Attempt to calculate LENA accuracy for ling/nonling

AC

Read data in

```
read.csv("../Derived_Data/result_final_lisa.csv")->metadata
metadata[1,]

csvs=dir("../Raw_Data/its_to_csv/",pattern="csv")

#count_present=NULL
all_segments=NULL
# extract info from csv's
for(thisits in levels(metadata$filename)){
  csv=grep(thisits,csvs)
  #count_present=c(count_present,length(csv))
  if(length(csv)==1){
    #read in full csv
    read.csv(paste0("../Raw_Data/its_to_csv/",csvs[csv]))->thisdat

    #select down to the lines used in zooniverse
    sel_segments_start=metadata[metadata$filename==thisits,"Starttime"]
    sel_segments_dat=thisdat[thisdat$startTime %in% sel_segments_start,]

    #the following is NOT right: we take the data in even if there is a mismatch in start/end time
    if(dim(sel_segments_dat)[1]!=length(sel_segments_start)) print(paste(thisits, csvs[csv], "have line
    if(dim(sel_segments_dat)[1]>0){

     #derive LENA classification at the segment level
    sel_segments_dat$LENA_type=NA
      #if ling info contained, then ling
    sel_segments_dat$LENA_type[is.na(sel_segments_dat$LENA_type) & sel_segments_dat$childUttLen>0 & !is
      #elsif there is crying
    sel_segments_dat$LENA_type[is.na(sel_segments_dat$LENA_type) &  !is.na(sel_segments_dat$endCry1)]<-
     #elsif there is fixed then laughing
    sel_segments_dat$LENA_type[is.na(sel_segments_dat$LENA_type) &  !is.na(sel_segments_dat$endVfx1)]<-
    #else junk
    sel_segments_dat$LENA_type[is.na(sel_segments_dat$LENA_type)]<-"Junk"

    sel_segments_dat=sel_segments_dat[,c("itsId","startTime", "endTime","LENA_type")]

    #combine with lab annotation
    sel_segments_dat=merge(sel_segments_dat,metadata[metadata$filename==thisits,],by.x="startTime",by.y=

    #add to full spreadsheet
    all_segments=rbind(all_segments,sel_segments_dat)
```

```
    }


  } else print(paste(thisits,"not found"))
}
write.csv(all_segments,"../Derived_Data/LENA_type_lab.csv",row.names = F)
```

Correspondence between LENA & lab annotation at the level of segments

Here we look at to what extent LENA and lab annotations match at the level of individual segments. Each data point is one segment (one "vocalization").

```
read.csv("../Derived_Data/LENA_type_lab.csv")->all_segments

## NOOOOOOOTE !!!! REMOVING Num_Agreement < 2
all_segments=all_segments[all_segments$Num_Agreement>=2,]

# create lab column with easier to read correspondance
all_segments$lab<-as.character(all_segments$Major_Choice)
all_segments$lab[all_segments$lab=="Non-canonical syllables"]<-"Non-Canonical"
all_segments$lab[all_segments$lab=="Canonical syllables"]<-"Canonical"
all_segments$lab[all_segments$lab=="Words"]<-"Canonical"  #!!! why didn't I need this before??
all_segments$lab[all_segments$lab %in% c("Don't mark","None")]<-"Junk"

#collapse acros can and non-can
all_segments$lab[all_segments$lab %in% c("Non-Canonical","Canonical")]<-"Can/non-can"


table(all_segments$lab)
```

```
##
## Can/non-can      Crying        Junk    Laughing
##        8502         598        2361         188
```

```
table(all_segments$LENA_type)
```

```
##
## Can/non-can      Crying        Junk    Laughing
##        8108        3028          30         483
```

```
all_segments$lab=factor(all_segments$lab,levels=c("Can/non-can","Laughing","Crying","Junk"))
all_segments$LENA_type=factor(all_segments$LENA_type,levels=c("Can/non-can","Laughing","Crying","Junk"))


mycf=confusionMatrix(all_segments$lab, all_segments$LENA_type, dnn = c("Lab","LENA"))
conf_tab=mycf$table
# this package uses sensitivity & specificity
#Sensitivity=recall
#Specificity=precision
mycf
```

```
## Confusion Matrix and Statistics
##
##              LENA
```

```
## Lab           Can/non-can Laughing Crying Junk
##   Can/non-can        6213      270   2012    7
##   Laughing             82       22     84    0
##   Crying              126        4    467    1
##   Junk               1687      187    465   22
##
## Overall Statistics
##
##                Accuracy : 0.5772
##                  95% CI : (0.5682, 0.5862)
##     No Information Rate : 0.696
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1145
##
##  Mcnemar's Test P-Value : <2e-16
##
## Statistics by Class:
##
##                      Class: Can/non-can Class: Laughing Class: Crying
## Sensitivity                      0.7663        0.045549       0.15423
## Specificity                      0.3536        0.985133       0.98480
## Pos Pred Value                   0.7308        0.117021       0.78094
## Neg Pred Value                   0.3978        0.959777       0.76826
## Prevalence                       0.6960        0.041463       0.25994
## Detection Rate                   0.5334        0.001889       0.04009
## Detection Prevalence             0.7298        0.016139       0.05133
## Balanced Accuracy                0.5599        0.515341       0.56952
##                      Class: Junk
## Sensitivity             0.733333
## Specificity             0.798692
## Pos Pred Value          0.009318
## Neg Pred Value          0.999139
## Prevalence              0.002575
## Detection Rate          0.001889
## Detection Prevalence    0.202678
## Balanced Accuracy       0.766013
```

Precision

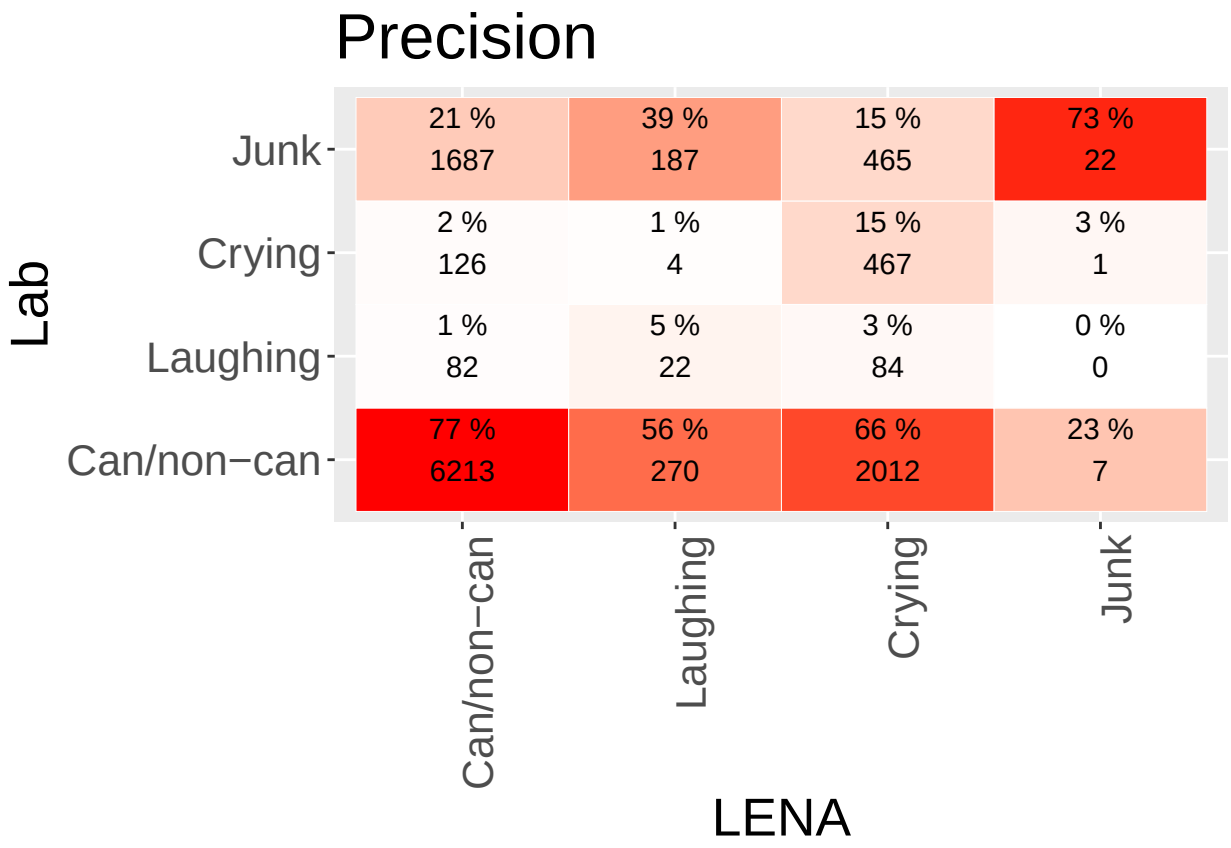
Precision means: If a segment was called X by LENA, what proportion of the time was it called X by lab coders?

```
colsums=colSums(conf_tab)
my_conf_tab=conf_tab
for(i in 1:dim(my_conf_tab)[2]) my_conf_tab[,i]=my_conf_tab[,i]/colsums[i]
colSums(my_conf_tab)
```

```
## Can/non-can    Laughing      Crying        Junk
##           1           1           1           1
```

```
prop_cat=data.frame(my_conf_tab*100) #generates precision because columms
prop_cat$id=paste(prop_cat$Lab,prop_cat$LENA)
colnames(prop_cat)[3]<-"pr"
```

```
data.frame(conf_tab)->stall
stall$id=paste(stall$Lab,stall$LENA)
stall=merge(stall,prop_cat[c("id","pr")])
ggplot(data = stall, mapping = aes(y = Lab, x=LENA)) +
 geom_tile(aes(fill= rescale(pr)), colour = "white") +
  geom_text(aes(label = paste(round(pr),"%")), vjust = -1) +
  geom_text(aes(label = Freq), vjust = 1) +
  scale_fill_gradient(low = "white", high = "red", name = "Proportion") +
     theme(legend.position = "none") +
  xlab("LENA") + ylab("Lab") +
  ggtitle("Precision")+theme(text = element_text(size=20),
        axis.text.x = element_text(angle=90, hjust=1))
```



Recall

Recall means: If a segment was called X by lab coders, what proportion of the time was it called X by LENA coders?

```
prop_cat=data.frame(conf_tab/rowSums(conf_tab)*100)  #generates recall because rows
prop_cat$id=paste(prop_cat$Lab,prop_cat$LENA)
colnames(prop_cat)[3]<-"rec"
data.frame(conf_tab)->stall
stall$id=paste(stall$Lab,stall$LENA)
stall=merge(stall,prop_cat[c("id","rec")])
ggplot(data = stall, mapping = aes(y = Lab, x=LENA)) +
 geom_tile(aes(fill= rescale(rec)), colour = "white") +
```

```
geom_text(aes(label = paste(round(rec),"%")), vjust = -1) +
geom_text(aes(label = Freq), vjust = 1) +
scale_fill_gradient(low = "white", high = "red", name = "Proportion") +
   theme(legend.position = "none") +
xlab("LENA") + ylab("Lab") +
ggtitle("Recall")+theme(text = element_text(size=20),
      axis.text.x = element_text(angle=90, hjust=1))
```