Topic Modeling in Word Prediction

Keith Trnka

University of Delaware
SIGNLP
September 25th, 2006

Who's on the project

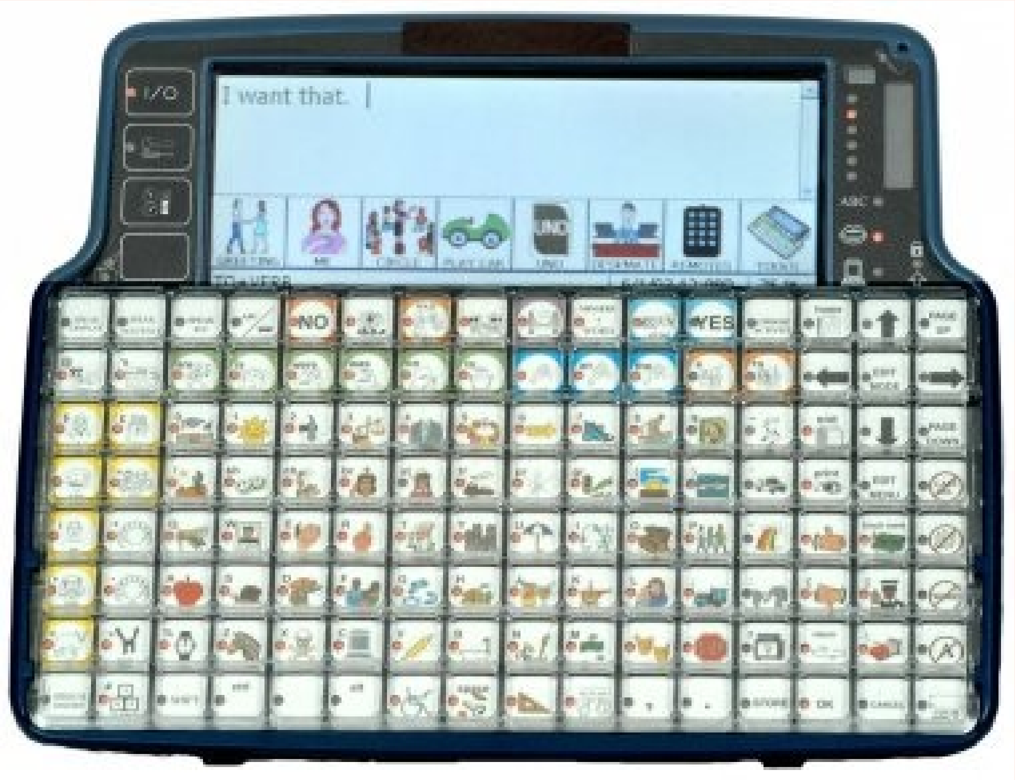
- Grad students
  - Keith Trnka
  - Debra Yarrington
  - John McCaw
- Everyone else
  - Kathy McCoy
  - Christopher Pennington (AgoraNet, Inc)

Augmentative and Alternative Communiction (AAC)

- People with communication disabilities
- Many are unable to speak
- Multiple disabilities common
  - Motor impairment
  - Cognitive impairment

Electronic Solutions

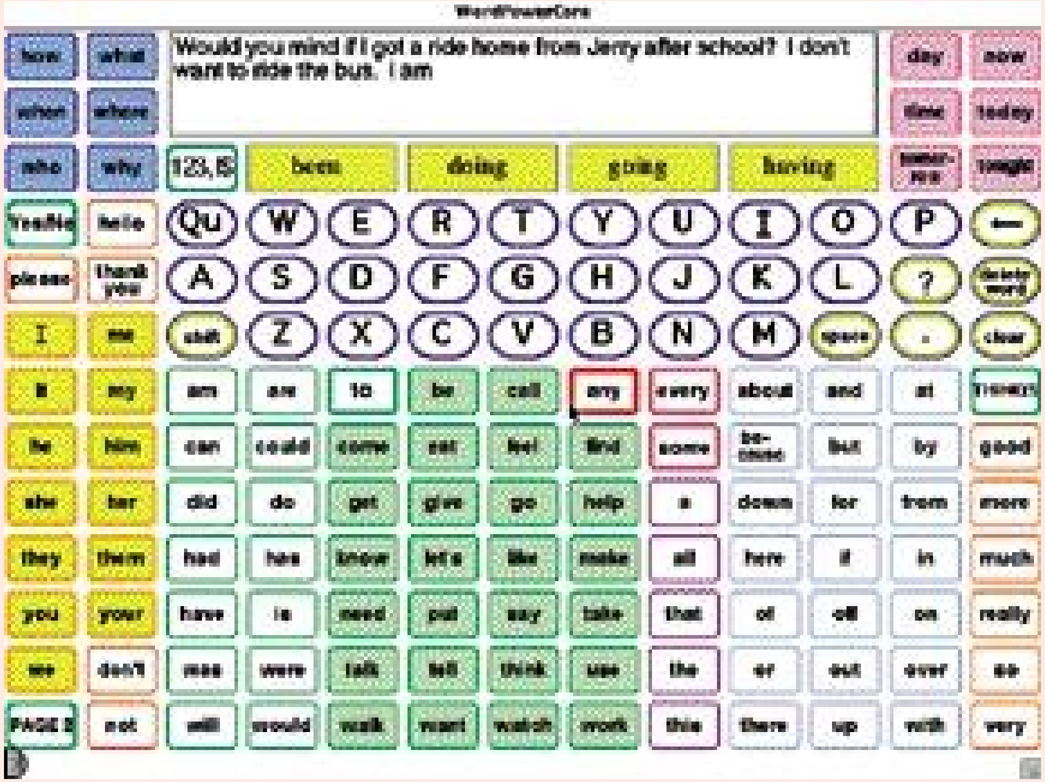
- Common:
  Text entry with speech synthesis



Prentke-Romich Company's *Pathfinder Plus*

Electronic Solutions

- User input – what do they input
  - Letter selection (semi-standard keyboard)
  - Word selection



Electronic Solutions

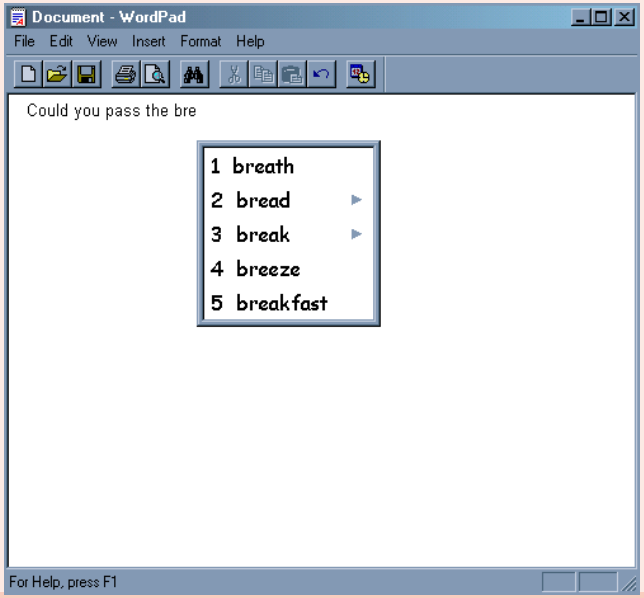
- User input – how do they input
  - Direct selection
    - Relatively fast
  - Row-column scanning
    - Very slow
  - Fingers, hands, head pointer, head motion, eye gaze
    - Varies

Electronic Solutions

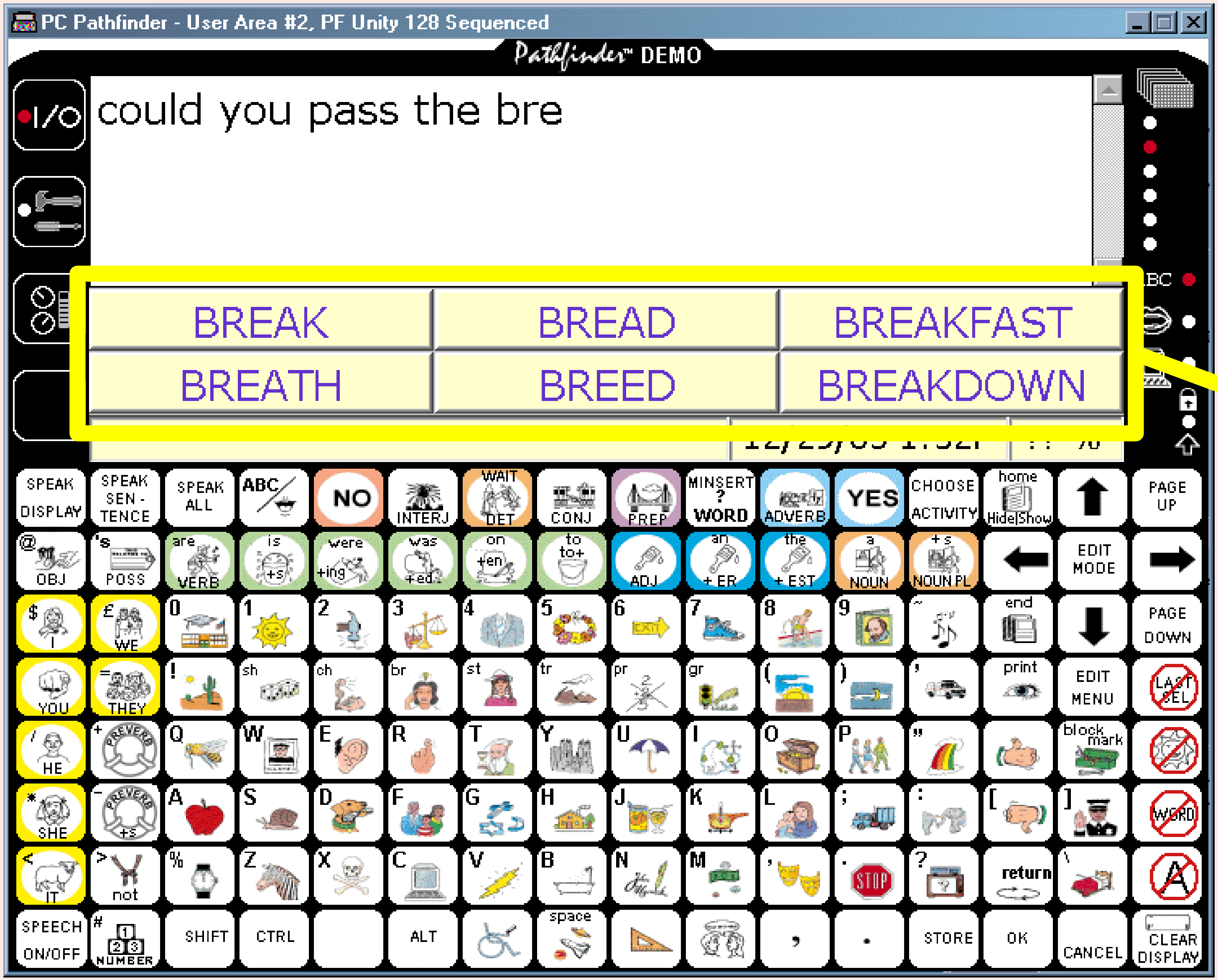
- Making it faster
  - Abbreviation expansion
    - Traditional/written: acronyms, contractions, etc
    - Instant Messaging: brb, afaik, iirc
    - Dynamic: dynmc, dnsr, airpl
    - Practice: abbreviation lists

Electronic Solutions

- Making it faster (cont'd)
  - Buttons/hierarchy for common phrases
  - Buttons for common words
    - Static list of common words – *core vocabulary*
    - Dynamic list of appropriate words – word prediction



Buttons for Common Words

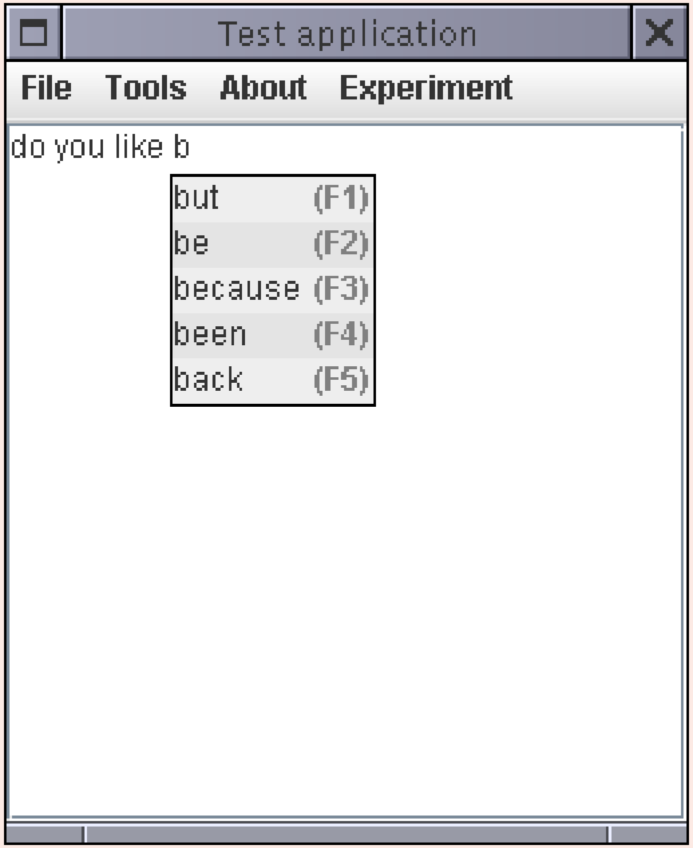


Prediction window (size 6)

Letter entry and core vocabulary via icon sequences

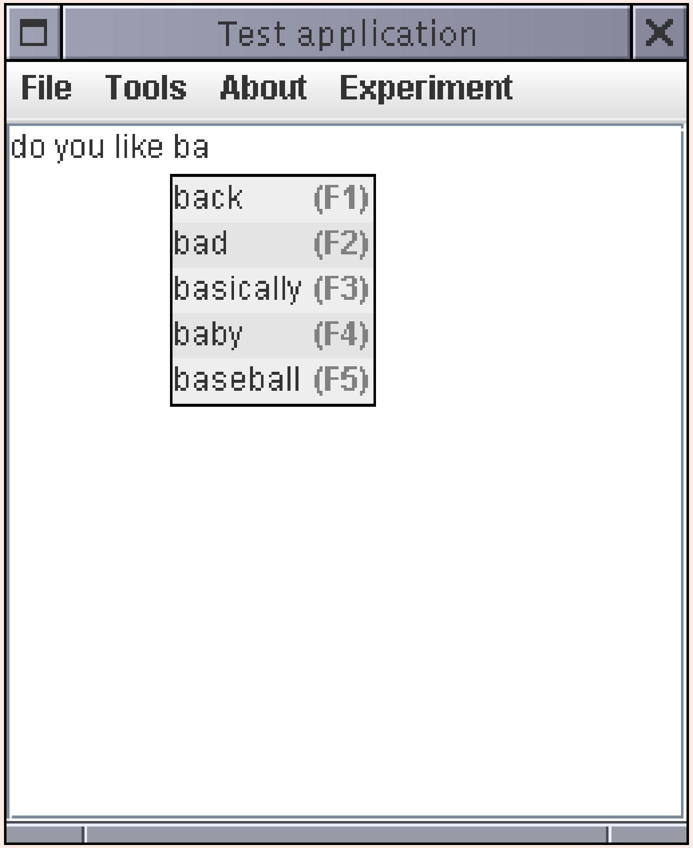
Word Prediction

- Suppose a user is asking a friend “do you like baseball games?”



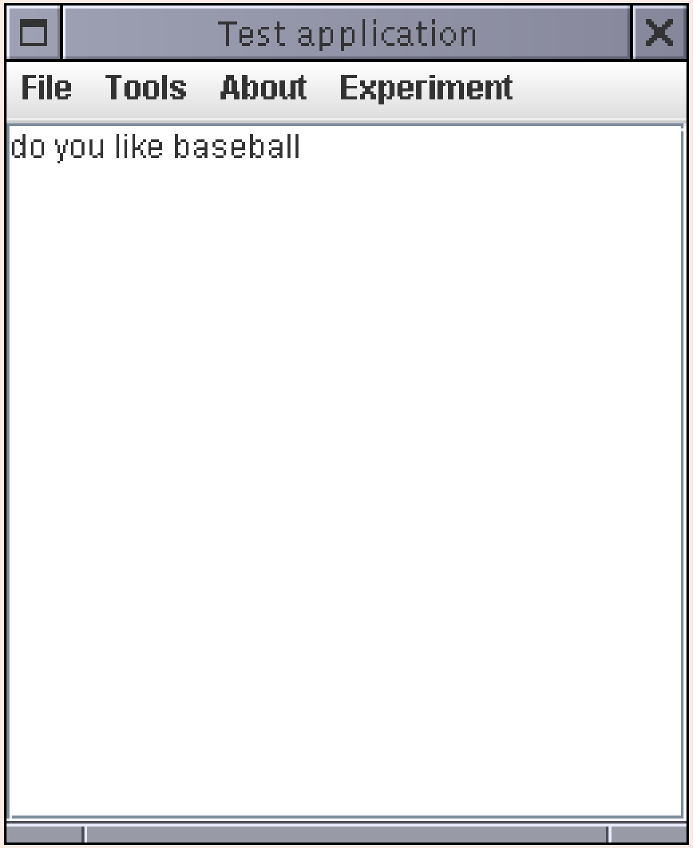
Word Prediction

- User pressed 'a'



Word Prediction

- User pressed 'F5'



Word Prediction

- Advantages
  - Low cognitive effort – very little time to learn
  - Very little screen real estate
  - Doesn't require memorization*
  - Reduces a user interface problem to a NLP problem
  - Can augment a core/fringe split system by focusing on fringe
  - Can speed communication rate significantly

Word Prediction

- Disadvantages
  - Requires same-domain training data to perform well
  - Requires some perceptual effort – distractions

Word Prediction

- Practical issues
  - Number of words to predict
    - 5-7 is common
  - Placement and orientation of the predictions
    - Vertical lists are easier to glance at quickly
    - Horizontal lists can be placed between the keyboard and editing area
    - Depends on the rest of the GUI

Word Prediction

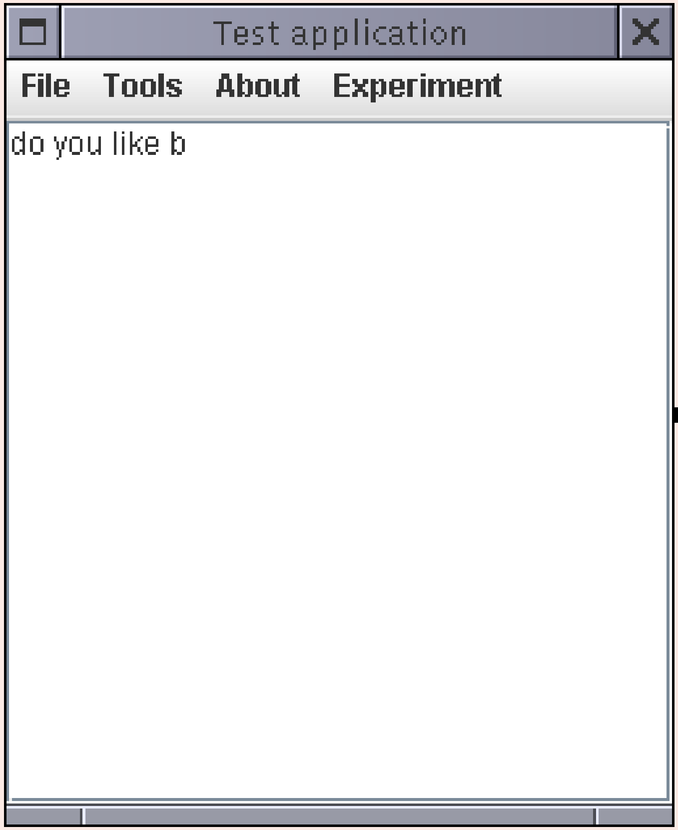
- Practical issues (cont'd)
  - Static vs. dynamic language model
    - Static – the language model doesn't adapt to the user
      - Lower cognitive demands
      - Allows for some memorization
    - Dynamic – the language model adapts to what the user types
      - Doesn't always allow for memorization
      - Higher computational demands (generally)
      - Often better predictions

Word Prediction

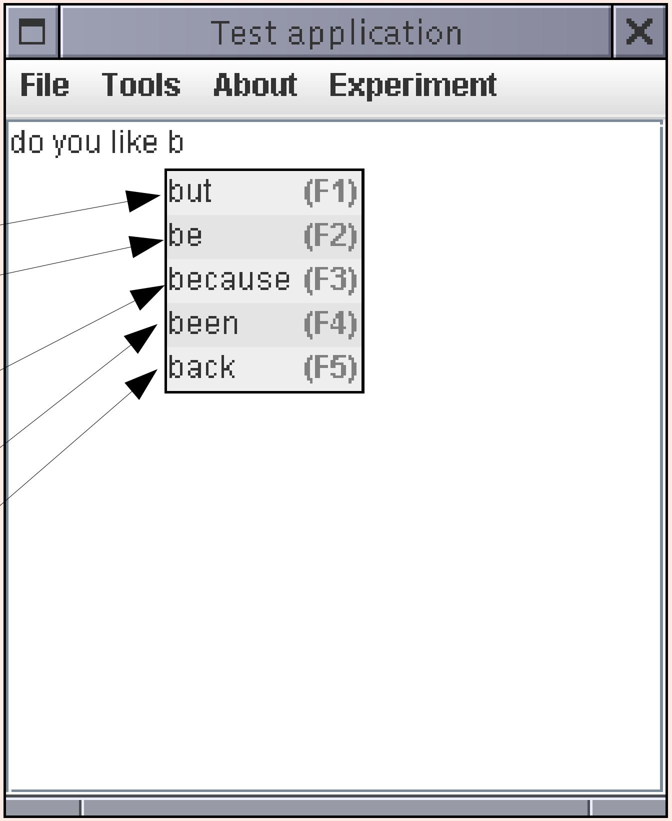
- Research issues
  - Company-researcher stances on word prediction
  - Core vs. Fringe vocabulary
    - Core vocabulary is company-specific
    - Fringe words are the rest of the vocabulary, often typed out using word prediction

Language Modeling for AAC

- A language model is used to generate the predictions

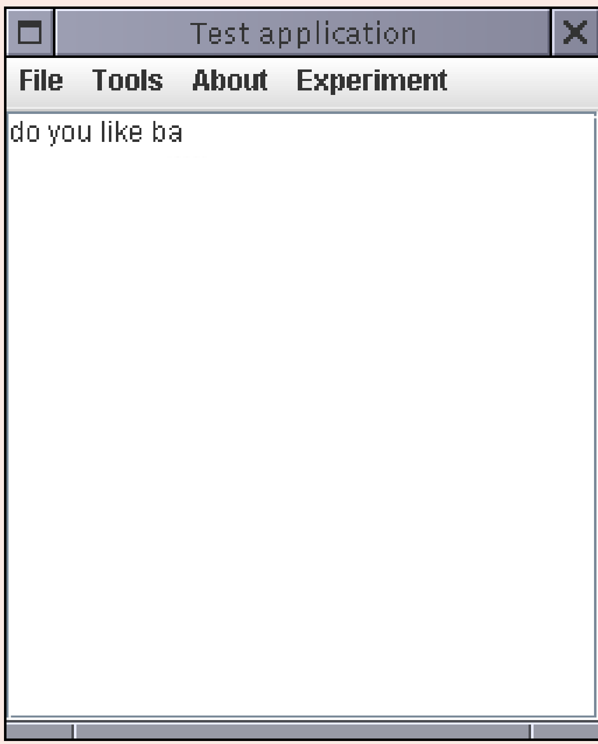


Word	$P(w \mid h)$
that	0.01000
a	0.00900
the	0.00850
but	0.00700
be	0.00400
green	0.00390
because	0.00398
ice	0.00350
been	0.00320
back	0.00310
bad	0.00301
basically	0.00270
baby	0.00250
baseball	0.00240

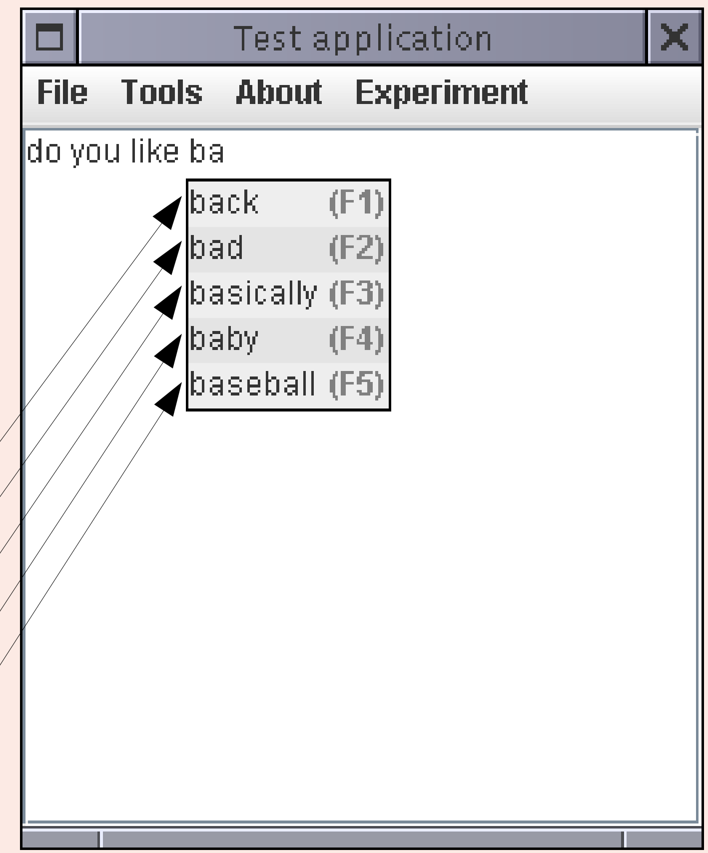


Language Modeling for AAC

- The user presses 'a'



Word	P(w | h)
that	0.01000
a	0.00900
the	0.00850
but	0.00700
be	0.00400
green	0.00390
because	0.00398
ice	0.00350
back	0.00310
bad	0.00301
basically	0.00270
baby	0.00250
baseball	0.00240



Language Modeling for AAC

- Tradition – unigrams and recency/cache
  - Low overlap between the NLP community and AAC community
- Language modeling baseline – trigrams with backoff

Project Goals

- To improve AAC devices by improving the language modeling used in fringe word prediction
- To increase the communication rate of AAC users given a constant rate of input

Evaluation: Keystroke Savings

- Formula

$$KS = \frac{keys_{orig} - keys_{\text{with prediction}}}{keys_{orig}} \times 100\,\%$$

- Issues
  - Do spaces count?
  - Does pressing enter count?
  - How many predictions?
  - Predict words before a letter is pressed or not? (delayed vs. immediate)

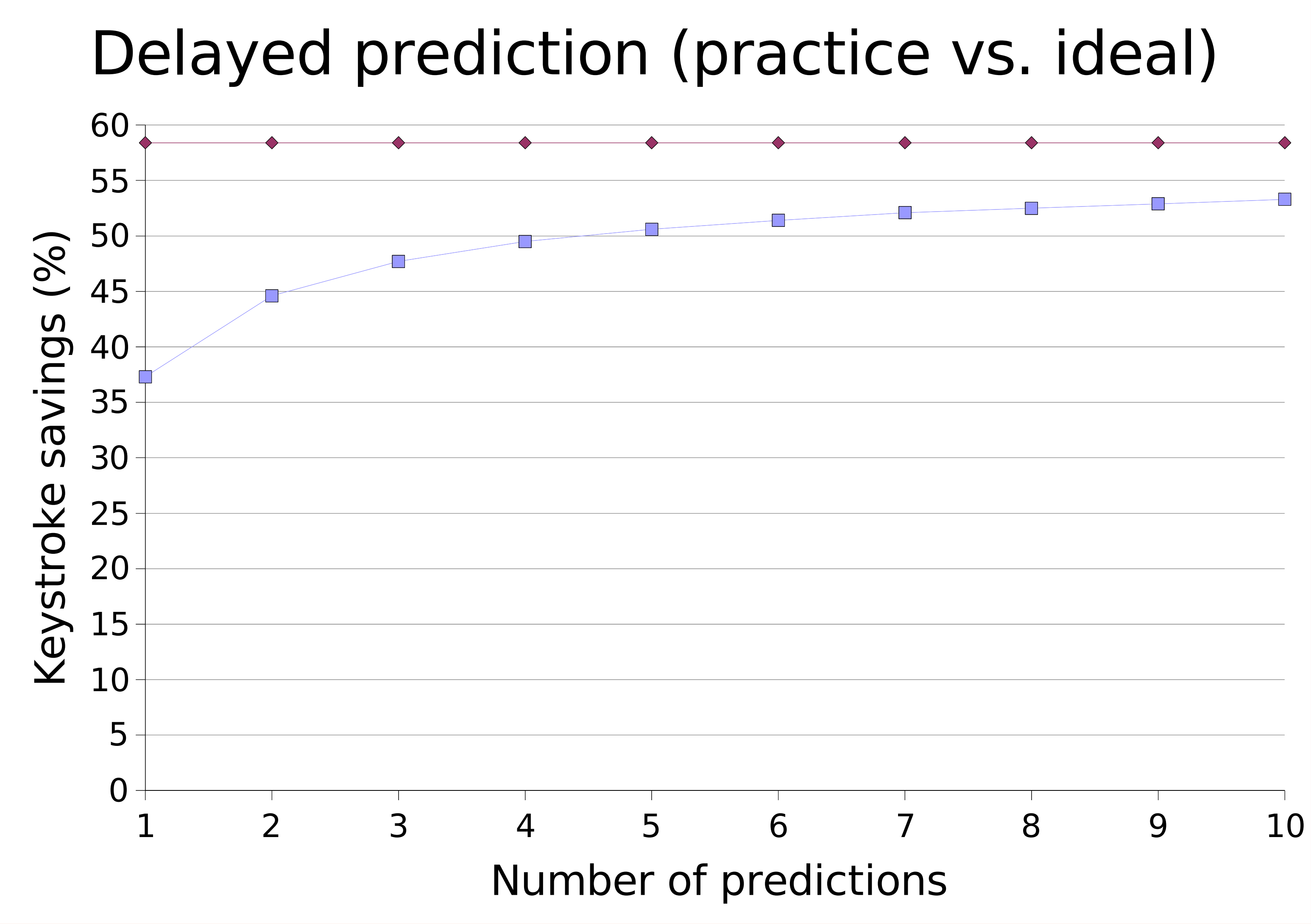
Evaluation: Keystroke Savings

- User simulation
  - A simulated user runs through the software typing the conversation using the fewest number of keystrokes possible
- User interface simulation
  - A space is automatically entered when selecting a predicted word
  - The user can't backspace
- Fringe words only

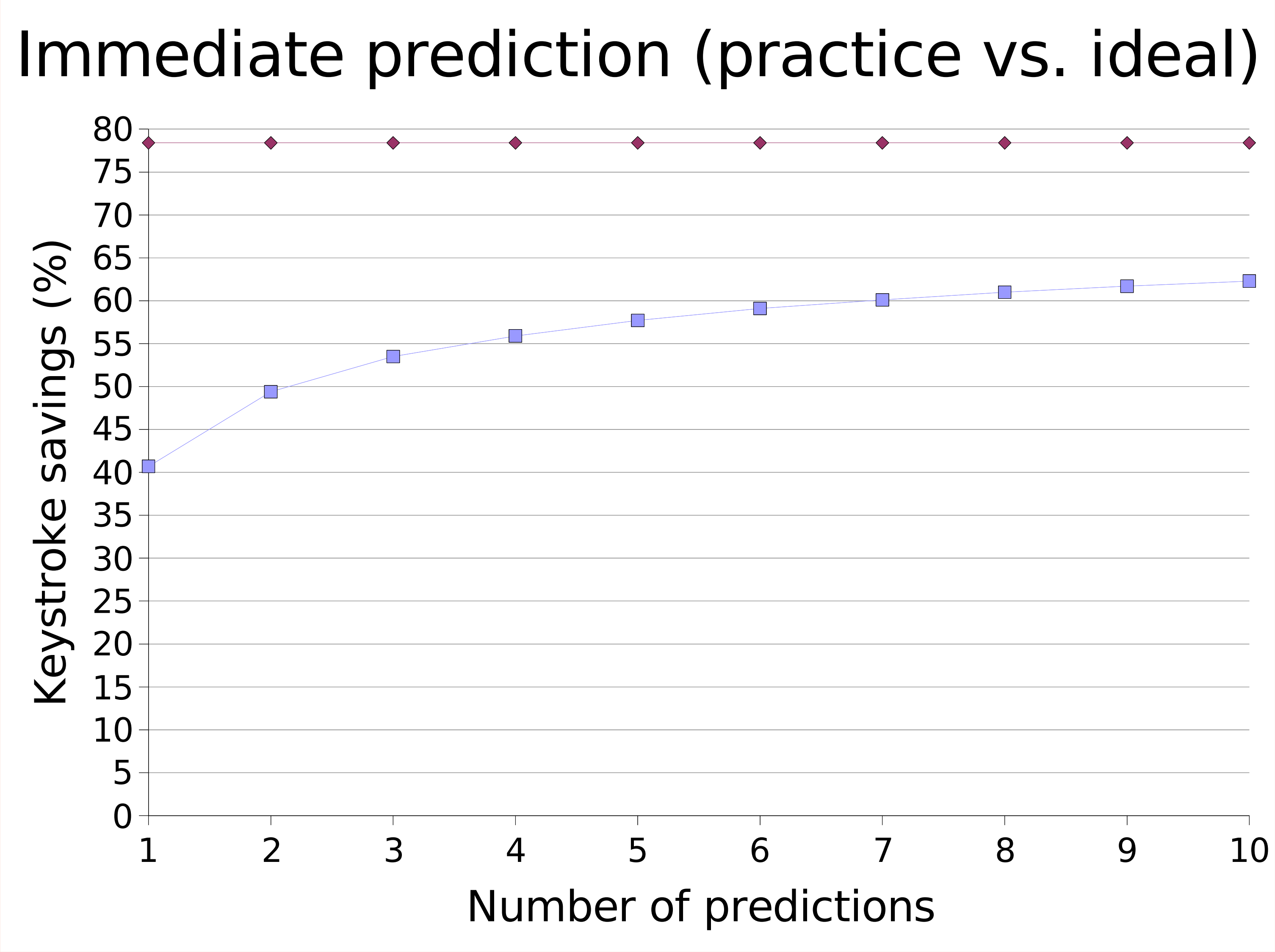
Evaluation – Limits

- Assumption: only single words are predicted
- There is a minimum amount of input required
- ***Delayed prediction*** – requires the first letter be pressed, plus one key to select the word (ideally)
- ***Immediate prediction*** – requires one key to select the word (ideally)
- Both require one key per utterance

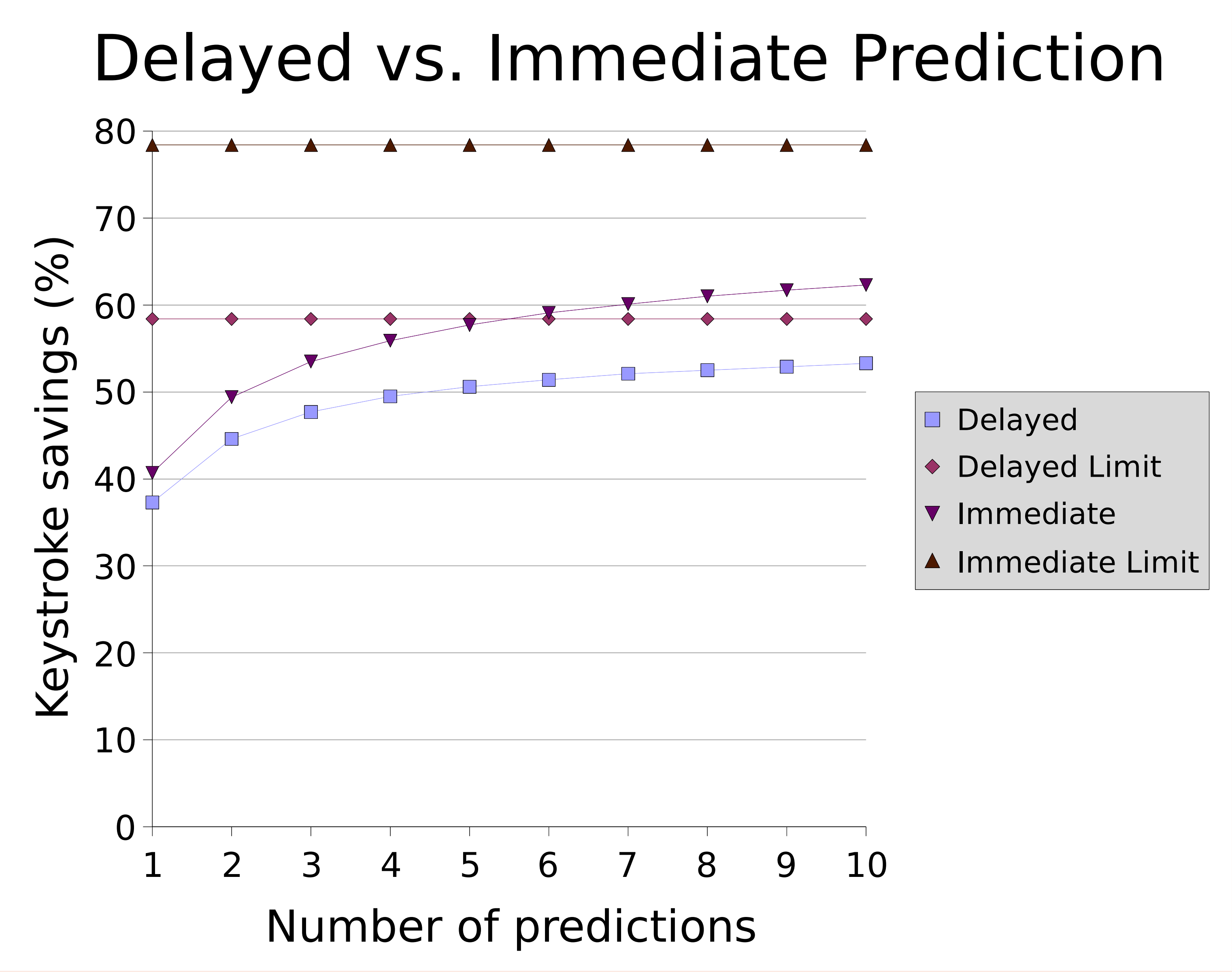
Evaluation – Switchboard Limits



Evaluation – Switchboard Limits



Evaluation – Switchboard Limits



Corpus

- Need a large collection of AAC user text
  - Doesn't exist
- AAC text is conversational
- Switchboard
  - Telephone conversations (transcribed)
  - ~3 million words, ~2,500 conversations
  - Preprocessing/cleanup example
    - Before: is there um an- is there [cough] a code of dress
    - After: is there a code of dress

Development of a Baseline

- Trigrams with backoff commonly accepted
- Original baseline: trigrams with custom backoff
  - Good-Turing smoothing or Witten-Bell for each conditional frequency distribution
- Ngram pruning
  - After hand tuning, no performance increase, but 12.6% decrease in language model size

Development of a Baseline

- Dictionary backoff
  - Improved KS from 57.7% to 57.8% (all words, W=5)
- Improved smoothing
  - Improved KS from 58.8% to 59.0% (fringe words, W=5)
  - No change at W=5 for all words
  - Katz backoff and new approximations close, but approximations are suitable for topic modeling

Topic Modeling

- Goal – adapt a language model to the topic of conversation
  - Boost probabilities of on-topic words
  - Depress probabilities of off-topic words
- Overview
  - Requires a corpus segmented by topic
  - Determine the topic of conversation based on what has been said so far
  - Create a language model for the current topic

Topic Representation

- In training – a collection of text, split by topic

Switchboard

Topic 1
dress code

wear
shirt
pants
suit
tie
jeans

Topic 2
air pollution

smog
Los Angeles
cars
coal
aerosol

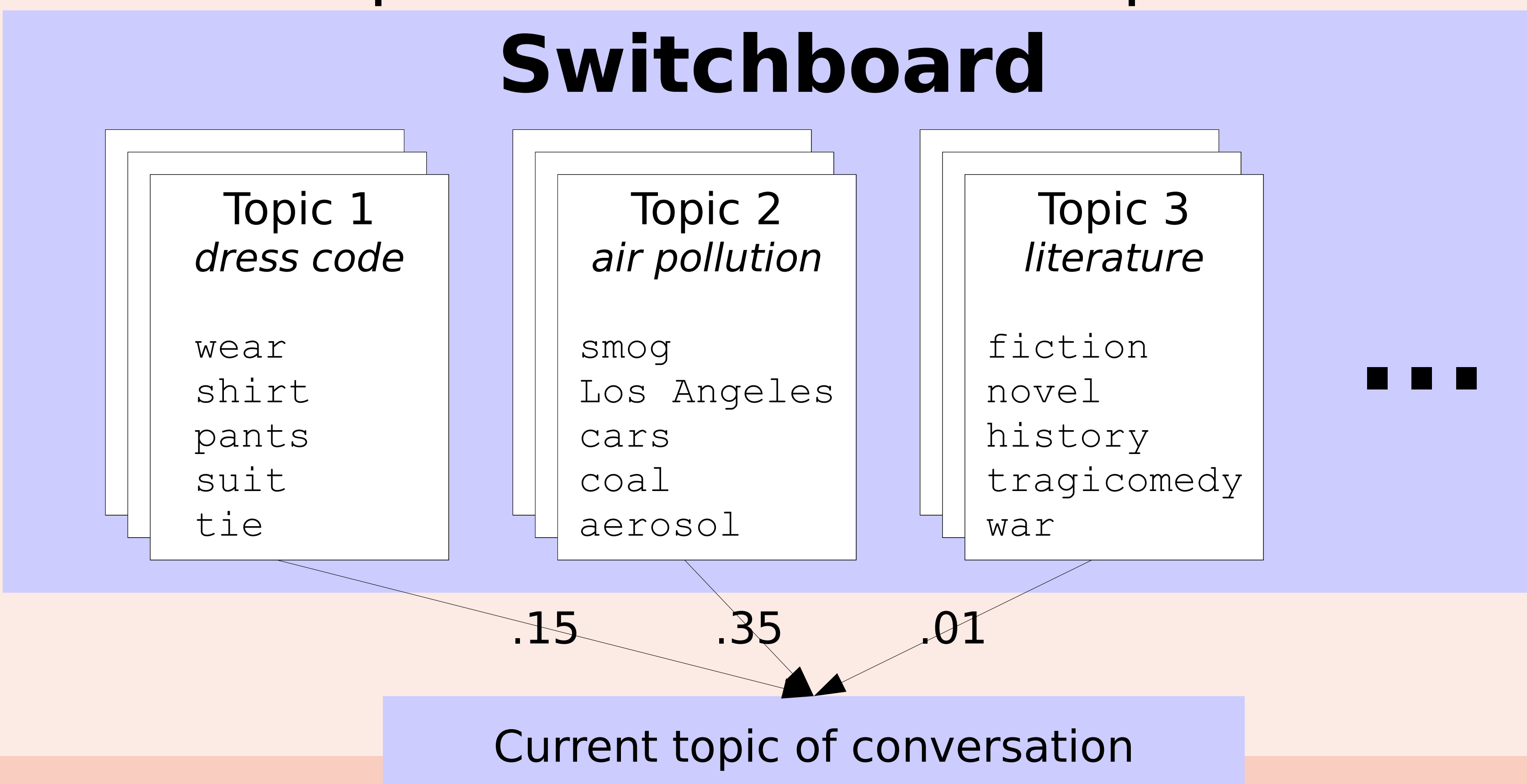
Topic 3
literature

fiction
novel
history
tragicomedy
war
romance

...

Topic Representation

- In testing – a mapping of training topics to weights
  - the compositional nature of topics



Topic Identification

- Cache representation
  - TF-IDF values
  - Exponential decay
    - multiply all weights by 0.975 after every update
  - Words with high IDF excluded (in 85%+ of documents)
  - IDF is really ITF – Inverse Topic Frequency

Topic Identification

Conversation 2001 – in progress

B: okay hi
A: hi yeah i'd like to talk about how you dress for work and what do you normally what type of outfit do you normally have to wear
B: well i work in corporate control so we have to dress kind of nice so i usually wear skirts and sweaters in the winter time slacks i guess and in the summer just dresses
A: um-hum

Topic Identification

Conversation 2001 – in progress

B: okay hi
A: hi yeah i'd like to talk about how you <u>dress</u> for <u>work</u> and what do you normally what type of <u>outfit</u> do you normally have to <u>wear</u>
B: well i work in <u>corporate</u> <u>control</u> so we have to <u>dress</u> kind of nice so i usually <u>wear</u> <u>skirts</u> and <u>sweaters</u> in the <u>winter</u> <u>time</u> <u>slacks</u> i guess and in the <u>summer</u> just <u>dresses</u>
A: um-hum

Word	**Weight**
sweaters	2.59
slacks	2.34
skirts	2.33
dresses	2.30
dress	1.85
outfit	1.70
hi	1.69
wear	1.09
corporate	1.00
winter	0.54
normally	0.47
summer	0.22
control	0.17

Topic Identification

- Similarity scores
  - Compare the cache and unigram models from each topic
  - Cosine similarity
  - Measure a topic's contribution to the final language model

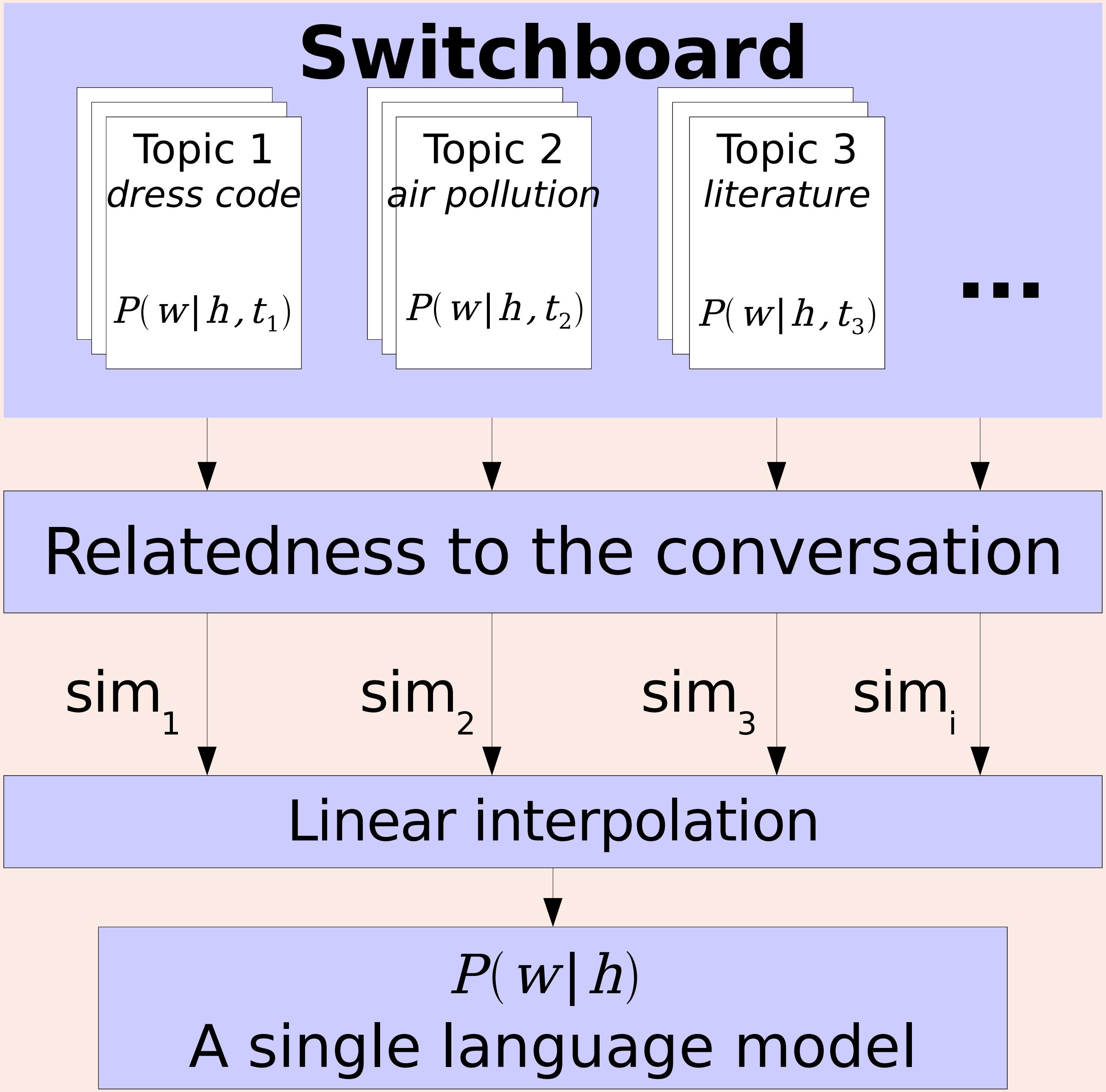
Topic Application

- Linear interpolation of each topic's language model
- General topic modeling equation

$$P(w|h)=\sum_{i\in topics} P(t_i|h)\times P(w|h,t_i)$$

- Topic likelihood approximation

$$P(t|h)\approx\frac{\mathrm{sim}(t,h)}{\sum_t \mathrm{sim}(t,h)}$$


Switchboard
Topic 1
dress code
$P(w|h,t_1)$
Topic 2
air pollution
$P(w|h,t_2)$
Topic 3
literature
$P(w|h,t_3)$
...
Relatedness to the conversation
sim_1
sim_2
sim_3
sim_i
Linear interpolation
$P(w|h)$
A single language model

Practical Issues

- Re-interpolating the language model can be slow
  - Solution: recompute the model less often, perform smoothing of bigrams on-demand
- Interpolation of frequencies allows for optimization
  - Interpolate frequencies, perform smoothing on-demand
- Smoothing and interpolating
  - Re-scale interpolated frequencies (0.4 impr*)

Two implementations

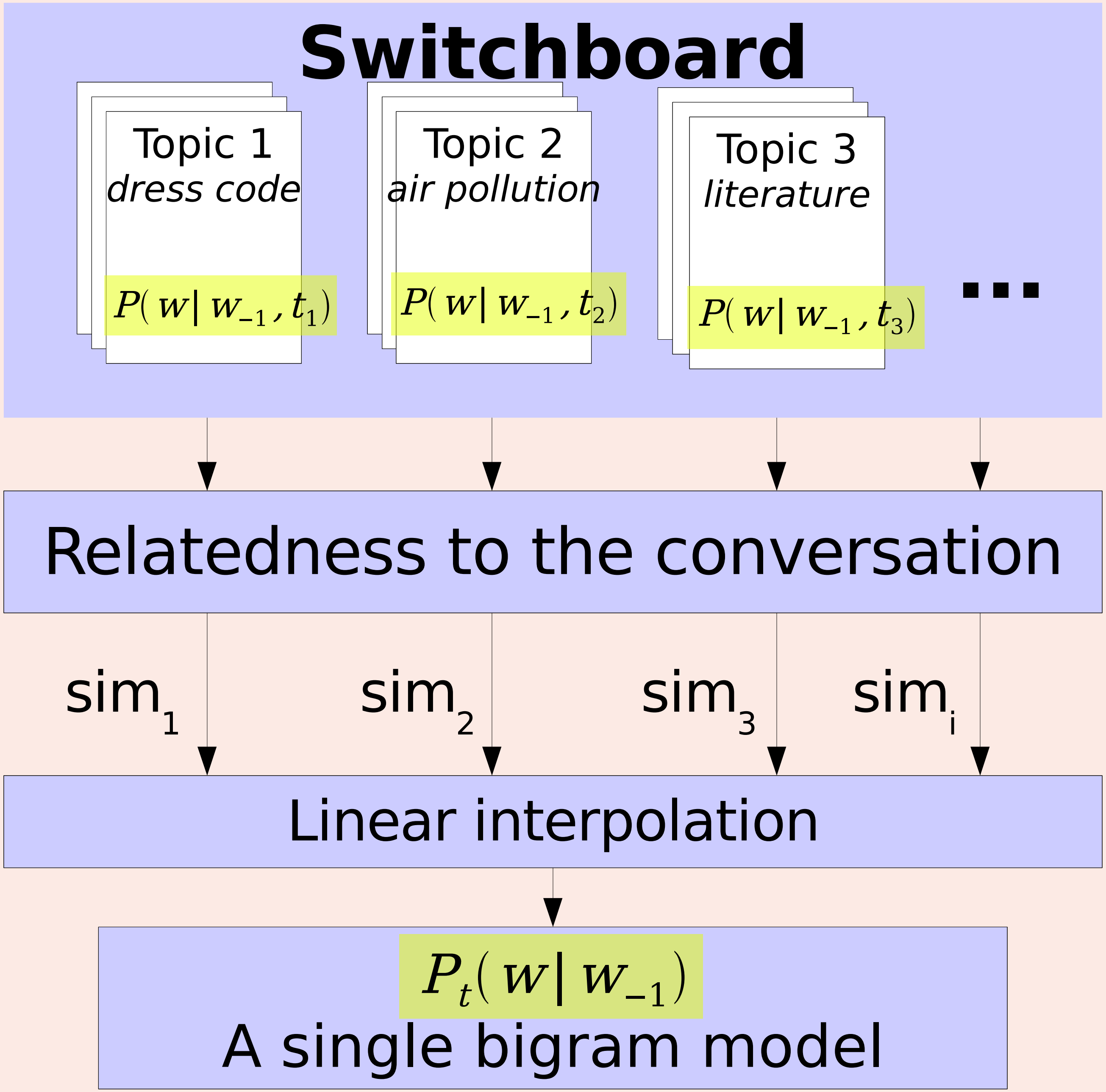
- Each topic model has a full-fledged ngram model
  - For computational reasons, bigrams
  - Method A
- Each topic model has a unigram model
  - Needs to be combined with a topic-independent context-aware model (trigrams)
  - Method B

Method A

- Each topic model is a bigram model
- Frequencies are interpolated and smoothed to probabilities on-demand

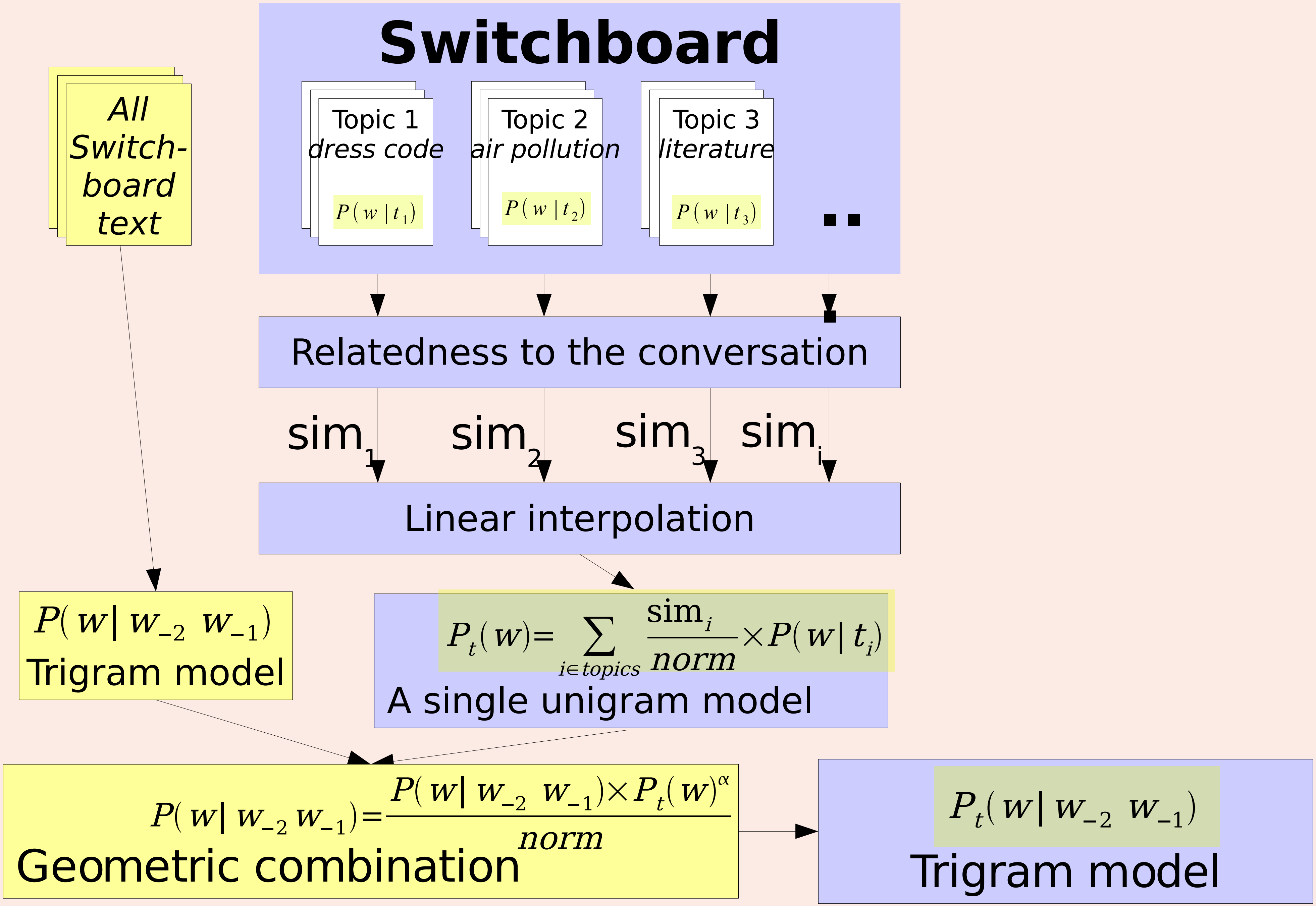
- Approximate equation

$$P(w \mid w_{-1}) = \sum_{i \in topics} \frac{\mathrm{sim}_i}{norm} \times P(w \mid w_{-1}, t_i)$$


Switchboard
Topic 1
dress code
$P(w|w_{-1}, t_1)$
Topic 2
air pollution
$P(w|w_{-1}, t_2)$
Topic 3
literature
$P(w|w_{-1}, t_3)$
...
Relatedness to the conversation
sim_1
sim_2
sim_3
sim_i
Linear interpolation
$P_t(w|w_{-1})$
A single bigram model

Method B

- Each topic model is a unigram model
- Frequencies are interpolated and smoothed when topic similarity is computed
- Geometric combination of topic-dependent and topic-independent parts
  - Following Bellegarda
  - Exponential weight on the topic component, hand-tuned to about 0.15


Switchboard
All Switch-board text
Topic 1 dress code
$P(w \mid t_1)$
Topic 2 air pollution
$P(w \mid t_2)$
Topic 3 literature
$P(w \mid t_3)$
..
Relatedness to the conversation
sim_1
sim_2
sim_3
sim_i
Linear interpolation
$P(w \mid w_{-2}\ w_{-1})$
Trigram model
$P_t(w) = \sum_{i \in topics} \frac{\text{sim}_i}{norm} \times P(w \mid t_i)$
A single unigram model
$P(w \mid w_{-2}\, w_{-1}) = \frac{P(w \mid w_{-2}\ w_{-1}) \times P_t(w)^{\alpha}}{norm}$
Geometric combination
$P_t(w \mid w_{-2}\ w_{-1})$
Trigram model

Comparison of Method A vs. B

- Baseline: 58.8%
- Method A: 60.2%
- Method B: 59.1%

- Approximate runtimes
  - Trigram baseline: 1,325 wpm
  - Method A: 32 wpm
  - Method B: 1,267 wpm

Improving Method A

- What if we treat each document as a topic?

Improving Method A

- What if we treat each document as a topic?
- Should we take all documents into account or only the most similar?
  - K-nn and All-nn

What is a good value for K?

- Early evaluation (fringe, W=5):
  - 250: 58.1%
  - 500: 59.0%
  - 750: 59.3%
  - 1000: 59.4%
  - **1250**: 59.4%
  - 1500: 59.4%
  - 1750: 59.4%
  - 2217 (all): 59.3%

What is a good value for K?

- 1250-nn has the best keystroke savings at all window sizes (1-10)
- Slight trend of smaller neighbors giving better performance at lower W
- Slight trend of larger neighbors giving better performance at high W

Stemming for Similarity

- Intuition
  If the similarity metric were good, all-nn should outperform k-nn for k < 2217
- Porter's stemmer on topic unigram models as well as the cache
- It may improve “true topic” Method A as well

Stemming for Similarity

- Fringe, W=5
- Without stemming
  - All-nn: 59.8%
  - True topic: 60.2%
- With stemming
  - All-nn: 60.0%
  - True topic: 60.1%
  - 1250-nn: 59.8%

Stemming for Similarity

- Desired trend with knn is shown!
- Stemming hurts true topic... why?

Glimpse of the Future

- Topic is one of many things an ngram model ignores
- What if we modeled formality, users, verb tense, agreement, and other things?
- How would we combine such language models and knowledge sources?
- How would we evaluate a combination model?

Glimpse of the Future

- How would we evaluate a combination model?
  - **Reasonable gold standard**
    What's the best keystroke savings if the desired word appeared as soon as the best of the language models to combine?

Glimpse of the Future

- Preliminary analysis
  - All words, W=5
  - Baseline: 57.67%
  - Recency: 31.52%
  - Method A: 57.76%
  - Method B: 57.91%

  - Reasonable gold standard: 62.52%

Glimpse of the Future

- But do all methods contribute?
  - **win percent** – the percent of words on which each method offers the maximum keystroke savings
  - Baseline: 80.66%
  - Recency: 32.08%
  - Method A: 80.86%
  - Method B: 81.68%

Conclusions

- NLP techniques can be applied to improve AAC devices.
- Bigram and trigram models predict words fairly well.
- Topic modeling improves the prediction of fringe words somewhat.
- Other language modeling improvements are likely to improve keystroke savings in word prediction.

Acknowledgments

- This work has been supported by the U.S. Department of Education Grant number: H133G040051, Field Initiated Development Project from the National Institutes on Disability and Rehabilitation Research.

Questions?