R Notebook for Categorization Analysis

Ari Dyckovsky

Categorization Analysis

Load packages

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyr)
library(ggplot2)
library(hash)
```

```
## hash-2.2.6.1 provided by Decision Patterns
```

Constants

```r
VALIDATION_THRESHOLD = 4

# Colors for plotting
COLORS <- hash()
COLORS[["BLACK"]] <- "#2F2F2F"
COLORS[["GRAY"]] <- "#5C6D70"
COLORS[["BLUE"]] <- "#0E79B2"
COLORS[["ORANGE"]] <- "#F39237"

# For formatting doubles
options(digits = 2)
```

Set datapath and load `shlab.imgct`

Currently pulling from local, but will make dynamic for other machines later.

```r
# Set the working directory to be part of S Drive (may make dynamic later?)
# Whilst not dynamic, change for own session if mount point is not equivalent on
# local machine
shared_dir <- "~/Projects/shlab/mounts/imgct"
```

```
package_dir <- "~/Projects/shlab"

datapath <- file.path(shared_dir, "csn_images")
imgct_package_path <- file.path(package_dir, "shlab.imgct")

# Make sure that devtools, tidyverse are installed before this call
devtools::load_all(imgct_package_path)
```

```
## Loading shlab.imgct
```

Load category data

```
counted_responses_cols = readr::cols(
  image_id = readr::col_character(),
  .default = readr::col_integer()
)

counted_df <- shlab.imgct::load_result(datapath,
                                       stringr::str_c("categorized_", VALIDATION_THRESHOLD, "_valid"),
                                       columns = counted_responses_cols)

# Determine category names from table
category_names <- counted_df %>%
  dplyr::select(-c(image_id, n_ratings)) %>%
  names()

# Include max_rating for further analyses
counted_df <- counted_df %>%
  dplyr::mutate(
    max_rating = pmax(!!!rlang::syms(category_names))
  )

head(counted_df)
```

```
## # A tibble: 6 x 8
##   image_id      Person `Animal/Plant` Object Place Other n_ratings max_rating
##   <chr>          <int>          <int>  <int> <int> <int>     <int>      <int>
## 1 IAPS_1033.jpg      0              6      0     0     0         6          6
## 2 IAPS_1310.jpg      0              7      0     0     0         7          7
## 3 IAPS_1390.jpg      0              7      0     0     0         7          7
## 4 IAPS_1617.jpg      0              7      0     0     0         7          7
## 5 IAPS_1660.jpg      0              7      0     0     0         7          7
## 6 IAPS_1750.jpg      0              7      0     0     0         7          7
```

Threshold-based Categorization

Functions for evaluating the harsh and semi-harsh threshold types of categorization for a given image based on number of responses.

```
# Function to choose category with harsh threshold for maximum
#   - Does not allow for ties, only unique maximum as named category or Other.
choose_category_max_threshold <- function(ratings) {

  # count number of ratings equal to maximum rating
  n_max <- length(which(ratings == max(ratings)))
  if (n_max > 1) {
    return("Other")
  } else {
    return(names(ratings)[which.max(ratings)])
  }

}

# Function to choose category with semi-harsh threshold for maximum
#   - Does allow for ties, so unique maximum as named category, tied maxima as list category, or Other.
choose_category_ties_threshold <- function(ratings) {
  return(stringr::str_c(
    names(ratings)[which(ratings == max(ratings))],
    collapse=" & "
  ))
}
```

Determine the category dataframe including both the harsh (category_max) and semi-harsh (category_ties) columns, which are character and list of character types respectively.

```
category_df <- counted_df %>%
  dplyr::mutate(

    category_max = dplyr::select(., category_names) %>%
      purrr::pmap_chr(
        .,
        ~ choose_category_max_threshold(c(...))
      ),

    category_ties = dplyr::select(., category_names) %>%
      purrr::pmap_chr(
        .,
        ~ choose_category_ties_threshold(c(...))
      )

  ) %>%
  dplyr::select(image_id, category_max, category_ties)
```

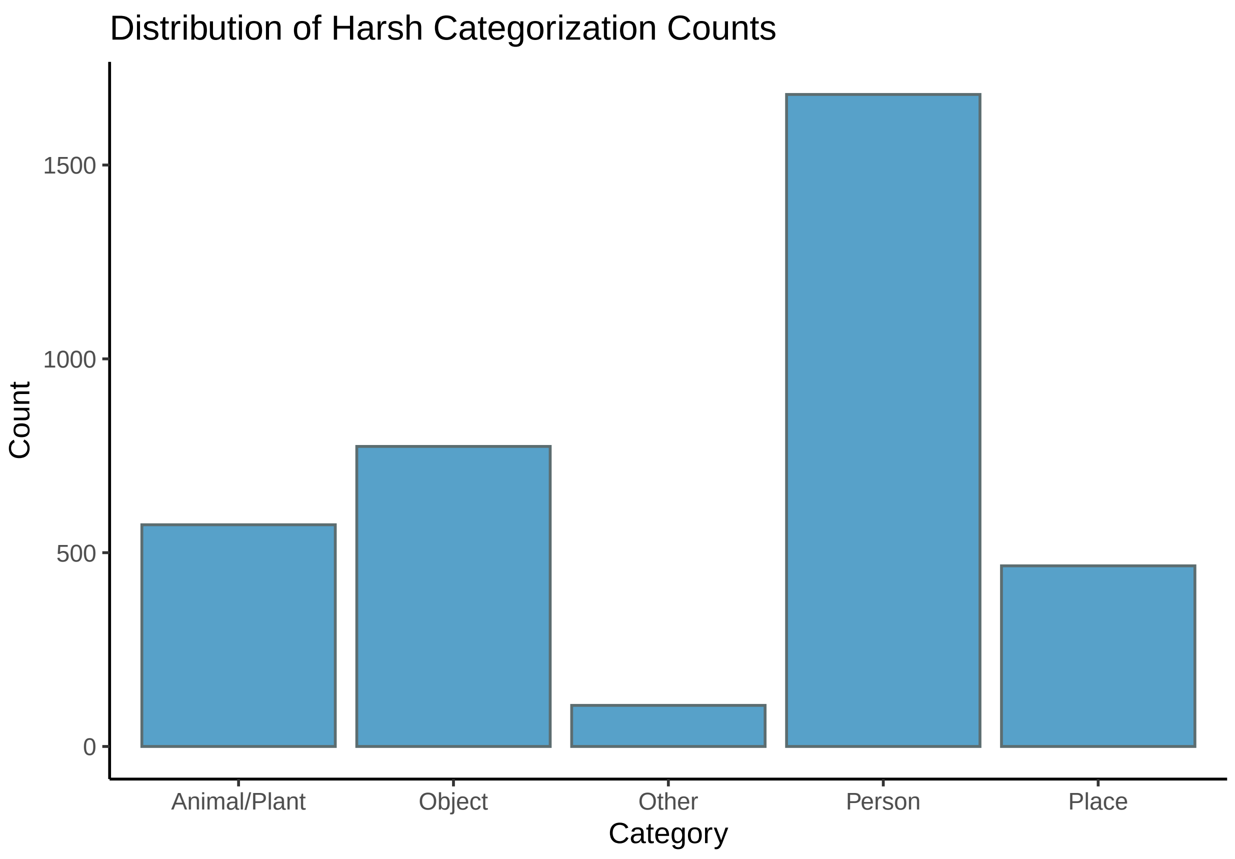
```
## Note: Using an external vector in selections is ambiguous.
## i Use `all_of(category_names)` instead of `category_names` to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
category_df
```

```
## # A tibble: 3,600 x 3
##    image_id      category_max category_ties
##    <chr>         <chr>        <chr>
##  1 IAPS_1033.jpg Animal/Plant Animal/Plant
##  2 IAPS_1310.jpg Animal/Plant Animal/Plant
##  3 IAPS_1390.jpg Animal/Plant Animal/Plant
##  4 IAPS_1617.jpg Animal/Plant Animal/Plant
##  5 IAPS_1660.jpg Animal/Plant Animal/Plant
##  6 IAPS_1750.jpg Animal/Plant Animal/Plant
##  7 IAPS_1900.jpg Animal/Plant Animal/Plant
##  8 IAPS_2045.jpg Person       Person
##  9 IAPS_2050.jpg Person       Person
## 10 IAPS_2095.jpg Person       Person
## # ... with 3,590 more rows
```

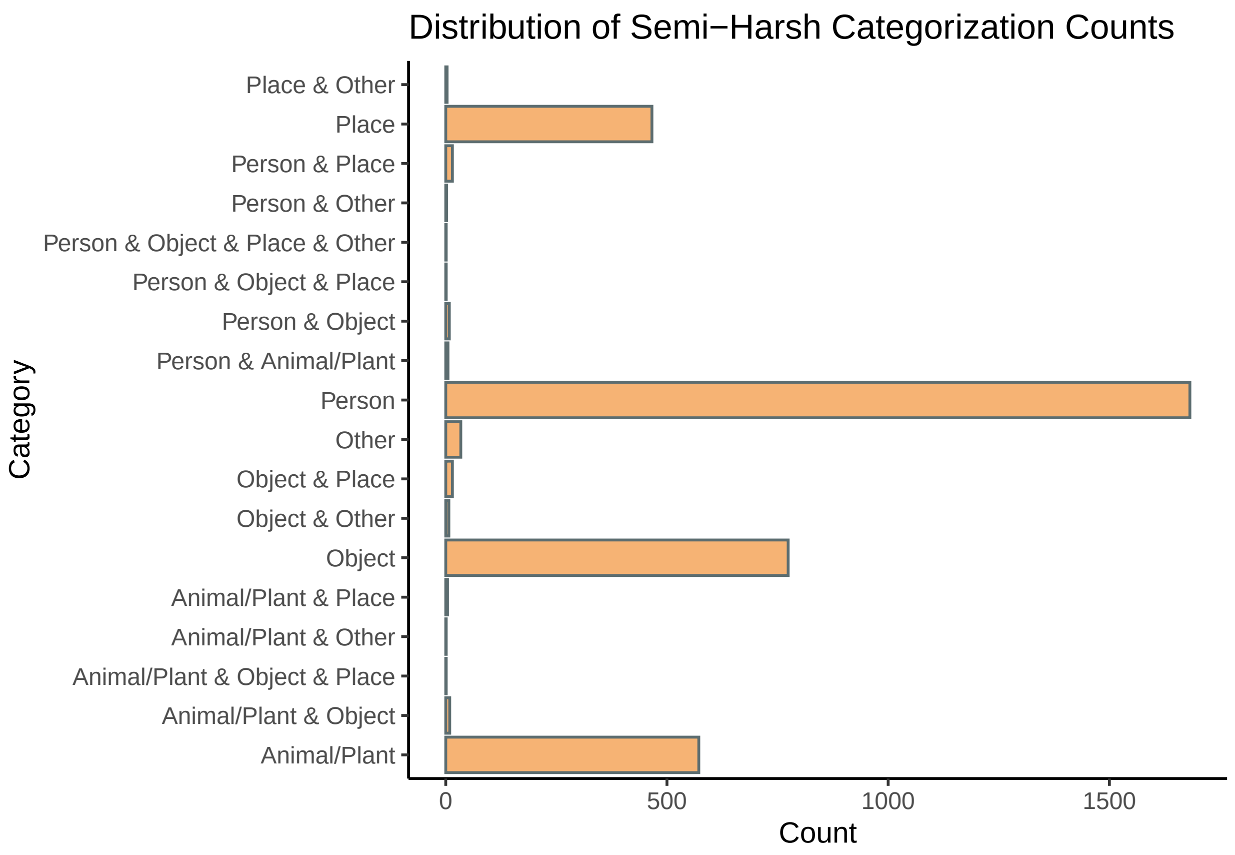
Plot Harsh Categories

```
p <- ggplot(category_df, aes(x=category_max)) +
  geom_bar(color=COLORS[["GRAY"]], fill=COLORS[["BLUE"]], alpha=0.7)

p + labs(title="Distribution of Harsh Categorization Counts", x="Category", y="Count") +
  theme_classic()
```

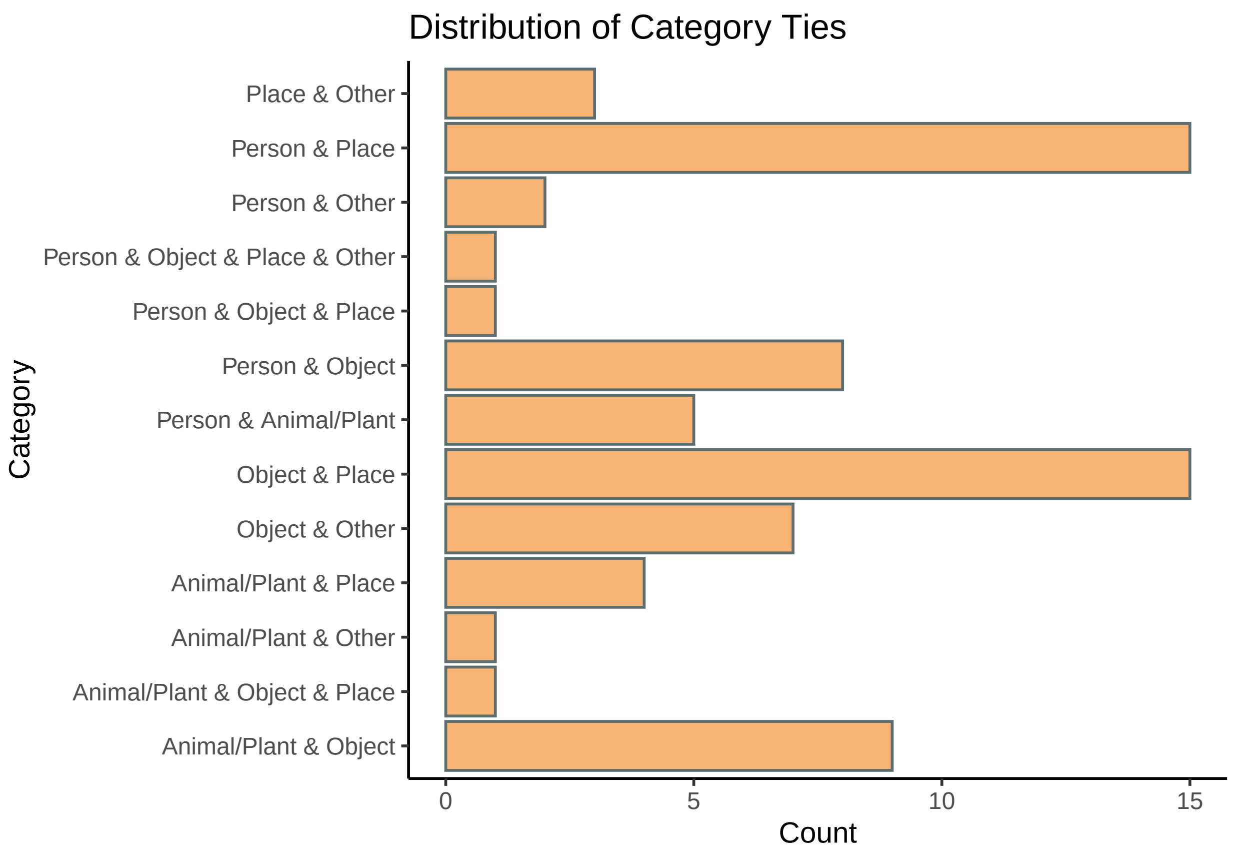


Plot Semi-Harsh Categories

```
p <- ggplot(category_df, aes(y=category_ties)) +
  geom_bar(color=COLORS[["GRAY"]], fill=COLORS[["ORANGE"]], alpha=0.7)

p + labs(title="Distribution of Semi-Harsh Categorization Counts", x="Count", y="Category") +
  theme_classic()
```



```r
only_ties_df <- category_df %>%
  dplyr::filter(!(category_ties %in% c(category_names)))

p <- ggplot(only_ties_df, aes(y=category_ties)) +
  geom_bar(color=COLORS[["GRAY"]], fill=COLORS[["ORANGE"]], alpha=0.7)

p + labs(title="Distribution of Category Ties", x="Count", y="Category") +
  theme_classic()
```



Normalization-based Categorization

Density normalization

```r
normalized_density_df <- counted_df %>%
  dplyr::mutate_at(vars(category_names), ~ . / n_ratings) %>%
  dplyr::select(-c(n_ratings, max_rating))

normalized_density_df
```

```
## # A tibble: 3,600 x 6
##    image_id      Person `Animal/Plant` Object Place Other
##    <chr>          <dbl>          <dbl>  <dbl> <dbl> <dbl>
##  1 IAPS_1033.jpg      0              1      0     0     0
##  2 IAPS_1310.jpg      0              1      0     0     0
##  3 IAPS_1390.jpg      0              1      0     0     0
##  4 IAPS_1617.jpg      0              1      0     0     0
##  5 IAPS_1660.jpg      0              1      0     0     0
##  6 IAPS_1750.jpg      0              1      0     0     0
##  7 IAPS_1900.jpg      0              1      0     0     0
##  8 IAPS_2045.jpg      1              0      0     0     0
##  9 IAPS_2050.jpg      1              0      0     0     0
## 10 IAPS_2095.jpg      1              0      0     0     0
## # ... with 3,590 more rows
```

Plot Density Normalization

```
density_long <- normalized_density_df %>%
  tidyr::pivot_longer(
    -image_id,
    names_to = "category",
    values_to = "density"
  )

density_long
```

```
## # A tibble: 18,000 x 3
##    image_id      category     density
##    <chr>         <chr>          <dbl>
##  1 IAPS_1033.jpg Person             0
##  2 IAPS_1033.jpg Animal/Plant       1
##  3 IAPS_1033.jpg Object             0
##  4 IAPS_1033.jpg Place              0
##  5 IAPS_1033.jpg Other              0
##  6 IAPS_1310.jpg Person             0
##  7 IAPS_1310.jpg Animal/Plant       1
##  8 IAPS_1310.jpg Object             0
##  9 IAPS_1310.jpg Place              0
## 10 IAPS_1310.jpg Other              0
## # ... with 17,990 more rows
```

Max normalization

```
normalized_max_df <- counted_df %>%
  dplyr::mutate_at(vars(category_names), ~ . / max_rating) %>%
  dplyr::select(-c(n_ratings, max_rating))

normalized_max_df
```

```
## # A tibble: 3,600 x 6
##    image_id      Person `Animal/Plant` Object Place Other
##    <chr>          <dbl>          <dbl>  <dbl> <dbl> <dbl>
##  1 IAPS_1033.jpg      0              1      0     0     0
##  2 IAPS_1310.jpg      0              1      0     0     0
##  3 IAPS_1390.jpg      0              1      0     0     0
##  4 IAPS_1617.jpg      0              1      0     0     0
##  5 IAPS_1660.jpg      0              1      0     0     0
##  6 IAPS_1750.jpg      0              1      0     0     0
##  7 IAPS_1900.jpg      0              1      0     0     0
##  8 IAPS_2045.jpg      1              0      0     0     0
##  9 IAPS_2050.jpg      1              0      0     0     0
## 10 IAPS_2095.jpg      1              0      0     0     0
## # ... with 3,590 more rows
```

Plot Max Normalization

```
max_long <- normalized_max_df %>%
  tidyr::pivot_longer(
    -image_id,
    names_to = "category",
    values_to = "density"
  )

max_long
```

```
## # A tibble: 18,000 x 3
##    image_id      category     density
##    <chr>         <chr>          <dbl>
##  1 IAPS_1033.jpg Person             0
##  2 IAPS_1033.jpg Animal/Plant       1
##  3 IAPS_1033.jpg Object             0
##  4 IAPS_1033.jpg Place              0
##  5 IAPS_1033.jpg Other              0
##  6 IAPS_1310.jpg Person             0
##  7 IAPS_1310.jpg Animal/Plant       1
##  8 IAPS_1310.jpg Object             0
##  9 IAPS_1310.jpg Place              0
## 10 IAPS_1310.jpg Other              0
## # ... with 17,990 more rows
```