ABCI Workshop 2013:
Language Model and Architecture for RSVP-iconCHAT

Karl Wiegand
Northeastern University
Boston, MA USA

January 14, 2013

My Background

- Computer science Ph.D. student
- Natural language processing (NLP)
- Artificial intelligence (AI)
- Applications to augmentative and alternative communication (AAC)

Outline

1. Constraints and approach
2. Interface and demo
3. Language model
4. Current architecture
5. Application to Unlock

Constraints and Approach

Constraints:

1. Single input signal (P300)
2. Icon-based AAC

Approach:

1. Event timer
2. Semantic frames

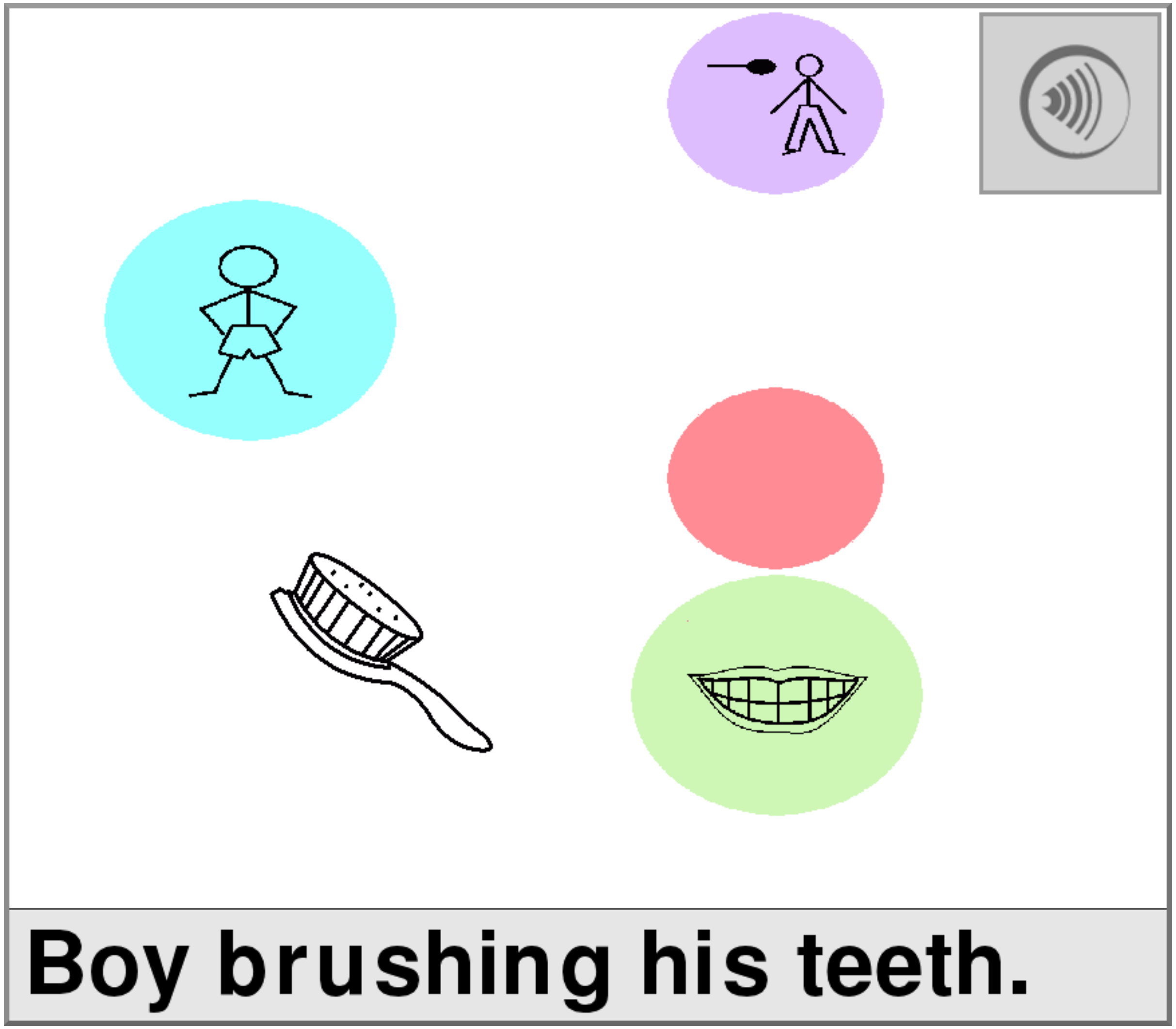
Semantic Frames

- Actions are central to messages (Fillmore, 1976)
- Verbs have "frames" with semantic roles:

Give (*Agent*, *Object*, *Beneficiary*)

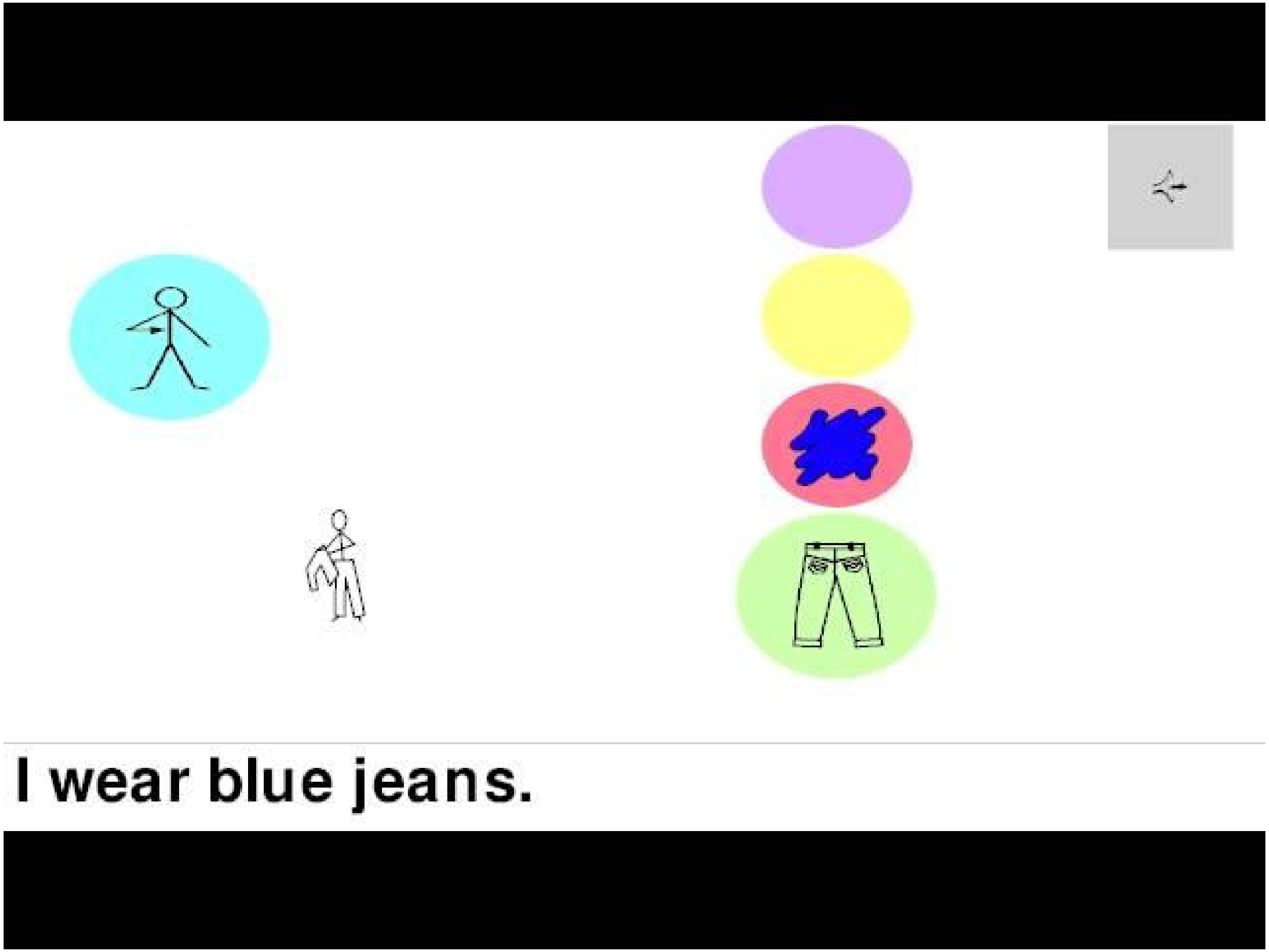
- WordNet, FrameNet, "Read the Web"
- Verb-first message construction (Patel et al, 2004)
- Any order in RSVP-iconCHAT

Interface



Boy brushing his teeth.

Demonstration



[redacted]

I wear blue jeans.

[redacted]

Language Model

- predict(role, state): listof([word, probability])

- Semantic grams (Wiegand and Patel, 2012)

"I like to play chess with my brother."

brother, chess	brother, i	brother, like
brother, play	chess, i	...

brother, chess, i	brother, chess, like
brother, chess, play	chess, i, like
chess, i, play	...

LM Training

1. Choose a corpus:

   "Blog Authorship Corpus"

   "Crowdsourced AAC-Like Corpus"

2. Split sentences and remove stop words

3. Count sentence lengths

4. Stem and count sem-grams

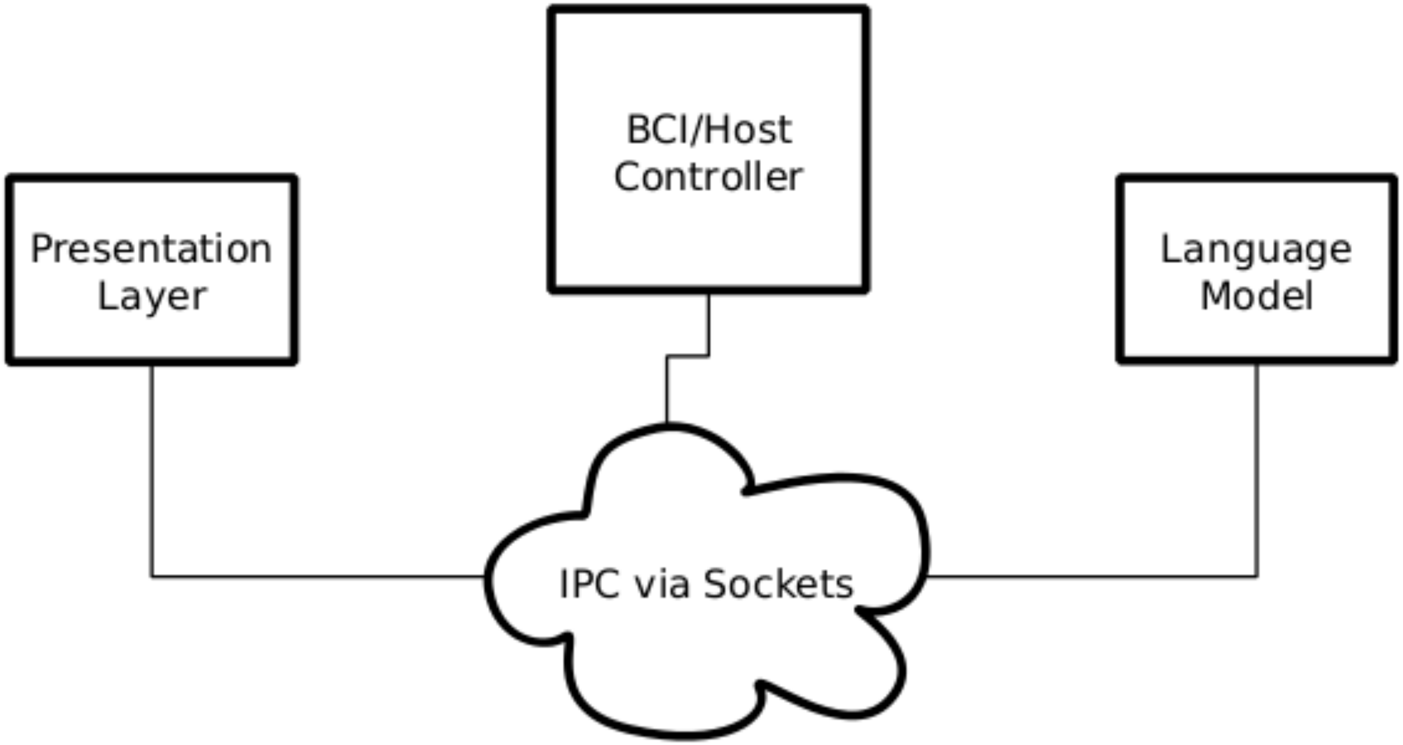
LM Algorithm

1. Tag closed vocabulary with possible roles
2. Select statistics for closed vocabulary
3. Get words from target role
4. Generate sem-grams from current roles
5. Convert sem-gram counts into probabilities
6. Reorder and return

More LM

- Semantic frames have syntactic forms
- First-word prediction is based on 1-grams
- Roles have uniform selection probability
- How do we detect wrong selections of a role? Of a word?

Current Architecture



Architecture Details

BCI/Host Controller ("The Brain")

- Control loop and signal processing

Presentation Layer (Client)

- User interaction -- images and keyboard

Language Model (Client)

- Oval and word prediction
- Semantic selections to syntactic phrase

Runtime Process: LM

1. Cache vocabulary statistics

2. Connect to the host controller

3. Wait for a request header:

   "Oval probabilities" -- None, current_state
   "Icon probabilities" -- oval, current_state
   "Syntactic utterance" -- current_state

Runtime Process: Presentation

1. Connect to host controller

2. Wait for a request header:

   "Start event loop" -- [oval/icon, bitcode]
   "Pause event loop"
   "Stop event loop"
   "Made decision" -- [oval/icon, bitcode]
   "Reset event loop" -- [oval/icon, bitcode]

Runtime Process: Host

1. Initialize gTec hardware
2. Initialize BCI modules
3. Receive connections from Client modules
4. Do:
    a. Query LM for oval probabilities
    b. Reorganize display order of ovals
    c. Send display order to Presentation
    d. Detect P300
    e. Query LM for icon probabilities
    f. Reorganize display order of icons
    g. Send decision to Presentation
    h. Repeat 4a - 4g until user selects Speak...

Runtime Process: Host (cont.)

i. Query LM for syntactic utterance
j. Send utterance to Presentation
k. Reset Presentation
l. Go to 4a...

Project Management

- Git repository on BitBucket
- Task management via Asana
- Schedules in Google Calendar
- Meeting notes in Google Drive
- Code backups, relevant papers, and meeting board photographs in CSLftp

Implementation Details

- IPC is via TCP/IP packets
- Shared network packet structure
- Controller uses Matlab
- Presentation and LM use Python, Twisted, and either Pygame+SDL or Pyglet+OpenGL
- Test Controller uses Python+Kivy+Twisted

Application to Unlock

- P300 design is different than SSVEP design
- Semantic frames:
  - Divide sentences
  - Free order construction
  - Semantic to syntactic mapping
- Semantic grams:
  - Free order prediction
  - Require applicable corpus
- IPC is nice with a standard packet structure

Thank you for listening!

Thanks to Dr. Rupal Patel, Dr. Deniz Erdogmus, and the National Science Foundation (Grant #0914808).