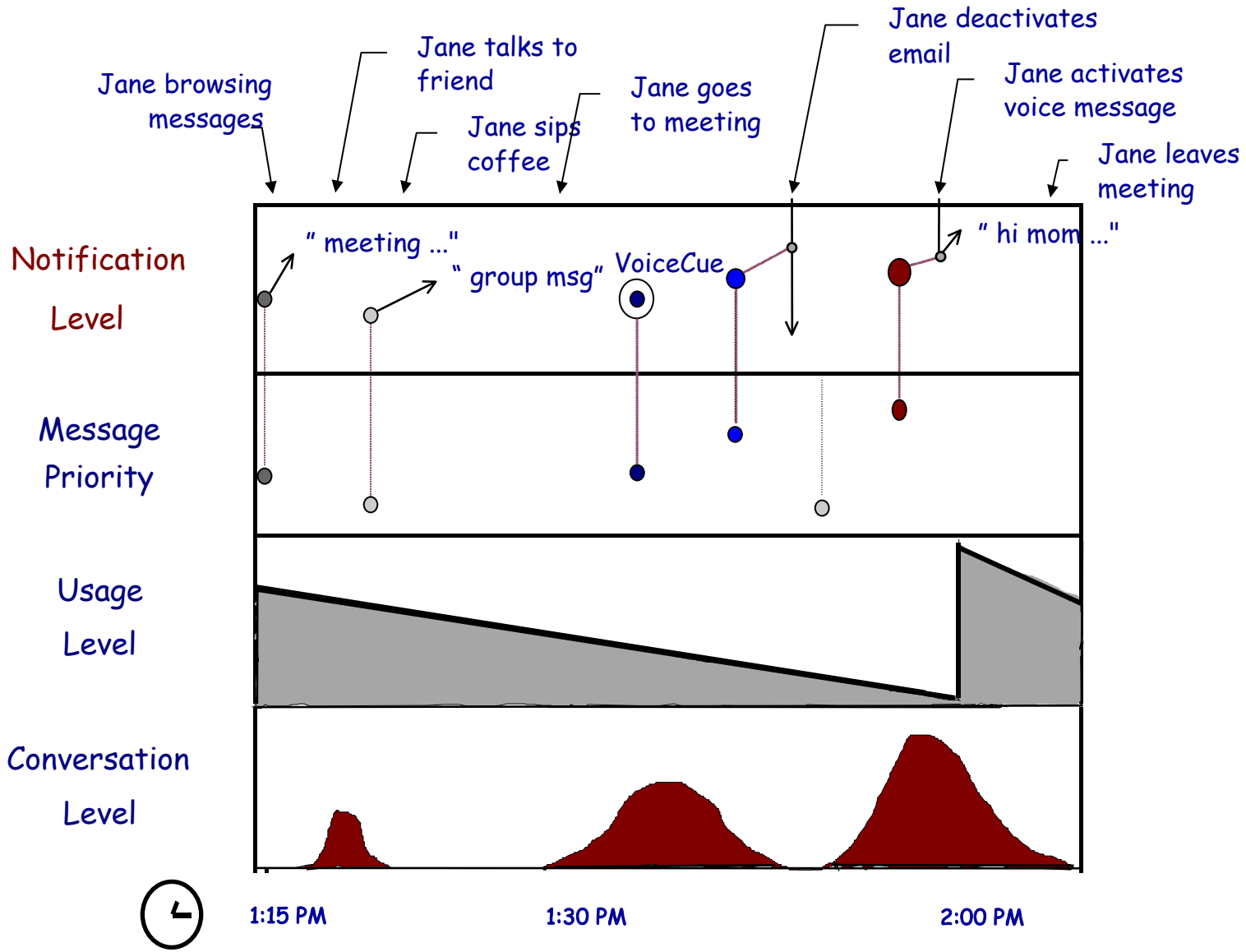


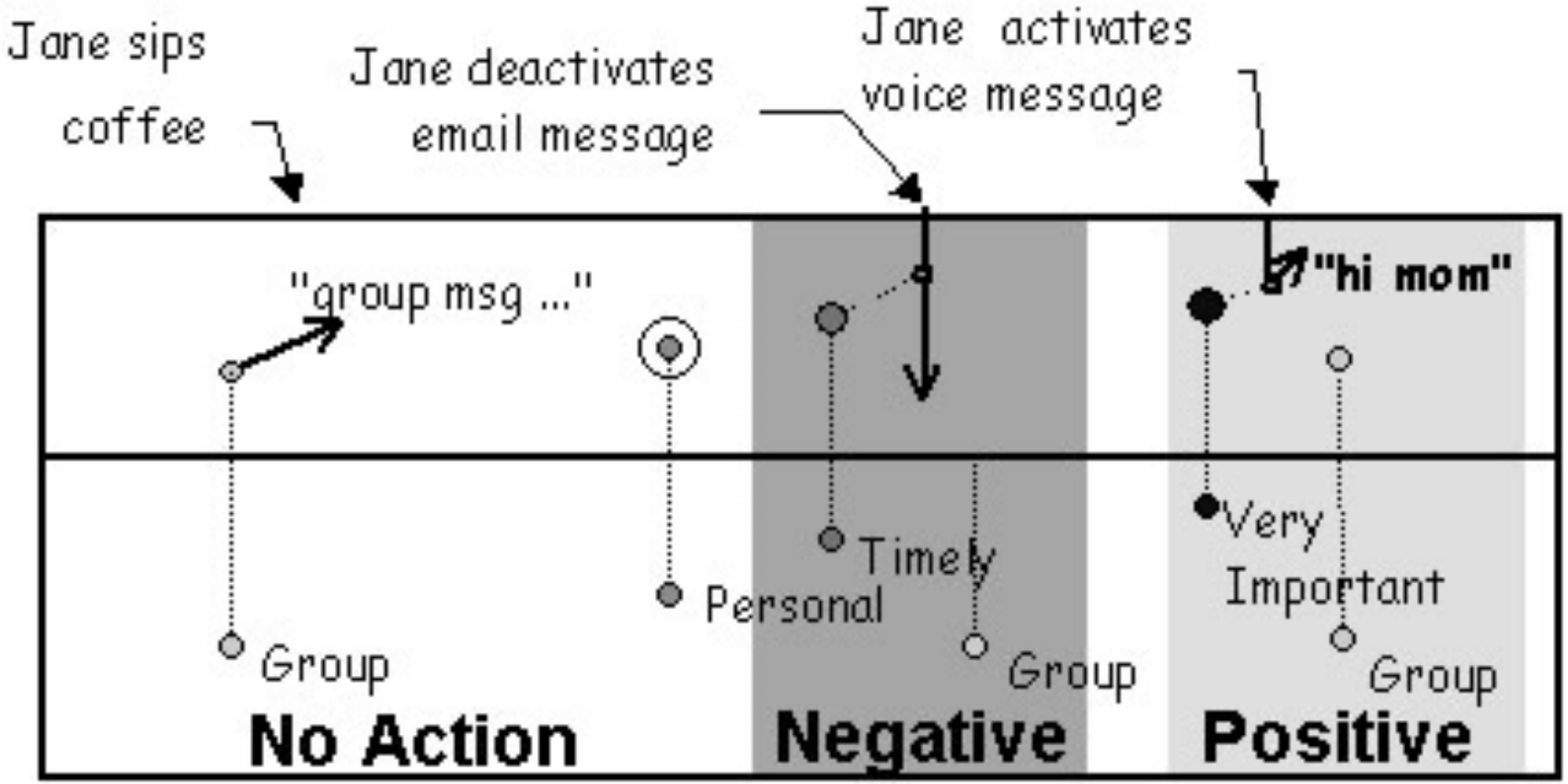
Spatialized Audio:
Simultaneous Listening & Message Scanning



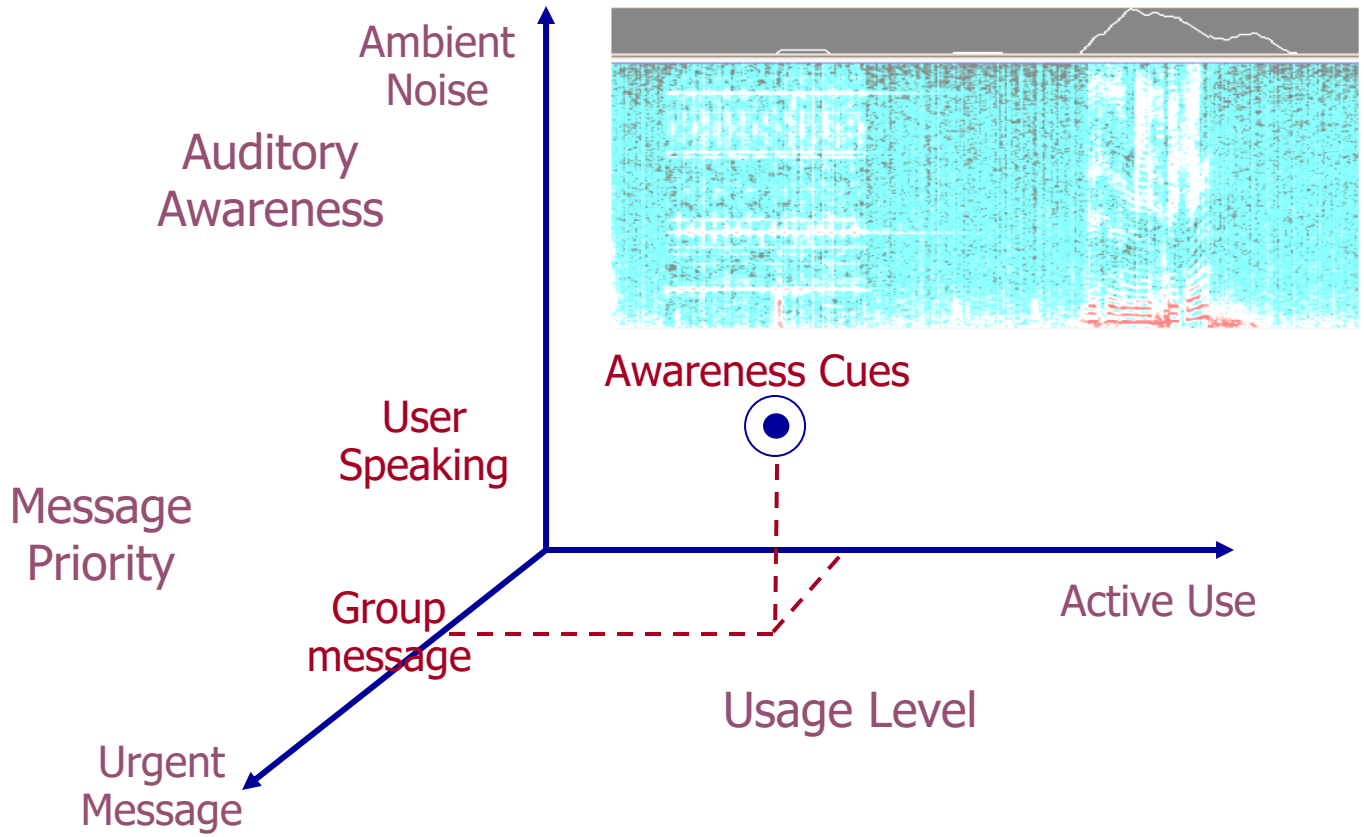

Applet Viewer: NomadicClient.cla...
Applet
12
9
3
6
[Ambient Mode]

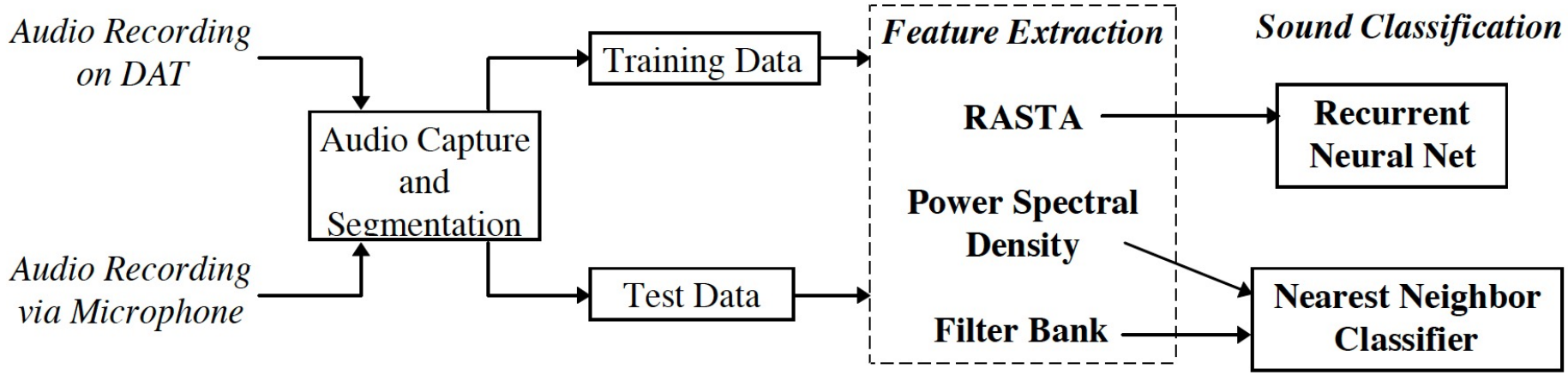
Contextual & Scalable Notifications with Ambient Awareness



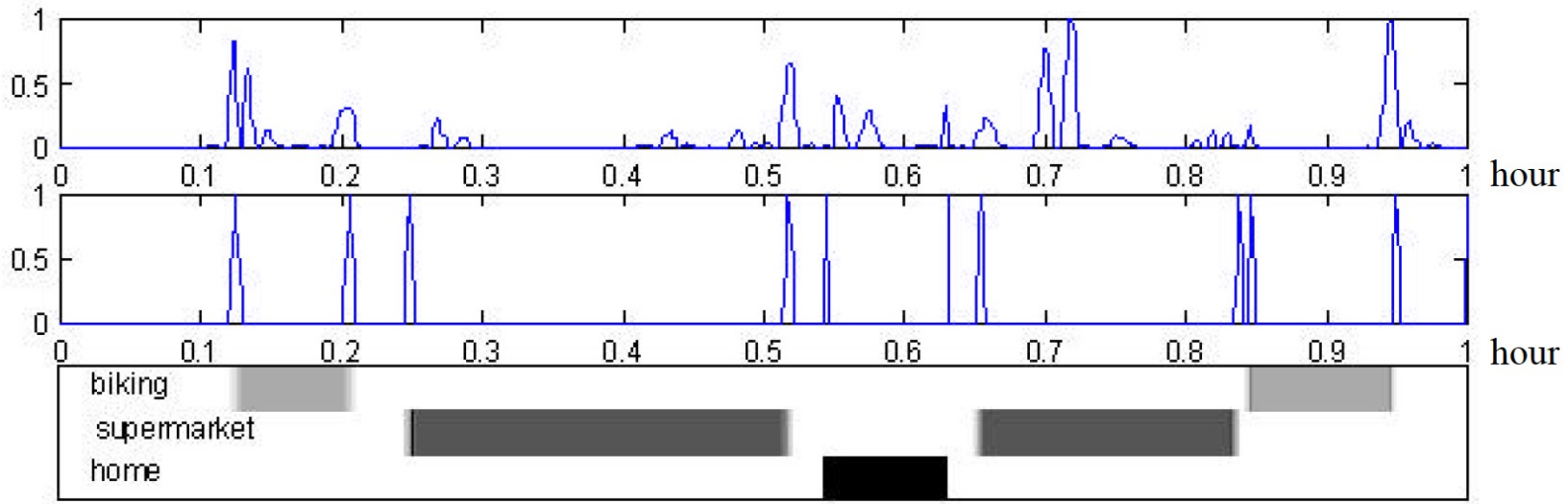

Jane sips coffee
Jane deactivates email message
Jane activates voice message
"group msg ..."
"hi mom"
Group
Personal
Timely
Group
Very Important
Group
No Action
Negative
Positive

Situational Awareness through Environmental Sounds





Overall process for capture, feature extraction and classification of audio data



Scene change detection via clustering (top), hand-labeled scene changes (middle), scene labels (bottom).

Experiences of Wearable Audio Computing

- Using distributed cognition to examine peripheral awareness vs. ambient auditory attention and speech rec. vs. conversational modes
- Spatialized **audio** not as effective in noisy outdoor settings and audio classification became intractable in messy sound environments
- Synthetic **speech** not natural sounding and inability to respond to people via free-form speech recognition was frustrating
- **Social norms** of speaking and listening to oneself while walking around seemed odd and intrusive to others at the time

>> Fast-forward:

- Siri, Google and Cortana on mobile phones today, but **no adaptive models** for auditory attention and awareness!
- Challenges of Privacy, Security and Social Etiquette remain.

Advanced Computational Speech processing

Aalto University
School of Arts, Design
and Architecture

SOPI Research Group

Wavenet

A generative model trained on speech samples.

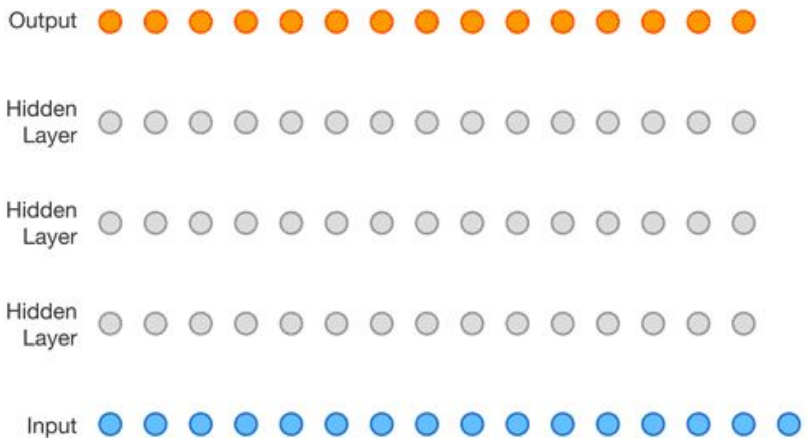
WaveNet automatically incorporates natural-sounding elements left out of earlier text-to-speech systems, such as lip-smacking and breathing patterns.

Examples:

- https://deepmind.com/blog/article/wavenet-generative-model-raw-audio

Related models:

- WaveRNN
- Tacotron 2



DDSP

Timbre transfer.

Uses simple interpretable DSP elements to create complex realistic signals by precisely controlling their many parameters with an ANN.

Whereas WaveNet and other neural audio synthesis models generate waveforms one sample at a time, DDSP passes parameters through known sound synthesis algorithms.

Examples:

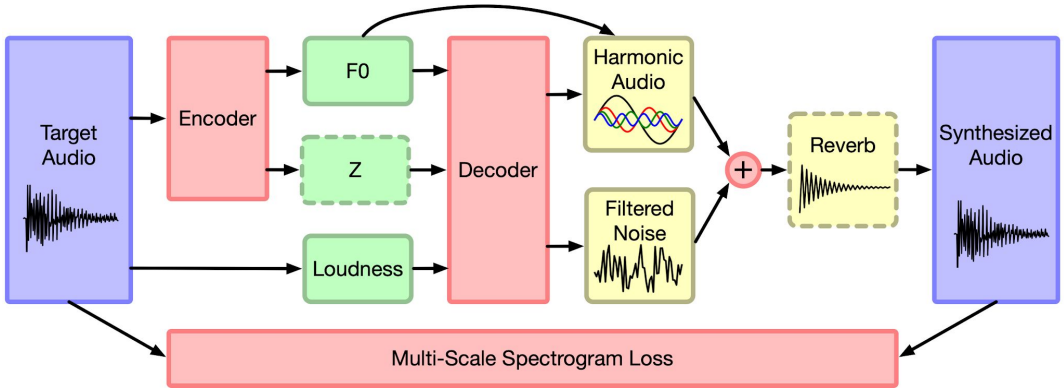
- https://storage.googleapis.com/ddsp/index.html#timbretransfer
- https://sites.research.google/tonetransfer

Trained for Voice Transfer:

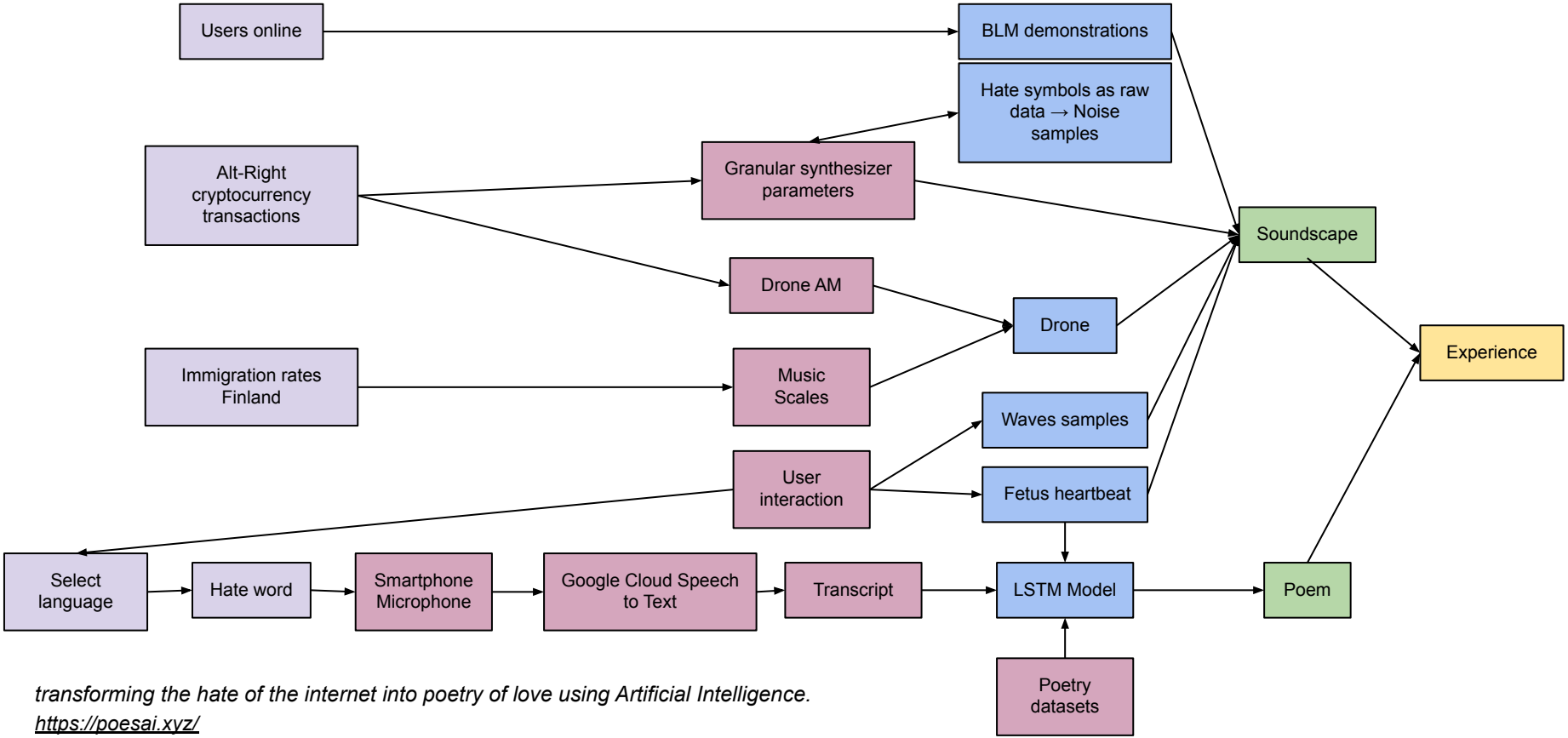
- https://juanalonso.github.io/DDSP-singing-experiments/#dd363fb5-8207-4408-aced-54bf3f7eae99

Bonus:

- SOPI DDSP Material: https://github.com/SopiMlab/DeepLearningWithAudio/tree/master/02_ddsp



{[[(爱 (ài) <3)*],[<3 爱 (ài))*],[(ài) <3 爱)*]]}



transforming the hate of the internet into poetry of love using Artificial Intelligence.
https://poesai.xyz/