A case for systematic sound symbolism in pragmatics: Corpus evidence

Introduction

This analysis find an optimal rule for predicting the pragmatic action of an upcoming turn based on the previous turn type and the initial phoneme of the current turn. It's based on data from:

> Roberts, S. G., Torreira, F., & Levinson, S. C. (2015). The effects of processing and sequence organization on the timing of turn taking: a corpus study. Frontiers in psychology, 6, 509.

Which is itslef based on the Switchboard corpus:

> Calhoun, S., Carletta, J., Brenier, J. M., Mayo, N., Jurafsky, D., Steedman, M., & Beaver, D. (2010). The NXT-format Switchboard Corpus: a rich resource for investigating the syntax, semantics, pragmatics and prosody of dialogue. Language resources and evaluation, 44(4), 387-419.

It uses decision trees as implemented in the `party` package:

> Torsten Hothorn, Kurt Hornik and Achim Zeileis (2006). Unbiased Recursive Partitioning: A Conditional Inference Framework. Journal of Computational and Graphical Statistics, 15(3), 651–674.

List of variables in the data:

- "id": id
- "time": time in recording
- "file": recording file
- "spkA": speaker id (prev turn)
- "spkB": speaker id (current turn)
- "turnA": Calhoun et al. dialog act types (whole turn, maybe multiple)
- "turnB": Same for B
- "A.PP": Sequence type for final TCU of A
- "B.PP": Same as above for B
- "A.Scat": Sequence category for TCU of A (see Roberts et al. 2015)
- "B.Scat": Same as above for B
- "dialActA2.last": Calhoun et al. dialog act type for final TCU of A's turn.: Content questions are marked 'wh_q'
- "dialActB2.first" Calhoun et al. dialog act type for *first* TCU of B's turn.
- "orthA": Orthographic form of A's turn, words separated by " "
- "orthB": Same as above for B.
- "turnPhonesA": Phonetic transcription of A's turn (phones separated by ".", syllables separated by "_", words separated by" ")
- "turnPhonesB": Same as above for B.
- "gapType": Turn transition type. 4 categories - less than -300ms (overlap), between -300 and 0 ms (overlap), between 0 and 300ms (short gap), more than 300ms (long gap)

Load libraries

```
library(party)
```

Load data

```
d = read.csv("../Data/fto_utt_V6_FOR_AS.csv",stringsAsFactors=F)
d$whq = d$dialActB2.first %in% c("wh_q","open_q")

d$prev.Act = as.factor(d$dialActA2.last)
```

Remove turns within first 5 seconds of the start of the conversation:

```
d = d[d$time > 5000,]
```

Identify the first phoneme of the turn. We ignore some pause markers such as 'ah', 'er', 'ahm', 'hhm', 'ow', 'uw' and 'aa'.

```
d$firstPhone = sapply(d$turnPhonesB,function(X){
  # split into words
  wd = strsplit(X," ")[[1]]
  # take out words we don't want
  words.we.donot.want = c('ah.m','er','ah',
                          'hh.m', 'ow','uw', 'aa ')
  wd = wd[!wd %in% words.we.donot.want]
  # take first word (now that the words we don't want are gone)
  firstWord = wd[1]
  # replace syllable delimiter with phoneme delimiter
  firstWord = gsub("_","\\.",firstWord)
  # split into phonemes
  phones = strsplit(firstWord,"\\.")[[1]]

  phones = phones[!is.na(phones)]

  # return first phoneme
  return(phones[1])
})

# convert to factor (ctree needs this)
d$firstPhone = as.factor(d$firstPhone)
```

Create binary variables for turn types:

```
d$prev.statement = d$prev.Act=='statement'
d$prev.affirm = d$prev.Act=='affirm'
d$prev.abandon = d$prev.Act=='abandon'
d$prev.agree = d$prev.Act=='agree'
d$prev.answer = d$prev.Act=='answer'
```

Identify initiating turns:

```
init_turn = c("yn_q","sum", "yn_decl_q", "wh_q", "open_q",
              "directive", "or", "open", "tag_q")
```

```
d$prev.Act2 = ! d$prev.Act %in% init_turn
no_init = d[!is.na(d$prev.Act2),]
```

Get rid of backchannels and overlaps:

```
#Previous initial pair with backchannels
d$Prev.InitialPair = d$A.PP=="1PP"
d = d[!is.na(d$A.PP) & !is.na(d$B.PP) &
        d$B.PP != "backchannel" &
        d$gapType != "(-300,0]" &
        d$gapType != "(-Inf,-300]",]

d$Prev.InitialPair_noBC = d$A.PP=="1PP"
```

Remove missing data:

```
wx = d[!is.na(d$whq) & !is.na(d$A.PP) &
         !is.na(d$prev.Act) & !is.na(d$firstPhone) &
         d$firstPhone!="NA" & !is.na(d$prev.Act2),]
# Recalculate factor to get rid of excluded phones
wx$firstPhone = factor(as.character(wx$firstPhone))
```

Decision tree

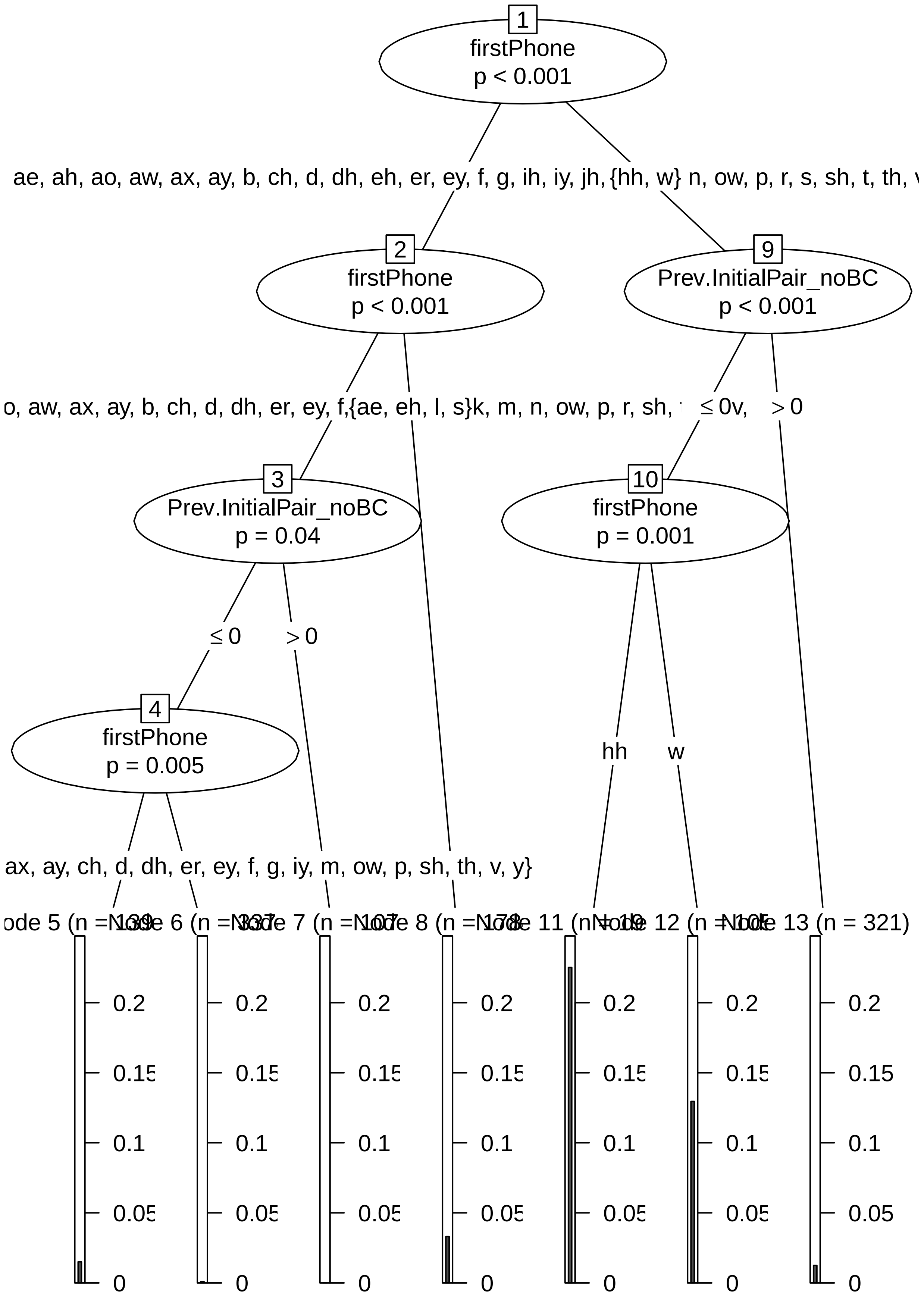
Select data and run tree:

```
Tree.Initial.Pair_noBC = wx[,c("whq",'firstPhone','Prev.InitialPair_noBC')]

Tree.Initial.Pair_noBC.ctree = ctree(whq~.,data=Tree.Initial.Pair_noBC)
```

Plot tree:

```r
plot(Tree.Initial.Pair_noBC.ctree,
     terminal_panel=node_barplot)
```


1
firstPhone
p < 0.001
ae, ah, ao, aw, ax, ay, b, ch, d, dh, eh, er, ey, f, g, ih, iy, jh,
{hh, w}
n, ow, p, r, s, sh, t, th,
2
firstPhone
p < 0.001
9
Prev.InitialPair_noBC
p < 0.001
o, aw, ax, ay, b, ch, d, dh, er, ey, f,
{ae, eh, l, s}
k, m, n, ow, p, r, sh,
≤ 0
v,
> 0
3
Prev.InitialPair_noBC
p = 0.04
10
firstPhone
p = 0.001
≤ 0
> 0
hh
w
4
firstPhone
p = 0.005
ax, ay, ch, d, dh, er, ey, f, g, iy, m, ow, p, sh, th, v, y}
ode 5 (n = 139)
Node 6 (n = 337)
Node 7 (n = 107)
Node 8 (n = 178)
Node 11 (n = 19)
Node 12 (n = 105)
Node 13 (n = 321)
0.2
0.15
0.1
0.05
0

Text representation of tree:

```
ctext = capture.output(print(Tree.Initial.Pair_noBC.ctree))
for(line in ctext){
  if(nchar(line)<60){
    cat(paste0(line))
  } else{
    bits = seq(1, nchar(line), 60)
    lbits = substring(line, bits, c(bits[2:length(bits)],nchar(line)))
    cat(paste(lbits,collapse="\n     "))
  }
  cat("\n")
}
```

```
##
##   Conditional inference tree with 7 terminal nodes
##
## Response:  whq
## Inputs:  firstPhone, Prev.InitialPair_noBC
## Number of observations:  9185
##
## 1) firstPhone == {aa, ae, ah, ao, aw, ax, ay, b, ch, d, dh, e
##      eh, er, ey, f, g, ih, iy, jh, k, l, m, n, ow, p, r, s, sh, t,
##      , th, v, y}; criterion = 1, statistic = 651.675
##   2) firstPhone == {aa, ah, ao, aw, ax, ay, b, ch, d, dh, er,
##      , ey, f, g, ih, iy, jh, k, m, n, ow, p, r, sh, t, th, v, y};
##       criterion = 1, statistic = 140.762
##     3) Prev.InitialPair_noBC <= 0; criterion = 0.96, statisti
##      ic = 43.682
##       4) firstPhone == {b, ih, jh, k, n, r, t}; criterion = 0
##      0.995, statistic = 52.52
##         5)*  weights = 1395
##       4) firstPhone == {aa, ah, ao, aw, ax, ay, ch, d, dh, er
##      r, ey, f, g, iy, m, ow, p, sh, th, v, y}
##         6)*  weights = 3374
##     3) Prev.InitialPair_noBC > 0
##       7)*  weights = 1070
##   2) firstPhone == {ae, eh, l, s}
##     8)*  weights = 1784
## 1) firstPhone == {hh, w}
##   9) Prev.InitialPair_noBC <= 0; criterion = 1, statistic = 4
##      42.79
##     10) firstPhone == {hh}; criterion = 0.999, statistic = 11
##      1.958
##       11)*  weights = 191
##     10) firstPhone == {w}
##       12)*  weights = 1050
##   9) Prev.InitialPair_noBC > 0
##     13)*  weights = 321
```

Work out goodness of fit:

```r
y = (as.numeric(Tree.Initial.Pair_noBC$whq))
rss <- sum(y - predict(Tree.Initial.Pair_noBC.ctree ))^2
tss <- sum((y-mean(y))^2)
r2 <- 1-(rss/tss)

1-rss/tss
```

```
## [1] 1
```

Print to svg for editing.

```r
svg(file='../results/InitialPair_noBC_YES.svg',width=7.5,height=5.5)
plot(Tree.Initial.Pair_noBC.ctree,
     inner_panel=node_inner(Tree.Initial.Pair_noBC.ctree,id=F),
     terminal_panel=node_barplot(Tree.Initial.Pair_noBC.ctree,
                                 id = F))
dev.off()
```

```
## pdf
##   2
```