Virtual Patient System Analysis

Han Lie

December 2018

Introduction

This document tries to give insight in the process of assessing the Subject Classification Robustness of the virtual patient. It does so by going through the necessary steps to calculate to what degree the virtual patient system overclassified and underclassified certain subjects with both speech input and transcript input compared to hand annotated input (which can be seen as a ground truth).

Pre-processing

A database query was done to retrieve the frequency for each subject that was discussed in each session.

```
SELECT  virtual_patient.input.keywords,
        COUNT(*) AS 'count'
FROM    virtual_patient.input
GROUP BY virtual_patient.input.keywords
```

Each query result was saved into either speech, transcript or handannotation folder as a .csv file. Each result was named based on its user and session: u followed by the user number, s followed by the session number. For example the csv file of the third session of the second user would get the name u2s3.csv.

All of the csv files were had to be parsed to create a file that was more workable with R. This included removing the headers of the .csv files and creating vectors of the subject names and subject frequencies. As a parser Python was used. The parser can be found in /system-robustness-analysis/parseRawData.py. It can be run by going to the directory and running: python parseRawData.py. Below is the parser code:

```
import csv
import os

for filename in os.listdir('/Users/fooyonghan/Downloads/final-analysis/
    system-robustness-analysis/csv-results/transcript/'):
    if filename.endswith(".csv"):
        output = {}

        with open(os.path.join('/Users/fooyonghan/Downloads/final-analysis/
            system-robustness-analysis/csv-results/transcript/', filename), 'rU') as csv_file:
            csv_reader = csv.reader(csv_file, delimiter='\t')
            for i, line in enumerate(csv_reader):
                row = line[0].split(',')

                if not not row[0]:
                    if i is not 0:
                        times_said = int(row[-1])

                        for item in row:
                            if not item.isdigit():
                                if item not in output:
                                    output[item] = times_said
```

```
                                else:
                                    output[item] += times_said

        newPath = os.path.join('/Users/fooyonghan/Downloads/final-analysis/
          system-robustness-analysis/transcript', filename)

        print newPath
        with open(newPath, 'wb') as f:
            w = csv.DictWriter(f, output.keys())
            w.writeheader()
            w.writerow(output)
        continue

for filename in os.listdir('/Users/fooyonghan/Downloads/final-analysis/
    system-robustness-analysis/csv-results/speech/'):
    if filename.endswith(".csv"):
        output = {}

        with open(os.path.join('/Users/fooyonghan/Downloads/final-analysis/
            system-robustness-analysis/csv-results/speech/', filename), 'rU') as csv_file:
            csv_reader = csv.reader(csv_file, delimiter='\t')
            for i, line in enumerate(csv_reader):
                row = line[0].split(',')

                if not not row[0]:
                    if i is not 0:
                        times_said = int(row[-1])

                        for item in row:
                            if not item.isdigit():
                                if item not in output:
                                    output[item] = times_said
                                else:
                                    output[item] += times_said

        newPath = os.path.join('/Users/fooyonghan/Downloads/final-analysis/
          system-robustness-analysis/speech', filename)

        print newPath
        with open(newPath, 'wb') as f:
            w = csv.DictWriter(f, output.keys())
            w.writeheader()
            w.writerow(output)
        continue

for filename in os.listdir('/Users/fooyonghan/Downloads/final-analysis/
    system-robustness-analysis/csv-results/handannotation/'):
    if filename.endswith(".csv"):
        output = {}

        with open(os.path.join('/Users/fooyonghan/Downloads/final-analysis/
            system-robustness-analysis/csv-results/handannotation/', filename), 'rU') as csv_file:
            csv_reader = csv.reader(csv_file, delimiter='\t')
            for i, line in enumerate(csv_reader):
```

```
                row = line[0].split(',')

                if not not row[0]:
                    if i is not 0:
                        times_said = int(row[-1])

                        for item in row:
                            if not item.isdigit():
                                if item not in output:
                                    output[item] = times_said
                                else:
                                    output[item] += times_said

        newPath = os.path.join('/Users/fooyonghan/Downloads/final-analysis/
          system-robustness-analysis/handannotation', filename)

        print newPath
        with open(newPath, 'wb') as f:
            w = csv.DictWriter(f, output.keys())
            w.writeheader()
            w.writerow(output)
        continue
```

Read all of the speech data into R as a list of lists.

```
folder <- paste("/Users/fooyonghan/Downloads/final-analysis/",
  "system-robustness-analysis/speech", sep="")
fileList <- list.files(path=folder, pattern="*.csv")

speechList <- list()

for (i in 1:length(fileList)){
  id = fileList[i]
  id = gsub(".csv", "", id)
  newPath <- file.path("/Users/fooyonghan/Downloads/final-analysis/",
    "system-robustness-analysis/speech/", id, ".csv", fsep = "")
  speech <- read.csv(file=newPath, header=TRUE, sep=",")
  speechList[[i]] <- speech
}
```

Read all of the transcript data into R as a list of lists.

```
folder <- paste("/Users/fooyonghan/Downloads/final-analysis/",
  "system-robustness-analysis/transcript", sep="")
fileList <- list.files(path=folder, pattern="*.csv")

transcriptList <- list()

for (i in 1:length(fileList)){
  id = fileList[i]
  id = gsub(".csv", "", id)
  newPath <- file.path("/Users/fooyonghan/Downloads/final-analysis/",
    "system-robustness-analysis/transcript/", id, ".csv", fsep = "")
  transcript <- read.csv(file=newPath, header=TRUE, sep=",")
  transcriptList[[i]] <- transcript
}
```

Read all of the handannotation data into R as a list of lists.

```
folder <- paste("/Users/fooyonghan/Downloads/final-analysis/",
  "system-robustness-analysis/handannotation", sep="")
fileList <- list.files(path=folder, pattern="*.csv")

handAnnotationList <- list()

for (i in 1:length(fileList)){
  id = fileList[i]
  id = gsub(".csv", "", id)
  newPath <- file.path("/Users/fooyonghan/Downloads/final-analysis/",
    "system-robustness-analysis/handannotation/", id, ".csv", fsep = "")
  handAnnotation <- read.csv(file=newPath, header=TRUE, sep=",")
  handAnnotationList[[i]] <- handAnnotation
}
```

Create a list of data frames, where each data frame contains speech, transcript and handannotation data. Note that each data frame has a column for all of the subjects detected in that user session for speech, transcript and handannotation. If for example in speech input the subject nightsweat was mentioned, but it was not mentioned in transcript input, the value for transcript in the column nightsweat is marked NA.

```
require(gtools)
systemAnalysisList <- list()

for (i in 1:length(speechList)){
  sessionList <- list(
    a <- speechList[[i]],
    b <- transcriptList[[i]],
    c <- handAnnotationList[[i]]
  )

  # Create a dataframe with all of the types of input and fill non existing values
  # in certain input types with NA
  systemAnalysis <- do.call(function(...) {
    fullList <- plyr::rbind.fill(...)
    rownames(fullList) <- c("speech", "transcript", "handannotation")
    return(fullList)
  }, sessionList)

  systemAnalysisList[[i]] <- systemAnalysis
}
```

Loop through each data frame and calculate the difference between speech and hand annotation, and transcript and hand annotation. This is to assess the difference in the amount of subjects that were classified between the different input types. These difference were then represented as a percentage of the hand annotation as that could be viewed as a ground truth.

```
subjectClassificationRobustness <- data.frame()

for (i in 1:length(systemAnalysisList)){
  systemAnalysis <- systemAnalysisList[[i]]

  # Replace NA values with 0
  systemAnalysis[is.na(systemAnalysis)] <- 0
  groundTruth <- sum(systemAnalysis["handannotation",])
```

```r
# Calculate the difference between speech and handannotation subject classification
diff_hand_speech <- systemAnalysis["handannotation",] - systemAnalysis["speech",]
rownames(diff_hand_speech) <- c("diff_hand_speech")

# Copy differences to new variables for further separate analysis
speech_over_ground_truth <- diff_hand_speech
speech_under_ground_truth <- diff_hand_speech

# For overclassification of speech check where there are negative values.
# Make all other values 0.
speech_over_ground_truth[speech_over_ground_truth>=0] = 0
speech_over_ground_truth <- abs(speech_over_ground_truth)
rownames(speech_over_ground_truth) <- c("speech_over_ground_truth")

# For underclassification of speech check where there are positive values.
# Make all other values 0.
speech_under_ground_truth[speech_under_ground_truth<0] = 0
rownames(speech_under_ground_truth) <- c("speech_under_ground_truth")

# Sum over all of the subjects for both over and under classification
sum_speech_over_ground_truth <- sum(
  speech_over_ground_truth["speech_over_ground_truth",]
)
sum_speech_under_ground_truth <- sum(
  speech_under_ground_truth["speech_under_ground_truth",]
)

# Repeat the same steps for transcript and handannotation
# Calculate the difference between transcript and handannotation classification
diff_hand_transcript <-
  systemAnalysis["handannotation",] - systemAnalysis["transcript",]
rownames(diff_hand_transcript) <- c("diff_hand_transcript")

# Copy differences to new variables for further separate analysis
transcript_over_ground_truth <- diff_hand_transcript
transcript_under_ground_truth <- diff_hand_transcript

# For overclassification of trancripts check where there are negative values.
# Make all other values 0.
transcript_over_ground_truth[transcript_over_ground_truth>=0] = 0
transcript_over_ground_truth <- abs(transcript_over_ground_truth)
rownames(transcript_over_ground_truth) <- c("transcript_over_ground_truth")

# For underclassification of transcripts check where there are positive values.
# Make all other values 0.
transcript_under_ground_truth[transcript_under_ground_truth<0] = 0
rownames(transcript_under_ground_truth) <- c("transcript_under_ground_truth")

# Sum over all of the subjects for both over and under classification
sum_transcript_over_ground_truth <- sum(
  transcript_over_ground_truth["transcript_over_ground_truth",]
)
sum_transcript_under_ground_truth <- sum(
```

```
    transcript_under_ground_truth["transcript_under_ground_truth",]
  )

  # Calculate the percentages of over and under classifcation
  speech_over_ground_truth_perc <- sum_speech_over_ground_truth/groundTruth*100
  speech_under_ground_truth_perc <- sum_speech_under_ground_truth/groundTruth*100
  transcript_over_ground_truth_perc <- sum_transcript_over_ground_truth/groundTruth*100
  transcript_under_ground_truth_perc <- sum_transcript_under_ground_truth/groundTruth*100

  # Add everything to a dataframe
  newRow <- data.frame(
    speech_over_ground_truth_perc=speech_over_ground_truth_perc,
    speech_under_ground_truth_perc=speech_under_ground_truth_perc,
    transcript_over_ground_truth_perc=transcript_over_ground_truth_perc,
    transcript_under_ground_truth_perc=transcript_under_ground_truth_perc
  )

  subjectClassificationRobustness <- rbind(subjectClassificationRobustness,newRow)
}
```

T-test:

```
t.test(subjectClassificationRobustness$speech_under_ground_truth_perc, mu=0)
```

```
##
##  One Sample t-test
##
## data:  subjectClassificationRobustness$speech_under_ground_truth_perc
## t = 16.398, df = 11, p-value = 4.447e-09
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  46.07915 60.36702
## sample estimates:
## mean of x
##  53.22309
```

```
t.test(subjectClassificationRobustness$transcript_under_ground_truth_perc, mu=0)
```

```
##
##  One Sample t-test
##
## data:  subjectClassificationRobustness$transcript_under_ground_truth_perc
## t = 11.904, df = 11, p-value = 1.264e-07
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  17.92564 26.05825
## sample estimates:
## mean of x
##  21.99195
```

Calculate the mean and standard deviation for over and under classification for each user and session:

```
means <- colMeans(x=subjectClassificationRobustness, na.rm = TRUE)
stds <- sapply(subjectClassificationRobustness, sd, na.rm = TRUE)

subjectClassificationRobustness <- rbind(subjectClassificationRobustness,means)
```

```
subjectClassificationRobustness <- rbind(subjectClassificationRobustness,stds)

# Add names to rows
rownames(subjectClassificationRobustness) <- c(
  "u1s1",
  "u1s2",
  "u1s3",
  "u2s1",
  "u2s2",
  "u2s3",
  "u3s1",
  "u3s2",
  "u3s3",
  "u4s1",
  "u4s2",
  "u4s3",
  "mean",
  "std"
)
```

Print final table:

```
subjectClassificationRobustness
```

```
##      speech_over_ground_truth_perc speech_under_ground_truth_perc
## u1s1                      3.669725                       64.22018
## u1s2                      1.234568                       59.25926
## u1s3                      6.329114                       63.29114
## u2s1                      1.052632                       66.31579
## u2s2                      0.000000                       38.88889
## u2s3                      2.020202                       49.49495
## u3s1                     14.516129                       62.90323
## u3s2                      4.054054                       56.75676
## u3s3                      8.163265                       40.81633
## u4s1                      2.083333                       41.66667
## u4s2                      7.246377                       59.42029
## u4s3                      2.970297                       35.64356
## mean                      4.444975                       53.22309
## std                       4.074416                       11.24374
##      transcript_over_ground_truth_perc transcript_under_ground_truth_perc
## u1s1                          3.669725                          26.605505
## u1s2                          1.234568                          27.160494
## u1s3                          7.594937                          32.911392
## u2s1                          2.105263                          14.736842
## u2s2                         12.222222                          13.333333
## u2s3                          6.060606                          14.141414
## u3s1                         19.354839                          22.580645
## u3s2                          6.756757                          24.324324
## u3s3                         12.244898                          22.448980
## u4s1                          4.166667                          20.833333
## u4s2                          8.695652                          28.985507
## u4s3                          3.960396                          15.841584
## mean                          7.338877                          21.991946
## std                           5.185397                           6.399906
```