PretendPlay_Gesture_Analysis_ALLyears

Kristen Johnson

2025-04-25

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
## Loading required package: carData
##
##
## Attaching package: 'car'
##
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
##
## The following object is masked from 'package:purrr':
##
##     some
##
##
##
## Attaching package: 'rstatix'
##
##
## The following objects are masked from 'package:effectsize':
##
##     cohens_d, eta_squared
##
##
## The following object is masked from 'package:stats':
##
##     filter
##
##
## here() starts at /Users/kristenjohnson/KristenWorkingDirectory/Play_Narrative
##
## Registering fonts with R
```

Read in each dataset and add Year column

```
child_summary_H10 <- read_csv(here("PN_Datasets", "Pretend_Play", "CSVs_of_Combined_Data_PP", "child_su
```

```
## Rows: 28 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): groupstatus
## dbl (8): participant_id, total_gestures, rep_gestures, total_pretend_episode...
## lgl (1): mlu_missing_flag
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
child_summary_H10$Year <- 10

child_summary_H8 <- read_csv(here("PN_Datasets", "Pretend_Play", "CSVs_of_Combined_Data_PP", "child_sum
```

```
## Rows: 23 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): groupstatus
## dbl (8): participant_id, total_gestures, rep_gestures, total_pretend_episode...
## lgl (1): mlu_missing_flag
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
child_summary_H8$Year <- 8

child_summary_H7 <- read_csv(here("PN_Datasets", "Pretend_Play", "CSVs_of_Combined_Data_PP", "child_sum
```

```
## Rows: 29 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): groupstatus
## dbl (8): participant_id, total_gestures, rep_gestures, total_pretend_episode...
## lgl (1): mlu_missing_flag
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
child_summary_H7$Year <- 7
```

Combine data across ALL THREE YEARS

```
# Combine all datasets into one data frame
pp_gesture_data_ALLyears <- bind_rows(child_summary_H10, child_summary_H8, child_summary_H7)

# View the combined dataset
head(pp_gesture_data_ALLyears)
```

```
## # A tibble: 6 x 11
##   participant_id groupstatus total_gestures rep_gestures total_pretend_episodes
##            <dbl> <chr>                <dbl>        <dbl>                  <dbl>
```

```
## 1             22 TD                       12           1                    112
## 2             25 TD                        2           2                     21
## 3             28 TD                        7           1                     17
## 4             29 TD                      100          26                    326
## 5             37 TD                        0           0                      4
## 6             38 TD                       10           2                     28
## # i 6 more variables: episodes_with_gesture <dbl>,
## #   prop_episodes_with_gesture <dbl>, prop_rep <dbl>, mlu <dbl>,
## #   mlu_missing_flag <lgl>, Year <dbl>
```

1. Conduct ANOVAs for gestures & summarize results

```
# For total gestures
total_anova <- aov(total_gestures ~ groupstatus, pp_gesture_data_ALLyears)
cat("\nTotal gestures ANOVA:\n")
```

```
##
## Total gestures ANOVA:
```

```
print(summary(total_anova))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## groupstatus  1     42   42.41   0.172  0.679
## Residuals   78  19237  246.63
```

```
# For representational gestures
rep_anova <- aov(rep_gestures ~ groupstatus, data = pp_gesture_data_ALLyears)
cat("\nRepresentational gestures ANOVA:\n")
```

```
##
## Representational gestures ANOVA:
```

```
print(summary(rep_anova))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## groupstatus  1    9.4   9.366   0.737  0.393
## Residuals   78  991.6  12.713
```

```
# For proportion of representational gestures to total gestures
prop_rep_anova <- aov(prop_rep ~ groupstatus, data = pp_gesture_data_ALLyears)
cat("\nProportion of representational gestures ANOVA:\n")
```

```
##
## Proportion of representational gestures ANOVA:
```

```
print(summary(prop_rep_anova))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## groupstatus  1  0.326  0.3259   5.091 0.0275 *
## Residuals   64  4.096  0.0640
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 14 observations deleted due to missingness
```

2. Calculate effect sizes

```
if(require(effectsize)) {
  cat("\nEffect sizes:\n")
  cat("Total gestures: ")
  print(eta_squared(total_anova))
  cat("Proportion representational: ")
  print(eta_squared(prop_rep_anova))
} else {
  # Manual calculation if package not available
  cat("\nEffect sizes calculated manually:\n")
  # Formula for eta-squared: SS_between / SS_total
  summary_total <- summary(total_anova)
  eta_sq_total <- summary_total[[1]]["GroupStatus", "Sum Sq"] /
                  sum(summary_total[[1]][, "Sum Sq"])
  cat("Total gestures eta-squared: ", eta_sq_total, "\n")
  # Repeat for other ANOVAs
}
```

```
##
## Effect sizes:
## Total gestures: groupstatus
## 0.002200016
## Proportion representational: groupstatus
##   0.0736867
```

3. Descriptive statistics by group

```
group_stats <- pp_gesture_data_ALLyears %>%
  group_by(groupstatus) %>%
  summarize(
    n = n(),
    total_mean = mean(total_gestures, na.rm = TRUE),
    total_sd = sd(total_gestures, na.rm = TRUE),
    rep_mean = mean(rep_gestures, na.rm = TRUE),
    rep_sd = sd(rep_gestures, na.rm = TRUE),
    prop_rep_mean = mean(prop_rep, na.rm = TRUE),
    prop_rep_sd = sd(prop_rep, na.rm = TRUE)
  )

print("Descriptive statistics by group:")
```

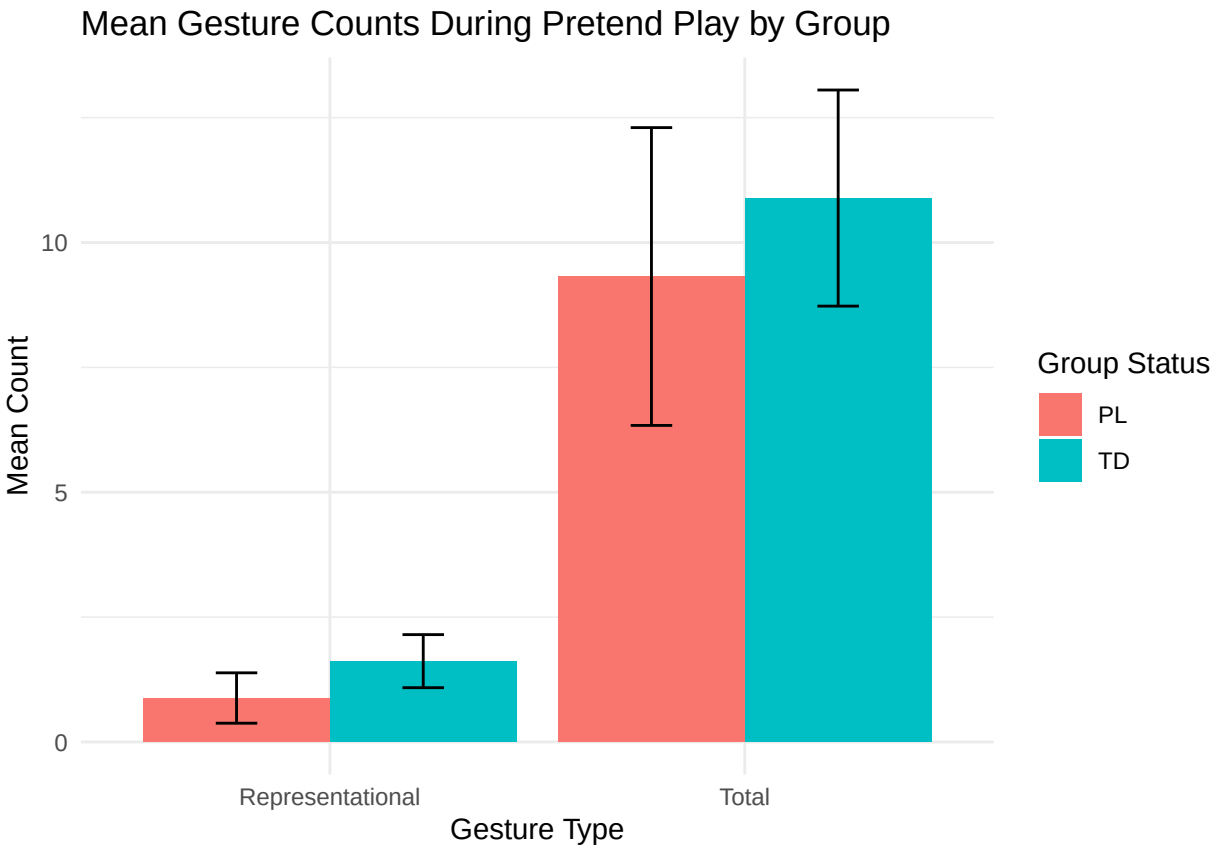
```
## [1] "Descriptive statistics by group:"
```

```
print(group_stats)
```

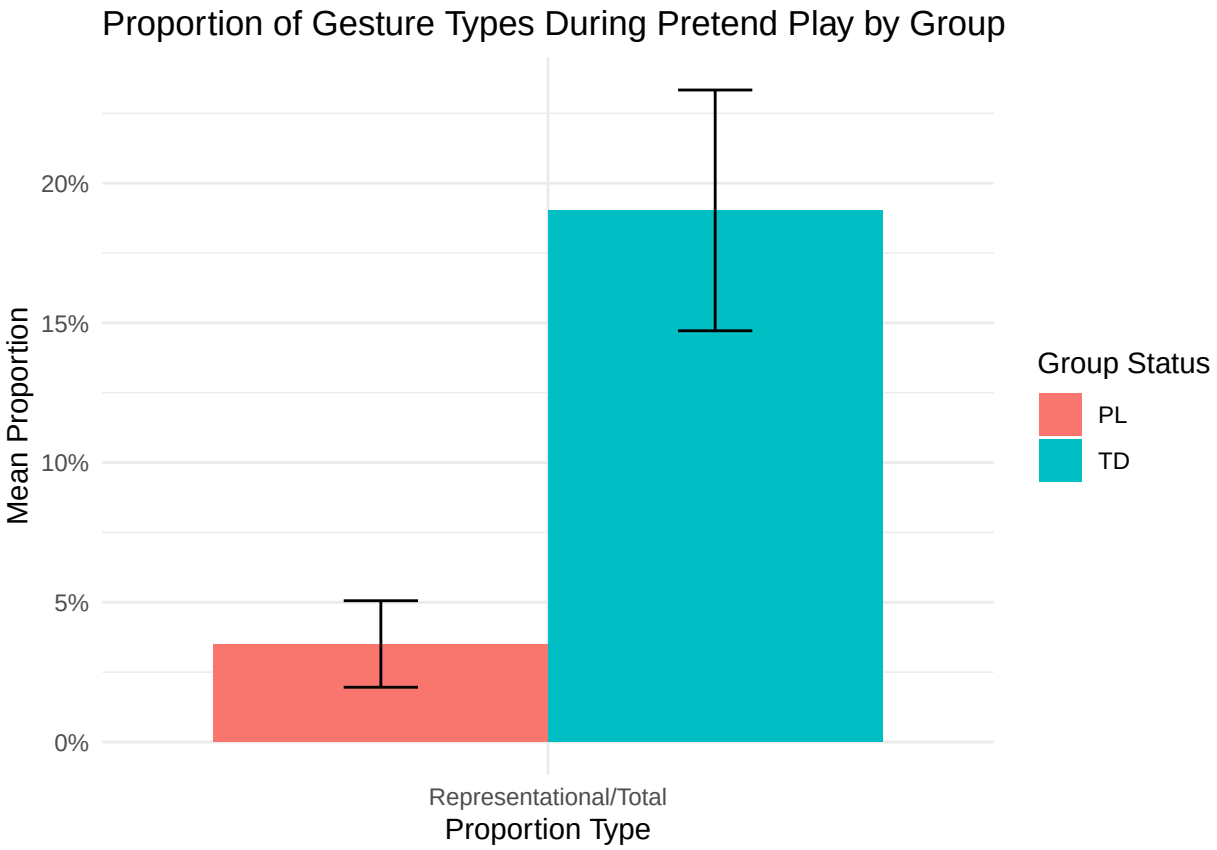
```
## # A tibble: 2 x 8
##   groupstatus     n total_mean total_sd rep_mean rep_sd prop_rep_mean
##   <chr>       <int>      <dbl>    <dbl>    <dbl>  <dbl>         <dbl>
## 1 PL             25       9.32     14.9     0.88   2.52        0.0351
## 2 TD             55      10.9      16.0     1.62   3.94        0.190
## # i 1 more variable: prop_rep_sd <dbl>
```

4. Visualization of results

```
if(require(tidyverse)) {
  # Raw counts visualization
  counts_long <- pp_gesture_data_ALLyears %>%
    select(groupstatus, total_gestures, rep_gestures) %>%
    pivot_longer(cols = c(total_gestures, rep_gestures),
                 names_to = "gesture_type",
                 values_to = "count")

  p1 <- ggplot(counts_long, aes(x = gesture_type, y = count, fill = groupstatus)) +
    stat_summary(fun = mean, geom = "bar", position = position_dodge(0.9)) +
    stat_summary(fun.data = function(x) {
      data.frame(y = mean(x, na.rm = TRUE),
                 ymin = mean(x, na.rm = TRUE) - sd(x, na.rm = TRUE)/sqrt(sum(!is.na(x))),
                 ymax = mean(x, na.rm = TRUE) + sd(x, na.rm = TRUE)/sqrt(sum(!is.na(x))))
    }, geom = "errorbar", width = 0.2, position = position_dodge(0.9)) +
    labs(title = "Mean Gesture Counts During Pretend Play by Group",
         x = "Gesture Type",
         y = "Mean Count",
         fill = "Group Status") +
    theme_minimal() +
    scale_x_discrete(labels = c("total_gestures" = "Total",
                                "rep_gestures" = "Representational"))
  print(p1)

  # Proportions visualization
  props_long <- pp_gesture_data_ALLyears %>%
    select(groupstatus, prop_rep) %>%
    pivot_longer(cols = c(prop_rep),
                 names_to = "proportion_type",
                 values_to = "proportion")

  p2 <- ggplot(props_long, aes(x = proportion_type, y = proportion, fill = groupstatus)) +
    stat_summary(fun = mean, geom = "bar", position = position_dodge(0.9)) +
    stat_summary(fun.data = function(x) {
      data.frame(y = mean(x, na.rm = TRUE),
                 ymin = mean(x, na.rm = TRUE) - sd(x, na.rm = TRUE)/sqrt(sum(!is.na(x))),
                 ymax = mean(x, na.rm = TRUE) + sd(x, na.rm = TRUE)/sqrt(sum(!is.na(x))))
    }, geom = "errorbar", width = 0.2, position = position_dodge(0.9)) +
    labs(title = "Proportion of Gesture Types During Pretend Play by Group",
         x = "Proportion Type",
         y = "Mean Proportion",
         fill = "Group Status") +
    theme_minimal() +
    scale_x_discrete(labels = c("prop_rep" = "Representational/Total")) +
    scale_y_continuous(labels = scales::percent)
  print(p2)
}
```




```
## Warning: Removed 14 rows containing non-finite outside the scale range
## (`stat_summary()`).
## Removed 14 rows containing non-finite outside the scale range
## (`stat_summary()`).
```



5. Statistical tests to directly compare TD and BI groups

```
# t-tests for each variable (alternative to ANOVA with only two groups)
t_total <- t.test(total_gestures ~ groupstatus, data = pp_gesture_data_ALLyears)
t_rep <- t.test(rep_gestures ~ groupstatus, data = pp_gesture_data_ALLyears)
t_prop_rep <- t.test(prop_rep ~ groupstatus, data = pp_gesture_data_ALLyears)

cat("\nt-test results (direct comparison between groups):\n")
```

```
##
## t-test results (direct comparison between groups):
```

```
cat("\nTotal gestures:\n")
```

```
##
## Total gestures:
```

```
print(t_total)
```

```
##
##  Welch Two Sample t-test
##
## data:  total_gestures by groupstatus
## t = -0.42644, df = 49.796, p-value = 0.6716
## alternative hypothesis: true difference in means between group PL and group TD is not equal to 0
## 95 percent confidence interval:
##  -8.970774  5.828956
```

```
## sample estimates:
## mean in group PL mean in group TD
##          9.32000         10.89091
```

```
cat("\nRepresentational gestures:\n")
```

```
##
## Representational gestures:
```

```
print(t_rep)
```

```
##
##  Welch Two Sample t-test
##
## data:  rep_gestures by groupstatus
## t = -1.0074, df = 69.051, p-value = 0.3172
## alternative hypothesis: true difference in means between group PL and group TD is not equal to 0
## 95 percent confidence interval:
##  -2.1999082  0.7235446
## sample estimates:
## mean in group PL mean in group TD
##         0.880000         1.618182
```

```
cat("\nProportion representational:\n")
```

```
##
## Proportion representational:
```

```
print(t_prop_rep)
```

```
##
##  Welch Two Sample t-test
##
## data:  prop_rep by groupstatus
## t = -3.3894, df = 56.246, p-value = 0.001286
## alternative hypothesis: true difference in means between group PL and group TD is not equal to 0
## 95 percent confidence interval:
##  -0.24690658 -0.06347617
## sample estimates:
## mean in group PL mean in group TD
##        0.0350700        0.1902614
```

6. Calculate Cohen's d effect sizes for t-tests with error handling

```
cat("\nCohen's d effect sizes:\n")
```

```
##
## Cohen's d effect sizes:
```

```
# Function to safely calculate Cohen's d
safe_cohens_d <- function(formula, data) {
  tryCatch({
    res <- cohens_d(formula, data = data)
    return(res)
  }, error = function(e) {
    # Extract variable name from formula
    var_name <- as.character(formula)[2]
```

```
    # Get means by group
    means <- aggregate(formula, data = data, FUN = mean, na.rm = TRUE)
    sds <- aggregate(formula, data = data, FUN = sd, na.rm = TRUE)

    cat("Error calculating Cohen's d for", var_name, ":\n")
    cat("Group means:", toString(means), "\n")
    cat("Group SDs:", toString(sds), "\n")
    cat("Error message:", e$message, "\n")
    return(NULL)
  })
}

# Apply the safe function to each variable
cat("\nTotal gestures:\n")
```

```
##
## Total gestures:
```

```
print(safe_cohens_d(total_gestures ~ groupstatus, data = pp_gesture_data_ALLyears))
```

```
## # A tibble: 1 x 7
##   .y.            group1 group2 effsize    n1    n2 magnitude
## * <chr>          <chr>  <chr>    <dbl> <int> <int> <ord>
## 1 total_gestures PL     TD      -0.101    25    55 negligible
```

```
cat("\nRepresentational gestures:\n")
```

```
##
## Representational gestures:
```

```
print(safe_cohens_d(rep_gestures ~ groupstatus, data = pp_gesture_data_ALLyears))
```

```
## # A tibble: 1 x 7
##   .y.          group1 group2 effsize    n1    n2 magnitude
## * <chr>        <chr>  <chr>    <dbl> <int> <int> <ord>
## 1 rep_gestures PL     TD      -0.223    25    55 small
```

```
cat("\nProportion representational:\n")
```

```
##
## Proportion representational:
```

```
print(safe_cohens_d(prop_rep ~ groupstatus, data = pp_gesture_data_ALLyears))
```

```
## # A tibble: 1 x 7
##   .y.      group1 group2 effsize    n1    n2 magnitude
## * <chr>    <chr>  <chr>    <dbl> <int> <int> <ord>
## 1 prop_rep PL     TD      -0.724    19    47 moderate
```

Make APA-formatted data visualizations

```
if(require(tidyverse)) {
  # Set APA theme
  apa_theme <- theme_bw() +
    theme(
      panel.grid.major = element_blank(),
```

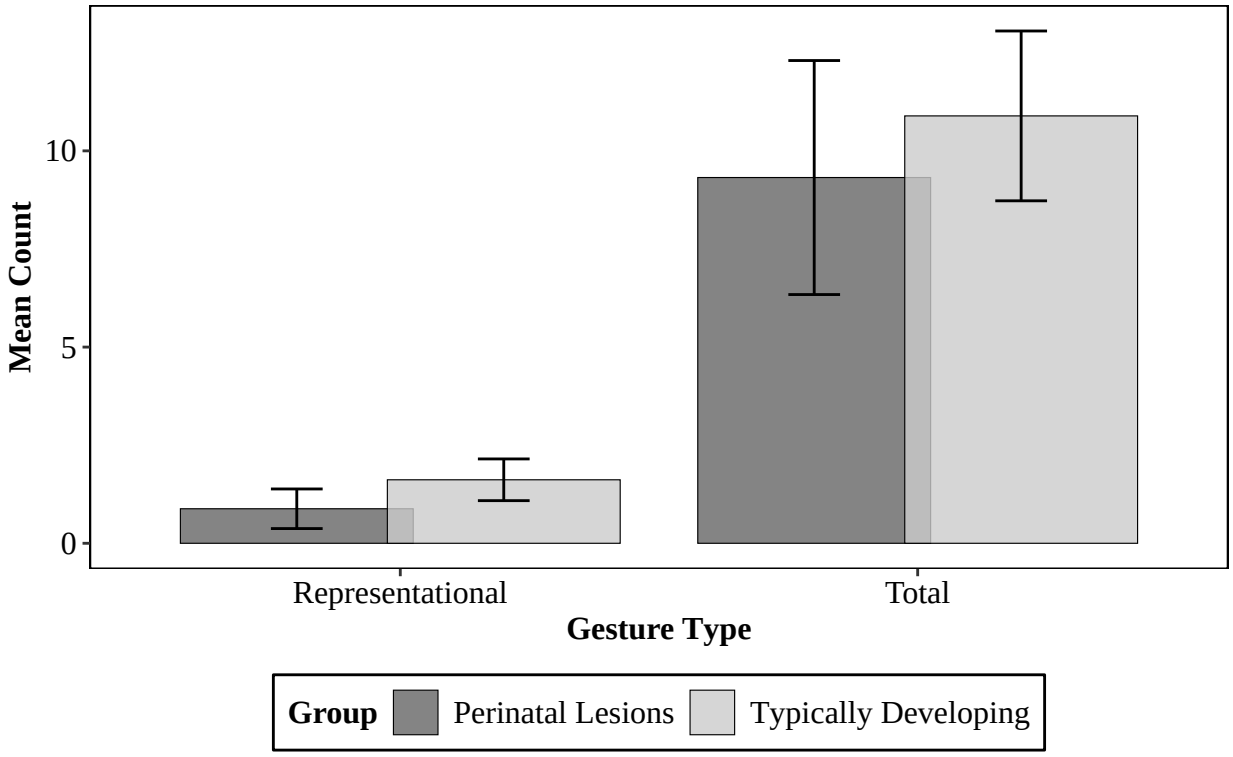
```
    panel.grid.minor = element_blank(),
    panel.border = element_rect(colour = "black", fill = NA),
    axis.text = element_text(colour = "black", size = 12),
    axis.title = element_text(size = 12, face = "bold"),
    legend.title = element_text(size = 12, face = "bold"),
    legend.text = element_text(size = 12),
    legend.position = "bottom",
    legend.background = element_rect(fill = "white", colour = "black"),
    plot.title = element_text(size = 14, hjust = 0.5, face = "bold"),
    text = element_text(family = "Times New Roman")
  )

# Raw counts visualization
counts_long <- pp_gesture_data_ALLyears %>%
  select(groupstatus, total_gestures, rep_gestures) %>%
  pivot_longer(cols = c(total_gestures, rep_gestures),
               names_to = "gesture_type",
               values_to = "count")

# Renaming GroupStatus labels to be more readable
counts_long$groupstatus <- factor(counts_long$groupstatus,
                                  levels = c("PL", "TD"),
                                  labels = c("Perinatal Lesions", "Typically Developing"))

p1 <- ggplot(counts_long, aes(x = gesture_type, y = count, fill = groupstatus)) +
  stat_summary(fun = mean, geom = "bar", position = position_dodge(0.8),
               alpha = 0.8, color = "black", size = 0.2) +
  stat_summary(fun.data = function(x) {
    data.frame(y = mean(x, na.rm = TRUE),
               ymin = mean(x, na.rm = TRUE) - sd(x, na.rm = TRUE)/sqrt(sum(!is.na(x))),
               ymax = mean(x, na.rm = TRUE) + sd(x, na.rm = TRUE)/sqrt(sum(!is.na(x))))
  }, geom = "errorbar", width = 0.2, position = position_dodge(0.8)) +
  labs(title = "Figure 1",
       subtitle = "Mean Gesture Counts During Pretend Play by Group",
       x = "Gesture Type",
       y = "Mean Count",
       fill = "Group") +
  scale_x_discrete(labels = c("total_gestures" = "Total",
                              "rep_gestures" = "Representational")) +
  scale_fill_grey(start = 0.4, end = 0.8) +
  apa_theme +
  theme(plot.subtitle = element_text(size = 12, hjust = 0.5))

print(p1)

# Save high-resolution figure
ggsave("Figure_1_Gesture_Counts.tiff", p1, width = 7, height = 5, dpi = 300)

# Proportions visualization
props_long <- pp_gesture_data_ALLyears %>%
  select(groupstatus, prop_rep) %>%
  pivot_longer(cols = c(prop_rep),
               names_to = "proportion_type",
```

```
                values_to = "proportion")

  # Renaming GroupStatus labels
  props_long$groupstatus <- factor(props_long$groupstatus,
                                   levels = c("PL", "TD"),
                                   labels = c("Perinatal Lesions", "Typically Developing"))

  p2 <- ggplot(props_long, aes(x = proportion_type, y = proportion, fill = groupstatus)) +
    stat_summary(fun = mean, geom = "bar", position = position_dodge(0.8),
                 alpha = 0.8, color = "black", size = 0.2) +
    stat_summary(fun.data = function(x) {
      data.frame(y = mean(x, na.rm = TRUE),
                 ymin = mean(x, na.rm = TRUE) - sd(x, na.rm = TRUE)/sqrt(sum(!is.na(x))),
                 ymax = mean(x, na.rm = TRUE) + sd(x, na.rm = TRUE)/sqrt(sum(!is.na(x))))
    }, geom = "errorbar", width = 0.2, position = position_dodge(0.8)) +
    labs(title = "Figure 2",
         subtitle = "Proportion of Gesture Types During Pretend Play by Group",
         x = "Proportion Type",
         y = "Mean Proportion",
         fill = "Group") +
    scale_x_discrete(labels = c("prop_rep" = "Representational/Total")) +
    scale_y_continuous(labels = scales::percent,
                       breaks = seq(0, 0.25, 0.05)) +
    scale_fill_grey(start = 0.4, end = 0.8) +
    apa_theme +
    theme(plot.subtitle = element_text(size = 12, hjust = 0.5))

  print(p2)

  # Save high-resolution figure
  ggsave("Figure_2_Gesture_Proportions.tiff", p2, width = 7, height = 5, dpi = 300)
}
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

Figure 1

Mean Gesture Counts During Pretend Play by Group




```
## Warning: Removed 14 rows containing non-finite outside the scale range
## (`stat_summary()`).
```

```
## Warning: Removed 14 rows containing non-finite outside the scale range
## (`stat_summary()`).
## Removed 14 rows containing non-finite outside the scale range
## (`stat_summary()`).
## Removed 14 rows containing non-finite outside the scale range
## (`stat_summary()`).
```

Figure 2

Proportion of Gesture Types During Pretend Play by Group