Image Clustering Project

Maciej Kasztelanic

Winter Semester 2023/2024 UL

Movie scenes clustering

In this article I am gonna show the usage of Unsupervised Learning methods for image clustering. Basis of this project is finding the dominant color of each movie frame and understand the flow of the movie.

Data aquicition

All the images used in this article were created and downloaded through online converter (https://www.onlineconverter.com/extract-image-from-video). "Fast and Furious 6" divided into 262 frames (image every 30 second)

Data preprocessing

Lets firstly load all needed packages

```
library(jpeg)
library(rasterImage)
```

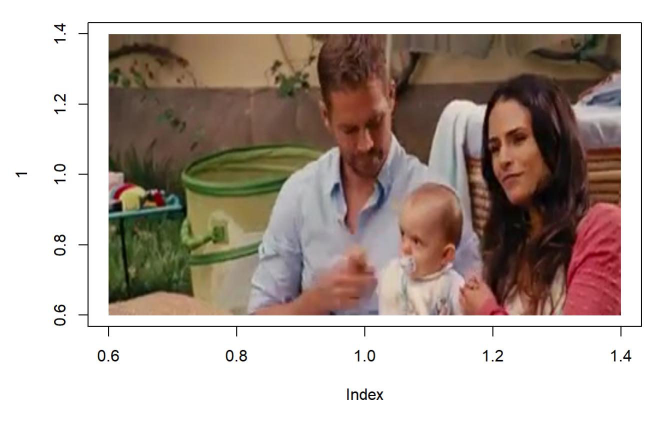
```
## Ładowanie wymaganego pakietu: plotrix
```

```
library(cluster)
library(ggplot2)
library(ggrepel)
```

Positive scene

Then lets load first image and perform all operations on it

```
positive_img <- readJPEG("images/0027.jpg")
plot(1, type="n")
rasterImage(positive_img, 0.6, 0.6, 1.4, 1.4)
```



Let's check the dimensions of the images

```
dm1 <- dim(positive_img);
dm1[1:2]
```

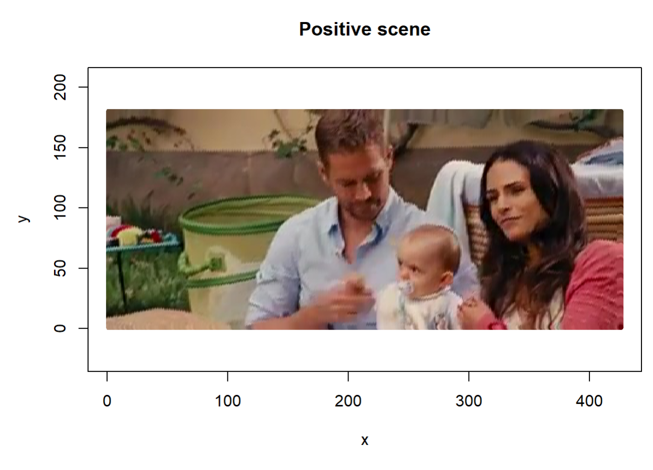
```
## [1] 180 426
```

For image processing we need to change the picture to a matrix of numbers. The RGB format proves to be highly convenient for this purpose. RGB stands for the amounts of red, green, and blue in each pixel of an image. This format simplifies processing, as it organizes the image into a matrix with three columns, where each corresponds to color intensity in range of 0 to 255.

```
rgb_positive_img <- data.frame(
  x=rep(1:dm1[2], each=dm1[1]),
  y=rep(dm1[1]:1, dm1[2]),
  r.value=as.vector(positive_img[,,1]),
  g.value=as.vector(positive_img[,,2]),
  b.value=as.vector(positive_img[,,3]))
```

Then we can plot the image again, but now with rgb values:

```
plot(y ~ x, data=rgb_positive_img, main="Positive scene",
     col = rgb(rgb_positive_img[c("r.value", "g.value", "b.value")]),
     asp = 1, pch = 20)
```



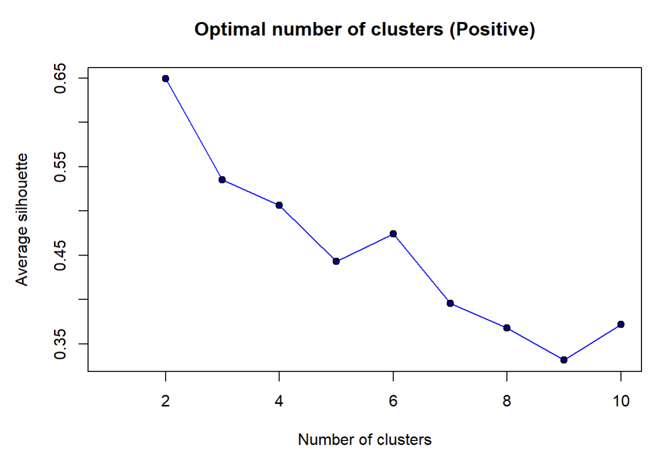
Clustering analysis

To find the optimal number of clusters I will use the silhouette score. It measures how well-defined and distinct the clusters are by providing a quantitative measure of how well each object has been classified.The silhouette score ranges from -1 to 1, where a high value indicates that the object is well-matched to its own cluster and poorly matched to neighboring clusters, while a low or negative value indicates that the object may be assigned to the wrong cluster[1].

```
silhouette_values <- c()
  for (i in 1:10) {
    cl <- clara(rgb_positive_img[, c("r.value", "g.value", "b.value")], i)
    silhouette_values[i] <- cl$silinfo$avg.width
  }
```

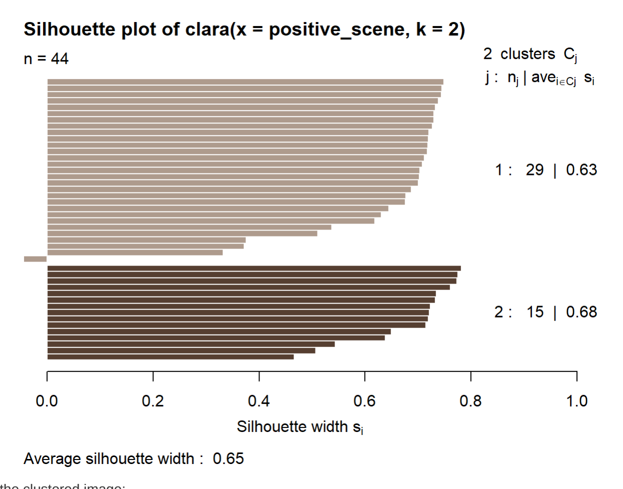
And then plot the silhouette score for each number of clusters:

```
plot_positive <- plot(silhouette_values, type='l', main="Optimal number of clusters (Positive)", xlab="Number of clusters", ylab="Average silhouette", col="blue")
points(silhouette_values, pch=21, bg="navyblue")
```



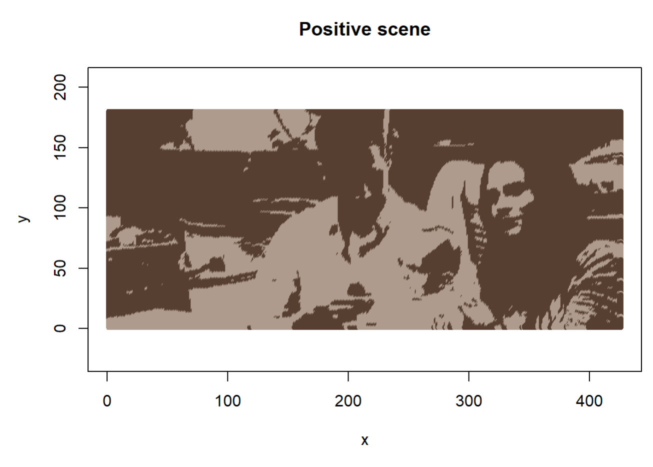
With 2 being the most optiomal number of clusters we can finally perform the CLARA clustering. CLARA (Clustering Large Applications) is an extension of the k-medoids algorithm designed to handle large datasets more efficiently.The algorithm works by selecting a small sample of the data with a fixed size (sampsize) and applying the PAM algorithm to this sample. The quality of the resulting medoids is measured by the average dissimilarity between every object in the dataset and the medoids, which is used as the cost function[2]. The main steps of the CLARA algorithm are as follows: 1. Create a randomly chosen sample from the original dataset. 2. Apply the PAM algorithm to the sample to find the medoids. 3. Calculate the mean (or sum) of the dissimilarities of the observations to their closest medoid. This value is used as a measure of the goodness of the clustering. 4. Retain the sub-dataset for which the mean (or sum) is minimal.

```
positive_scene <- rgb_positive_img[, c("r.value", "g.value", "b.value")]
clara_p <- clara(positive_scene, 2)
plot(silhouette(clara_p), col=c("#AE9B8D", "#563F31"))
```



And plot the clustered image:

```
colors_p <- rgb(clara_p$medoids[clara_p$clustering, ])
plot(y ~ x, data=rgb_positive_img, main="Positive scene",
     col = colors_p,
     asp = 1, pch = 20)
```



The last part of the process is finding the dominant color in the picture. As the analysis was performed for two clusters im trying to find 2 colors.

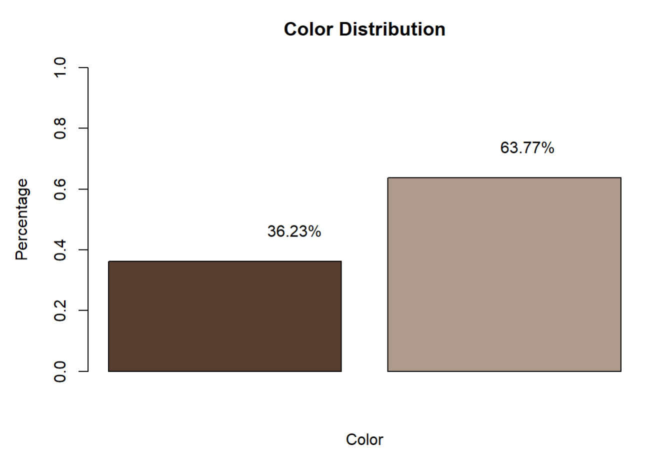
```
dominantColours <- as.data.frame(table(colors_p))

max_col <- max(dominantColours$Freq)/sum(dominantColours$Freq)
min_col <- min(dominantColours$Freq)/sum(dominantColours$Freq)

dominantColours$distribution <- round((c(min_col, max_col) * 100), 2)
dominantColours$colors_p <- as.character(dominantColours$colors_p)
dominantColours
```

```
##   colors_p  Freq distribution
## 1  #563F31 48901        36.23
## 2  #AE9B8D 27779        63.77
```

```
barplot(c(min_col, max_col), col = unique(dominantColours$colors_p), main = "Color Distribution",
        xlab = "Color", ylab = "Percentage", ylim = c(0, 1))
text(c(1, 2), c(min_col, max_col) + 0.05, labels = paste0(dominantColours$distribution, "%"), pos = 3, col = "black")
```



With such Analysis we can find the dominating color on a picture, which can help us detect easily what is happening in the movie without knowing it exactly.

As the main goal of this article is to find dominant colors of the pictures and not improve quality of clustered images I won't consider other algorythms, as all would result in same colors.

Funtions

For further analysis I will combine the code above into funtions:

1.

```
calculate_silhouette <- function(data) {
  silhouette_values <- c()
  for (i in 1:10) {
    cl <- clara(data[, c("r.value", "g.value", "b.value")], i)
    silhouette_values[i] <- cl$silinfo$avg.width
  }
  return(silhouette_values)
}
```

Which calculates optimal number of clusters.

2.

```
image_to_rgb <- function(image) {
  dm <- dim(image);
  rgb_image <- data.frame(
    x=rep(1:dm[2], each=dm[1]),
    y=rep(dm[1]:1, dm[2]),
    r.value=as.vector(image[,,1]),
    g.value=as.vector(image[,,2]),
    b.value=as.vector(image[,,3]))

  return(rgb_image)
}
```

Which changes the image from jpg to data.frame storing rgb values

3.

```
analyze_color_distribution <- function(rgb_image) {

  scene <- rgb_image[, c("r.value", "g.value", "b.value")]
  clara <- clara(scene, 2)
  colors <- rgb(clara$medoids[clara$clustering, ])

  dominantColours <- as.data.frame(table(colors))

  max_col <- max(dominantColours$Freq) / sum(dominantColours$Freq)
  min_col <- min(dominantColours$Freq) / sum(dominantColours$Freq)

  dominantColours$distribution <- round((c(min_col, max_col) * 100), 2)

  max_distribution_color <- dominantColours$colors[which.max(dominantColours$distribution)]

  return(max_distribution_color)
}
```

Which finds the most common (dominant color) on each provided image

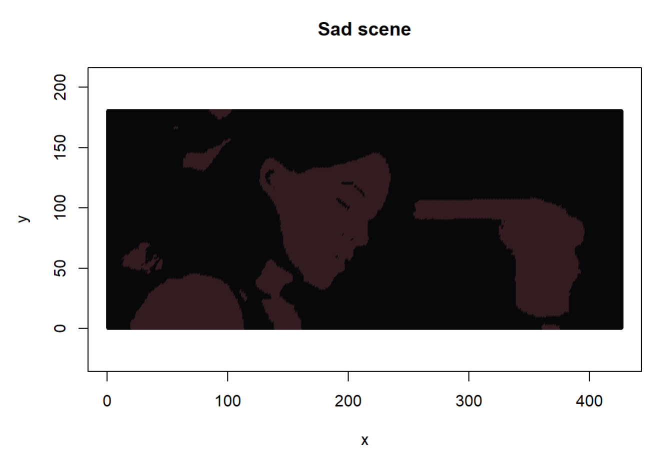
Sad scene

With all that I am gonna find the dominant colors for negative scenes (negative emotions)

```
sad_img <-readJPEG("images/0143.jpg")
dm2 <- dim(sad_img);
rgb_sad_img <- image_to_rgb(sad_img)

sad_scene <- rgb_sad_img[, c("r.value", "g.value", "b.value")]
clara_s <- clara(sad_scene, 2)

colors_s <- rgb(clara_s$medoids[clara_s$clustering, ])
plot(y ~ x, data=rgb_sad_img, main="Sad scene",
     col = colors_s,
     asp = 1, pch = 20)
```



After previewing the image we can see the most dominant color:

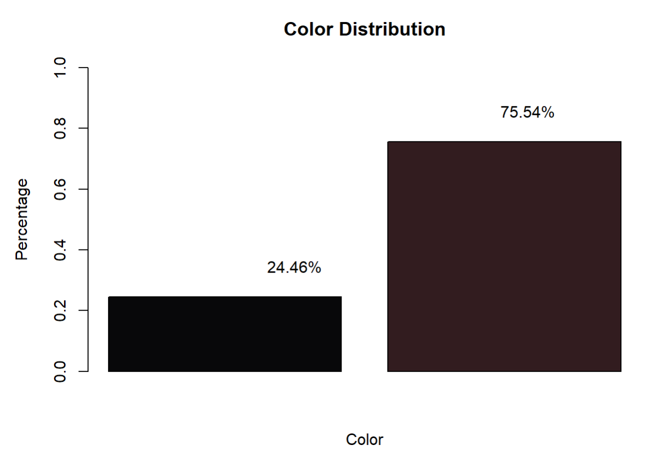
```
dominantColours <- as.data.frame(table(colors_s))

max_col <- max(dominantColours$Freq)/sum(dominantColours$Freq)
min_col <- min(dominantColours$Freq)/sum(dominantColours$Freq)

dominantColours$distribution <- round((c(min_col, max_col) * 100), 2)
dominantColours$colors_s <- as.character(dominantColours$colors_s)
dominantColours
```

```
##   colors_s  Freq distribution
## 1  #08080A 57924        24.46
## 2  #321C1F 18756        75.54
```

```
barplot(c(min_col, max_col), col = unique(dominantColours$colors_s), main = "Color Distribution",
        xlab = "Color", ylab = "Percentage", ylim = c(0, 1))
text(c(1, 2), c(min_col, max_col) + 0.05, labels = paste0(dominantColours$distribution, "%"), pos = 3, col = "black")
```



The colors of the Negative scenes are dark.

In order to excel emotions of a viewer, scenes in movies are usually well organized with every color taken into considerations. Emotions are situational, arising from the brain processing conscious experiences and translating them into feelings, and that conscious and unconscious color symbolism plays an important role in the type of emotion that arises[3]. So usually in action movies the scenery of positively associated scenes will have more light - brighter colors, and one with negative will shift towards the darker theme.

Clustering dominant colors for all of the movie scenes

Lets first create a path and data.farme that will store results

```
image_path <- "images/"
result_df <- data.frame(scene_id = character(), max_distribution_color = character(), stringsAsFactors = FALSE)
```

Secondly lets loop through all the images that are actually from a movie (exclude end credits)

```
for (i in 2:249) {
  # Generate file name
  file_number <- sprintf("%04d", i)
  file_name <- paste0(image_path, file_number, ".jpg")

  # Read the image
  image <- readJPEG(file_name)

  # Convert image to RGB
  rgb_image <- image_to_rgb(image)

  # Perform clara clustering and find the most common color
  max_dist_color <- analyze_color_distribution(rgb_image)

  # Save results in the dataframe
  result_df <- rbind(result_df, data.frame(scene_id = file_number, max_distribution_color = max_dist_color))
}
```

And lastly plot all the dominant colors on a timeline:

```
ggplot(result_df, aes(x = factor(scene_id), fill = max_distribution_color)) +
  geom_bar(stat = "count") +
  scale_fill_identity() +
  labs(title = "Distribution of max colors for each scene",
       x = "Scene ID",
       y = "Count") +

theme(axis.text.x = element_blank(),
      axis.text.y = element_blank(),
      axis.title.x = element_blank(),
      axis.title.y = element_blank())
```

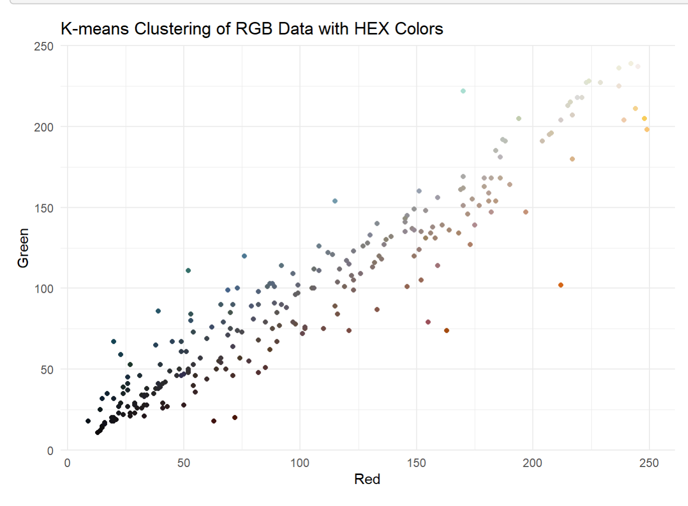
With such distributin we can clearly see which scenes can be considered as positive and negative. What is also interesting we can clearly see yellow and orange bars at the end of the movie which can be assumed to show explosions or/and fire.

```
hex_to_rgb <- function(hex) {
  rgb_values <- col2rgb(hex)
  return(data.frame(
    R = rgb_values[1, ],
    G = rgb_values[2, ],
    B = rgb_values[3, ]
  ))
}
rgb_values_df <- do.call(rbind, lapply(result_df$max_distribution_color, hex_to_rgb))

rgb_df <- cbind(result_df$scene_id, rgb_values_df)
colnames(rgb_df) <- c("scene_id", "R", "G", "B")
```

```
rgb_data <- rgb_df[, c("R", "G", "B")]
merged_df <- merge(rgb_df, result_df, by = "scene_id")
merged_df$max_distribution_color <- factor(merged_df$max_distribution_color, levels = unique(merged_df$max_distri
bution_color))

kmeans_result <- kmeans(rgb_data, centers = 3)
cluster_assignments <- kmeans_result$cluster
rgb_df$cluster <- cluster_assignments
ggplot(merged_df, aes(x = R, y = G, color = max_distribution_color)) +
  geom_point() +
  scale_color_identity() +  # Use the specified colors
  labs(title = "K-means Clustering of RGB Data with HEX Colors",
       x = "Red", y = "Green",
       color = "Max Distribution Color") +
  theme_minimal()
```



The last graph shows us that the entire movie is kept in rather constant color palette, with few outliers breaking the straight line of colors.

Conclusion

In summary, this project utilized unsupervised learning methods to analyze the dominant colors in movie scenes, focusing on "Fast and Furious 6." Beginning with data acquisition and preprocessing, we applied the CLARA clustering algorithm to efficiently categorize frames into distinct clusters. The subsequent identification of dominant colors revealed a compelling pattern: positive scenes predominantly featured lighter hues, while negative scenes gravitated towards darker tones.Extending the analysis to all movie scenes created a visual timeline showcasing the emotional dynamics throughout the film. Peaks and shifts in dominant colors aligned with key moments, offering a unique perspective on the cinematic narrative.

1. https://bookdown.org/brittany_davidson1993/bookdown-demo/cluster-analysis-part-1.html↩
2. https://www.datanovia.com/en/lessons/clara-in-r-clustering-large-applications/↩
3. "Color and Emotion in Movies" - Yoshanka Samarawira↩