Integrated Network Analysis

Jinxi_Hu-48528608, Samarth_Grover-38220463

2025-11-09

1. Environment and Dependencies

```
library(readr)
library(igraph)
library(RColorBrewer)
library(ggplot2)
library(reshape2)
library(scales)
library(gridExtra)
set.seed(48528608)
```

2. Raw Data Loading and Preprocessing (before removing isolated nodes)

```
# Load raw node and edge data
a_nodes <- read.csv("data/nodes.csv")
a_edges <- read.csv("data/edges.csv")

# Keep nodes with non-empty titles
a_nodes_clean <- subset(a_nodes, !(is.na(title) | trimws(title) == ""))

# Filter valid edges (source and target both in node set)
a_edges_clean <- subset(a_edges, source %in% a_nodes_clean$local_id &
                                 target %in% a_nodes_clean$local_id)

cat("=== RAW DATA ===\n")
```

```
## === RAW DATA ===
```

```
cat("Total nodes (including potential isolated):", nrow(a_nodes_clean), "\n")
```

```
## Total nodes (including potential isolated): 2610
```

```
cat("Valid citation edges:", nrow(a_edges_clean), "\n\n")
```

```
## Valid citation edges: 3757
```

```r
# Build undirected base graph (for overview and isolated node detection)
graph_raw <- graph_from_data_frame(a_edges_clean, vertices = a_nodes_clean, directed = FALSE)
# Remove multi-edges and self-loops
graph_raw <- simplify(graph_raw, remove.multiple = TRUE, remove.loops = TRUE)

cat("Raw undirected graph nodes:", vcount(graph_raw), "\n")
```

```
## Raw undirected graph nodes: 2610
```

```r
cat("Raw undirected graph edges:", ecount(graph_raw), "\n\n")
```

```
## Raw undirected graph edges: 3722
```

```r
# Compute isolated nodes
deg_all_raw <- degree(graph_raw, mode = "all")
raw_isolated <- V(graph_raw)[deg_all_raw == 0]
cat("Isolated nodes count:", length(raw_isolated), " (", round(length(raw_isolated)/vcount(graph_raw)*1
```

```
## Isolated nodes count: 1043  ( 39.96 %)
```

3. Connected Data After Removing Isolated Nodes (for directed advanced analysis)

```r
# Load cleaned connected node/edge data (already separated from isolated set)
nodes_connected <- read.csv("data/nodes_connected.csv")
edges_connected <- read.csv("data/edges_connected.csv")

cat("=== CONNECTED DATA ===\n")
```

```
## === CONNECTED DATA ===
```

```r
cat("Connected nodes:", nrow(nodes_connected), "\n")
```

```
## Connected nodes: 1582
```

```r
cat("Connected edges:", nrow(edges_connected), "\n\n")
```

```
## Connected edges: 3757
```

```r
# Build directed graph
graph_connected <- graph_from_data_frame(edges_connected, vertices = nodes_connected, directed = TRUE)
graph_connected <- simplify(graph_connected, remove.multiple = TRUE, remove.loops = TRUE)

cat("Directed graph nodes:", vcount(graph_connected), "\n")
```

```
## Directed graph nodes: 1582
```

```r
cat("Directed graph edges:", ecount(graph_connected), "\n")
```

```
## Directed graph edges: 3730
```

```r
cat("Network density:", round(edge_density(graph_connected), 6), "\n")
```

```
## Network density: 0.001491
```

```r
cat("Is directed:", is_directed(graph_connected), " | Is weighted:", is_weighted(graph_connected), "\n\n
```

```
## Is directed: TRUE  | Is weighted: FALSE
```

4. Global Degree Statistics and Distributions (directed connected graph)

```r
all_deg <- degree(graph_connected, mode = "all")
in_deg <- degree(graph_connected, mode = "in")
out_deg <- degree(graph_connected, mode = "out")

cat("=== GLOBAL DEGREE STATISTICS ===\n")
```

```
## === GLOBAL DEGREE STATISTICS ===
```

```r
cat("Mean total degree:", round(mean(all_deg), 2), " | Range:", min(all_deg), "-", max(all_deg), " | SD
```

```
## Mean total degree: 4.72  | Range: 0 - 110  | SD: 6.74
```

```r
cat("Mean in-degree:", round(mean(in_deg), 2), " | Range:", min(in_deg), "-", max(in_deg), " | SD:", rou
```

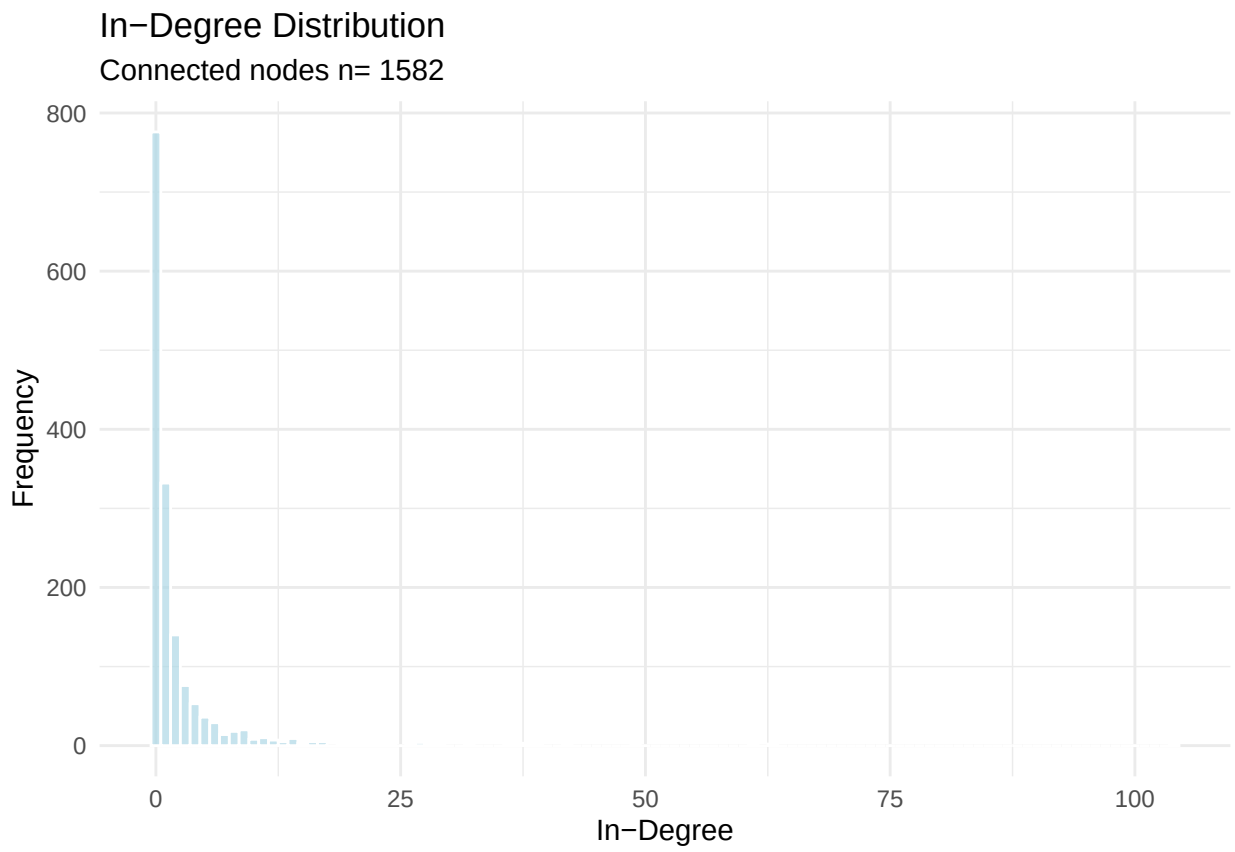
```
## Mean in-degree: 2.36  | Range: 0 - 104  | SD: 6.2
```

```r
cat("Mean out-degree:", round(mean(out_deg), 2), " | Range:", min(out_deg), "-", max(out_deg), " | SD:"
```

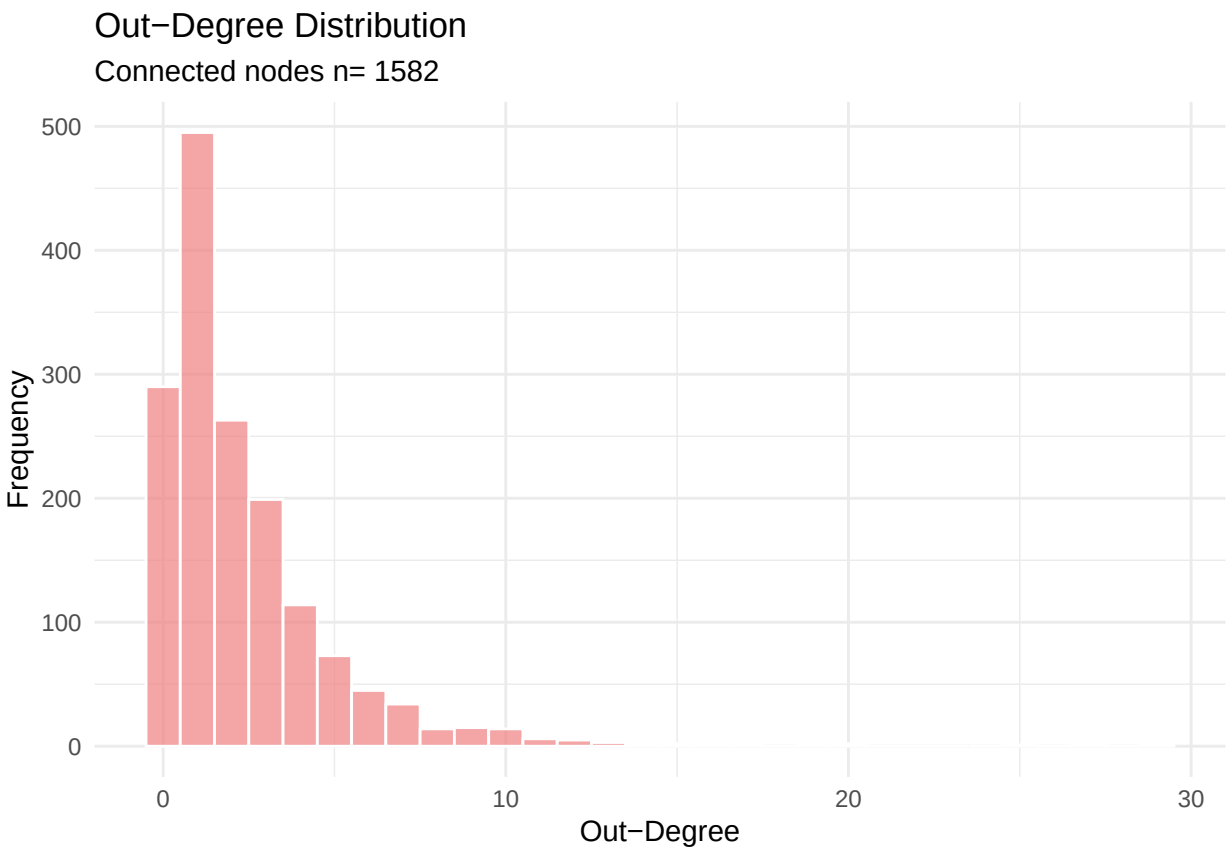
```
## Mean out-degree: 2.36  | Range: 0 - 29  | SD: 2.72
```

```r
degree_data <- data.frame(
  node_id = V(graph_connected)$name,
  total_degree = all_deg,
  in_degree = in_deg,
  out_degree = out_deg,
  institution = V(graph_connected)$institution,
  subtopic = V(graph_connected)$subtopic,
  year = V(graph_connected)$year,
  citations = V(graph_connected)$citations
)
```

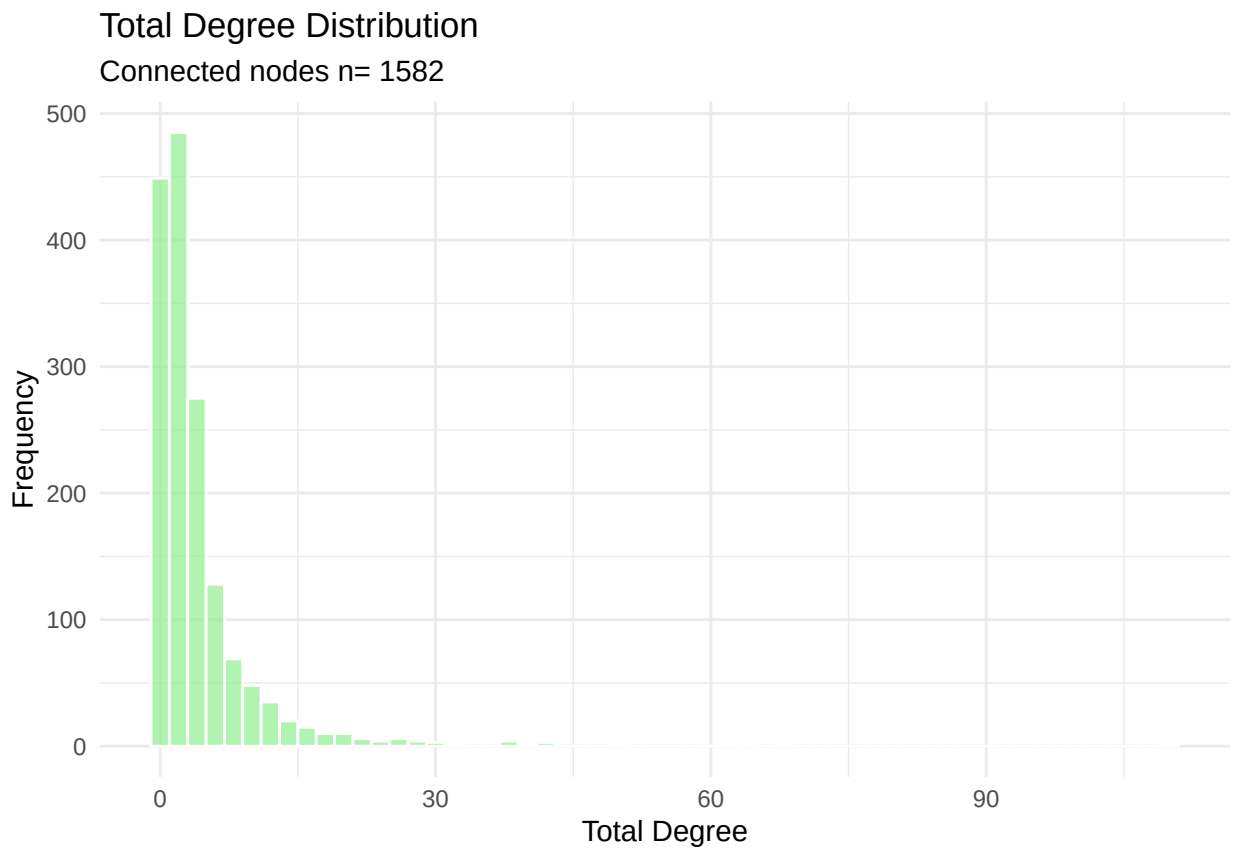
```r
# In-degree distribution
ggplot(degree_data, aes(x = in_degree)) +
  geom_histogram(binwidth = 1, fill = "lightblue", alpha = 0.7, color = "white") +
  labs(title = "In-Degree Distribution", subtitle = paste("Connected nodes n=", vcount(graph_connected)),
       x = "In-Degree", y = "Frequency") + theme_minimal()
```




```r
# Out-degree distribution
ggplot(degree_data, aes(x = out_degree)) +
  geom_histogram(binwidth = 1, fill = "lightcoral", alpha = 0.7, color = "white") +
  labs(title = "Out-Degree Distribution", subtitle = paste("Connected nodes n=", vcount(graph_connected)),
       x = "Out-Degree", y = "Frequency") + theme_minimal()
```



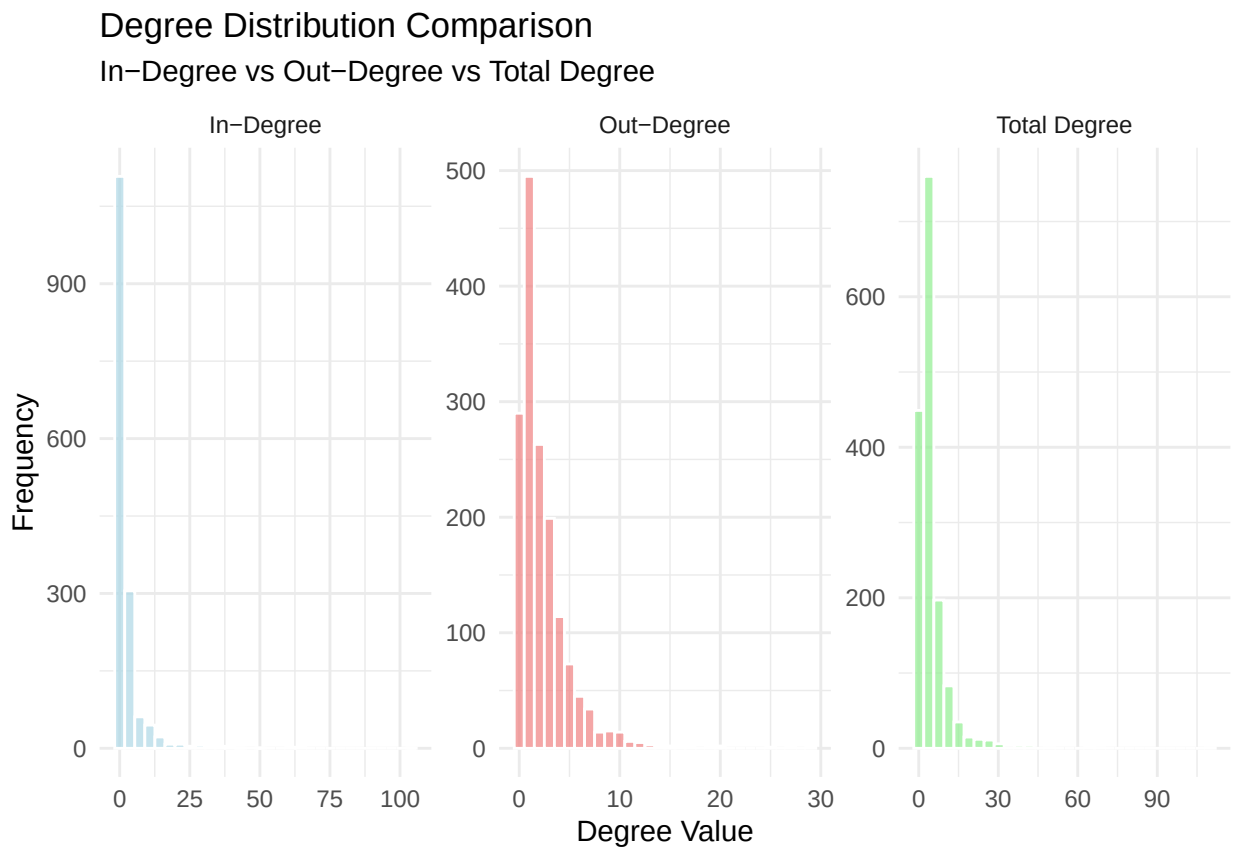

```r
# Total-degree distribution
ggplot(degree_data, aes(x = total_degree)) +
  geom_histogram(binwidth = 2, fill = "lightgreen", alpha = 0.7, color = "white") +
  labs(title = "Total Degree Distribution", subtitle = paste("Connected nodes n=", vcount(graph_connecte
       x = "Total Degree", y = "Frequency") + theme_minimal()
```




```
degree_long <- melt(degree_data[, c("in_degree", "out_degree", "total_degree")],
                    variable.name = "degree_type", value.name = "degree_value")
```

```
## No id variables; using all as measure variables
```

```
degree_long$degree_type <- factor(degree_long$degree_type,
                                  levels = c("in_degree", "out_degree", "total_degree"),
                                  labels = c("In-Degree", "Out-Degree", "Total Degree"))

ggplot(degree_long, aes(x = degree_value, fill = degree_type)) +
  geom_histogram(alpha = 0.7, color = "white", bins = 30) +
  facet_wrap(~degree_type, scales = "free") +
  scale_fill_manual(values = c("In-Degree" = "lightblue", "Out-Degree" = "lightcoral", "Total Degree" =
  labs(title = "Degree Distribution Comparison", subtitle = "In-Degree vs Out-Degree vs Total Degree",
  theme_minimal() + theme(legend.position = "none")
```



5. Component Analysis (directed graph)

```
cat("=== COMPONENT ANALYSIS (DIRECTED GRAPH) ===\n")
```

```
## === COMPONENT ANALYSIS (DIRECTED GRAPH) ===
```

```
weak_comp <- components(graph_connected, mode = "weak")
strong_comp <- components(graph_connected, mode = "strong")

cat("Weakly connected components:", weak_comp$no, " | Largest weak component size:", max(weak_comp$csiz
```

```
## Weakly connected components: 58  | Largest weak component size: 1433  ( 90.58 %)
```

```
cat("Strongly connected components:", strong_comp$no, " | Largest strong component size:", max(strong_c
```

```
## Strongly connected components: 1569  | Largest strong component size: 6  ( 0.38 %)
```

```
cat("Weak component sizes (Top 10):\n")
```

```
## Weak component sizes (Top 10):
```

```r
print(head(sort(weak_comp$csize, decreasing = TRUE), 10))
```

```
##  [1] 1433   27    9    5    5    4    3    3    3    3
```

```r
cat("\nStrong component sizes (Top 10):\n")
```

```
##
## Strong component sizes (Top 10):
```

```r
print(head(sort(strong_comp$csize, decreasing = TRUE), 10))
```

```
##  [1] 6 2 2 2 2 2 2 2 2 1
```

```r
largest_nodes <- which(weak_comp$membership == which.max(weak_comp$csize))
largest_component <- induced_subgraph(graph_connected, largest_nodes)
cat("\n=== LARGEST WEAK COMPONENT STATISTICS ===\n")
```

```
##
## === LARGEST WEAK COMPONENT STATISTICS ===
```

```r
cat("Nodes:", vcount(largest_component), " | Edges:", ecount(largest_component), "\n")
```

```
## Nodes: 1433  | Edges: 3634
```

```r
cat("Density:", round(edge_density(largest_component), 6), "\n")
```

```
## Density: 0.001771
```

```r
cat("Average in-degree:", round(mean(degree(largest_component, mode = "in")), 2), " | Average out-degree
```

```
## Average in-degree: 2.54  | Average out-degree: 2.54
```

```r
cat("Average total degree:", round(mean(degree(largest_component, mode = "all")), 2), "\n")
```

```
## Average total degree: 5.07
```

```r
cat("Diameter:", diameter(largest_component, directed = FALSE), " | Average path length:", round(mean_d
```

```
## Diameter: 13  | Average path length: 4.5
```

6. Centrality Analysis of Largest Component

```r
cat("=== CENTRALITY (LARGEST COMPONENT) ===\n")
```

```
## === CENTRALITY (LARGEST COMPONENT) ===
```

```r
cat("Calculating centralities...\n")
```

```
## Calculating centralities...
```

```r
c_deg_in <- degree(largest_component, mode = "in", normalized = TRUE)
c_deg_out <- degree(largest_component, mode = "out", normalized = TRUE)
c_deg_all <- degree(largest_component, mode = "all", normalized = TRUE)
c_between <- betweenness(largest_component, directed = TRUE, normalized = TRUE)
c_close_in <- closeness(largest_component, mode = "in", normalized = TRUE)
c_close_out <- closeness(largest_component, mode = "out", normalized = TRUE)
c_eigen <- eigen_centrality(largest_component, directed = TRUE, scale = TRUE)$vector
```

```
## Warning: The `scale` argument of `eigen_centrality()` is deprecated as of igraph 2.1.1.
## i eigen_centrality() will always behave as if scale=TRUE were used.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
c_pagerank <- page_rank(largest_component, directed = TRUE)$vector
c_hits_hub <- hub_score(largest_component)$vector
```

```
## Warning: `hub_score()` was deprecated in igraph 2.0.3.
## i Please use `hits_scores()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
c_hits_auth <- authority_score(largest_component)$vector
```

```
## Warning: `authority_score()` was deprecated in igraph 2.1.0.
## i Please use `hits_scores()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
centrality_summary <- data.frame(
  node_id = V(largest_component)$name,
  degree_in = c_deg_in,
  degree_out = c_deg_out,
  degree_all = c_deg_all,
  betweenness = c_between,
  closeness_in = c_close_in,
  closeness_out = c_close_out,
  eigenvector = c_eigen,
  pagerank = c_pagerank,
  hub = c_hits_hub,
  authority = c_hits_auth
)

cat("Top 5 In-Degree Centrality:\n"); print(head(centrality_summary[order(centrality_summary$degree_in,
```

```
## Top 5 In-Degree Centrality:

##       node_id  degree_in
## P0060   P0060 0.07262570
## P0001   P0001 0.04399441
## P0009   P0009 0.04399441
## P0002   P0002 0.04259777
## P0199   P0199 0.04259777
```

```
cat("\nTop 5 PageRank:\n"); print(head(centrality_summary[order(centrality_summary$pagerank, decreasing=
```

```
##
## Top 5 PageRank:

##       node_id   pagerank
## P0199   P0199 0.04046559
## P0031   P0031 0.03232105
## P0741   P0741 0.01974340
## P0116   P0116 0.01746391
## P0535   P0535 0.01572970
```

```
cat("\nTop 5 Betweenness Centrality:\n"); print(head(centrality_summary[order(centrality_summary$betwee
```

```
##
## Top 5 Betweenness Centrality:

##       node_id betweenness
## P0011   P0011 0.002580361
## P0060   P0060 0.002094395
## P0003   P0003 0.001932865
## P0002   P0002 0.001732924
## P0001   P0001 0.001339179
```

```
cat("\nTop 5 Authority Score:\n"); print(head(centrality_summary[order(centrality_summary$authority, de
```

```
##
## Top 5 Authority Score:

##       node_id authority
## P0009   P0009 1.0000000
## P0060   P0060 0.9173024
## P0017   P0017 0.6726492
## P0014   P0014 0.6544079
## P0006   P0006 0.6283538
```

```
cat("=== Betweenness Centrality Distribution (Largest Component) ===\n")
```

```
## === Betweenness Centrality Distribution (Largest Component) ===
```

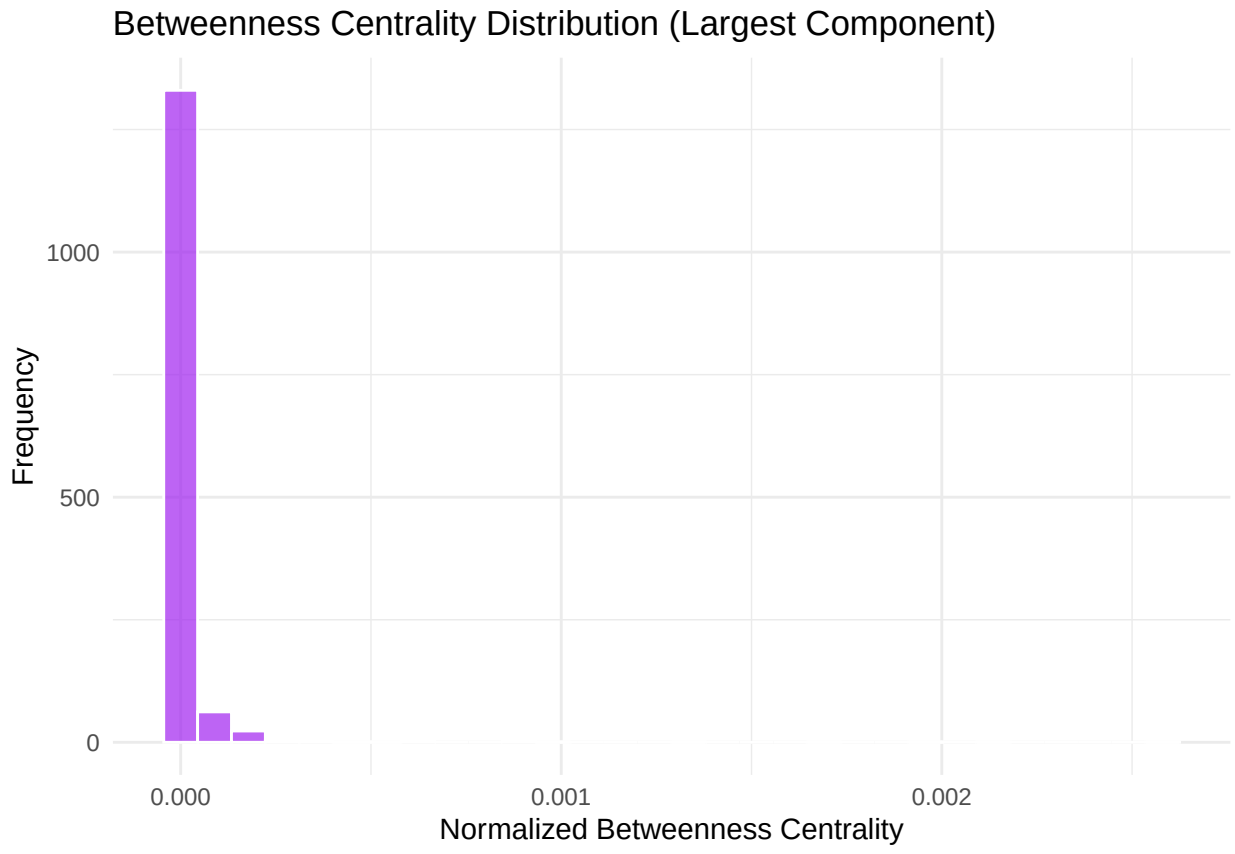
```r
bet_stats <- summary(c_between)
print(bet_stats)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.00e+00 0.00e+00 0.00e+00 2.12e-05 3.66e-06 2.58e-03
```

```r
q95 <- quantile(c_between, 0.95)
high_between_ids <- which(c_between >= q95)
cat("Nodes in top 5% betweenness:", length(high_between_ids), "\n")
```

```
## Nodes in top 5% betweenness: 72
```

```r
bet_df <- data.frame(node_id = V(largest_component)$name,
                     betweenness = c_between,
                     institution = V(largest_component)$institution,
                     subtopic = V(largest_component)$subtopic)

ggplot(bet_df, aes(x = betweenness)) +
  geom_histogram(bins = 30, fill = "purple", alpha = 0.7, color = "white") +
  labs(title = "Betweenness Centrality Distribution (Largest Component)", x = "Normalized Betweenness C
```




```r
cat("\nTop 10 bridging nodes (betweenness):\n")
```

```
##
## Top 10 bridging nodes (betweenness):
```

```
print(head(bet_df[order(bet_df$betweenness, decreasing = TRUE), c("node_id","betweenness","subtopic","in
```

```
##       node_id  betweenness                                              subtopic
## P0011   P0011 0.0025803608                       Machine Learning in Healthcare
## P0060   P0060 0.0020943953 Artificial Intelligence in Healthcare and Education
## P0003   P0003 0.0019328651 Artificial Intelligence in Healthcare and Education
## P0002   P0002 0.0017329242 Artificial Intelligence in Healthcare and Education
## P0001   P0001 0.0013391788                       Machine Learning in Healthcare
## P0192   P0192 0.0009925863 Artificial Intelligence in Healthcare and Education
## P0484   P0484 0.0008632584 Artificial Intelligence in Healthcare and Education
## P0054   P0054 0.0008526921 Artificial Intelligence in Healthcare and Education
## P0136   P0136 0.0008494446 Artificial Intelligence in Healthcare and Education
## P0021   P0021 0.0006611386 Artificial Intelligence in Healthcare and Education
##                   institution
## P0011     Stanford University
## P0060 University Of Cambridge
## P0003         Yale University
## P0002      Harvard University
## P0001     Stanford University
## P0192  Mayo Clinic In Arizona
## P0484    Stanford Health Care
## P0054     Stanford University
## P0136     Stanford University
## P0021        Emory University
```

```
cat("=== Edge Betweenness (Largest Component) ===\n")
```

```
## === Edge Betweenness (Largest Component) ===
```

```
edge_bet <- edge_betweenness(largest_component, directed = TRUE)
cat("Number of edges:", length(edge_bet), "\n")
```

```
## Number of edges: 3634
```

```
print(summary(edge_bet))
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    1.000    2.000    5.667   25.071   17.462 2536.235
```

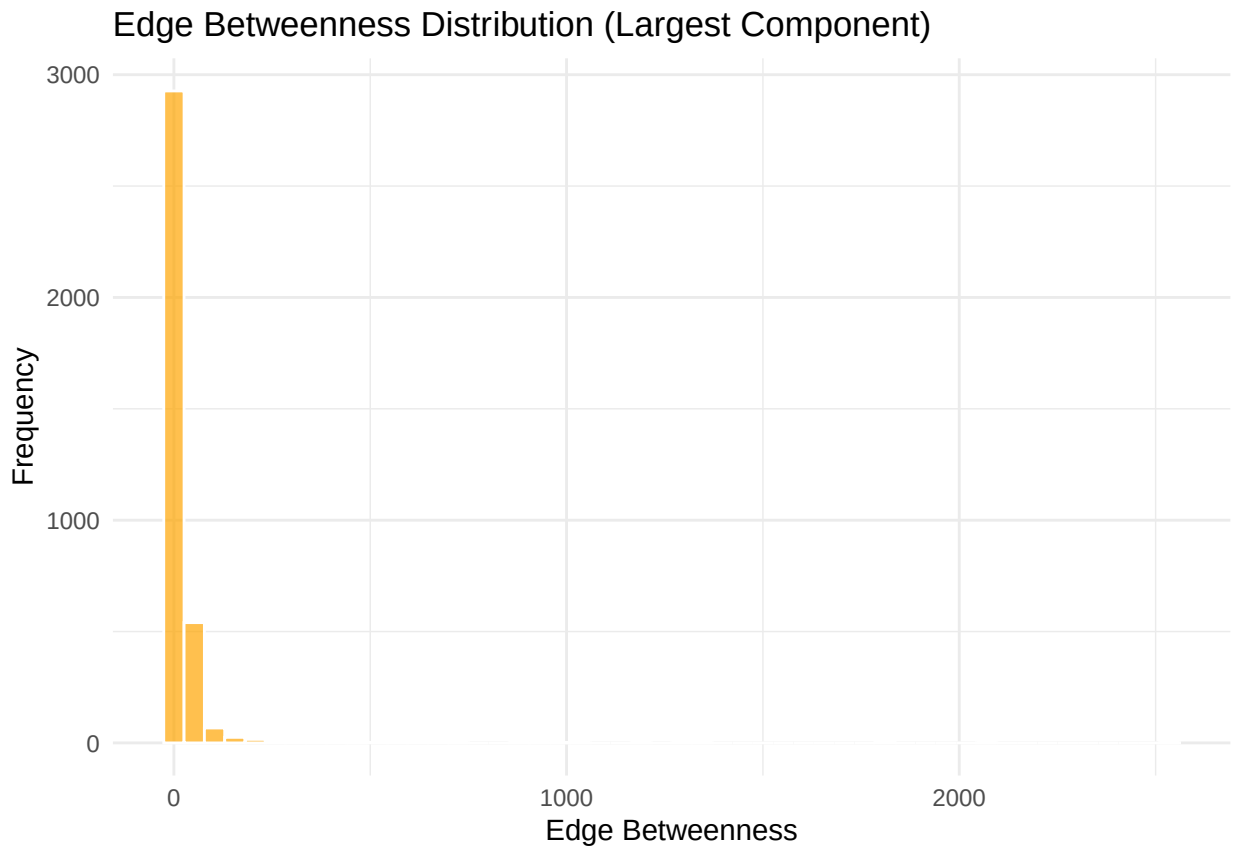
```
idx_top_e <- order(edge_bet, decreasing = TRUE)[1:10]
edge_pairs <- get.edges(largest_component, idx_top_e)
cat("\nTop 10 bridging edges: \n")
```

```
##
## Top 10 bridging edges:
```

```
for(i in seq_along(idx_top_e)){
  from_n <- V(largest_component)$name[edge_pairs[i,1]]
  to_n <- V(largest_component)$name[edge_pairs[i,2]]
  cat(sprintf("%d. %s -> %s (%.4f)\n", i, from_n, to_n, edge_bet[idx_top_e[i]]))
}
```

```
## 1. P0001 -> P0003 (2536.2345)
## 2. P0003 -> P0054 (2051.3298)
## 3. P0484 -> P0011 (1825.9821)
## 4. P0192 -> P0136 (1811.0000)
## 5. P0011 -> P0192 (1789.1667)
## 6. P0003 -> P0021 (1346.8333)
## 7. P0002 -> P0107 (1293.0250)
## 8. P0060 -> P0001 (1211.1667)
## 9. P0060 -> P0002 (1039.8810)
## 10. P0011 -> P0003 (1002.4452)
```

```
ggplot(data.frame(edge_betweenness = edge_bet), aes(x = edge_betweenness)) +
  geom_histogram(bins = 50, fill = "orange", alpha = 0.7, color = "white") +
  labs(title = "Edge Betweenness Distribution (Largest Component)", x = "Edge Betweenness", y = "Freque
```



7. Community Detection Comparison (Largest Component)

```
cat("=== COMMUNITY DETECTION (Louvain vs Edge Betweenness) ===\n")
```

```
## === COMMUNITY DETECTION (Louvain vs Edge Betweenness) ===
```

```
largest_undirected <- as.undirected(largest_component, mode = "collapse")
```

```
## Warning: `as.undirected()` was deprecated in igraph 2.1.0.
## i Please use `as_undirected()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

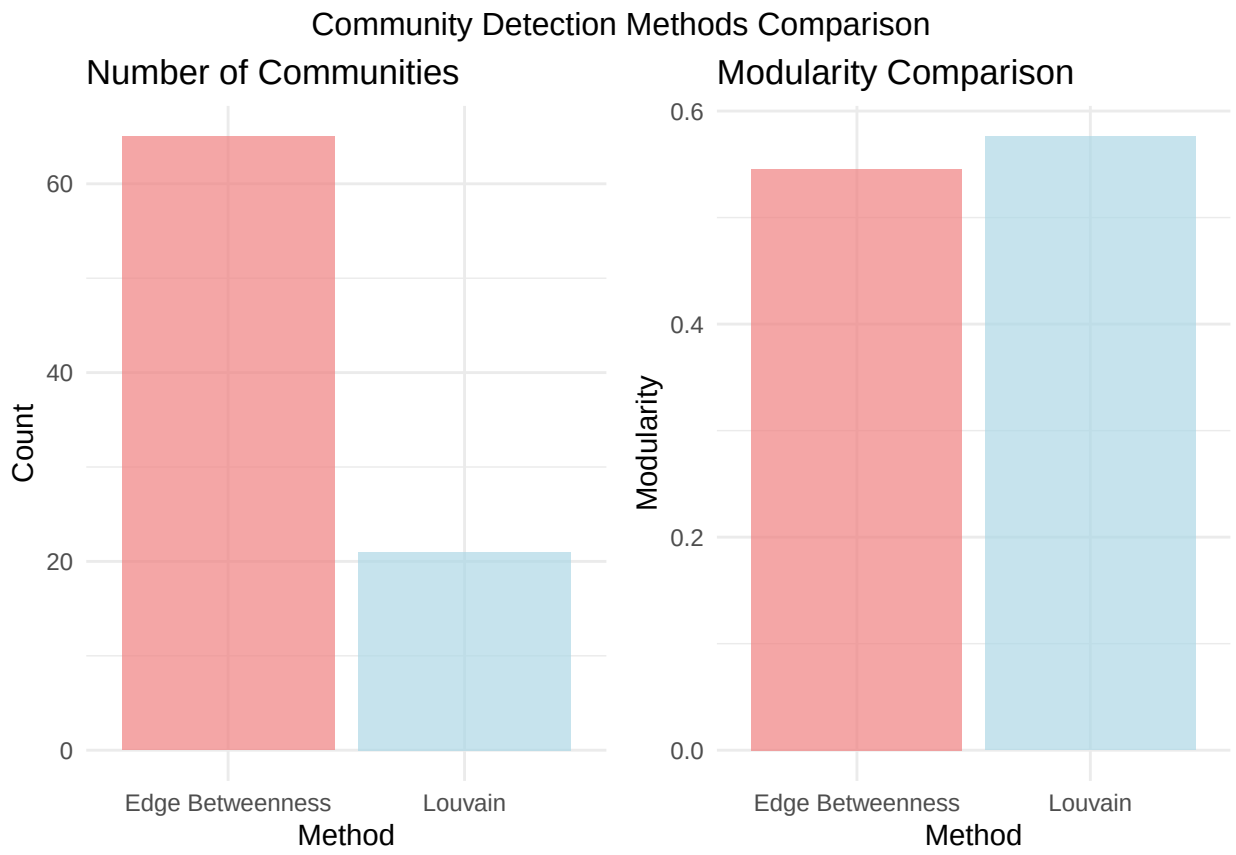
```
louvain_comm <- cluster_louvain(largest_undirected)
edgebet_comm <- cluster_edge_betweenness(largest_undirected, directed = FALSE)

cat("Louvain communities:", length(louvain_comm), " | Modularity:", round(modularity(louvain_comm),4), "
```

```
## Louvain communities: 21  | Modularity: 0.576
```

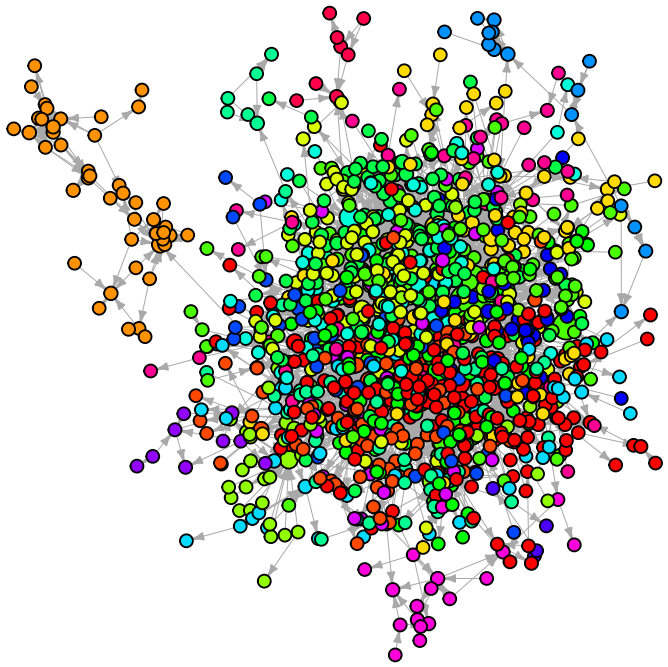
```
cat("Edge Betweenness communities:", length(edgebet_comm), " | Modularity:", round(modularity(edgebet_c
```

```
## Edge Betweenness communities: 65  | Modularity: 0.5455
```

```
comm_compare <- data.frame(
  Method = c("Louvain", "Edge Betweenness"),
  Communities = c(length(louvain_comm), length(edgebet_comm)),
  Modularity = c(modularity(louvain_comm), modularity(edgebet_comm))
)

p1 <- ggplot(comm_compare, aes(x = Method, y = Communities, fill = Method)) +
  geom_bar(stat = "identity", alpha = 0.7) +
  scale_fill_manual(values = c("Louvain" = "lightblue", "Edge Betweenness" = "lightcoral")) +
  labs(title = "Number of Communities", y = "Count") + theme_minimal() + theme(legend.position = "none")

p2 <- ggplot(comm_compare, aes(x = Method, y = Modularity, fill = Method)) +
  geom_bar(stat = "identity", alpha = 0.7) +
  scale_fill_manual(values = c("Louvain" = "lightblue", "Edge Betweenness" = "lightcoral")) +
  labs(title = "Modularity Comparison", y = "Modularity") + theme_minimal() + theme(legend.position = "

grid.arrange(p1, p2, ncol = 2, top = "Community Detection Methods Comparison")
```




```r
# Louvain visualization + PageRank node sizing
pagerank_vals <- page_rank(largest_component, directed = TRUE)$vector
size_scaled <- 4 + 10 * (pagerank_vals - min(pagerank_vals)) / (max(pagerank_vals) - min(pagerank_vals))
comm_mem <- membership(louvain_comm)
if(length(louvain_comm) <= 12){
  comm_cols <- brewer.pal(max(3, length(louvain_comm)), "Set1")
} else {
  comm_cols <- rainbow(length(louvain_comm))
}
vertex_cols_comm <- comm_cols[comm_mem]
layout_large <- layout_with_fr(largest_component, niter = 800)

plot(largest_component, layout = layout_large,
     vertex.size = size_scaled,
     vertex.label = NA,
     edge.arrow.size = 0.3,
     edge.width = 0.5,
     vertex.color = vertex_cols_comm,
    main = "Largest Component: Louvain Communities + PageRank Node Size",
    sub = paste("Communities:", length(louvain_comm), " Modularity:", round(modularity(louvain_comm),4))
```



8. Centrality Correlations

```
cat("=== CENTRALITY CORRELATION MATRIX ===\n")
```

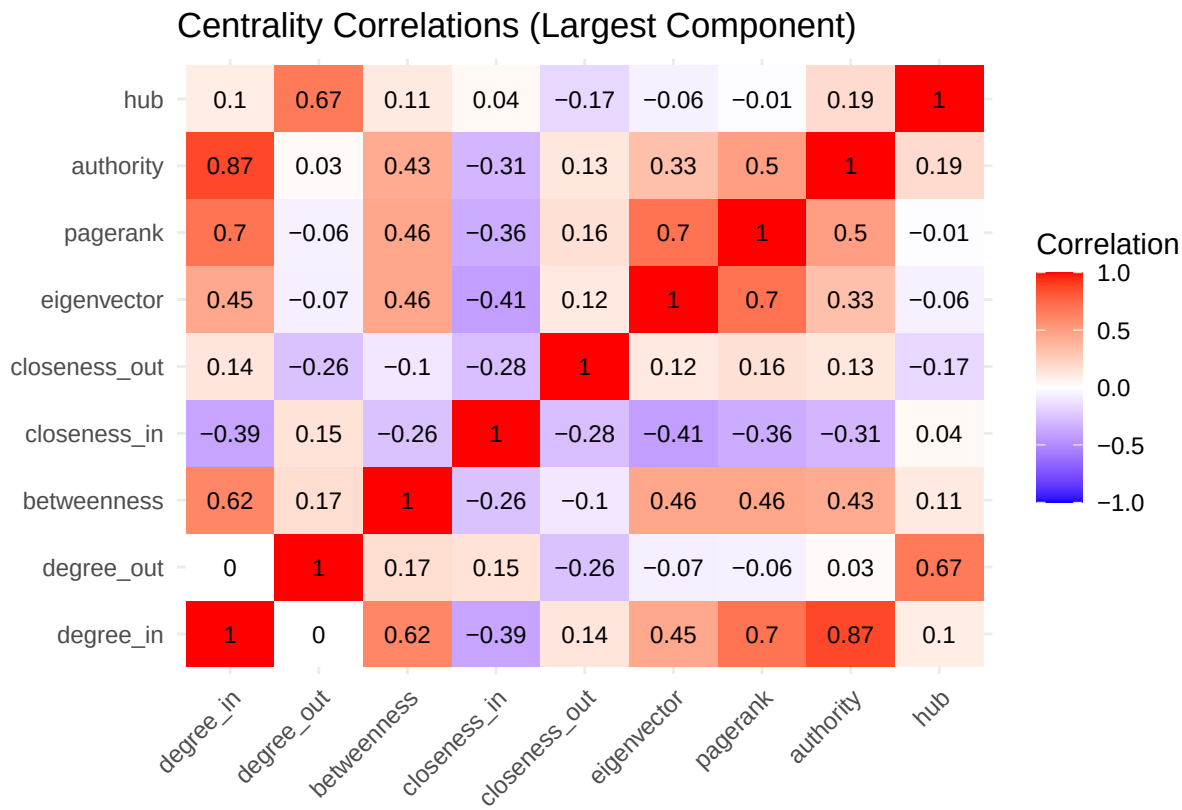
```
## === CENTRALITY CORRELATION MATRIX ===
```

```
cor_data <- centrality_summary[, c("degree_in","degree_out","betweenness","closeness_in","closeness_out
cor_mat <- cor(cor_data, use = "complete.obs")
print(round(cor_mat,3))
```

```
##               degree_in degree_out betweenness closeness_in closeness_out
## degree_in         1.000      0.002       0.616       -0.390          0.138
## degree_out        0.002      1.000       0.173        0.152         -0.262
## betweenness       0.616      0.173       1.000       -0.260         -0.101
## closeness_in     -0.390      0.152      -0.260        1.000         -0.277
## closeness_out     0.138     -0.262      -0.101       -0.277          1.000
## eigenvector       0.449     -0.068       0.458       -0.412          0.115
## pagerank          0.696     -0.057       0.460       -0.356          0.162
## authority         0.869      0.029       0.435       -0.311          0.125
```

```
## hub               0.096      0.670       0.110        0.036        -0.170
##               eigenvector pagerank authority    hub
## degree_in           0.449    0.696     0.869  0.096
## degree_out         -0.068   -0.057     0.029  0.670
## betweenness         0.458    0.460     0.435  0.110
## closeness_in       -0.412   -0.356    -0.311  0.036
## closeness_out       0.115    0.162     0.125 -0.170
## eigenvector         1.000    0.701     0.331 -0.060
## pagerank            0.701    1.000     0.498 -0.012
## authority           0.331    0.498     1.000  0.187
## hub                -0.060   -0.012     0.187  1.000
```

```r
cor_melt <- melt(cor_mat)

ggplot(cor_melt, aes(Var1, Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0, limit = c(-1,1), name =
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = "Centrality Correlations (Largest Component)", x = "", y = "") +
  geom_text(aes(label = round(value,2)), size = 3)
```



9. Isolated vs Connected Nodes Comparison

```r
# Load isolated nodes data
nodes_isolated <- read.csv("data/nodes_isolated.csv")
cat("=== ISOLATED NODE STATISTICS ===\n")
```

```
## === ISOLATED NODE STATISTICS ===
```

```r
cat("Isolated nodes:", nrow(nodes_isolated), " | Year range:", min(nodes_isolated$year,na.rm=TRUE), "-"
```

```
## Isolated nodes: 1028  | Year range: 2015 - 2025
```

```r
cat("Mean citations (isolated):", round(mean(nodes_isolated$citations, na.rm=TRUE),2), " | Mean referen
```

```
## Mean citations (isolated): 9.97  | Mean references (isolated): 22.48
```

```r
#
cat("=== CONNECTED NODE STATISTICS ===\n")
```

```
## === CONNECTED NODE STATISTICS ===
```

```r
cat("Connected nodes:", nrow(nodes_connected), " | Year range:", min(nodes_connected$year,na.rm=TRUE),
```

```
## Connected nodes: 1582  | Year range: 2015 - 2025
```

```r
cat("Mean citations (connected):", round(mean(nodes_connected$citations, na.rm=TRUE),2), " | Mean refer
```

```
## Mean citations (connected): 50.88  | Mean references (connected): 45.12
```

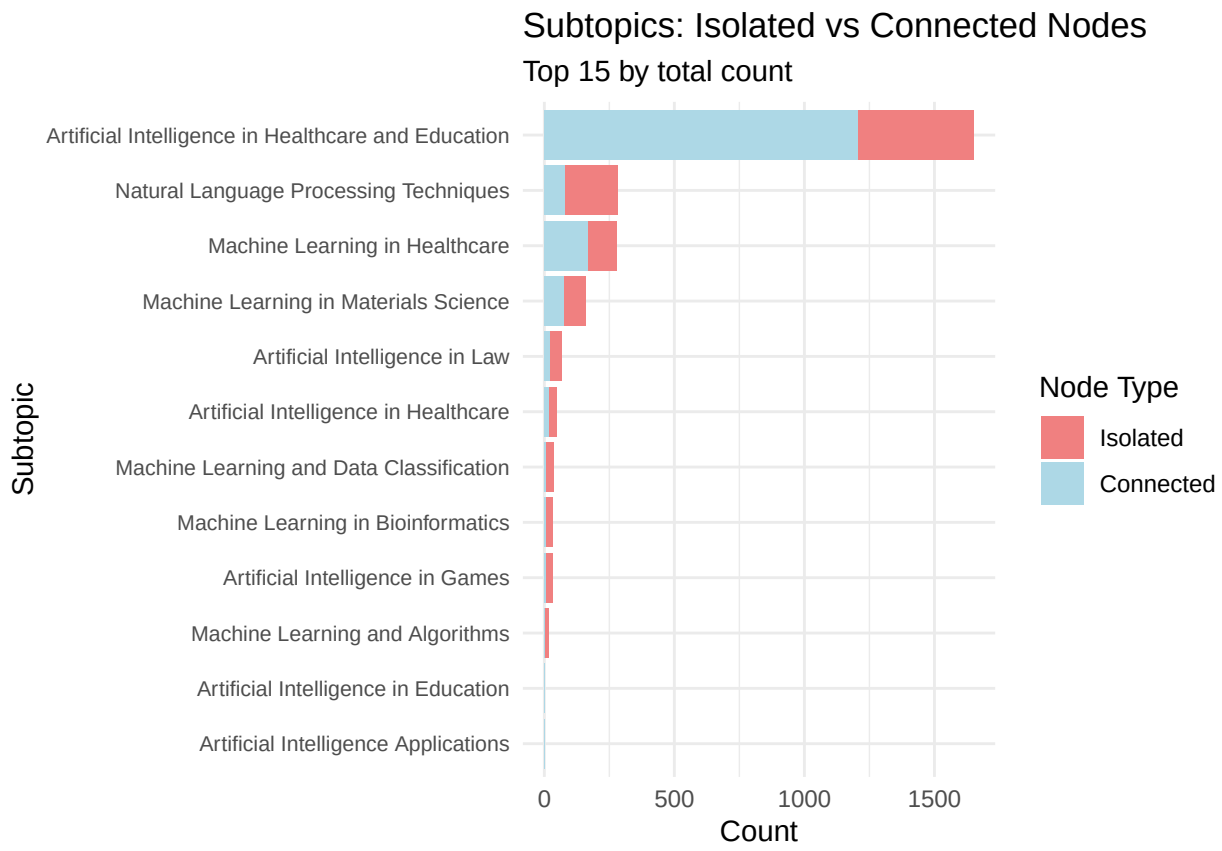
```r
# Subtopic frequency comparison
sub_isolated <- table(nodes_isolated$subtopic)
sub_connected <- table(nodes_connected$subtopic)

sub_union <- unique(c(names(sub_isolated), names(sub_connected)))
comp_df <- data.frame(subtopic = sub_union,
                      isolated = as.numeric(sub_isolated[sub_union]),
                      connected = as.numeric(sub_connected[sub_union]))
comp_df$isolated[is.na(comp_df$isolated)] <- 0
comp_df$connected[is.na(comp_df$connected)] <- 0
comp_df$total <- comp_df$isolated + comp_df$connected
comp_df$isolated_prop <- ifelse(comp_df$total>0, comp_df$isolated/comp_df$total, NA)
comp_df_ord <- comp_df[order(comp_df$isolated_prop, decreasing = TRUE), ]

cat("Subtopics with highest isolation proportion (total>=5) Top 10:\n")
```

```
## Subtopics with highest isolation proportion (total>=5) Top 10:
```

```r
print(head(comp_df_ord[comp_df_ord$total>=5, c("subtopic","isolated","connected","isolated_prop")],10))
```

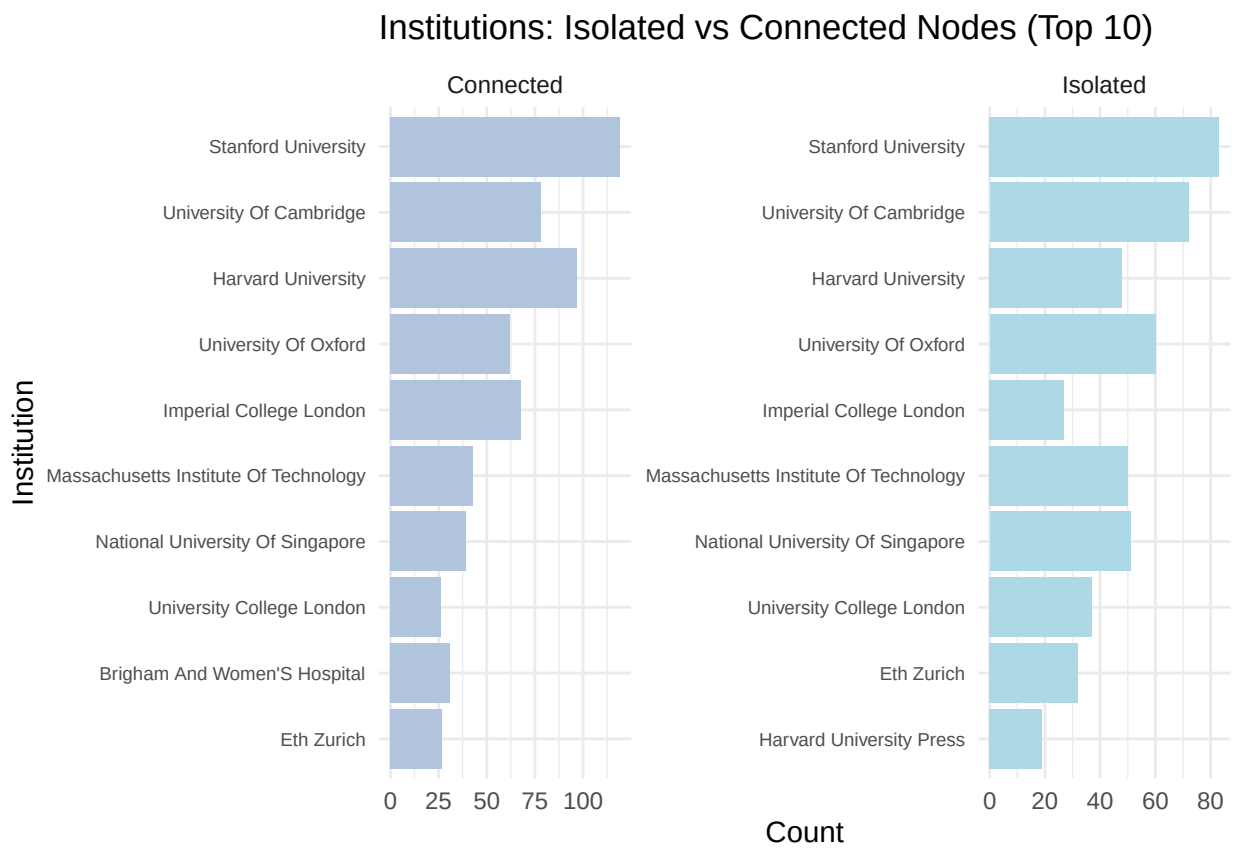
```
##                                                subtopic isolated connected
## 7                         Machine Learning and Algorithms       15         1
## 8               Machine Learning and Data Classification       34         4
## 3                        Artificial Intelligence in Games       28         4
## 9                     Machine Learning in Bioinformatics       26         7
## 12                Natural Language Processing Techniques      206        78
## 6                         Artificial Intelligence in Law       44        23
## 4                  Artificial Intelligence in Healthcare       30        18
## 11                 Machine Learning in Materials Science       87        74
## 10                        Machine Learning in Healthcare      111       167
## 5  Artificial Intelligence in Healthcare and Education      445      1206
##    isolated_prop
## 7      0.9375000
## 8      0.8947368
## 3      0.8750000
## 9      0.7878788
## 12     0.7253521
## 6      0.6567164
## 4      0.6250000
## 11     0.5403727
## 10     0.3992806
## 5      0.2695336
```

```r
# Visualize top 15 subtopics by total count
comp_top15 <- head(comp_df[order(comp_df$total, decreasing = TRUE), ], 15)
plot_df <- melt(comp_top15[, c("subtopic","isolated","connected")], id.vars = "subtopic", variable.name

ggplot(plot_df, aes(x = reorder(subtopic, count), y = count, fill = type)) +
  geom_bar(stat = "identity") + coord_flip() +
  scale_fill_manual(values = c("isolated" = "lightcoral", "connected" = "lightblue"), labels = c("Isola
  labs(title = "Subtopics: Isolated vs Connected Nodes", subtitle = "Top 15 by total count", x = "Subto
  theme_minimal() + theme(axis.text.y = element_text(size = 8))
```




```r
# Institution frequency comparison
inst_iso <- table(nodes_isolated$institution)
inst_conn <- table(nodes_connected$institution)
inst_union <- unique(c(names(inst_iso), names(inst_conn)))
inst_df <- data.frame(institution = inst_union,
                      isolated = as.numeric(inst_iso[inst_union]),
                      connected = as.numeric(inst_conn[inst_union]))
inst_df$isolated[is.na(inst_df$isolated)] <- 0
inst_df$connected[is.na(inst_df$connected)] <- 0
inst_df$total <- inst_df$isolated + inst_df$connected

iso_top10 <- head(inst_df[order(inst_df$isolated, decreasing=TRUE), ],10)
conn_top10 <- head(inst_df[order(inst_df$connected, decreasing=TRUE), ],10)

iso_plot <- data.frame(
  institution = iso_top10$institution,
  frequency = iso_top10$isolated,
  type = "Isolated"
)
conn_plot <- data.frame(
  institution = conn_top10$institution,
  frequency = conn_top10$connected,
  type = "Connected"
)
inst_plot_df <- rbind(iso_plot, conn_plot)
```

```
ggplot(inst_plot_df, aes(x = reorder(institution, frequency), y = frequency, fill = type)) +
  geom_bar(stat = "identity") + coord_flip() + facet_wrap(~type, scales = "free") +
  scale_fill_manual(values = c("Isolated" = "lightblue", "Connected" = "lightsteelblue")) +
  labs(title = "Institutions: Isolated vs Connected Nodes (Top 10)", x = "Institution", y = "Count") +
  theme(axis.text.y = element_text(size = 7), legend.position = "none")
```



10. Overall Summary

```
cat("=== SUMMARY ===\n")
```

```
## === SUMMARY ===
```

```
cat("Raw graph nodes:", vcount(graph_raw), " | Isolated nodes:", length(raw_isolated), " (", round(leng
```

```
## Raw graph nodes: 2610  | Isolated nodes: 1043  ( 39.96 %)
```

```
cat("Connected directed graph nodes:", vcount(graph_connected), " | Edges:", ecount(graph_connected), "
```

```
## Connected directed graph nodes: 1582  | Edges: 3730  | Density: 0.001491
```

```r
cat("Largest component nodes:", vcount(largest_component), " | Louvain communities:", length(louvain_co
```

```
## Largest component nodes: 1433  | Louvain communities: 21  | Modularity: 0.576
```

```r
cat("Highest PageRank node:", V(largest_component)$name[which.max(c_pagerank)], " (", round(max(c_pagera
```

```
## Highest PageRank node: P0199  ( 0.0405 )
```

```r
cat("Highest betweenness node:", V(largest_component)$name[which.max(c_between)], " (", round(max(c_bet
```

```
## Highest betweenness node: P0011  ( 0.0026 )
```