Design Challenge

Team He, Khan, Kishor, Young

We met several times as a team, and decided to try out several different visualization ideas instead of focusing on just one. We created a total of five visualizations based on the ideas we discussed during our team meetings. Four visualizations are described below, and the fifth visualization will be submitted separately.

Heat Map created in R

This visualization shows association strength of cell/pair variable for each participant. Originally, each participant has a symmetric matrix. We removed the matrix structure by treating each cell as one variable, which we call a pair-variable. Hence, we formed a vector of pair-variables for each participant. In a case of 8-by-8 symmetric matrix, we have unique 8*7/2=28 pair-variables for each participant. I'll call these variables as pair-variables to reduce the confusion with the original variables which would pair up with each other and of whose pair has an association strength. We display one participant as one row and one pair-variable as one column. In this display, each cell (a pair-variable of a participant) has its corresponding association strength and the association strength is encoded by the lightness of color. The stronger the strength is, the redder the cell is. The weaker the strength is, the whiter the cell is. These cells make a big rectangle with colors varying from red to white, main part of the visualization. On the right side of this main rectangle, a participant's information is written - venue, story, and id. One the left side of the main rectangle, venues are encoded with different colors. One the bottom, pair-variables are labeled. One the top of the main rectangle, a pair-variable (e.g. K_reporting - K_story) is un-paired into two variables (K_reporting and K_story), encoded with color. A dendrogram is drawn by hierarchical clustering which joins the two most similar pair-variables together in Euclidean distance. Columns are ordered accordingly by the result of dendrogram. Rows maintain the actual groupings of venue and story. It is made by modified function of "heatmap" function in "R". To run, refer to "DCheatmap.R". "DCheatmap.R" use "funcheatmp.R" and "all.csv", generating "heatmap.png".

This visualization can be useful when there are differences in some pair-variables between groups. Practicum and Game are more associated overall than Peer and Course. Course seems to associate V_engaging_reader_story and S_detailed_description, and V_engaging_reader_story and K_reader more than other venues do. It can also compare pair-variables in each group. In Game venue, left-side pair-variables seem to have more strength than the right-hand-side pair-variables. It can also detect some odd behaviors like Practicum with story 2.

This visualization can display all the strength information in a network as one row. We can display multiple networks by adding more rows. In this way, we could quickly explore whole data. By adding several participants in the same group, it naturally makes a pattern and can be compared with other group in column-wise. Also, we could take into account variability for each pair-variable from participant to participant in the same group. Since hierarchical clustering spatially puts similar pair-variable next to each other, it would get easier to find a pattern across venues by tying similar pair-variables together.

One weakness of this visualization is that it does not have the original structure of the symmetric matrix. Hence, we cannot see the linkages among variables. Top-hand-side bars with several colors was implemented to overcome this shortcomings a bit.

The feedback and the response are"

1. This visualization requires explanation before - One of the confusing part of this visualization is due to transforming symmetric matrix into a vector of pair-variables.

2. Overload / snapshot of data / zoom-in - When looking at multiple networks, this visualization gives us an overview of the data, although this visualization can overwhelm us. Depending on the contrast and ordering, we might not get any meaningful information. We could select some pair-variables and participants and zoom-in.

3. Bright colors - Different color scheme could be used with less difference in color between cells.

4. Dendrogram - A dendrogram would suggest grouping of pair-variables. Pair-variables in the same cluster would act similarly. If there is a difference between venues, pair-variables in the same cluster might show same pattern over venues. Analogously in microarray, venues are diseased/normal with multiple individuals in them and pair-variables in the same cluster are genes in the same pathway. I think it it hard to compare dendrograms because dendrogram itself seems to be hard to compare and there could be several solutions for dendrograms.

5. Small label size - The labels on the bottom were made bigger.

```
## modified version of heatmap
# funcheatmap.R
myheatmap <-
function (x, Rowv = NULL, Colv = if (symm) "Rowv" else NULL,
    distfun = dist, hclustfun = hclust, reorderfun = function(d,
        w) reorder(d, w), add.expr, symm = FALSE, revC = identical(Colv,
        "Rowv"), scale = c("row", "column", "none"), na.rm = TRUE,
    margins = c(5, 5), ColSideColors, RowSideColors, cVenue, cexRow = 0.2 +
        1/log10(nr), cexCol = 0.2 + 1/log10(nc), labRow = NULL,
    labCol = NULL, main = NULL, xlab = NULL, ylab = NULL, keep.dendro = FALSE,
    verbose = getOption("verbose"), ...) {
    scale <- if (symm && missing(scale))
        "none"
    else match.arg(scale)
    if (length(di <- dim(x)) != 2 || !is.numeric(x))
        stop("'x' must be a numeric matrix")
    nr <- di[1]
    nc <- di[2]
    if (nr <= 1 || nc <= 1)
```

```
        stop("'x' must have at least 2 rows and 2 columns")
    if (!is.numeric(margins) || length(margins) != 2)
        stop("'margins' must be a numeric vector of length 2")
    doRdend <- !identical(Rowv, NA)
    doCdend <- !identical(Colv, NA)
    if (is.null(Rowv))
        Rowv <- rowMeans(x, na.rm = na.rm)
    if (is.null(Colv))
        Colv <- colMeans(x, na.rm = na.rm)
    if (doRdend) {
        if (inherits(Rowv, "dendrogram"))
            ddr <- Rowv
        else {
            hcr <- hclustfun(distfun(x))
            ddr <- as.dendrogram(hcr)
            if (!is.logical(Rowv) || Rowv)
                ddr <- reorderfun(ddr, Rowv)
        }
        if (nr != length(rowInd <- order.dendrogram(ddr)))
            stop("row dendrogram ordering gave index of wrong length")
    }
    else rowInd <- 1L:nr
    if (doCdend) {
        if (inherits(Colv, "dendrogram"))
            ddc <- Colv
        else if (identical(Colv, "Rowv")) {
            if (nr != nc)
                stop("Colv = \"Rowv\" but nrow(x) != ncol(x)")
            ddc <- ddr
        }
        else {
            hcc <- hclustfun(distfun(if (symm)
                x
            else t(x)))
            ddc <- as.dendrogram(hcc)
            if (!is.logical(Colv) || Colv)
                ddc <- reorderfun(ddc, Colv)
```

```
        }
        if (nc != length(colInd <- order.dendrogram(ddc)))
            stop("column dendrogram ordering gave index of wrong length")
    }
    else colInd <- 1L:nc
    x <- x[rowInd, colInd]
    labRow <- if (is.null(labRow))
        if (is.null(rownames(x)))
            (1L:nr)[rowInd]
        else rownames(x)
    else labRow[rowInd]
    labCol <- if (is.null(labCol))
        if (is.null(colnames(x)))
            (1L:nc)[colInd]
        else colnames(x)
    else labCol[colInd]
    if (scale == "row") {
        x <- sweep(x, 1, rowMeans(x, na.rm = na.rm), check.margin = FALSE)
        sx <- apply(x, 1, sd, na.rm = na.rm)
        x <- sweep(x, 1, sx, "/", check.margin = FALSE)
    }
    else if (scale == "column") {
        x <- sweep(x, 2, colMeans(x, na.rm = na.rm), check.margin = FALSE)
        sx <- apply(x, 2, sd, na.rm = na.rm)
        x <- sweep(x, 2, sx, "/", check.margin = FALSE)
    }
    lmat <- rbind(c(NA, 3), 2:1)
    lwid <- c(if (doRdend) 1 else 0.05, 4)
    lhei <- c((if (doCdend) 1 else 0.05) + if (!is.null(main)) 0.2 else 0,
        4)
    if (!missing(ColSideColors)) {
        if (!is.character(ColSideColors))
            stop("'ColSideColors' must be a character vector of length ncol(x)")
        lmat <- rbind(lmat[1, ] + 1, c(NA, 1), lmat[2, ] + 1)
        lhei <- c(lhei[1], 0.2, lhei[2])
    }
    if (!missing(RowSideColors)) {
```

```
        if (!is.character(RowSideColors) || length(RowSideColors) !=
            nr)
            stop("'RowSideColors' must be a character vector of length nrow(x)")
        lmat <- cbind(lmat[, 1] + 1, c(rep(NA, nrow(lmat) - 1),
            1), lmat[, 2] + 1)
        lwid <- c(lwid[1], 0.2, lwid[2])
    }
    lmat[is.na(lmat)] <- 0
    if (verbose) {
        cat("layout: widths = ", lwid, ", heights = ", lhei,
            "; lmat=\n")
        print(lmat)
    }
    op <- par(no.readonly = TRUE)
    on.exit(par(op))
    layout(lmat, widths = lwid, heights = lhei, respect = TRUE)
    if (!missing(RowSideColors)) {
        par(mar = c(margins[1], 0, 0, 0.5))
        image(rbind(1L:nr), col = RowSideColors[rowInd], axes = FALSE)
    }
    if (!missing(ColSideColors)) {
        par(mar = c(0.5, 0, 0, margins[2]))
        nCategory<-length(ColSideColors)
        K=rainbow(nCategory)
        J<-labCol
        J<-unlist(strsplit(J, split=' - '))
        J<-factor(J)
        levels(J)<-seq(nCategory)
        J<-as.numeric(J)
        J<-matrix(J, nrow=2)
        image(t(J), col=rainbow(nCategory), axes=FALSE)
     }
    par(mar = c(margins[1], 0, 0, margins[2]))
    if (!symm || scale != "none")
        x <- t(x)
    if (revC) {
        iy <- nr:1
```

```
        ddr <- rev(ddr)
        x <- x[, iy]
    }
    else iy <- 1L:nr
    image(1L:nc, 1L:nr, x, xlim = 0.5 + c(0, nc), ylim = 0.5 +
        c(0, nr), axes = FALSE, xlab = "", ylab = "", ...)
    abline(h=cumsum(cVenue)+.5, lwd=2)
    #abline(h=52+.5, lwd=2)
    #abline(h=52+30+.5, lwd=2)
    #abline(h=52+30+22+.5, lwd=2)
    #abline(h=52+30+22+32+.5, lwd=2)

    axis(1, 1L:nc, labels = labCol, las = 2, line = -0.5, tick = 0,
        cex.axis = cexCol)
    if (!is.null(xlab))
        mtext(xlab, side = 1, line = margins[1] - 1.25)
    axis(4, iy, labels = labRow, las = 2, line = -0.5, tick = 0,
        cex.axis = cexRow)
    if (!is.null(ylab))
        mtext(ylab, side = 4, line = margins[2] - 1.25)
    if (!missing(add.expr))
        eval(substitute(add.expr))
    par(mar = c(margins[1], 0, 0, 0))
    if (doRdend)
        plot(ddr, horiz = TRUE, axes = FALSE, yaxs = "i", leaflab = "none")
    else frame()
    par(mar = c(0, 0, if (!is.null(main)) 1 else 0, margins[2]))
    if (doCdend)
        plot(ddc, axes = FALSE, xaxs = "i", leaflab = "none")
    else if (!is.null(main))
        frame()
    if (!is.null(main)) {
        par(xpd = NA)
        title(main, cex.main = 1.5 * op[["cex.main"]])
    }
    invisible(list(rowInd = rowInd, colInd = colInd, Rowv = if (keep.dendro &&
        doRdend) ddr, Colv = if (keep.dendro && doCdend) ddc))
```

```
}
```

```
source('funcheatmap.R')
```

```
## data and input
# DCheatmap.R
data<-read.csv('path/all.csv', as.is=TRUE)
```

```
subsetCol<-8:15
nCategory<-length(subsetCol)
n<-dim(data)[1]/nCategory
Venues<-unique(data$Venue)
nVenue<-length(Venues)
cVenue<-table(data$Venue)[Venues]/nCategory
```

```
## Transforming symetric matrix into a vector
X<-matrix(0, nrow=n, ncol=nCategory*(nCategory-1)/2)
i<-1
a<-upper.tri(data[((i-1)*nCategory+1):(i*nCategory),subsetCol])
```

```
for (i in 1:n){
  X[i,]<-data[((i-1)*nCategory+1):(i*nCategory),subsetCol][a]
}
```

```
## Labels
code <- colnames(data)[subsetCol]
```

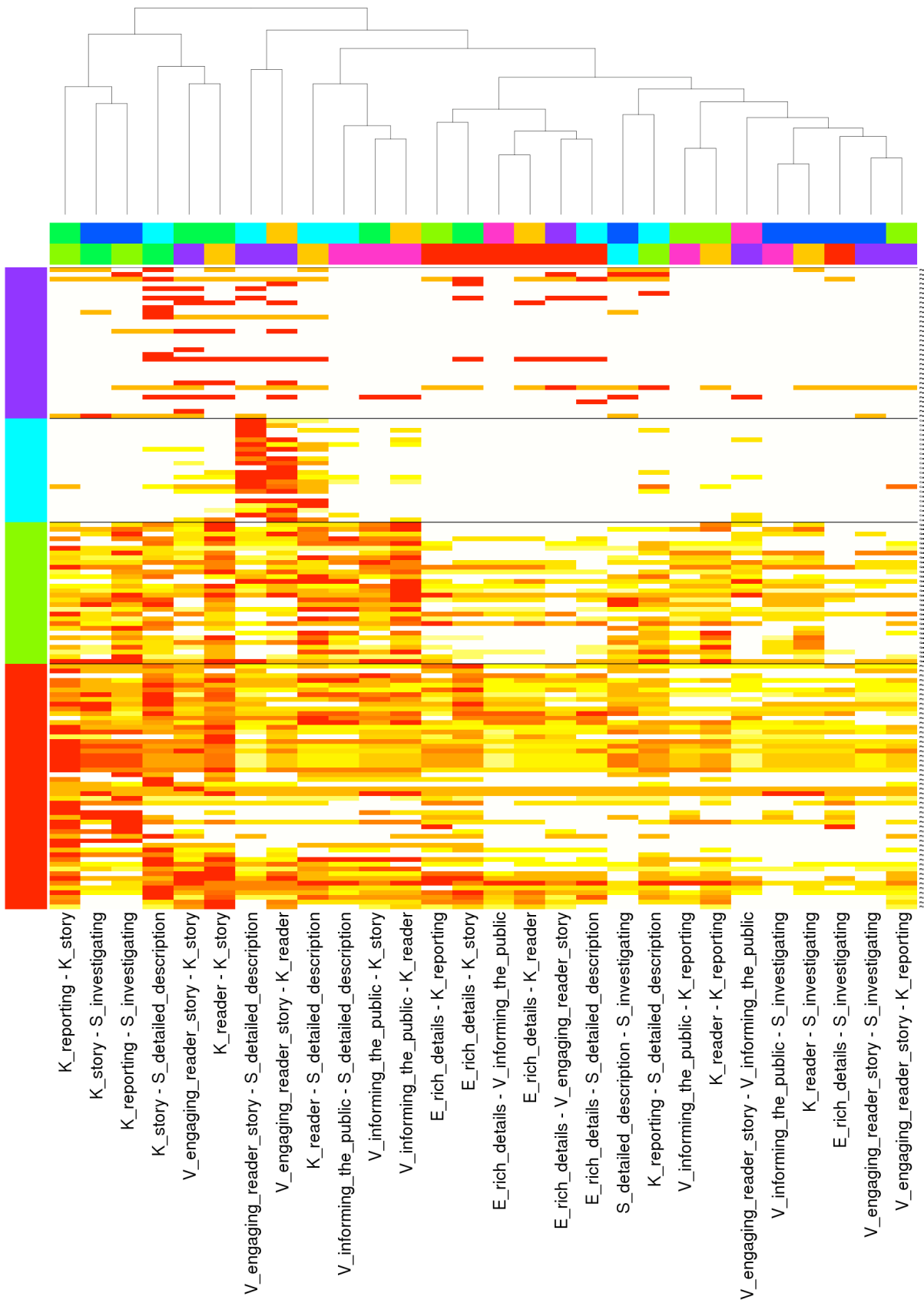
```
xcolname<-c()
for(i in 2:nCategory){
  for(j in 1:(i-1)){
    xcolname<-c(xcolname, paste(code[i], code[j], sep=' - '))
   }
}
```

```
xrowname<-c()
for(i in 1:n){
```

```
  xrowname<-c(xrowname, paste(data[(i*nCategory),'Venue'], data[(i*nCategory),
3],data[(i*nCategory),1], sep='-'))
}

colnames(X)<-xcolname
rownames(X)<-xrowname

## Draw heatmap
RowSideColors<-rainbow(nVenue)
RowSideColors<-rep(RowSideColors, time=cVenue)

png('heatmap.png', height=3000, width=1800)
myheatmap(X, Rowv=NA, scale='none',  margins=c(50,5), col=rev(heat.colors(100)), cexCol=4,
RowSideColors=RowSideColors, ColSideColors=rainbow(nCategory), cVenue=cVenue)
dev.off()
```

Sample data (see attached "all.csv" for the complete dataset)

```
"RefTag_Participant_ID","Venue","Story_Number","Root_Code","Story_Key","context-
code_key","Story_Code_Key","S_investigating","S_detailed_description","K_story","K_re
porting","K_reader","V_informing_the_public","V_engaging_reader_story","E_rich_detail
s"
51,"Practicum",1,"S_investigating","Practicum-1","Practicum-
S_investigating","Practicum-1-S_investigating",0,0,0,0,0,0,0,0
51,"Practicum",1,"S_detailed_description","Practicum-1","Practicum-
S_detailed_description","Practicum-1-S_detailed_description",0,0,20,0,0,0,0,0
51,"Practicum",1,"K_story","Practicum-1","Practicum-K_story","Practicum-1-K_story",
0,20,0,20,100,20,60,40
51,"Practicum",1,"K_reporting","Practicum-1","Practicum-K_reporting","Practicum-1-
K_reporting",0,0,20,0,20,0,20,20
51,"Practicum",1,"K_reader","Practicum-1","Practicum-K_reader","Practicum-1-
K_reader",0,0,100,20,0,40,60,40
51,"Practicum",1,"V_informing_the_public","Practicum-1","Practicum-
V_informing_the_public","Practicum-1-V_informing_the_public",0,0,20,0,40,0,0,20
51,"Practicum",1,"V_engaging_reader_story","Practicum-1","Practicum-
V_engaging_reader_story","Practicum-1-V_engaging_reader_story",0,0,60,20,60,0,0,20
51,"Practicum",1,"E_rich_details","Practicum-1","Practicum-
E_rich_details","Practicum-1-E_rich_details",0,0,40,20,40,20,20,0
52,"Practicum",1,"S_investigating","Practicum-1","Practicum-
S_investigating","Practicum-1-S_investigating",0,0,0,0,0,0,0,0
```


K_reporting - K_story
K_story - S_investigating
K_reporting - S_investigating
K_story - S_detailed_description
V_engaging_reader_story - K_story
K_reader - K_story
V_engaging_reader_story - S_detailed_description
V_engaging_reader_story - K_reader
K_reader - S_detailed_description
V_informing_the_public - S_detailed_description
V_informing_the_public - K_story
V_informing_the_public - K_reader
E_rich_details - K_reporting
E_rich_details - K_story
E_rich_details - V_informing_the_public
E_rich_details - K_reader
E_rich_details - V_engaging_reader_story
E_rich_details - S_detailed_description
S_detailed_description - S_investigating
K_reporting - S_detailed_description
V_informing_the_public - K_reporting
K_reader - K_reporting
V_engaging_reader_story - V_informing_the_public
V_informing_the_public - S_investigating
K_reader - S_investigating
E_rich_details - S_investigating
V_engaging_reader_story - S_investigating
V_engaging_reader_story - K_reporting

Cascading Triangles

This visualization came up by mixing following experiences, group work, personal perspectives and shallow understanding of domain problem.

- The first attempt was to draw "cascading triangle," (see the second screenshot below) to see how it works. But, this visualization does not compare networks very well and it could not convey all the nodes at the same time. Use of lightness by values seemed to improve the visualization.

- Others talked about using color for association strength but with structure of symmetric matrix maintained. Because of the structure, each cell needs to be comprised of multiple network values.

- A careful look at the data that the clients provided shows that a venue has several participants and each participant has the same form of symmetric matrix.

- It was not clear why the clients used co-occurrence instead of occurrence.

We decide to treating each cell in the matrix as one variable and to use color for association strength. This visualization was not tested at all levels. It was only tried with different color scheme. It turned out that it is difficult to compare between juxtaposed networks by the thickness and darkness as many people pointed out. Also, as the number of nodes grows, it would be more difficult to compare. Even with 2 networks with mirror image, the contrast does not pop out.

In a respect of Tufte's design principles, comparison of association strength between groups can be done in this visualization. This tried to integrate multiple networks by adding rows and pair information by encoding each variable with different color on the top of the main plot. It tried documentation by labeling but it could be improved by including data source information and signal strength legend, and labeling right-hand-side with venues and top-side with each variable. In the parallel axis, nodes are placed. For a fixed node A, the pairs are presented by linking the left and right parallel axis with the width and lightness proportional to the association strength. We can add a network by linking to the next parallel axis. On the rightmost side, scatterplot is presented for each pair grouped by venues. This visualization is sketched with processing. Since it is done only up to prototype, the code is somewhat incomplete (see "sketch_parallel_multiple.pde").

```
//sketch_parallel_multiple
```

```
Car myCar; Car myCar2; Car myCar3; Car myCar4;
```

```
Tri myTriG_12; Tri myTriG_13; Tri myTriG_14;
Tri myTriG_15; Tri myTriG_16; Tri myTriG_17;
Tri myTriG_18;
```

```
Tri myTriP_12; Tri myTriP_13; Tri myTriP_14; Tri myTriP_15;
Tri myTriP_16; Tri myTriP_17; Tri myTriP_18;
```

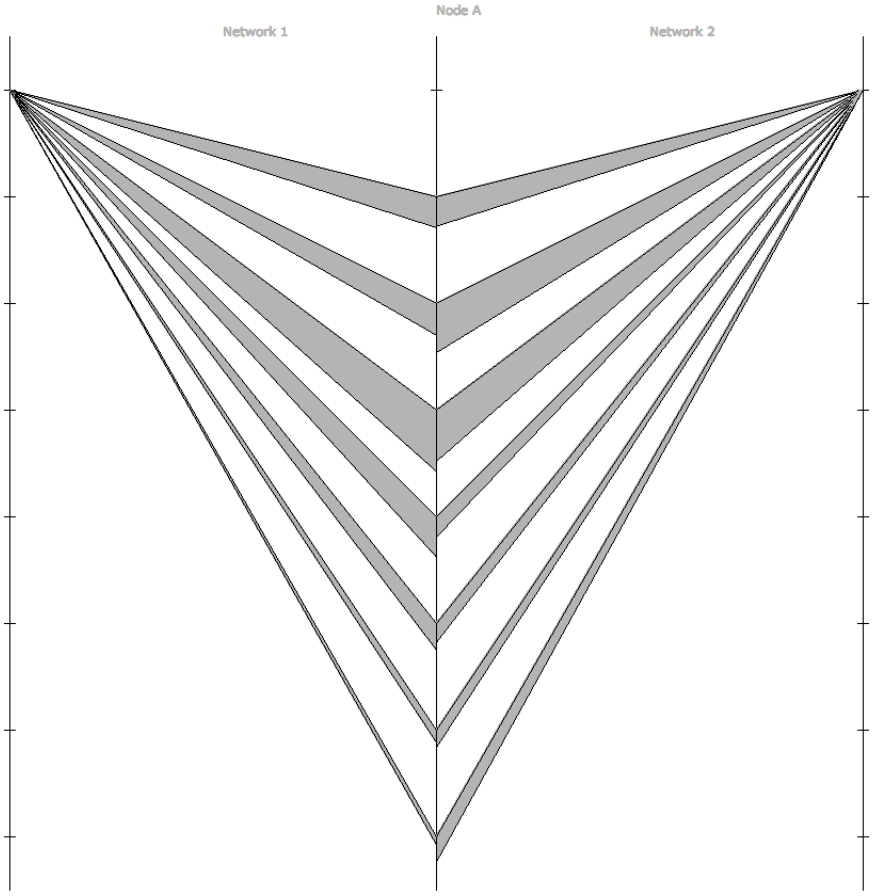
```
Tri myTriPeer_12; Tri myTriPeer_13; Tri myTriPeer_14;
Tri myTriPeer_15; Tri myTriPeer_16; Tri myTriPeer_17;
```

```
Tri myTriPeer_18;

PFont myFont;

void setup(){
  size(1600,900);
  background(255);
  float Net0=50; float Net1=450; float Net2=850; float Net3=1250;

  myCar = new Car(450); myCar2 = new Car(50); myCar3 = new Car(850);
  myCar4 = new Car(1250);

  //Game
  myTriG_12 = new Tri(Net0,Net1,1,2,29.01); myTriG_13 = new Tri(Net0,Net1,1,3,29.8);
  myTriG_14 = new Tri(Net0,Net1,1,4,57.54);myTriG_15 = new Tri(Net0,Net1,1,5,37.63);
  myTriG_16 = new Tri(Net0,Net1,1,6,24.78);myTriG_17 = new Tri(Net0,Net1,1,7, 11.57);
  myTriG_18 = new Tri(Net0,Net1,1,8,7.36);

  //Practicum
  myTriP_12 = new Tri(Net1,Net2,1,2,27.58);myTriP_13 = new Tri(Net1,Net2,1,3,46.07);
  myTriP_14 = new Tri(Net1,Net2,1,4,47.94);myTriP_15 = new Tri(Net1,Net2,1,5,18.98);
  myTriP_16 = new Tri(Net1,Net2,1,6,17.75);myTriP_17 = new Tri(Net1,Net2,1,7,16.18);
  myTriP_18 = new Tri(Net1,Net2,1,8,23.45);

  //Peer
  myTriPeer_12 = new Tri(Net2,Net3,1,2,47.37);
  myTriPeer_13 = new Tri(Net2,Net3, 1,3,72.18);
  myTriPeer_14 = new Tri(Net2,Net3,1,4,71.43);
  myTriPeer_15 = new Tri(Net2,Net3,1,5,31.58);
  myTriPeer_16 = new Tri(Net2,Net3,1,6,28.57);
  myTriPeer_17 = new Tri(Net2,Net3,1,7,24.06);
  myTriPeer_18 = new Tri(Net2,Net3,1,8,35.34);

  myFont = createFont("verdana",12);
  textFont(myFont);
}
```

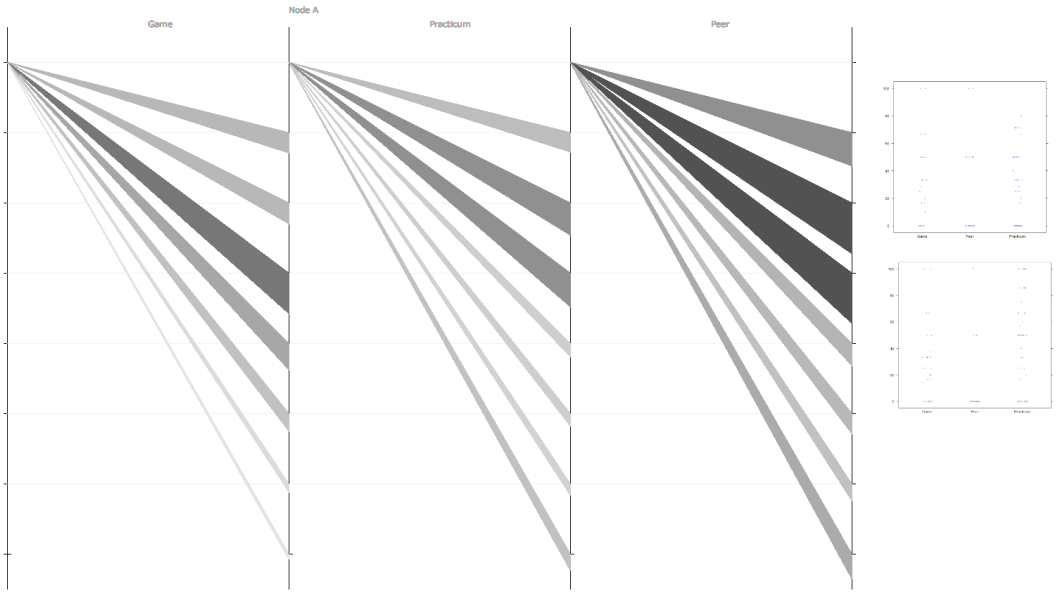
```
void draw(){
  stroke(0);
  myCar.display(); myCar2.display(); myCar3.display(); myCar4.display();

  float Net0=50; float Net1=450; float Net2=850; float Net3=1250;

  stroke(240);
  line(Net0, 100, Net3, 100); line(Net0, 200, Net3, 200);
  line(Net0, 300, Net3, 300); line(Net0, 400, Net3, 400);
  line(Net0, 500, Net3, 500); line(Net0, 600, Net3, 600);
  line(Net0, 700, Net3, 700);

  myTriG_12.display(); myTriG_13.display();
  myTriG_14.display(); myTriG_15.display();
  myTriG_16.display(); myTriG_17.display();
  myTriG_18.display();

  myTriP_12.display(); myTriP_13.display();
  myTriP_14.display(); myTriP_15.display();
  myTriP_16.display(); myTriP_17.display();
  myTriP_18.display();

  myTriPeer_12.display(); myTriPeer_13.display();
  myTriPeer_14.display(); myTriPeer_15.display();
  myTriPeer_16.display(); myTriPeer_17.display();
  myTriPeer_18.display();

  text("Node A", 450, 30);
  text("Game", (450+50)/2, 50);
  text("Practicum", (450+850)/2, 50);
  text("Peer", (850+1250)/2, 50);

  saveFrame("outputs.png");
}


//Vertical Line
```

```
class Car{
  float a1;

  Car(float b1){
    a1=b1;
  }

  void display(){
    line(a1,50,a1,850);
    float pos=100;
    while (pos<900){
       line(a1-5,pos,a1+5,pos);
       pos=pos+100;
    }
  }
}

//Triangle
class Tri{
  float x;
  float y;
  float association;
  int xnode;
  int ynode;
  int strength;
  color c;

  Tri(float Net1, float Net2, int Net1node, int Net2node, float Netassociation){
    x=Net1;
    y=Net2;
    xnode=Net1node;
    ynode=Net2node;
    association=Netassociation;
    strength= (int)(association);
    //int strength = (int)(strength);
    c = (int)(-2.4*(strength-100));
  }
```

```
  void display(){
    stroke(c);
    fill(c);
    triangle(x,xnode*100, y, ynode*100, y, ynode*100+association);
  }

}
```



The same information in other network is presented as linked to the next parallel axis.


Node A
Game
Practicum
Peer

A node-circle visualization created in Processing

Going over the domain paper from David et.al., we realized that one important visual comparison is showing how a node's centrality value (or weight) in a network relate to its strength with other elements of the network. The second important consideration that we had in mind was to have a solution that scale to more than 6 network. In this particular design, we tried to facilitate comparison at two different abstractions related to epistemic frame data. These are (i) a relative centrality of a node in different networks (ii) the strength of relationship between a node and other elements in a network.

Each circle shows a node from epistemic frame data. The area around the circle is divided in portions (wedges) based on number of networks to compare. The boundary of each wedge is color coded using saturation value of blue color. The saturation value shows the relative centrality (or weight) of a node in a given network. These values are normalized (divided by maximum weight across all networks being compared). The more saturated a color is a higher weight (close to 1) it has in that network.

Additionally, a node's individual comparison to other elements in a network are done using spikes. The length of the spikes indicates the relative strength value. These values are also normalized within a network. These spikes are always drawn from node A to the last node in a clock-wise direction. The comparison of the node to itself is not shown.

Design Principles:

Overall, we tried to based our decision choices in the design phase around specific design principles. Specially, the encoding was chosen based on Munzner's guidelines regarding the relative strength of visual channels for a given category of data. Below is a summary of some of the design choices made for this visualization.

(i) Length is one of the strong visual channel for quantitate value comparison. The length of spikes around the circle provide visual comparisons among node elements.

(ii) Circles are useful in saving space and providing proximity to different visual encodings.

(iii) Facilitate a global (centrality values across networks) as well as local comparisons (link strength between nodes).

(iv) Color saturation is a strong visual channel when comparing ordered data. It was used for approximately the same reason. However, we are not sure the way we are using relative centrality values comes under the category of ordered data.

Prototype Implementation

The prototype was implemented in Processing. The source code is attached.

Strength & Weakness

Strength: This particular visualization is simple and scale fairly well to multiple networks. Depending upon how relevant the measure of relative centrality value is for the domain scientist, a good mix of global and local comparison is available.

Weakness: It probably still lacks what Tufte's call "tell a story". The pop-out effect is limited. But that might have required more time and effort and a better visual design. Although, the visualization covers many different aspects of the domain problem but still misses few of the important ones e.g. progression of nodes across networks etc.

Further Improvements

Concluded from Comments

(i) The visualization need to have some extra information e.g. showing the name of the network and also making it explicit if networks and nodes are drawn clock-wise or anti-clockwise. They are drawn clock-wise. There is also need for a suitable legend for color saturation values. Also, we need to come up with a labeling scheme. In order to avoid clutter, perhaps, displaying labels on mouse over is preferred.

(ii) Distribution in spikes should be evenly distributed. Actually the distance between spikes of a network/node combination is equally distributed. The unevenness is due to the presence of zero value of a node in certain network.

(iii) Size of circle can be adjusted to save ink.

(iv) The alternative encoding for node strength with other elements using weight of spikes (instead of length) makes visual comparison harder. So length encoding is more preferred.

Other Improvements:

(v) Equipped with the recent knowledge about color specially the ColorBrewer we might be able to comes up with a better color encoding for centrality values.

(vi) A more interactive solution where users could decide placements of networks on specific wedges.

```
/**
nodecircle
 */

int wsize=600;
int radius= 100;
int normmode = 0;
int networks = 4;

float[][][] matrices = new float[4][][];
float[][] weights = new float[8][];
float max_elements[] = new float[matrices.length];
float centrality_range = 0.0;

String defaultMatrix1 = "./lsm_course.csv";
```

```
String defaultMatrix2 = "./lsm_game.csv";
String defaultMatrix3 = "./lsm_gff.csv";
String defaultMatrix4 = "./lsm_pract.csv";

void setup()
{

  matrices[0] = readmatrix(defaultMatrix1);
  matrices[1] = readmatrix(defaultMatrix2);
  matrices[2] = readmatrix(defaultMatrix3);
  matrices[3] = readmatrix(defaultMatrix4);

  for (int i=0; i<max_elements.length; i++) {
    max_elements[i] = minmaxmat(matrices[i])[1];
  }

  //calculate the weight matrix number of nodes * networks
  for (int i=0; i<weights.length; i++) {
    weights[i] = new float[networks];
    for (int j=0; j<networks; j++) {
      weights[i][j] = compute(i, matrices[j])/max_elements[j];
    }
  }

  float minmax_centrality[] = minmaxmat(weights);
  centrality_range = minmax_centrality[1] - minmax_centrality[0];
  size(wsize,wsize);
  noLoop();
}

float compute(int node, float[][] matrix) {
  float weight = 0.0;

  for (int i=0; i<matrix[node].length; i++) {
    weight += matrix[node][i];
  }
```

```
  return weight;
}

void draw()
{
  background(255);

  stroke(70,70,90);
  noFill();
  smooth();
  int angle = 360/networks;

  int cx = radius;
  int cy = radius;
  for (int n=0; n<8; n++) {
    int region = 0;
    ellipse(cx, cy, radius, radius);
    ellipse(cx, cy, radius+15, radius+15);

    for (int i=0; i<360; i+=angle) {
      float normalized_weight = weights[n][region]/centrality_range;
      drawfill(cx, cy, radius/2, (radius+15)/2, i, i+angle, normalized_weight, n, region);
      region++;
    }

    cx +=  (2*radius);
    if (cx > wsize) {
        cx = radius;
        cy += (2*radius);
    }
  }
}

void drawfill(int cx, int cy, int radius1, int radius2, int sangle, int eangle, float g, int
   node, int network) {

  colorMode(HSB, 360, 1, 150);
```

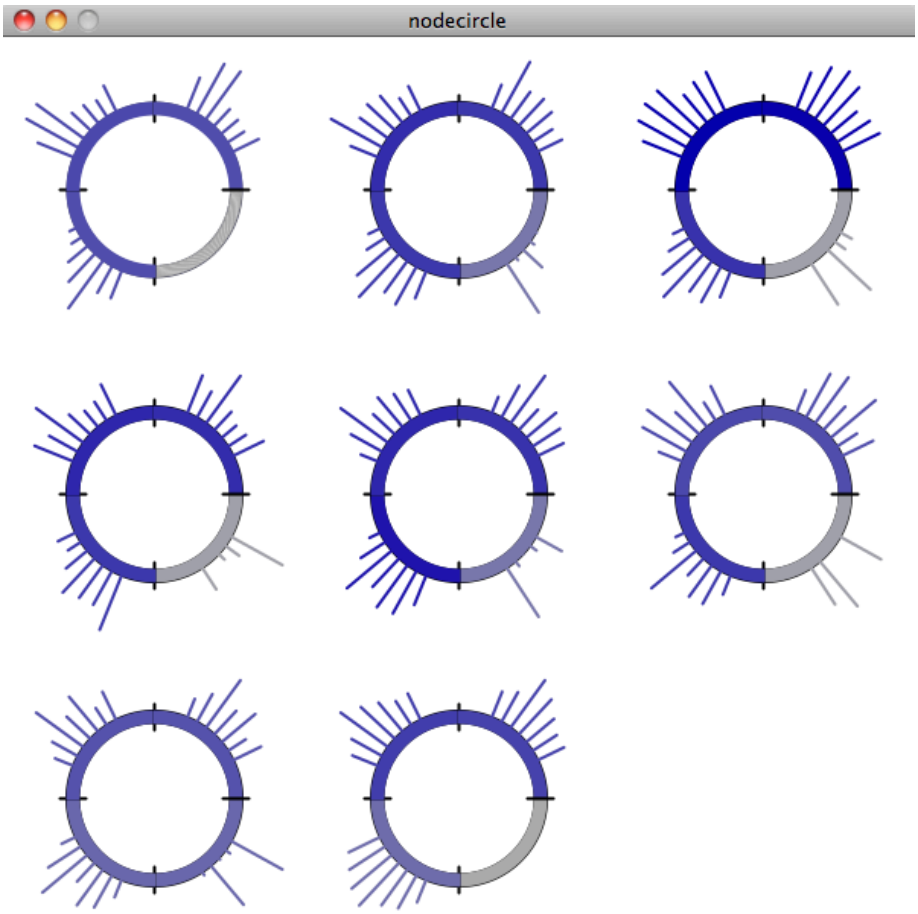
```
  for (int i=sangle; i<eangle; i++) {
    float rad = i * (PI/180);
    float x1 = cx + (radius1  * cos(rad));
    float y1 = cy + (radius1  * sin(rad));
    float x2 = cx + (radius2  * cos(rad));
    float y2 = cy + (radius2 * sin(rad));
    stroke(240.0, g, 100.0);
    line(x1, y1, x2, y2);
  }

  drawbars2(cx, cy, radius2, sangle, eangle, node, network);

  //draw a boundary line for this circle region.
  float rad = eangle * (PI/180);
  float bx1 = cx + ((radius1 - 5) * cos(rad));
  float by1 = cy + ((radius1 - 5) * sin(rad));
  float bx2 = cx + ((radius2 + 5) * cos(rad));
  float by2 = cy + ((radius2 + 5) * sin(rad));
  stroke(0);
  strokeWeight(2);
  line(bx1, by1, bx2, by2);

}

void drawbars(int cx, int cy, int radius2, int sangle, int eangle, int node, int network) {
  //compute the strength of the connection
  float nn[][] = new float[1][8];

  for (int i=0; i<8; i++) {
    if (i == node)
      continue;

    nn[0][i] = matrices[network][node][i];
  }

  float[] minmax = minmaxmat(nn);
```

```
  int subangles = (eangle - sangle) / 4;
  int angle = sangle + subangles;

  for (int j=0; j<8; j++) {
    if (node == j)
      continue;

    float rad = angle * (PI/180);
    float x1 = cx +  (radius2 * cos(rad));
    float y1 = cy + (radius2 * sin(rad));
    float x2 = cx + ((radius2+25) * cos(rad));
    float y2 = cy + ((radius2+25) * sin(rad));
    println(matrices[network][node][j]);
    strokeWeight(matrices[network][node][j]/(minmax[1]-minmax[0]) * 7);
    line(x1, y1, x2, y2);
    angle += 7;
  }
}

void drawbars2(int cx, int cy, int radius2, int sangle, int eangle, int node, int network) {
  //compute the strength of the connection
  float nn[][] = new float[1][8];

  for (int i=0; i<8; i++) {
    if (i == node)
      continue;

    nn[0][i] = matrices[network][node][i];
  }

  float[] minmax = minmaxmat(nn);


  int subangles = (eangle - sangle) / 4;
  int angle = sangle + subangles;

```

```
  for (int j=0; j<8; j++) {
    if (node == j)
      continue;

    float rad = angle * (PI/180);
    float x1 = cx +  (radius2 * cos(rad));
    float y1 = cy + (radius2 * sin(rad));

    float rad2 = radius2 + (matrices[network][node][j]/(minmax[1]-minmax[0]) * 39);
    println(rad2);

    float x2 = cx + (rad2 * cos(rad));
    float y2 = cy + (rad2 * sin(rad));
    println(matrices[network][node][j]);
    //strokeWeight(matrices[network][node][j]/(minmax[1]-minmax[0]) * 7);
    line(x1, y1, x2, y2);
    angle += 7;
  }
}
// matrixtools: simple tools for working with matrices
//From Michael's Original Visualization

public float[][] readmatrix(String filename)
{
  String[] lines = loadStrings(filename);

  if (lines != null) {
    print("File has "+lines.length+" lines"); println();
  } else
    return null;

  float[][] newmat = new float[lines.length][];

  for (int row = 0; row<newmat.length; row++) {
    String[] words = split(lines[row],',');
    newmat[row] = new float[words.length];
    for(int col=0; col<words.length; col++) {
```

```
      newmat[row][col] = float(words[col]);
    }
  }
  return newmat;
}

public float[] minmaxmat(float m[][])
{
  float[] result = new float[2];
  result[0] = m[0][0];
  result[1] = m[0][0];

  for(int row=0; row<m.length; row++) {
    for(int col=0; col<m[row].length; col++) {
      if (m[row][col] < result[0]) {
        result[0] = m[row][col];
      }
      if (m[row][col] > result[1]) {
        result[1] = m[row][col];
      }
    }
  }
  return result;
}

public float maxoffd(float m[][])
{
  float d=0;
  for(int i=0; i<m.length; i++)
    for(int j=i+1; j<m.length; j++)
      if (m[i][j] > d)
        d=m[i][j];
  return d;
}
```

The screen shot below is using four networks (lsm_source, lsm_game, lsm_gff, lsm_pract) from the EpistemicNet folder given by Michael. The visual elements on each circle are drawn in clock-wise direction.



An horizontal stacked chart animation created in a spreadsheet and Quicktime

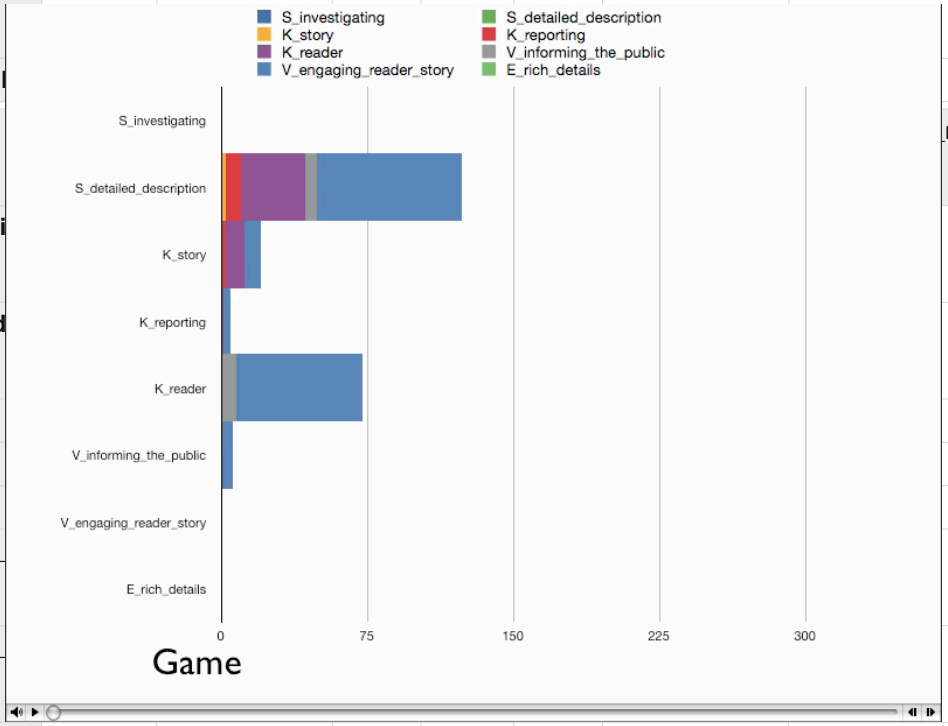
Each scene is a "venue" showing each of the stories plotted against rest of the stories. Stacked horizontal bar charts were created for each venue using the same color scheme for consistency of recall. The charts were imported into Apple Keynote as slides, slide transitions and timings were applied, then the presentation was exported to a Quicktime movie which was fine tuned in iMovie.

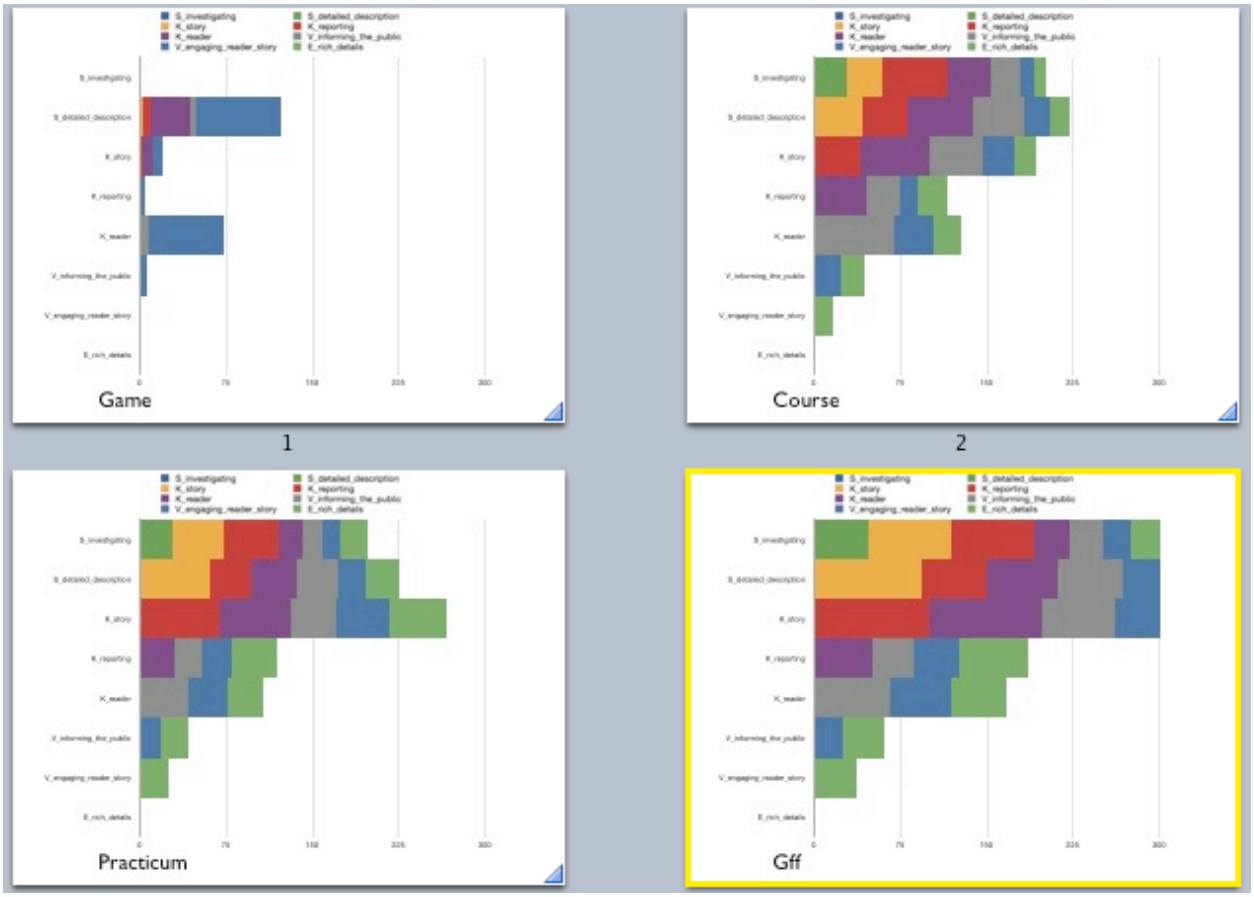
quality	S_investigating	S_detailed_description	K_story	K_reporting	K_reader	V_informing_the_public	V_engaging_reader_story	E_rich_details
S_investigating	0							
S_detailed_description	0	0						
K_story	0	2.272727273	0					
K_reporting	0	7.575757576	2.2727	0				
K_reader	0	33.21969697	9.6591	1.1364	0			
V_informing_the_public	0	6.060606061	0	0	7.5758	0		
V_engaging_reader_story	0	74.01515152	8.1061	3.4091	64.545	5.681818182	0	
E_rich_details	0	0	0	0	0	0	0	0

quality	S_investigating	S_detailed_description	K_story	K_reporting	K_reader	V_informing_the_public	V_engaging_reader_story	E_rich_details
S_investigating	0							
S_detailed_description	47.3684211	0						
K_story	72.1804511	93.9849624	0					
K_reporting	71.4285714	56.3909774	100	0				
K_reader	31.5789474	60.9022556	98.4962406	50.3759399	0			
V_informing_the_public	28.5714286	56.3909774	62.406015	36.8421053	65.4135338	0		

quality	S_investigating	S_detailed_description	K_story	K_reporting	K_reader	V_informing_the_public	V_engaging_reader_story	E_rich_details
V_engaging_reader_story	24.0601504	35.3383459	69.924812	39.0977444	54.1353384	24.8120301	0	
E_rich_details	35.3383459	45.112782	74.4360902	59.3984962	48.1203008	36.8421053	37.593985	0

quality	S_investigating	S_detailed_description	K_story	K_reporting	K_reader	V_informing_the_public	V_engaging_reader_story	E_rich_details
S_investigating	0							
S_detailed_description	29.00661	0						
K_story	29.80192	42.49239418	0					
K_reporting	57.53671	38.29365079	39.971	0				
K_reader	37.63294	57.24537037	61.013	46.151	0			
V_informing_the_public	24.77877	44.5462963	45.937	27.702	68.623	0		
V_engaging_reader_story	11.57407	22.22420635	26.631	15.764	35.06	23.52513228	0	
E_rich_details	23.45238	28.70421245	48.897	40.101	30.659	24.54212454	24.73901099	0




S_investigating
S_detailed_description
K_story
K_reporting
K_reader
V_informing_the_public
V_engaging_reader_story
E_rich_details
Game
1
Course
2
Practicum
Gff