Virtual Patient SDM Performance Analysis

Han Lie

December 2018

Introduction

In this document an overview is given of all the data that has been collected in an experiment done in a collaboration between the TU Delft, AMC-UvA, UvA and LUMC. The experiment consisted of testing a virtual patient system that teaches shared decision making.

In total four participants were asked to do a shared decision making conversation with three different virtual patients. The goal was to find out whether test subjects improved their knowledge and skill with regards to shared decision making through this process.

This document consists of several chunks of code that are accompanied by pieces of text that try to give insight in the data collected throughout this experiment. It roughly consists of the following parts:

- Subject Coverage - Show in what degree each conversation subject (in this case the different therapies) were discussed according to the system.
- Shared decision making coverage - Show in what degree each step in the shared decision making process was discussed according to the system.

Both the Subject Coverage and the SDM Performance analyses were done for talked input, transcripsted input and hand annotated input.

Speech Results

Subject Coverage

The system outputs raw conversation data to a MySQL database. To filter out the necessary parts SQL queries are used. The resulting data chunks were saved to .csv files which were read into R, where data analysis was continued.

After each shared decision conversation the database was exported and cleared for the next conversation. On each exported database the following query was used to retrieve the necessary item coverage data:

```
SELECT  virtual_patient.effect_on_therapy.therapyId,
        virtual_patient.effect_on_desire.desireId,
        virtual_patient.effect_on_therapy.frequency,
        virtual_patient.desire.importanceToPatient
FROM virtual_patient.effect_on_desire
    JOIN virtual_patient.effect_on_therapy
        ON virtual_patient.effect_on_desire.effectId = virtual_patient.effect_on_therapy.effectId
    JOIN virtual_patient.therapy
        ON virtual_patient.effect_on_therapy.therapyId = virtual_patient.therapy.id
    JOIN virtual_patient.desire
        ON virtual_patient.desire.id = virtual_patient.effect_on_desire.desireId
```

Read all conversation data into R and add columns that contain the userId and sessionId of each conversation:

```
folder <- paste("/Users/fooyonghan/Downloads/final-analysis/sdm-coverage-analysis/",
                "speech-results/formatted-results/item-coverage/", sep="")
fileList <- list.files(path=folder, pattern="*.csv")

itemCoverageList <- list()

for (i in 1:length(fileList)){
  id = fileList[i]
  id = gsub(".csv", "", id)

  userId = as.integer(substring(id, 2, 2))
  sessionId = as.integer(substring(id, 4, 4))

  itemCoverage = read.csv(paste(folder, fileList[i], sep=''))
  itemCoverage$userId <- rep(userId, nrow(itemCoverage))
  itemCoverage$sessionId <- rep(sessionId, nrow(itemCoverage))

  itemCoverageList[[i]] <- itemCoverage
}
```

When the .csv files have been loaded into R, a table is made representing to what degree each subject has been discussed for each therapy in every session for all of the test subjects. To create these tables first add these packages:

First count the frequency each desire is mentioned in each therapy:

```
aggregatedList <- list()

for (i in 1:length(itemCoverageList)){
  aggregatedItemCoverage <- aggregate(frequency ~ userId + sessionId + therapyId +
    desireId + importanceToPatient, data = itemCoverageList[[i]], sum)

  aggregatedList[[i]] <- aggregatedItemCoverage
}
```

Remove therapyId = 1, as these are pre-filled values representing doing no therapy. It has no significance for gathering speech data that is parsed to knowledge by the system. Subsequently sort them based on therapyId:

```
sortedList <- list()

for (i in 1:length(aggregatedList)){
  aggregatedItemCoverage <- aggregatedList[[i]]
  aggregatedItemCoverage <- aggregatedItemCoverage[aggregatedItemCoverage$therapyId!=1,]
  aggregatedItemCoverage <- aggregatedItemCoverage[order(aggregatedItemCoverage$therapyId,
    aggregatedItemCoverage$desireId),]

    sortedList[[i]] <- aggregatedItemCoverage
}
```

Now split the resulting dataframe into separate columns, so that aggregated frequency can be viewed per therapy:

```
splitList <- list()

for (i in 1:length(sortedList)){
  splitList[[i]] <- split( sortedList[[i]] , f = sortedList[[i]]$therapyId )
```

```
}
```

Format the data into dataframes that each contains a user' session:

```
sessionList <- list()

for (i in 1:length(splitList)){
  userId <- splitList[[i]][[1]]$userId
  sessionId <- splitList[[i]][[1]]$sessionId
  desireId <- splitList[[i]][[1]]$desireId
  importanceToPatient <- splitList[[i]][[1]]$importanceToPatient

  session <- data.frame(userId, sessionId, desireId, importanceToPatient)

  for (j in 1:length(splitList[[i]])){
    therapyResult <- splitList[[i]][[j]]

    columnName <- paste0("frequency-", j)
    frequency <- therapyResult$frequency
    session[[columnName]] <- frequency
  }

  sessionList[[i]] <- session
}
```

A table needs to be created that represents this data per user in percentages.

This table is created by counting the amount of times each important desire has been discussed for each therapy throughout each session:

```
countList <- setNames(data.frame(matrix(ncol = 6, nrow = 0)),
  c("userId", "sessionId", "step2"))

for (i in 1:length(sessionList)){
  therapy1Satisfaction <- count(sessionList[[i]],
    (sessionList[[i]]$`frequency-1` > 0 &
    sessionList[[i]]$importanceToPatient > mean(sessionList[[i]]$importanceToPatient))
  )
  therapy2Satisfaction <- count(sessionList[[i]],
    (sessionList[[i]]$`frequency-2` > 0 &
    sessionList[[i]]$importanceToPatient > mean(sessionList[[i]]$importanceToPatient))
  )
  therapy3Satisfaction <- count(sessionList[[i]],
    (sessionList[[i]]$`frequency-3` > 0 &
    sessionList[[i]]$importanceToPatient > mean(sessionList[[i]]$importanceToPatient))
  )

  numOfImportantTherapies <- count(sessionList[[i]],
    (sessionList[[i]]$importanceToPatient > mean(sessionList[[i]]$importanceToPatient))
  )

  userId <- sessionList[[i]]$userId[1]
  sessionId <- sessionList[[i]]$sessionId[1]
  t2Count <- therapy2Satisfaction[[2]][2]
  t2Count[is.na(t2Count)] <- 0
```

```
  t3Count <- therapy3Satisfaction[[2]][2]
  t3Count[is.na(t3Count)] <- 0
  numOfWantedDesires <- (numOfImportantTherapies[[2]][2])*2
  step2 <- (t2Count+t3Count)/numOfWantedDesires

  countList[nrow(countList) + 1,] = list(userId, sessionId, step2)
}
```

SDM Performance

The previously calculated percentage respresents the second step in the shared decision making protocol, namely to what degree the desires of the virtual patient were discussed. This column should now be added to a table that contains all the shared decision making steps for each user and each user' session. To create this dataframe, a new SQL query needs to be done on the database:

```
SELECT  virtual_patient.conversation.id,
        virtual_patient.conversation.dateTime,
        virtual_patient.conversation.message,
        virtual_patient.conversation.source,
        virtual_patient.question.questionSubject,
        virtual_patient.input.keywords
FROM    virtual_patient.conversation
    LEFT JOIN virtual_patient.question
        ON virtual_patient.conversation.message = virtual_patient.question.question
    LEFT JOIN virtual_patient.input
        ON virtual_patient.conversation.message = virtual_patient.input.message
WHERE virtual_patient.question.questionSubject IS NOT NULL
    OR virtual_patient.input.keywords = 'shareddecisionmaking'
```

This SQL query returns information about steps 1, 3 and 4 of the SDM protocol. Below these csv files are read and put into a dataframe.

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

sdmCoverageList <- list()

for (i in 1:length(fileList)){
  id = fileList[i]
  id = gsub(".csv", "", id)

  userId = as.integer(substring(id, 2, 2))
  sessionId = as.integer(substring(id, 4, 4))

  sdmCoverage = read.csv(paste(folder, fileList[i], sep=''))
  sdmCoverage$userId <- rep(userId, nrow(sdmCoverage))
  sdmCoverage$sessionId <- rep(sessionId, nrow(sdmCoverage))

  sdmCoverageList[[i]] <- sdmCoverage
}
```

Now that a dataframe with the right information has been made, an check needs to be made whether steps 1,3 and 4 of the protocol have been done. For step 1 this means 'shareddecisionmaking' has to be mentioned

as a keyword. For step 2 the questionsSubject 'desires' has to be mentioned. For step 3 the questionSubject 'chemoChoice' or hormonesChoice' has to be mentioned. The below chunk of code will check these steps accordingly.

```
threeStepsList <- setNames(data.frame(matrix(ncol = 5, nrow = 0)),
  c("userId", "sessionId", "step1", "step3", "step4"))

for (i in 1:length(sdmCoverageList)){
  userId <- sdmCoverageList[[i]]$userId[1]
  sessionId <- sdmCoverageList[[i]]$sessionId[1]
  step1 <- 'shareddecisionmaking' %in% sdmCoverageList[[i]]$keywords
  step3 <- 'desires' %in% sdmCoverageList[[i]]$questionSubject
  step4 <- 'chemoChoice' %in% sdmCoverageList[[i]]$questionSubject |
    'hormonesChoice'%in% sdmCoverageList[[i]]$questionSubject

  threeStepsList[nrow(threeStepsList) + 1,] = list(userId, sessionId, step1, step3, step4)
}
```

Add step 2 from the previously calculated itemCoverage:

```
mergedList <- merge(x=threeStepsList, y=countList[, c("userId","sessionId", "step2")],
  all.x=TRUE, by = c("userId", "sessionId"))

mergedList <- mergedList[ , c("userId", "sessionId", "step1", "step2", "step3", "step4")]
```

Replace boolean values with 1 / 0:

```
cols <- sapply(mergedList, is.logical)
mergedList[,cols] <- lapply(mergedList[,cols], as.numeric)
```

Calculate sdm satisfaction percentage:

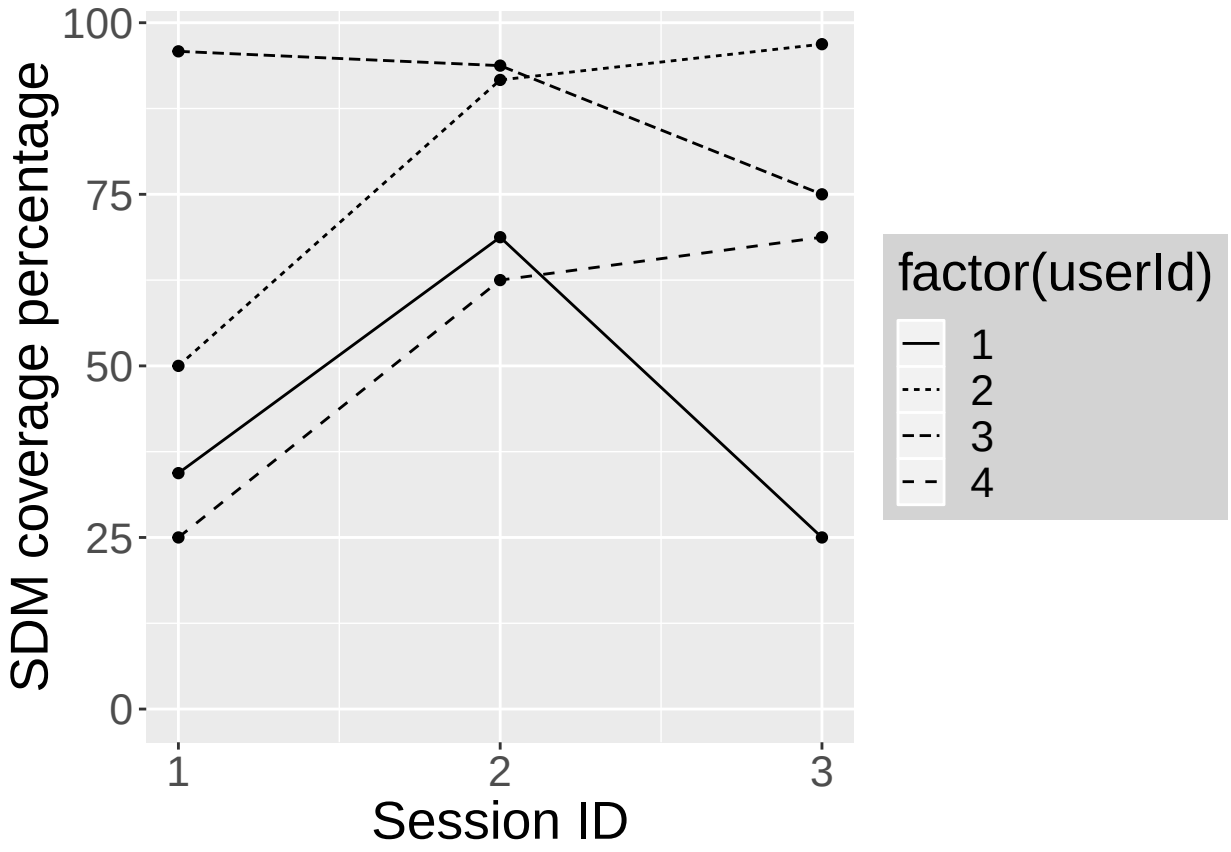
```
mergedList$total <- mergedList$step1 + mergedList$step2 + mergedList$step3 +
  mergedList$step4

mergedList$sdmSatisfaction <- mergedList$total/4*100
```

Only keep the necessary values for our regression analysis:

```
speechSDMPerformance <- mergedList[ , c("userId", "sessionId", "sdmSatisfaction")]
```

Create a plot of the SDM satisfaction data:

```
library(ggplot2)
ggplot(data=speechSDMPerformance, aes(x=sessionId, y=sdmSatisfaction, group=userId)) +
  geom_line(aes(linetype=factor(userId))) +
  geom_point() +
  xlab("Session ID") +
  ylab("SDM coverage percentage") +
  scale_x_continuous(breaks=c(1,2,3)) +
  theme(legend.background = element_rect(fill="lightgrey", size=0.5, linetype="solid")) +
  theme(text = element_text(size=20)) +
  expand_limits(y = 0)
```



Transcript Results

Repeat analysis for transcript results: ## Subject Coverage

```
folder <- paste("/Users/fooyonghan/Downloads/final-analysis/sdm-coverage-analysis/",
  "transcript-results/formatted-results/item-coverage/",sep="")
fileList <- list.files(path=folder, pattern="*.csv")

itemCoverageList <- list()

for (i in 1:length(fileList)){
  id = fileList[i]
  id = gsub(".csv", "", id)

  userId = as.integer(substring(id, 2, 2))
  sessionId = as.integer(substring(id, 4, 4))

  itemCoverage = read.csv(paste(folder, fileList[i], sep=''))
  itemCoverage$userId <- rep(userId, nrow(itemCoverage))
  itemCoverage$sessionId <- rep(sessionId, nrow(itemCoverage))

  itemCoverageList[[i]] <- itemCoverage
}

aggregatedList <- list()
```

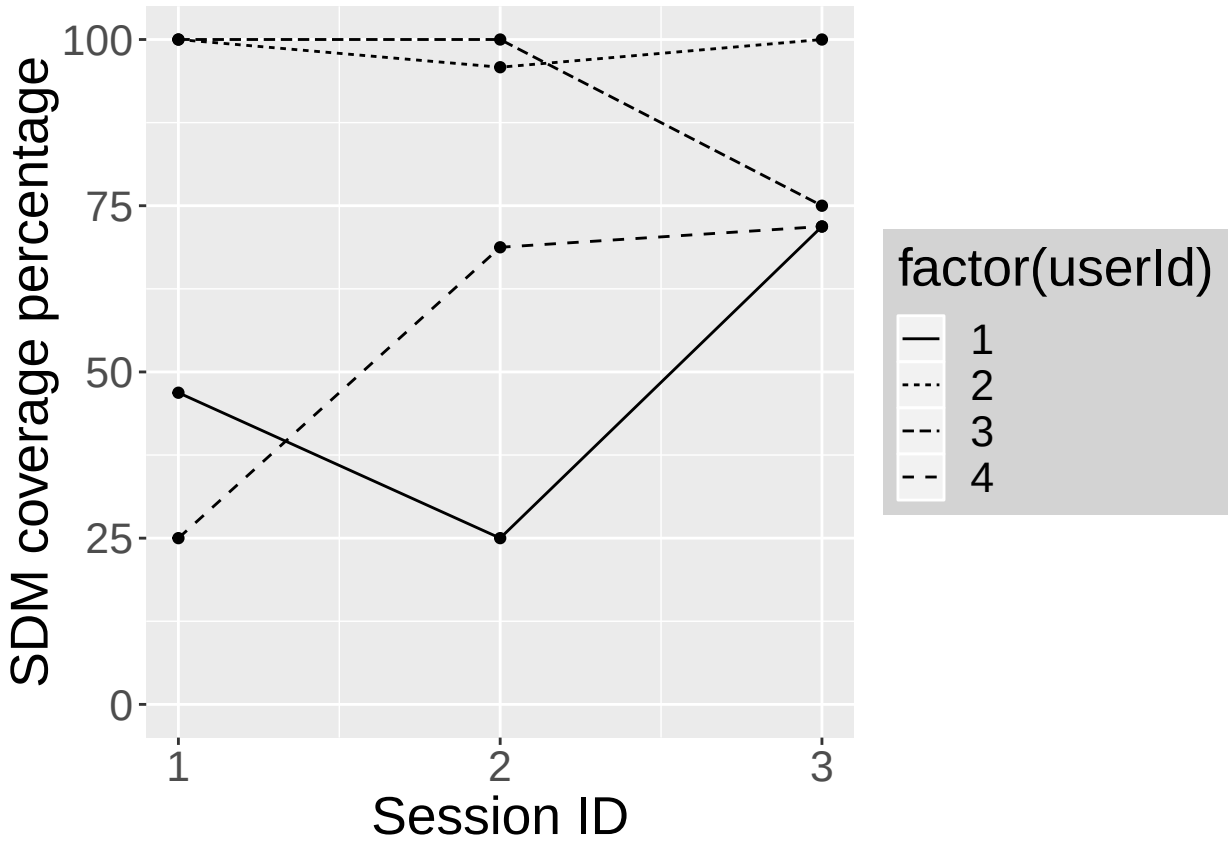
```
for (i in 1:length(itemCoverageList)){
  aggregatedItemCoverage <- aggregate(frequency ~ userId + sessionId + therapyId +
    desireId + importanceToPatient, data = itemCoverageList[[i]], sum)

  aggregatedList[[i]] <- aggregatedItemCoverage
}

sortedList <- list()

for (i in 1:length(aggregatedList)){
  aggregatedItemCoverage <- aggregatedList[[i]]
  aggregatedItemCoverage <- aggregatedItemCoverage[aggregatedItemCoverage$therapyId!=1,]
  aggregatedItemCoverage <- aggregatedItemCoverage[order(aggregatedItemCoverage$therapyId,
    aggregatedItemCoverage$desireId),]

    sortedList[[i]] <- aggregatedItemCoverage
}

splitList <- list()

for (i in 1:length(sortedList)){
  splitList[[i]] <- split( sortedList[[i]] , f = sortedList[[i]]$therapyId )
}

sessionList <- list()

for (i in 1:length(splitList)){
  userId <- splitList[[i]][[1]]$userId
  sessionId <- splitList[[i]][[1]]$sessionId
  desireId <- splitList[[i]][[1]]$desireId
  importanceToPatient <- splitList[[i]][[1]]$importanceToPatient

  session <- data.frame(userId, sessionId, desireId, importanceToPatient)

  for (j in 1:length(splitList[[i]])){
    therapyResult <- splitList[[i]][[j]]

    columnName <- paste0("frequency-", j)
    frequency <- therapyResult$frequency
    session[[columnName]] <- frequency
  }

  sessionList[[i]] <- session
}

countList <- setNames(data.frame(matrix(ncol = 6, nrow = 0)),
  c("userId", "sessionId", "step2"))

for (i in 1:length(sessionList)){
  therapy1Satisfaction <- count(sessionList[[i]],
    (sessionList[[i]]$`frequency-1` > 0 &
    sessionList[[i]]$importanceToPatient > mean(sessionList[[i]]$importanceToPatient))
  )
```

```
  therapy2Satisfaction <- count(sessionList[[i]],
    (sessionList[[i]]$`frequency-2` > 0 &
    sessionList[[i]]$importanceToPatient > mean(sessionList[[i]]$importanceToPatient))
  )
  therapy3Satisfaction <- count(sessionList[[i]],
    (sessionList[[i]]$`frequency-3` > 0 &
    sessionList[[i]]$importanceToPatient > mean(sessionList[[i]]$importanceToPatient))
  )

  numOfImportantTherapies <- count(sessionList[[i]],
    (sessionList[[i]]$importanceToPatient > mean(sessionList[[i]]$importanceToPatient))
  )

  userId <- sessionList[[i]]$userId[1]
  sessionId <- sessionList[[i]]$sessionId[1]
  t2Count <- therapy2Satisfaction[[2]][2]
  t2Count[is.na(t2Count)] <- 0
  t3Count <- therapy3Satisfaction[[2]][2]
  t3Count[is.na(t3Count)] <- 0
  numOfWantedDesires <- (numOfImportantTherapies[[2]][2])*2
  step2 <- (t2Count+t3Count)/numOfWantedDesires

  countList[nrow(countList) + 1,] = list(userId, sessionId, step2)
}
```

SDM Performance

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

sdmCoverageList <- list()

for (i in 1:length(fileList)){
  id = fileList[i]
  id = gsub(".csv", "", id)

  userId = as.integer(substring(id, 2, 2))
  sessionId = as.integer(substring(id, 4, 4))

  sdmCoverage = read.csv(paste(folder, fileList[i], sep=''))
  sdmCoverage$userId <- rep(userId, nrow(sdmCoverage))
  sdmCoverage$sessionId <- rep(sessionId, nrow(sdmCoverage))

  sdmCoverageList[[i]] <- sdmCoverage
}

threeStepsList <- setNames(data.frame(matrix(ncol = 5, nrow = 0)),
  c("userId", "sessionId", "step1", "step3", "step4"))

for (i in 1:length(sdmCoverageList)){
  userId <- sdmCoverageList[[i]]$userId[1]
```

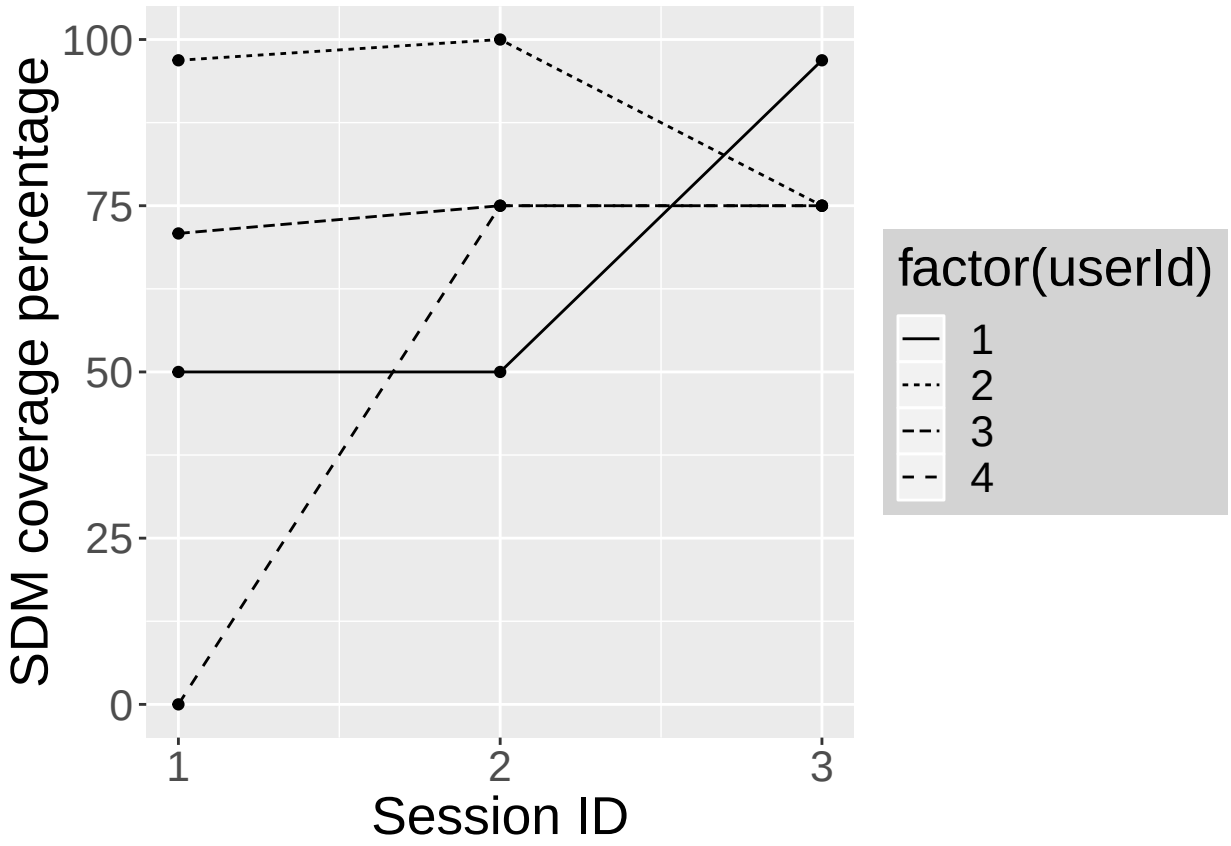
```
  sessionId <- sdmCoverageList[[i]]$sessionId[1]
  step1 <- 'shareddecisionmaking' %in% sdmCoverageList[[i]]$keywords
  step3 <- 'desires' %in% sdmCoverageList[[i]]$questionSubject
  step4 <- 'chemoChoice' %in% sdmCoverageList[[i]]$questionSubject |
    'hormonesChoice' %in% sdmCoverageList[[i]]$questionSubject

  threeStepsList[nrow(threeStepsList) + 1,] = list(userId, sessionId, step1, step3, step4)
}

mergedList <- merge(x=threeStepsList, y=countList[, c("userId","sessionId", "step2")],
  all.x=TRUE, by = c("userId", "sessionId"))
mergedList <- mergedList[ , c("userId", "sessionId", "step1", "step2", "step3", "step4")]
cols <- sapply(mergedList, is.logical)
mergedList[,cols] <- lapply(mergedList[,cols], as.numeric)
mergedList$total <- mergedList$step1 + mergedList$step2 + mergedList$step3 +
  mergedList$step4
mergedList$sdmSatisfaction <- mergedList$total/4*100

transcriptSDMPerformance <- mergedList[ , c("userId", "sessionId", "sdmSatisfaction")]

ggplot(data=transcriptSDMPerformance, aes(x=sessionId, y=sdmSatisfaction, group=userId)) +
  geom_line(aes(linetype=factor(userId))) +
  geom_point() +
  xlab("Session ID") +
  ylab("SDM coverage percentage") +
  scale_x_continuous(breaks=c(1,2,3)) +
  theme(legend.background = element_rect(fill="lightgrey", size=0.5, linetype="solid")) +
  theme(text = element_text(size=20)) +
  expand_limits(y = 0)
```



Hand Annotation Results

The hand annotated results, were based on a coder going through each transcript and countin how many times each subject was discussed. These results were put into a csv file. For the hand annotated graph the following code can be used. Load hand annotated values and plot them.

```
handAnnotationSDMPerformance <- read.csv(file=paste("/Users/fooyonghan/Downloads/final-analysis/",
  "sdm-coverage-analysis/handannotation-results/sdm-coverage-hand.csv",sep=""), header=TRUE, sep=";")

# Replace commas with dots
handAnnotationSDMPerformance$sdmSatisfaction <- as.numeric(gsub(",", ".", gsub("\\.", "",
  handAnnotationSDMPerformance$sdmSatisfaction)))

library(ggplot2)
ggplot(data=handAnnotationSDMPerformance, aes(x=sessionId, y=sdmSatisfaction, group=userId)) +
  geom_line(aes(linetype=factor(userId))) +
  geom_point() +
  xlab("Session ID") +
  ylab("SDM coverage percentage") +
  scale_x_continuous(breaks=c(1,2,3)) +
  theme(legend.background = element_rect(fill="lightgrey", size=0.5, linetype="solid")) +
  theme(text = element_text(size=20)) +
  expand_limits(y = 0)
```



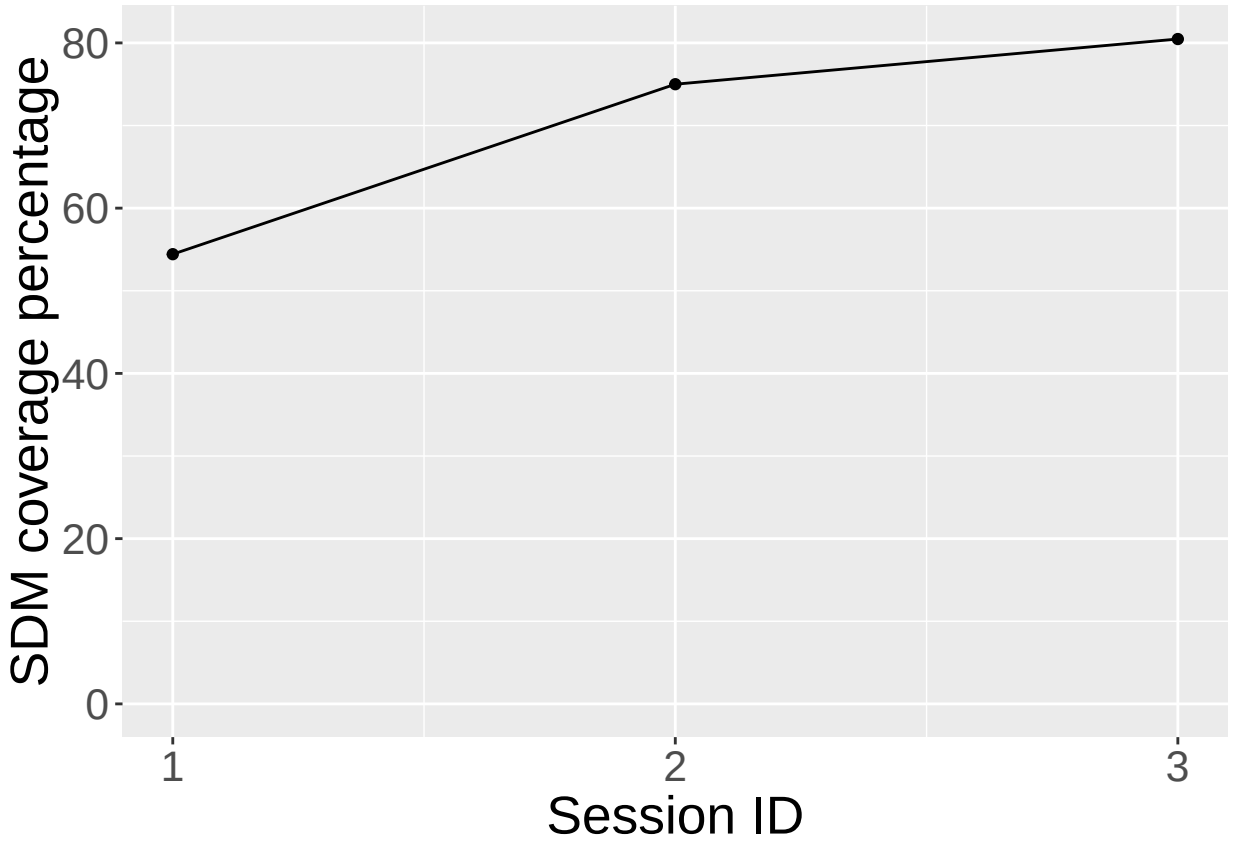
Calculate Spearman correlation coefficient:

```
#sccTalk <- cor(x=finalList$sessionId, y=finalList$sdmSatisfaction, method="spearman")
#sccHand <- cor(x=handAnnotated$sessionId,
#  y=handAnnotated$sdmSatisfaction, method="spearman")

#sccTalk
#sccHand
```

A graph could also be made that takes the hand annotated data and plots the average of all the users:

```
handAnnotatedAverage <- read.csv(file="/Users/fooyonghan/Downloads/averageSDM.csv",
  header=TRUE, sep=";")

# Replace commas with dots
handAnnotatedAverage$sdmSatisfaction <- as.numeric(gsub(",", ".", gsub("\\.", "",
  handAnnotatedAverage$sdmSatisfaction)))

library(ggplot2)
ggplot(data=handAnnotatedAverage, aes(x=sessionId, y=sdmSatisfaction)) +
  geom_line(aes(linetype="solid")) +
  geom_point() +
  xlab("Session ID") +
  ylab("SDM coverage percentage") +
  scale_x_continuous(breaks=c(1,2,3)) +
  theme(legend.background = element_rect(fill="lightgrey", size=0.5, linetype="solid")) +
  theme(text = element_text(size=20)) +
  expand_limits(y = 0) +
```

```
  theme(legend.position="none")
```



Feedback Robustness

To assess whether the feedback component was reliable, a further analysis was done on the SDM performance of speech, transcript and handannotation. Per user and session it was analysed to what degree speech input and transcript input resulted in a higher or lower SDM performance compared to handannotation input.

To do this the SDM performance scores of the three different types of input were put into one dataframe:

```
speechResults <- speechSDMPerformance[,3]
transcriptResults <- transcriptSDMPerformance[,3]
handAnnotationResults <- handAnnotationSDMPerformance[,3]

speechLabel <- "speech"
transcriptLabel <- "transcript"
handAnnotationLabel <- "handannotation"

sdmPerformance <- data.frame(speechResults,transcriptResults,handAnnotationResults)
names(sdmPerformance) <- c(speechLabel, transcriptLabel, handAnnotationLabel)
rownames(sdmPerformance) <- c(
  "u1s1",
  "u1s2",
  "u1s3",
  "u2s1",
  "u2s2",
```

```
  "u2s3",
  "u3s1",
  "u3s2",
  "u3s3",
  "u4s1",
  "u4s2",
  "u4s3"
)
```

Add columns that show when speech under- and overperformed compared to handannotation, and when transcript under- and overperformed compared to handannotation:

```
sdmPerformance["diff_hand_speech"] <-
  sdmPerformance$handannotation - sdmPerformance$speech

sdmPerformance["speech_under_ground_truth"] <- sdmPerformance$diff_hand_speech
sdmPerformance$speech_under_ground_truth[sdmPerformance$diff_hand_speech>=0] = 0
sdmPerformance$speech_under_ground_truth <- abs(sdmPerformance$speech_under_ground_truth)

sdmPerformance["speech_over_ground_truth"] <- sdmPerformance$diff_hand_speech
sdmPerformance$speech_over_ground_truth[sdmPerformance$diff_hand_speech<0] = 0
sdmPerformance$speech_over_ground_truth <- abs(sdmPerformance$speech_over_ground_truth)

sdmPerformance["diff_hand_transcript"] <-
  sdmPerformance$handannotation - sdmPerformance$transcript

sdmPerformance["transcript_under_ground_truth"] <- sdmPerformance$diff_hand_transcript
sdmPerformance$transcript_under_ground_truth[sdmPerformance$diff_hand_transcript>=0] = 0
sdmPerformance$transcript_under_ground_truth <-
  abs(sdmPerformance$transcript_under_ground_truth)

sdmPerformance["transcript_over_ground_truth"] <- sdmPerformance$diff_hand_transcript
sdmPerformance$transcript_over_ground_truth[sdmPerformance$diff_hand_transcript<0] = 0
sdmPerformance$transcript_over_ground_truth <-
  abs(sdmPerformance$transcript_over_ground_truth)
```

Calculate the average discrepancy of speech and transcript:

```
sdmPerformance["avg_speech"] <- abs((sdmPerformance$speech_under_ground_truth -
  sdmPerformance$speech_over_ground_truth) / 2)

sdmPerformance["avg_transcript"] <- abs((sdmPerformance$transcript_under_ground_truth -
  sdmPerformance$transcript_over_ground_truth) / 2)
```

Calculate correlation and t-test between hand-speech, and transcript-hand:

```
cor.test(sdmPerformance$speech, sdmPerformance$handannotation,method="pearson")
```

```
    Pearson's product-moment correlation

data:  sdmPerformance$speech and sdmPerformance$handannotation
t = 1.2859, df = 10, p-value = 0.2275
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.2516212  0.7816118
```

```
sample estimates:
      cor
0.3766743
```

```
cor.test(sdmPerformance$transcript, sdmPerformance$handannotation,method="pearson")
```

```

    Pearson's product-moment correlation

data:  sdmPerformance$transcript and sdmPerformance$handannotation
t = 3.7988, df = 10, p-value = 0.003492
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.3482776 0.9315682
sample estimates:
      cor
0.7685614
```

```
(cor(sdmPerformance$speech, sdmPerformance$handannotation))^2
```

```
[1] 0.1418836
```

```
(cor(sdmPerformance$transcript, sdmPerformance$handannotation))^2
```

```
[1] 0.5906867
```

T-test:

```
t.test(sdmPerformance$speech, sdmPerformance$handannotation, paired = TRUE)
```

```

    Paired t-test

data:  sdmPerformance$speech and sdmPerformance$handannotation
t = -0.49673, df = 11, p-value = 0.6292
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -23.57174  14.89118
sample estimates:
mean of the differences
              -4.340278
```

```
t.test(sdmPerformance$transcript, sdmPerformance$handannotation, paired = TRUE)
```

```

    Paired t-test

data:  sdmPerformance$transcript and sdmPerformance$handannotation
t = 0.62093, df = 11, p-value = 0.5473
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -8.614774 15.385607
sample estimates:
mean of the differences
               3.385417
```

Calculate means and stds:

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

sdmPerformance <- rbind(sdmPerformance,means)
sdmPerformance <- rbind(sdmPerformance,stds)

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

Print final table:

```r
print(sdmPerformance)
```

```
       speech transcript handannotation diff_hand_speech
u1s1 34.37500   46.87500       50.00000        15.625000
u1s2 68.75000   25.00000       50.00000       -18.750000
u1s3 25.00000   71.87500       96.87500        71.875000
u2s1 50.00000  100.00000       96.87500        46.875000
u2s2 91.66667   95.83333      100.00000         8.333333
u2s3 96.87500  100.00000       75.00000       -21.875000
u3s1 95.83333  100.00000       70.83333       -25.000000
u3s2 93.75000  100.00000       75.00000       -18.750000
u3s3 75.00000   75.00000       75.00000         0.000000
u4s1 25.00000   25.00000        0.00000       -25.000000
u4s2 62.50000   68.75000       75.00000        12.500000
u4s3 68.75000   71.87500       75.00000         6.250000
mean 65.62500   73.35069       69.96528         4.340278
std  26.96287   28.20541       27.25338        30.268140
     speech_under_ground_truth speech_over_ground_truth diff_hand_transcript
u1s1                  0.000000                15.625000             3.125000
u1s2                 18.750000                 0.000000            25.000000
u1s3                  0.000000                71.875000            25.000000
u2s1                  0.000000                46.875000            -3.125000
u2s2                  0.000000                 8.333333             4.166667
u2s3                 21.875000                 0.000000           -25.000000
u3s1                 25.000000                 0.000000           -29.166667
u3s2                 18.750000                 0.000000           -25.000000
u3s3                  0.000000                 0.000000             0.000000
u4s1                 25.000000                 0.000000           -25.000000
```

u4s2	0.000000	12.500000	6.250000
u4s3	0.000000	6.250000	3.125000
mean	9.114583	13.454861	-3.385417
std	11.420608	22.763167	18.886942

	transcript_under_ground_truth	transcript_over_ground_truth	avg_speech	avg_transcript
u1s1	0.000000	3.125000	7.812500	1.562500
u1s2	0.000000	25.000000	9.375000	12.500000
u1s3	0.000000	25.000000	35.937500	12.500000
u2s1	3.125000	0.000000	23.437500	1.562500
u2s2	0.000000	4.166667	4.166667	2.083333
u2s3	25.000000	0.000000	10.937500	12.500000
u3s1	29.166667	0.000000	12.500000	14.583333
u3s2	25.000000	0.000000	9.375000	12.500000
u3s3	0.000000	0.000000	0.000000	0.000000
u4s1	25.000000	0.000000	12.500000	12.500000
u4s2	0.000000	6.250000	6.250000	3.125000
u4s3	0.000000	3.125000	3.125000	1.562500
mean	8.940972	5.555556	11.284722	7.248264
std	12.707078	9.320478	9.759915	5.915356