# Early blindness does not impair the detection of sound symbolic associations in natural language

Marco Barilari, Minah Chang and Roberto Bottini

## Data Ananlysys for the paper xxx (under review)

### Data preparation

- All Distractor trials were removed (4186 values) and not considered for further analyses.

#### Load merged data

- 1. subID - subject id
- 2. group - 'EB' either early or congenital blind, 'SC' sighted controls
- 3. session - it refers to the experiment version , either 'matlab' or 'gorilla'
- 4. category [integers 1, 2, 3, 4, 5, 6, 7]
  - category 1: size
  - category 2: brightness
  - category 3: loudness
  - category 4: velocity
  - category 5: movement
  - category 6: space
  - category 7: distance
- 5. wordcode - it refers to the identifier code of each stimulus, the code structure is speaker's gender + lamguage + xxx
- 6. language - language of origin of the stimulus
- 7. stimulus - sound transcription of the stimulus
- 8. meaning - meaning of the sitmulus refered the general synonim
- 9. value - 'good' or 'good_op', see note below.
- 10. response - participants response as generla synonim
- 11. grade - response coded as correct (1) or incorrect (0) compared to the 'meaning' column

*Note about 'value' header (native accuracy): given that some words in specific native languages groups can "sound" with the opposite meaning as presented in Tzeng et al., (2017), we recompute the accuracy for those stimuli that in our prestudy were mapped systematically to the opposite meaning. For these subset of stimuli, we consider the response correct if it does not match the original meaning and incorrect if it does. The 9th column "value" indicates whether the accurcy should be calculated as the word's meaning (good), or the opposite meaning (good_op). Raw accuracy rate is calculated without taking into account the value of good or goodop.

```r
allData<-read.csv("allData.csv", header = TRUE)
```

## Descriptive analysis

### Compute the over all raw and then native accuracy

```
allData$rawAccuracy <- as.integer(allData$response == allData$meaning)

allData$nativeAccuracy <- ifelse(allData$value == 'goodop',
                                 abs(allData$rawAccuracy - 1),
                                 allData$rawAccuracy + 0)
allData %>%
  group_by(group) %>%
  summarize(native_acc = mean(nativeAccuracy),
            .groups = 'keep')
```

```
## # A tibble: 2 x 2
## # Groups:   group [2]
##   group native_acc
##   <fct>      <dbl>
## 1 EB         0.624
## 2 SC         0.626
```

### Accuracy per subjects and category

```
categoryAccuracySubj_tidy <- allData %>%
  group_by(subID, group, category) %>%
  summarize(nativeAccuracy_mean = mean(nativeAccuracy),
            nativeAccuracy_sd = sd(nativeAccuracy),
            n = n(),
            nativeAccuracy_se = nativeAccuracy_sd / sqrt(n),
            .groups = 'keep')

print(categoryAccuracySubj_tidy)
```

```
## # A tibble: 322 x 7
## # Groups:   subID, group, category [322]
##    subID group category nativeAccuracy_~ nativeAccuracy_~     n nativeAccuracy_~
##    <fct> <fct>    <int>            <dbl>            <dbl> <int>            <dbl>
##   1 EB01  EB              1            0.885            0.323    52           0.0447
##   2 EB01  EB              2            0.708            0.459    48           0.0663
##   3 EB01  EB              3            0.611            0.502    18           0.118
##   4 EB01  EB              4            0.8              0.407    30           0.0743
##   5 EB01  EB              5            0.545            0.504    44           0.0759
##   6 EB01  EB              6            0.639            0.487    36           0.0812
##   7 EB01  EB              7            0.682            0.471    44           0.0710
##   8 EB02  EB              1            0.692            0.466    52           0.0646
##   9 EB02  EB              2            0.5              0.505    48           0.0729
## 10 EB02  EB              3            0.333            0.485    18           0.114
## # ... with 312 more rows
```

### Accuracy per group and category

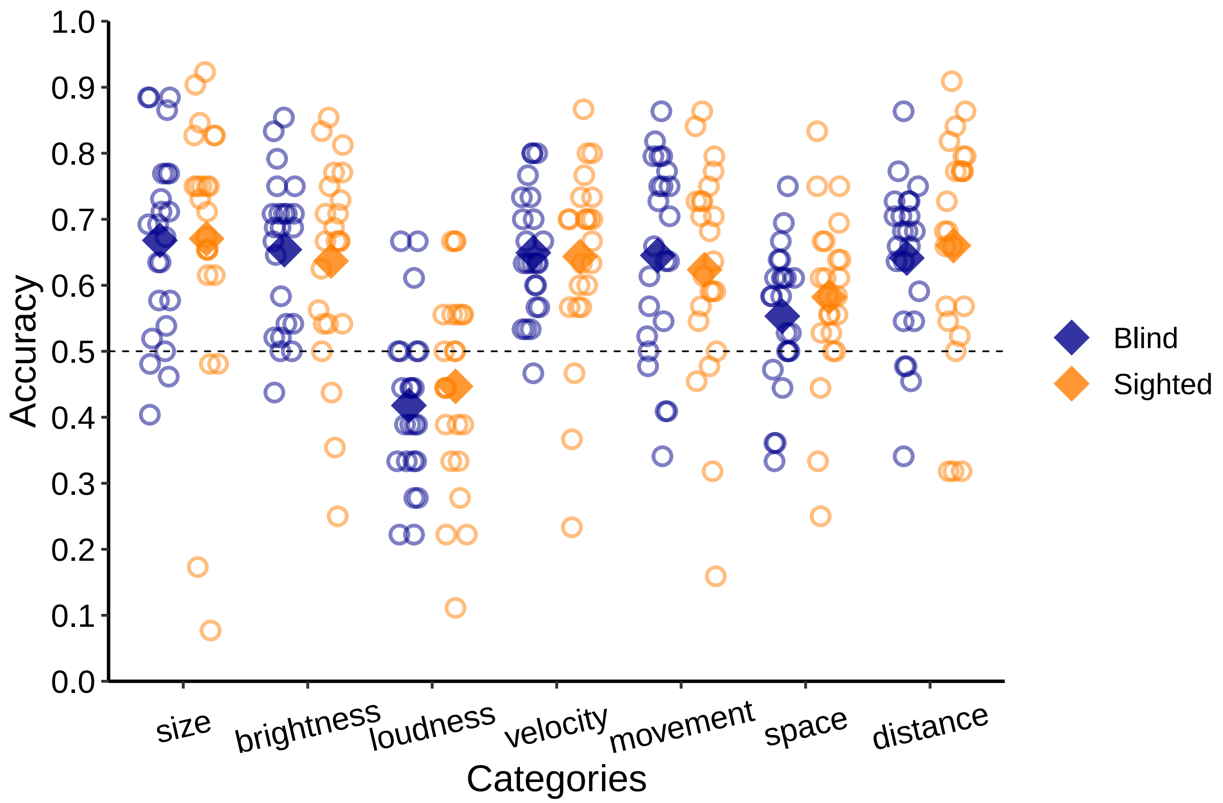
```
categoryAccuracyGroup_tidy <- categoryAccuracySubj_tidy %>%
  group_by(group, category) %>%
  summarize(nativeAccuracy_mean = mean(nativeAccuracy_mean),
            nativeAccuracy_sd = sd(nativeAccuracy_mean),
            n = n(),
            nativeAccuracy_se = nativeAccuracy_sd / sqrt(n),
            .groups = 'keep')

print(categoryAccuracyGroup_tidy)
```

```
## # A tibble: 14 x 6
## # Groups:   group, category [14]
##    group category nativeAccuracy_mean nativeAccuracy_sd     n nativeAccuracy_se
##    <fct>    <int>               <dbl>             <dbl> <int>             <dbl>
##  1 EB           1               0.668                NA    23                NA
##  2 EB           2               0.654                NA    23                NA
##  3 EB           3               0.418                NA    23                NA
##  4 EB           4               0.649                NA    23                NA
##  5 EB           5               0.645                NA    23                NA
##  6 EB           6               0.553                NA    23                NA
##  7 EB           7               0.641                NA    23                NA
##  8 SC           1               0.671                NA    23                NA
##  9 SC           2               0.637                NA    23                NA
## 10 SC           3               0.447                NA    23                NA
## 11 SC           4               0.643                NA    23                NA
## 12 SC           5               0.624                NA    23                NA
## 13 SC           6               0.582                NA    23                NA
## 14 SC           7               0.660                NA    23                NA
```

**Plot of the native accuracy per category.**

```
ggplot() +
  geom_jitter(data = categoryAccuracySubj_tidy,
              aes(x = as.factor(category),
                  y = nativeAccuracy_mean,
                  color = group),
              position = position_jitterdodge(),
              shape = 1,
              size = 2.5,
              stroke = 1,
              alpha = .5,
              show.legend = F) +
  geom_errorbar(data = categoryAccuracyGroup_tidy,
                aes(y = nativeAccuracy_mean,
                    x = as.factor(category),
                    ymin = nativeAccuracy_mean - nativeAccuracy_se,
                    ymax = nativeAccuracy_mean + nativeAccuracy_se,
                    color = group),
                width = .08,
                alpha = .6,
                position = position_jitterdodge(0),
                show.legend = F) +
  geom_point(data = categoryAccuracyGroup_tidy,
              aes(y = nativeAccuracy_mean,
                  x = as.factor(category),
                  color = group),
              position = position_jitterdodge(0),
              shape = 18,
              size = 6,
              alpha = .8) +
  theme_classic() +
  scale_color_manual(values=c('darkblue','darkorange1'), labels=c('Blind', 'Sighted')) +
  scale_y_continuous(limits=c(0, 1), breaks=seq(0, 1, 0.10), expand = c(0,0)) +
  scale_x_discrete(limits=c("1", "2", "3", "4", "5", "6", "7"),
                   labels = c("size", "brightness", "loudness", "velocity", "movement", "space", "distan
  labs(x="Categories", y="Accuracy") +
  ggtitle("") +
  theme(
    text=element_text(size=14),
    axis.line = element_line(size = 0.6),
    axis.text.x = element_text(size=12,colour="black",
                               angle = 13,
                               vjust = .5,
                               hjust = 0.5),
    axis.text.y = element_text(size=12,
                               colour='black'),
    legend.title=element_blank())+
  geom_hline(yintercept=c(0.5), linetype="dashed", colour="black", size=0.3)
```



```
# ggsave('categories_accuracy.tiff',
#        device="tiff",
#        units="in",
#         width=7.54,
#         height=4.54,
#         dpi=300)
```

## Statistical analyses

### Prepare the data

```
summary(allData)
```

```
##       subID        group        session        category        wordcode
##  EB01   :  272   EB:6256   gorilla:6800   Min.   :1.000   am1nu1  :   46
##  EB02   :  272   SC:6256   matlab :5712   1st Qu.:2.000   am1nu102:   46
##  EB03   :  272                            Median :4.000   am1nu109:   46
##  EB04   :  272                            Mean   :3.919   am1nu114:   46
##  EB05   :  272                            3rd Qu.:6.000   am1nu117:   46
##  EB06   :  272                            Max.   :7.000   am1nu118:   46
##  (Other):10880                                            (Other) :12236
##         language        stimulus        meaning        value         response
##  mandarin  :2024   xia     :   92   big    :1196   good  :7912   small  :1281
##  korean    :1610   a cobori:   46   dark   :1196   goodop:4600   dark   :1123
##  turkish   :1518   a le    :   46   small  :1196                 big    :1111
##  tamil     :1380   aasu    :   46   near   :1058                 bright :1085
##  indonesian:1334   acaru   :   46   bright :1012                 moving :1046
##  romanian  :1242   acele   :   46   moving :1012                 far    :1021
##  (Other)   :3404   (Other) :12190   (Other):5842                 (Other):5845
##       grade         rawAccuracy     nativeAccuracy
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000   Median :1.0000   Median :1.0000
##  Mean   :0.6248   Mean   :0.5624   Mean   :0.6248
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##
```

```
allData$subID <- factor(allData$subID)
allData$category <- factor(allData$category)
allData$group <- factor(allData$group)
allData$session <- factor(allData$session)
allData$language <- factor(allData$language)
allData$stimulus <- factor(allData$stimulus)
```

### Groups accuracy agaisnt chance

```
native_EB <- subset(allData, group == 'EB')
binom.test(sum(native_EB$grade), nrow(native_EB), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(native_EB$grade) and nrow(native_EB)
## number of successes = 3902, number of trials = 6256, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
```

```
##  0.6115812 0.6357458
## sample estimates:
## probability of success
##              0.6237212
```

```
native_SC <- subset(allData, group == 'SC')
binom.test(sum(native_SC$grade), nrow(native_SC), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(native_SC$grade) and nrow(native_SC)
## number of successes = 3915, number of trials = 6256, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6136715 0.6378096
## sample estimates:
## probability of success
##              0.6257992
```

### Accuracy per category against chance

### Category 1: size

```
catOne_EB <- subset(allData, category == 1 & group == 'EB')
binom.test(sum(catOne_EB$grade), nrow(catOne_EB), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catOne_EB$grade) and nrow(catOne_EB)
## number of successes = 799, number of trials = 1196, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6405655 0.6947258
## sample estimates:
## probability of success
##              0.6680602
```

```
catOne_SC <- subset(allData, category == 1 & group == 'SC')
binom.test(sum(catOne_SC$grade), nrow(catOne_SC), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catOne_SC$grade) and nrow(catOne_SC)
## number of successes = 802, number of trials = 1196, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6431187 0.6971769
## sample estimates:
## probability of success
##              0.6705686
```

### Category 2: brightness

```r
catThree_EB <- subset(allData, category == 2 & group == 'EB')
binom.test(sum(catThree_EB$grade), nrow(catThree_EB), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catThree_EB$grade) and nrow(catThree_EB)
## number of successes = 722, number of trials = 1104, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6250906 0.6820569
## sample estimates:
## probability of success
##              0.6539855
```

```r
catThree_SC <- subset(allData, category == 2 & group == 'SC')
binom.test(sum(catThree_SC$grade), nrow(catThree_SC), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catThree_SC$grade) and nrow(catThree_SC)
## number of successes = 703, number of trials = 1104, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6076187 0.6652007
## sample estimates:
## probability of success
##              0.6367754
```

### Category 3: loudness

```r
catFour_EB <- subset(allData, category == 3 & group == 'EB')
binom.test(sum(catFour_EB$grade), nrow(catFour_EB), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catFour_EB$grade) and nrow(catFour_EB)
## number of successes = 173, number of trials = 414, p-value = 0.0009687
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.3698935 0.4670376
## sample estimates:
## probability of success
##              0.4178744
```

```r
catFour_SC <- subset(allData, category == 3 & group == 'SC')
binom.test(sum(catFour_SC$grade), nrow(catFour_SC), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catFour_SC$grade) and nrow(catFour_SC)
## number of successes = 185, number of trials = 414, p-value = 0.03445
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.3982912 0.4961934
## sample estimates:
## probability of success
##              0.4468599
```

### Category 4: velocity

```
catSix_EB <- subset(allData, category == 4 & group == 'EB')
binom.test(sum(catSix_EB$grade), nrow(catSix_EB), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catSix_EB$grade) and nrow(catSix_EB)
## number of successes = 448, number of trials = 690, p-value = 3.764e-15
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6123655 0.6849022
## sample estimates:
## probability of success
##              0.6492754
```

```
catSix_SC <- subset(allData, category == 4 & group == 'SC')
binom.test(sum(catSix_SC$grade), nrow(catSix_SC), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catSix_SC$grade) and nrow(catSix_SC)
## number of successes = 444, number of trials = 690, p-value = 4.312e-14
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6064614 0.6792620
## sample estimates:
## probability of success
##              0.6434783
```

### Category 5: movement

```
catSeven_EB <- subset(allData, category == 5 & group == 'EB')
binom.test(sum(catSeven_EB$grade), nrow(catSeven_EB), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catSeven_EB$grade) and nrow(catSeven_EB)
## number of successes = 653, number of trials = 1012, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6148969 0.6747690
## sample estimates:
## probability of success
##              0.6452569
```

```
catSeven_SC <- subset(allData, category == 5 & group == 'SC')
binom.test(sum(catSeven_SC$grade), nrow(catSeven_SC), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catSeven_SC$grade) and nrow(catSeven_SC)
## number of successes = 631, number of trials = 1012, p-value = 3.633e-15
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5928484 0.6534663
## sample estimates:
## probability of success
##              0.6235178
```

### Category 6: space

```
catEight_EB <- subset(allData, category == 6 & group == 'EB')
binom.test(sum(catEight_EB$grade), nrow(catEight_EB), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catEight_EB$grade) and nrow(catEight_EB)
## number of successes = 458, number of trials = 828, p-value = 0.002479
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5185339 0.5873668
## sample estimates:
## probability of success
##              0.5531401
```

```
catEight_SC <- subset(allData, category == 6 & group == 'SC')
binom.test(sum(catEight_SC$grade), nrow(catEight_SC), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catEight_SC$grade) and nrow(catEight_SC)
## number of successes = 482, number of trials = 828, p-value = 2.581e-06
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5476847 0.6159799
## sample estimates:
## probability of success
##              0.5821256
```

### Category 7: distance

```
catNine_EB <- subset(allData, category == 7 & group == 'EB')
binom.test(sum(catNine_EB$grade), nrow(catNine_EB), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catNine_EB$grade) and nrow(catNine_EB)
## number of successes = 649, number of trials = 1012, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6108832 0.6709006
## sample estimates:
## probability of success
##              0.6413043
```

```
catNine_SC <- subset(allData, category == 7 & group == 'SC')
binom.test(sum(catNine_SC$grade), nrow(catNine_SC), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(catNine_SC$grade) and nrow(catNine_SC)
## number of successes = 668, number of trials = 1012, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6299678 0.6892556
## sample estimates:
## probability of success
##              0.6600791
```

### Session by Category by Group interaction maximal model

```
nativelModel <- glmer(nativeAccuracy ~ session + category + group + session*category*group +
        (1 + category | subID) + (1 + category | language:stimulus),
      data = allData,
      family = binomial,
      control = glmerControl(optimizer="bobyqa"))
```

```
## Warning in commonArgs(par, fn, control, environment()): maxfun < 10 *
## length(par)^2 is not recommended.

## Warning in commonArgs(par, fn, control, environment()): maxfun < 10 *
## length(par)^2 is not recommended.

## Warning in optwrap(optimizer, devfun, start, rho$lower, control = control, :
## convergence code 1 from bobyqa: bobyqa -- maximum number of function evaluations
## exceeded

## boundary (singular) fit: see help('isSingular')
```

```
summary(nativelModel)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: nativeAccuracy ~ session + category + group + session * category *
##     group + (1 + category | subID) + (1 + category | language:stimulus)
##    Data: allData
## Control: glmerControl(optimizer = "bobyqa")
##
##      AIC      BIC   logLik deviance df.resid
##  15768.1  16392.6  -7800.0  15600.1    12428
##
```

```
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.1440 -1.0133  0.5446  0.7239  3.2797
##
## Random effects:
##  Groups            Name        Variance Std.Dev. Corr
##  language:stimulus (Intercept) 0.2238   0.4731
##                    category2   0.3345   0.5784   -0.79
##                    category3   0.3529   0.5940   -0.79  0.75
##                    category4   0.2886   0.5372   -0.44  0.34  0.44
##                    category5   0.2959   0.5440   -0.85  0.65  0.71  0.20
##                    category6   0.2714   0.5209   -0.67  0.53  0.57  0.29  0.63
##                    category7   0.2057   0.4535   -0.91  0.72  0.74  0.40  0.80
##  subID             (Intercept) 0.6982   0.8356
##                    category2   0.1593   0.3991   -0.94
##                    category3   1.1500   1.0724   -0.96  0.90
##                    category4   0.4006   0.6329   -0.90  0.83  0.92
##                    category5   0.3534   0.5945   -0.68  0.74  0.60  0.42
##                    category6   0.2605   0.5104   -0.96  0.85  0.99  0.91  0.59
##                    category7   0.3140   0.5603   -0.73  0.66  0.57  0.72  0.53
##
##
##
##
##
##
##
##   0.92
##
##
##
##
##
##
##   0.61
## Number of obs: 12512, groups:  language:stimulus, 271; subID, 46
##
## Fixed effects:
##                                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)                       0.69697    0.26571   2.623  0.00871 **
## sessionmatlab                     0.21829    0.37199   0.587  0.55733
## category2                         0.10352    0.19153   0.540  0.58888
## category3                        -0.93127    0.36812  -2.530  0.01141 *
## category4                        -0.08204    0.26168  -0.313  0.75390
## category5                         0.12748    0.23084   0.552  0.58079
## category6                        -0.37566    0.22050  -1.704  0.08844 .
## category7                        -0.24827    0.21866  -1.135  0.25622
## groupSC                           0.25977    0.35778   0.726  0.46780
## sessionmatlab:category2          -0.47260    0.24890  -1.899  0.05760 .
## sessionmatlab:category3          -0.45770    0.50871  -0.900  0.36826
## sessionmatlab:category4          -0.11171    0.33845  -0.330  0.74136
## sessionmatlab:category5          -0.57005    0.31300  -1.821  0.06857 .
## sessionmatlab:category6          -0.42425    0.28881  -1.469  0.14184
## sessionmatlab:category7           0.12617    0.30049   0.420  0.67458
## sessionmatlab:groupSC           -0.55803    0.52978  -1.053  0.29219
## category2:groupSC               -0.23697    0.23987  -0.988  0.32320
## category3:groupSC               -0.31442    0.48871  -0.643  0.51999
## category4:groupSC               -0.08615    0.32553  -0.265  0.79128
## category5:groupSC               -0.38380    0.30048  -1.277  0.20150
## category6:groupSC               -0.18328    0.27758  -0.660  0.50908
## category7:groupSC                0.13438    0.28880   0.465  0.64171
## sessionmatlab:category2:groupSC  0.30251    0.35708   0.847  0.39690
## sessionmatlab:category3:groupSC  0.94370    0.72251   1.306  0.19150
## sessionmatlab:category4:groupSC  0.12643    0.48271   0.262  0.79338
## sessionmatlab:category5:groupSC  0.55671    0.44676   1.246  0.21272
## sessionmatlab:category6:groupSC  0.64856    0.41352   1.568  0.11679
## sessionmatlab:category7:groupSC -0.05194    0.43112  -0.120  0.90410
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Correlation matrix not shown by default, as p = 28 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

```
## optimizer (bobyqa) convergence code: 1 (bobyqa -- maximum number of function evaluations exceeded)
## boundary (singular) fit: see help('isSingular')
## maxfun < 10 * length(par)^2 is not recommended.
```

```
Anova(nativelModel)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: nativeAccuracy
##                          Chisq Df Pr(>Chisq)
## session                 4.3757  1    0.03646 *
## category               40.4008  6  3.799e-07 ***
## group                   0.3677  1    0.54426
## session:category        9.4287  6    0.15086
## session:group           0.0225  1    0.88067
## category:group          3.6683  6    0.72146
## session:category:group  5.5108  6    0.48015
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

## Analyses not considering those words that rquired inversion mapping

```
allDataNoInversion <- subset(allData, value == 'good')

allDataNoInversion$rawAccuracy <- as.integer(allDataNoInversion$response == allDataNoInversion$meaning)

allDataNoInversion$nativeAccuracy <- ifelse(allDataNoInversion$value == 'goodop',
                                 abs(allDataNoInversion$rawAccuracy - 1),
                                 allDataNoInversion$rawAccuracy + 0)
allDataNoInversion %>%
  group_by(group) %>%
  summarize(native_acc = mean(nativeAccuracy),
            .groups = 'keep')
```

```
## # A tibble: 2 x 2
## # Groups:   group [2]
##   group native_acc
##   <fct>      <dbl>
## 1 EB         0.647
## 2 SC         0.649
```

### Groups accuracy against chance

```
native_EB <- subset(allDataNoInversion, group == 'EB')
binom.test(sum(native_EB$grade), nrow(native_EB), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(native_EB$grade) and nrow(native_EB)
## number of successes = 2561, number of trials = 3956, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6322515 0.6622727
## sample estimates:
## probability of success
##              0.6473711
```

```
native_SC <- subset(allDataNoInversion, group == 'SC')
binom.test(sum(native_SC$grade), nrow(native_SC), p = .5)
```

```
##
##  Exact binomial test
##
## data:  sum(native_SC$grade) and nrow(native_SC)
## number of successes = 2566, number of trials = 3956, p-value < 2.2e-16
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6335266 0.6635235
## sample estimates:
## probability of success
##               0.648635
```

### Session by Category by Group interaction maximal model on the subset of words that did not require inversion mapping

```
nativelNoInversionModel <- glmer(nativeAccuracy ~ session + category + group + session*category*group +
        (1 + category | subID) + (1 + category | language:stimulus),
      data = subset(allData, value == 'good'),
      family = binomial,
      control = glmerControl(optimizer="bobyqa"))
```

```
## Warning in commonArgs(par, fn, control, environment()): maxfun < 10 *
## length(par)^2 is not recommended.

## Warning in commonArgs(par, fn, control, environment()): maxfun < 10 *
## length(par)^2 is not recommended.
```

```
## boundary (singular) fit: see help('isSingular')
```

```r
summary(nativelNoInversionModel)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: nativeAccuracy ~ session + category + group + session * category *
##     group + (1 + category | subID) + (1 + category | language:stimulus)
##    Data: subset(allData, value == "good")
## Control: glmerControl(optimizer = "bobyqa")
##
##      AIC      BIC   logLik deviance df.resid
##   9726.2  10312.2  -4779.1   9558.2     7828
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.5740 -0.9855  0.5109  0.6880  3.3217
##
## Random effects:
##  Groups            Name        Variance Std.Dev. Corr
##  language:stimulus (Intercept) 0.22132  0.4704
##                    category2   0.20819  0.4563   -0.58
##                    category3   0.22132  0.4704   -1.00  0.58
##                    category4   0.17742  0.4212   -0.80  0.53  0.80
##                    category5   0.12241  0.3499   -0.83  0.53  0.83  0.70
##                    category6   0.17478  0.4181   -0.68  0.32  0.68  0.55  0.50
##                    category7   0.11568  0.3401   -0.91  0.67  0.91  0.77  0.75
##  subID             (Intercept) 0.73750  0.8588
##                    category2   0.09885  0.3144   -0.83
##                    category3   1.10801  1.0526   -0.96  0.86
##                    category4   0.24703  0.4970   -0.69  0.66  0.80
##                    category5   0.20221  0.4497   -0.62  0.77  0.53  0.05
##                    category6   0.27720  0.5265   -0.90  0.64  0.94  0.72  0.33
##                    category7   0.19172  0.4379   -0.75  0.48  0.56  0.35  0.51
##
##
##
##
##
##
##
##   0.47
##
##
##
##
##
##
##   0.49
## Number of obs: 7912, groups:  language:stimulus, 172; subID, 46
##
## Fixed effects:
##                                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)                       0.76041    0.27663   2.749  0.00598 **
## sessionmatlab                     0.20231    0.38764   0.522  0.60175
## category2                        0.18502    0.20599   0.898  0.36906
## category3                       -1.06032    0.38887  -2.727  0.00640 **
## category4                       -0.16192    0.25087  -0.645  0.51864
## category5                        0.11233    0.23611   0.476  0.63424
## category6                       -0.23984    0.24665  -0.972  0.33085
## category7                       -0.16935    0.21424  -0.790  0.42925
## groupSC                          0.30148    0.37132   0.812  0.41685
## sessionmatlab:category2         -0.38052    0.25685  -1.481  0.13848
## sessionmatlab:category3         -0.27560    0.55372  -0.498  0.61868
## sessionmatlab:category4          0.04413    0.33776   0.131  0.89605
## sessionmatlab:category5         -0.58667    0.31084  -1.887  0.05911 .
## sessionmatlab:category6         -0.57826    0.32349  -1.788  0.07385 .
## sessionmatlab:category7          0.05013    0.28905   0.173  0.86232
## sessionmatlab:groupSC           -0.63262    0.55336  -1.143  0.25295
## category2:groupSC               -0.26949    0.24757  -1.089  0.27635
## category3:groupSC               -0.36045    0.53074  -0.679  0.49705
## category4:groupSC                0.28282    0.32804   0.862  0.38860
## category5:groupSC               -0.31528    0.30077  -1.048  0.29453
## category6:groupSC               -0.31218    0.31052  -1.005  0.31472
## category7:groupSC               -0.05236    0.27702  -0.189  0.85008
## sessionmatlab:category2:groupSC  0.18208    0.37019   0.492  0.62281
## sessionmatlab:category3:groupSC  0.64505    0.78903   0.818  0.41363
## sessionmatlab:category4:groupSC -0.04349    0.49165  -0.088  0.92951
## sessionmatlab:category5:groupSC  0.78503    0.44791   1.753  0.07966 .
## sessionmatlab:category6:groupSC  0.80993    0.46387   1.746  0.08081 .
## sessionmatlab:category7:groupSC -0.01972    0.41552  -0.047  0.96215
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Correlation matrix not shown by default, as p = 28 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

```
## optimizer (bobyqa) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
## maxfun < 10 * length(par)^2 is not recommended.
```

```
Anova(nativelNoInversionModel)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: nativeAccuracy
##                           Chisq Df Pr(>Chisq)
## session                  3.0913  1    0.07871 .
## category                50.1817  6  4.323e-09 ***
## group                    0.0014  1    0.96991
## session:category         5.4997  6    0.48150
## session:group            0.0771  1    0.78123
## category:group           5.1447  6    0.52539
## session:category:group   7.6695  6    0.26333
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```