Modern Football Analysis Using Data Science

What can football statistics show us?

StatsBomb

StatsBoom is a data provider that collects accurate metrics of sport events, allowing for a very indepth technical analysis of how players and teams perform

We are fetching information for our specific match from the StatsBomb dataset from their github repository using:

```
devtools::install_github("statsbomb/StatsBombR")
library(StatsBombR)
devtools::install_github("FCrSTATS/SBpitch")
library(SBpitch)
```

01

Data fetching and filtering

02

Histogram and scatterplot

03

Heat map and Goal representation map

04

Pass Distribution Neural Network

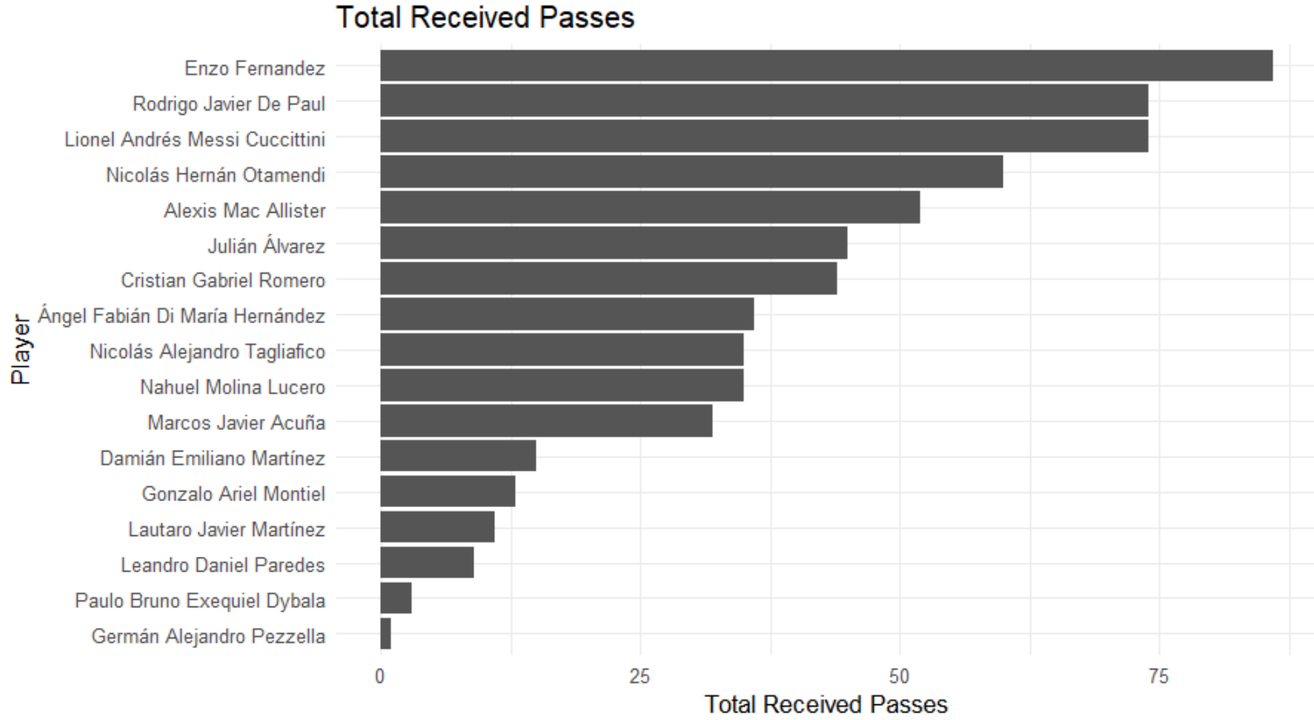
Data Fetching and filtering

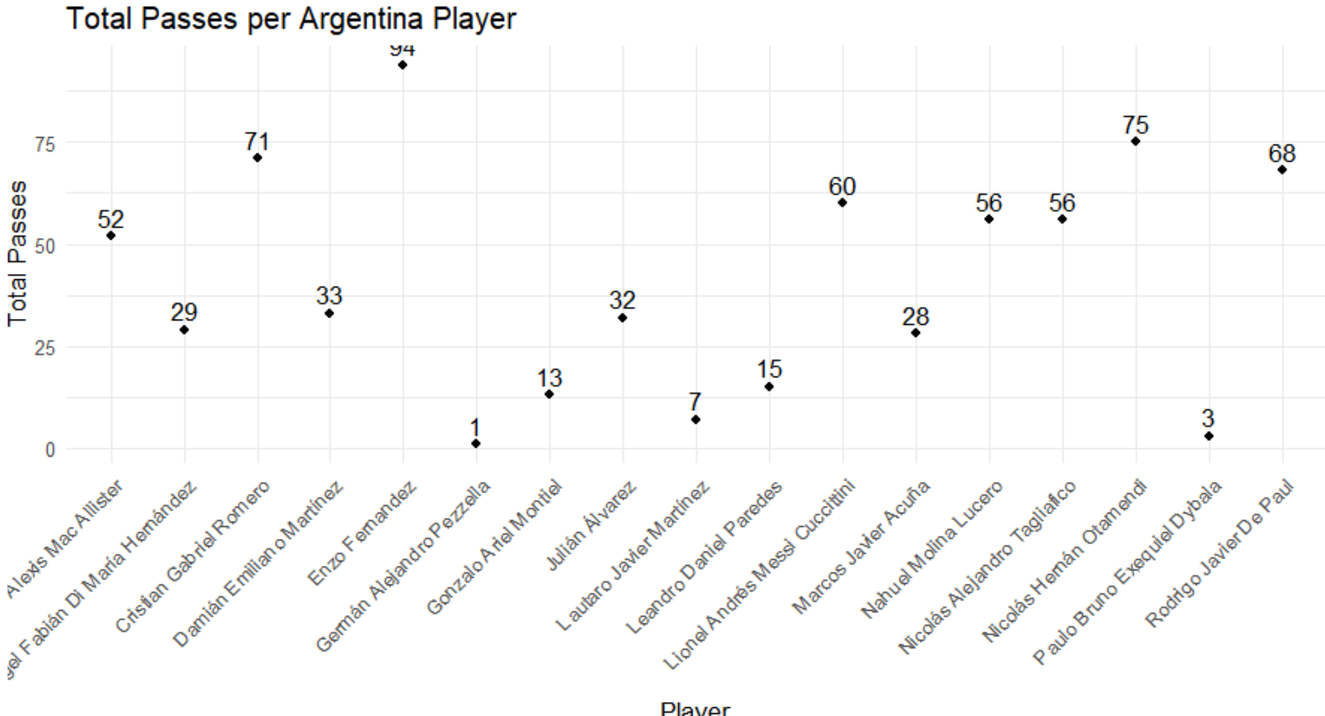
-Retrieve data on available competitions
- Fetche match data for the specified competition and match
-Filter data only for Argentina players

```r
# Fetch the data
comps <- FreeCompetitions() %>%
 filter(competition_name == "FIFA World Cup" & season_name == "2022")
comp_data <- FreeMatches(comps)


# Filter for the specific match
specific_match <- comp_data%>% filter(match_id == 3869685)

events <- free_allevents(MatchesDF = specific_match,Parallel = T)
events <- allclean(events)

# Filter for Argentina passes in the specific match
Argentina_passes <- events %>% filter(team.name == "Argentina",
 type.name == "Pass", is.na(pass.outcome.name))
```

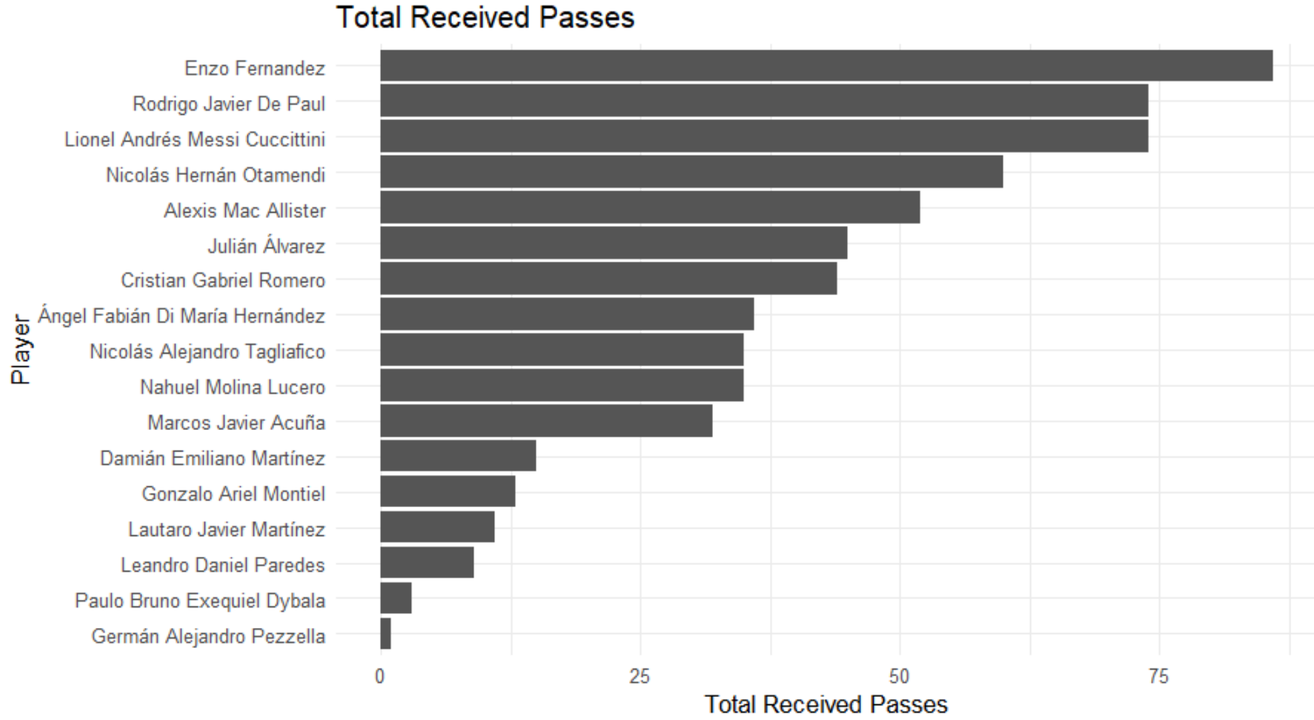
Passes received

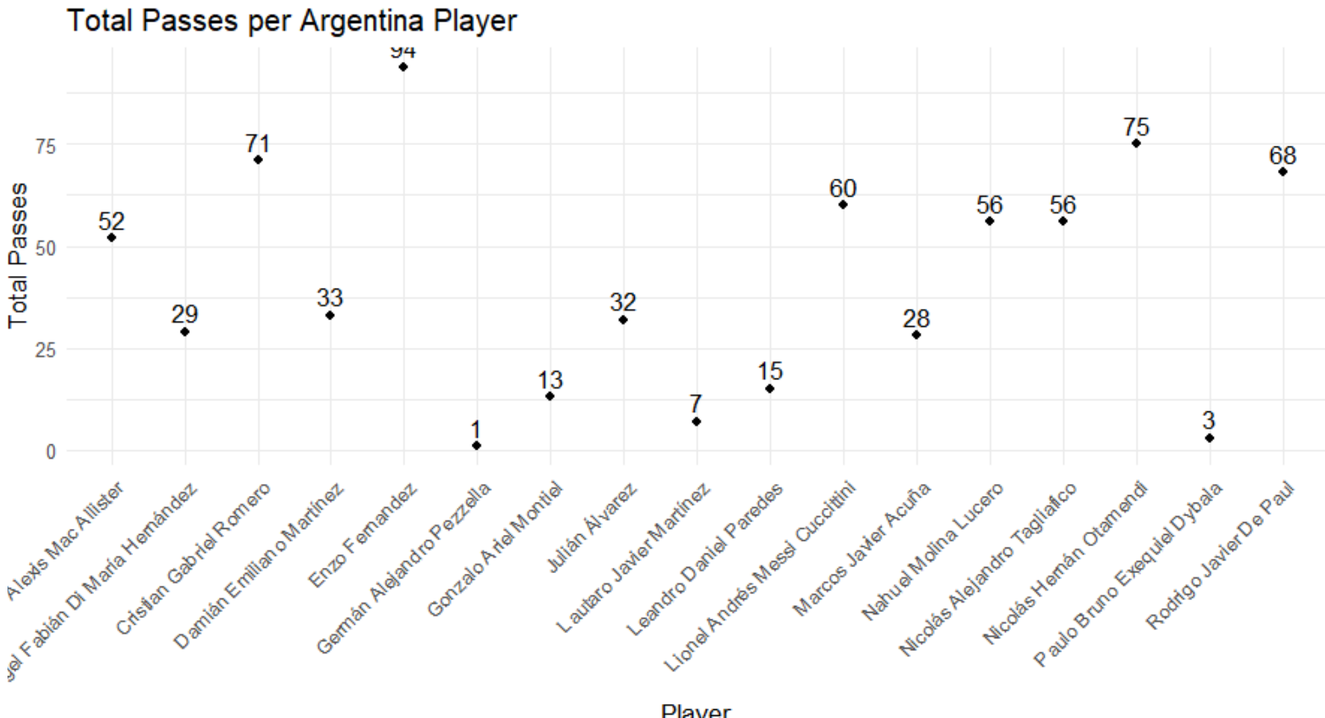
Passes made



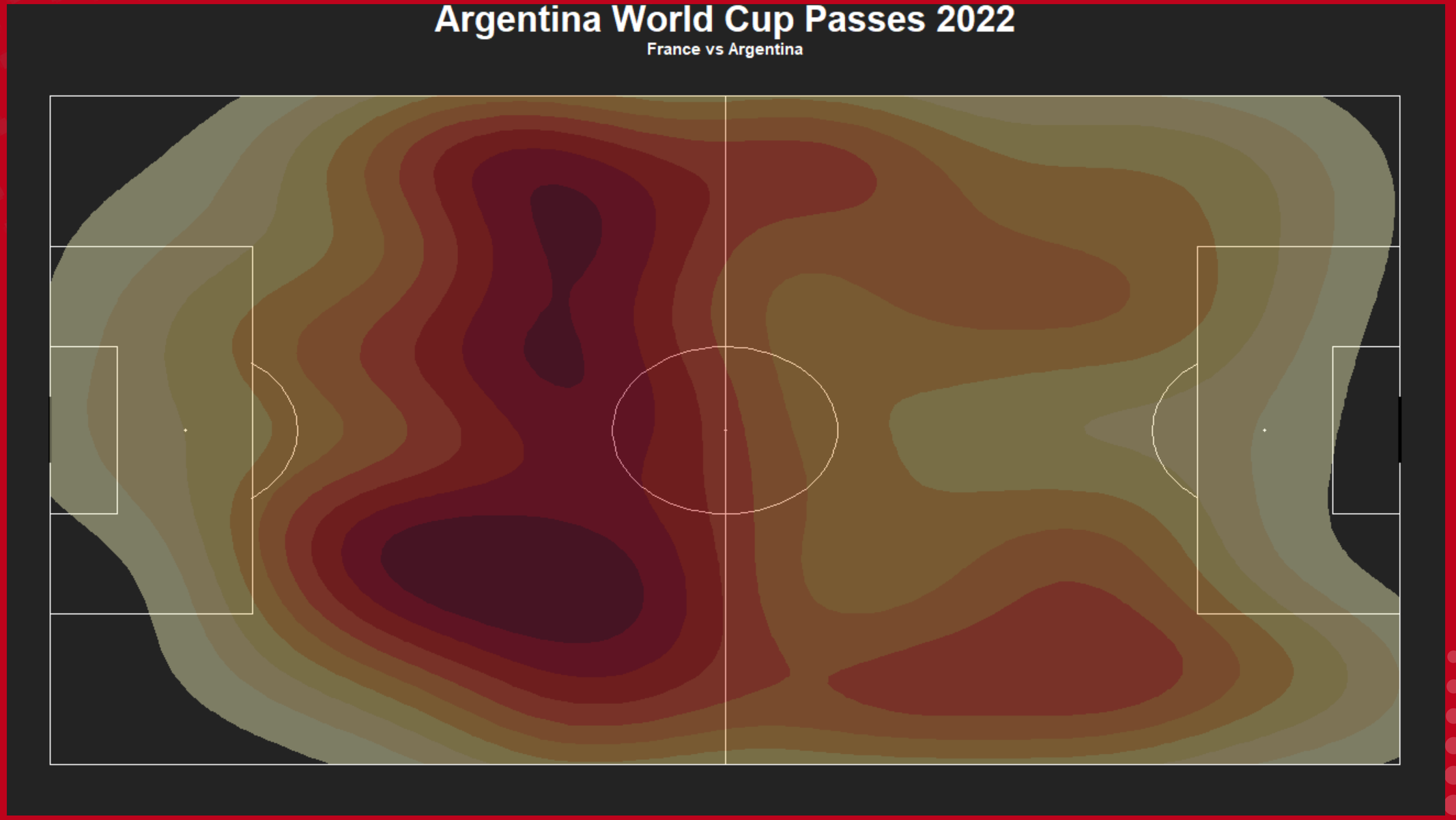


Can you see the correlation?





HEAT MAP OF PASS DISTRIBUTION

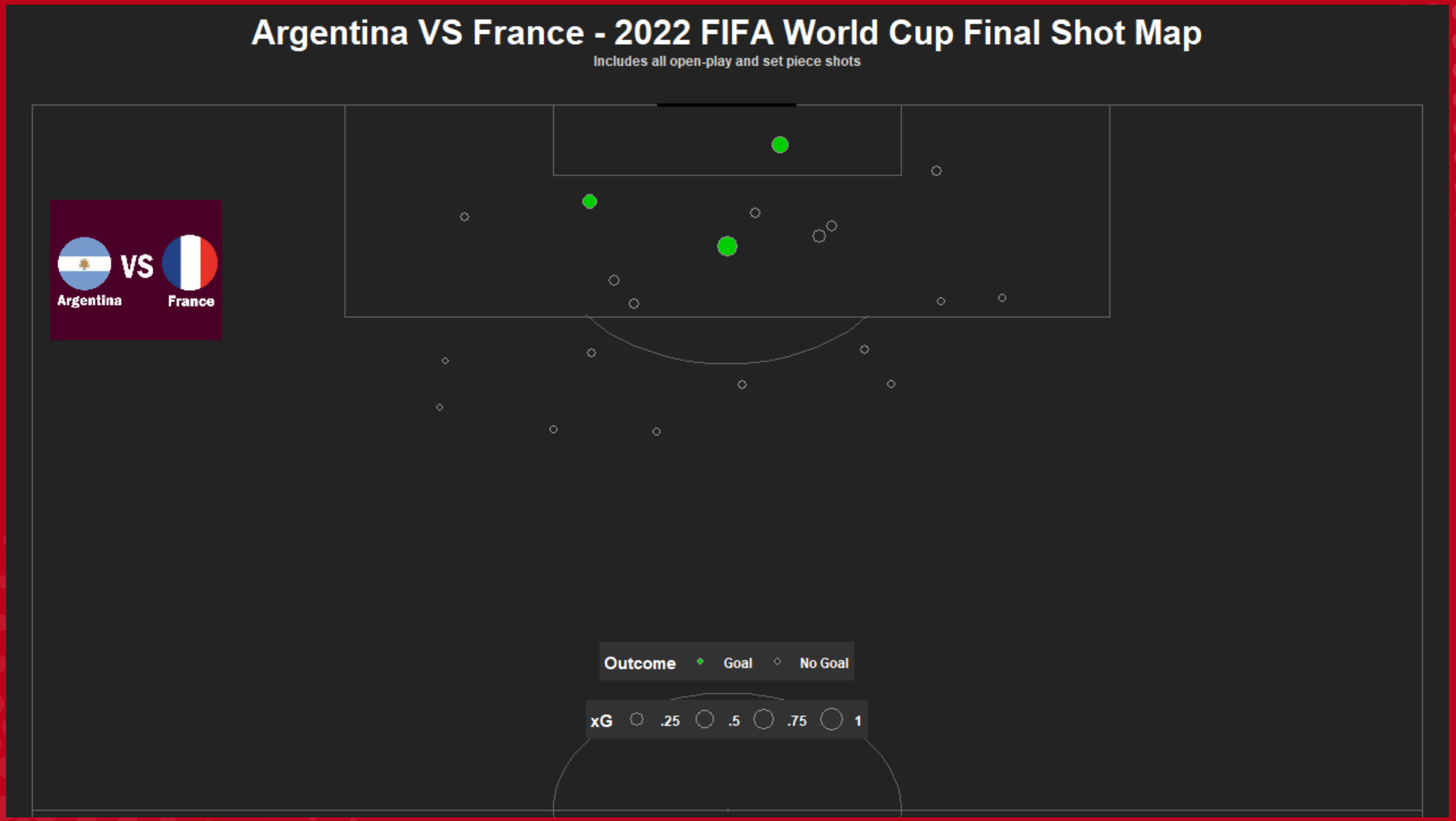


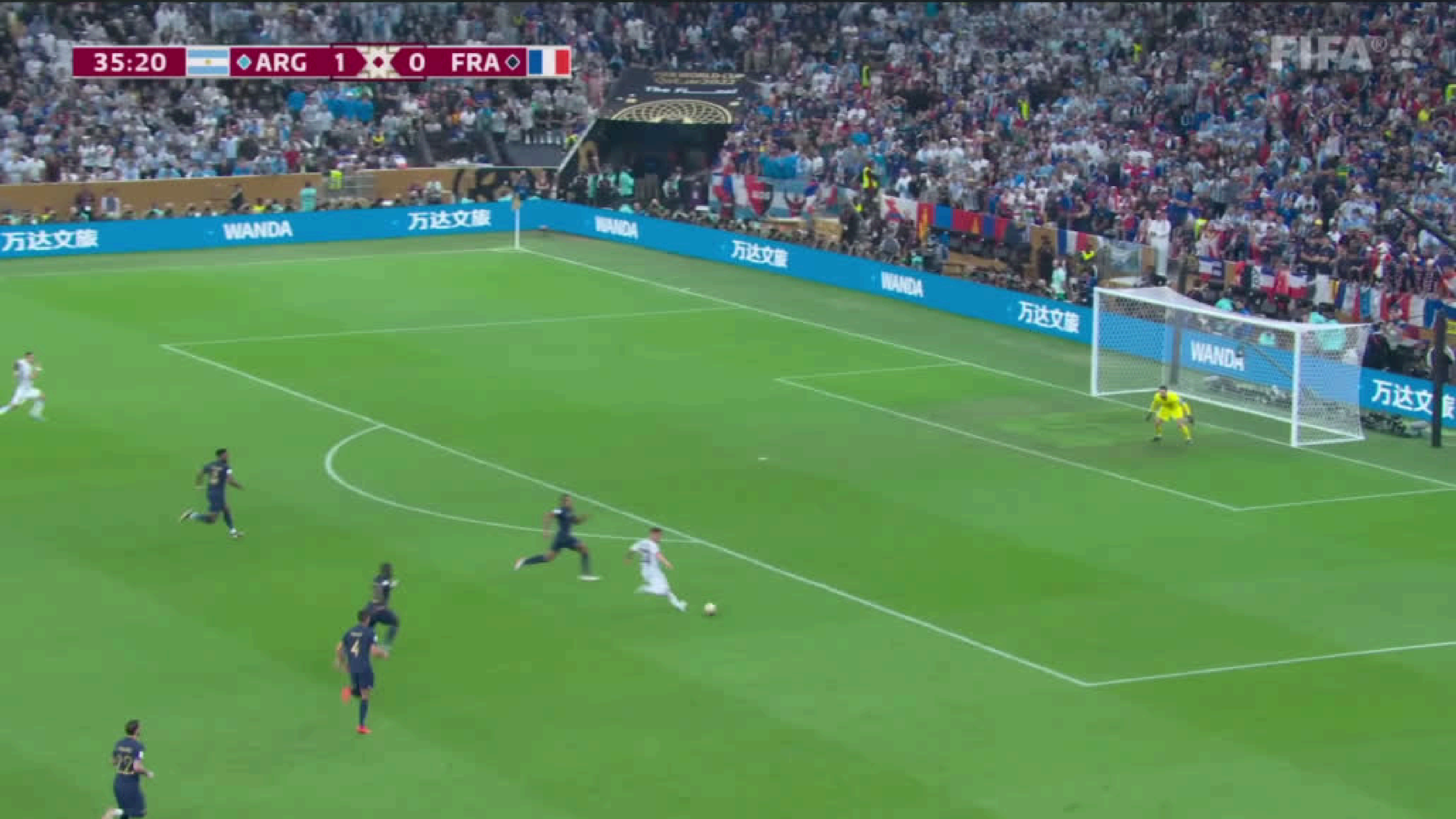

FIFA+
GO TO FIFAPLUS.COM
FIFA+
GO TO FIFAPLUS.COM

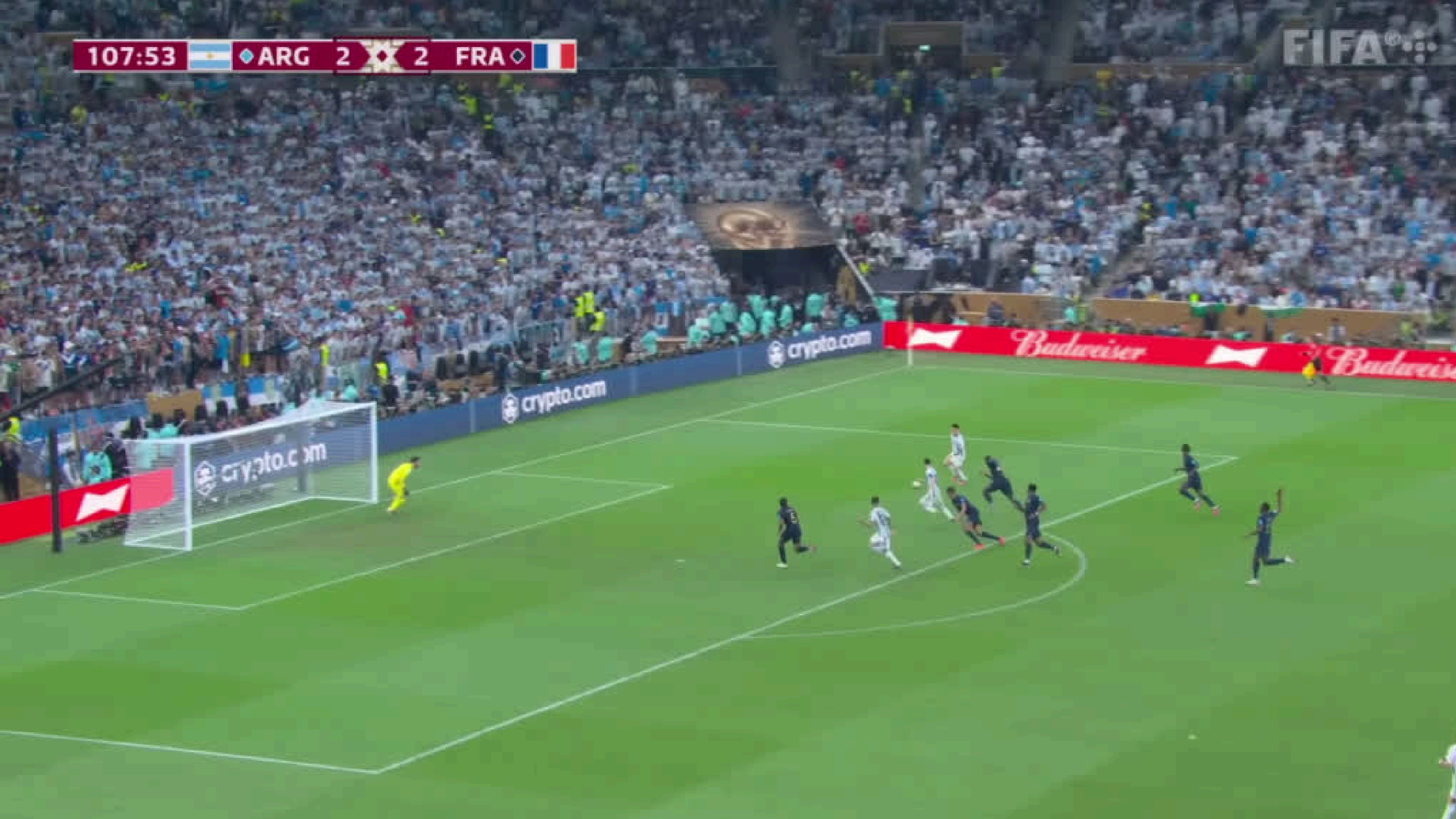
```r
#Select necessary columns
Argentina_passes <- Argentina_passes %>% select(period, minute, type.name, pass.length, pass.angle, player.name, position.name, pass.recipient.name,
                                        pass.outcome.name, pass.height.name, pass.body_part.name, location.x, location.y,
                                        pass.end_location.x, pass.end_location.y, carry.end_location.x, carry.end_location.y,
                                        shot.end_location.x, shot.end_location.y, shot.end_location.z)

# Define color palette
palette <- paletteer::paletteer_d("RColorBrewer::YlOrRd", direction = 1)

# Create the heatmap
p1 <- create_Pitch(grass_colour = "gray15", background_colour = "gray15", line_colour = "white") +
  geom_density_2d_filled(data = Argentina_passes, aes(x = pass.end_location.x, y = pass.end_location.y, fill = ..level..),
                         alpha = .4, contour_var = "ndensity", breaks = seq(0.1, 1.0, length.out = 10)) +
  theme(legend.position = "none") +
  scale_x_continuous(limits = c(0, 120)) +
  scale_y_continuous(limits = c(0, 80)) +
  scale_fill_manual(values = c(palette), aesthetics = c("fill", "color")) +
  theme(legend.position = "none",
        plot.background = element_rect(colour = "gray15", fill = "gray15"),
        plot.title = element_text(color = "white", hjust = .5, size = 22, family = "Comic Sans MS", face = "bold", vjust = -1),
        plot.subtitle = element_text(color = "white", hjust = .5, size = 10, family = "Comic Sans MS", face = "bold", vjust = -4),
        plot.caption = element_text(color = "white", hjust = .5, size = 10, family = "Comic Sans MS", face = "bold", vjust = 4)) +
  labs(title = "Argentina World Cup Passes 2022",
       subtitle = "France vs Argentina",
       caption = "")

# Plot the heatmap
ggdraw(p1) + theme(plot.background = element_rect(fill = "gray15", color = NA))
```

-We use library(SBpitch) and (ggplot2) to plot and draw the heatmap.We visualize the density of pass locations on the pitch for Argentina players

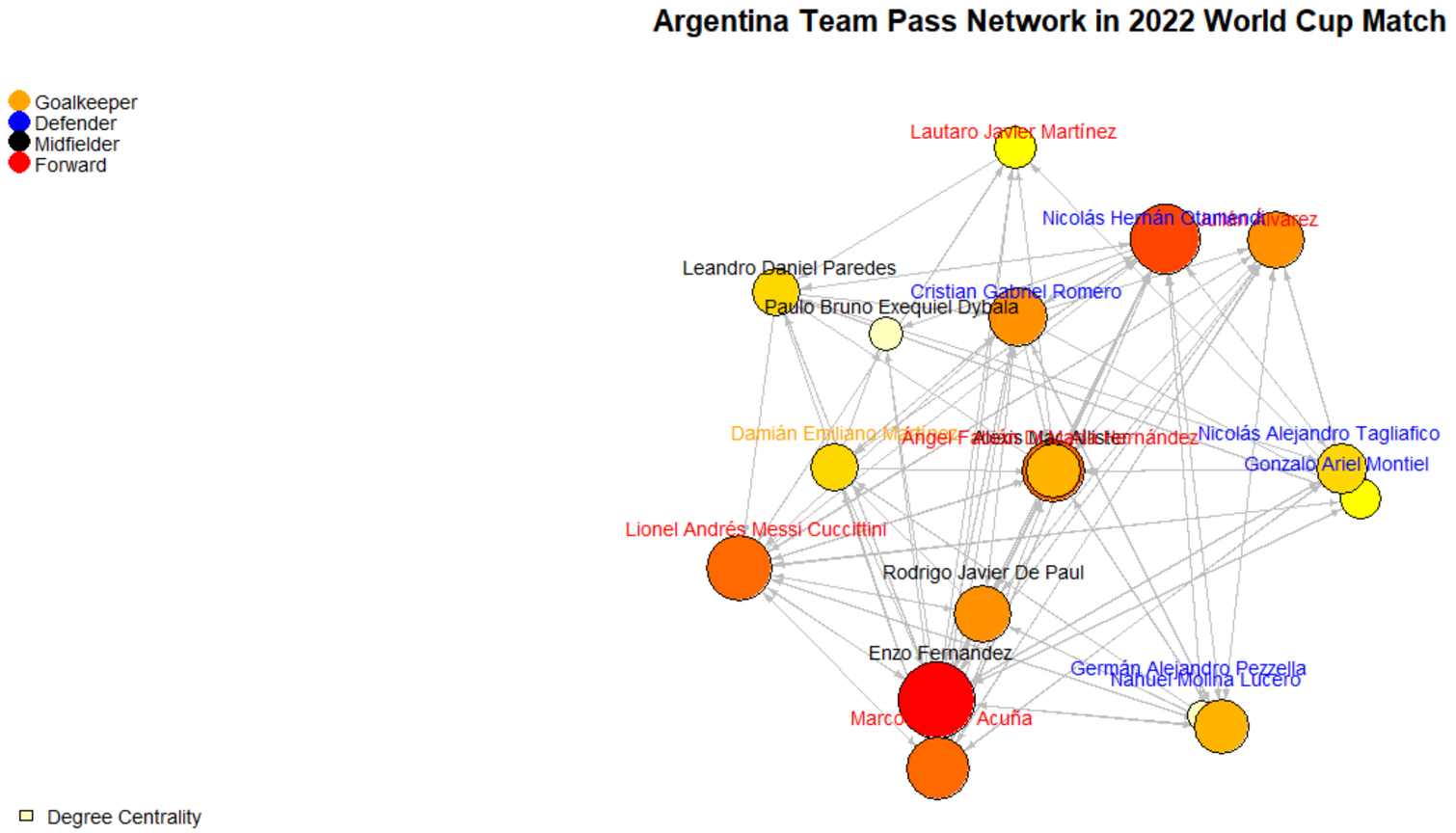
Goal representation map




35:20
ARG 1 0 FRA
FIFA
WANDA
万达文旅


107:53
ARG 2 2 FRA
FIFA
crypto.com
Budweiser

PASS DISTRIBUTION NEURAL NETWORK



```r
# Calculate node colors based on degree centrality
node_colors <- rev(heat.colors(10))  # Generate a gradient from blue to red

# Assign colors based on degree centrality
V(g)$color <- node_colors[rescale(node_degrees, to = c(1, 10))]

# Create a 3D spherical layout
l <- layout_on_sphere(g)

# Plotting in 3D using igraph without labels
plot(g, layout = l, vertex.size = node_sizes, vertex.color = V(g)$color, edge.color = "gray", edge.width = 1,
     vertex.label = NA,  # Disable default vertex labels
     main = "Argentina Team Pass Network in 2022 World Cup Match",
     sub = "Nodes are colored from (Yellow to Red) based on the players passing abilities",
     edge.arrow.size = 0.4)  # Adjust arrow size here (smaller tips))

# Add custom vertex labels with resized text and color coding
for (i in 1:vcount(g)) {
  player_name <- V(g)$name[i]
  role <- V(g)$role[i]
  text(l[i, 1], l[i, 2], player_name, col = ifelse(role == "Goalkeeper", "orange",
                                                   ifelse(role == "Defender", "blue",
                                                          ifelse(role == "Midfielder", "black", "red"))), cex = 0.8, pos = 3)
}
```

- We Define position groups for different roles (Goalkeeper, Defender, Midfielder, Forward).
- Add custom vertex labels for each position and color code them
- Maps each player to their respective role based on their position.

```r
#Neural Network representation
# Load necessary libraries
library(igraph)
library(ggraph)
library(tidyverse)
library(scales)

edge_list <- Argentina_passes %>%
  filter(!is.na(pass.recipient.name)) %>%
  group_by(player.name, pass.recipient.name) %>%
  summarise(weight = n()) %>%
  ungroup()

# Create graph object
g <- graph_from_data_frame(d = edge_list, directed = TRUE)
```

- we use igraph: For creating and plotting graph objects.
- ggraph: For enhanced graph plotting.
- tidyverse: For data manipulation and cleaning.
- scales: For rescaling numeric values.

- We create an Edge List and filters out passes with no recipient.
- Groups the data by player and their pass recipient.
- Counts the number of passes between each player pair to use as edge weights.

```r
# Calculate node sizes based on the total number of passes each player is involved in
node_degrees <- degree(g, mode = "all")
node_sizes <- rescale(node_degrees, to = c(10, 25))  # Rescale using scales package
```

- We computes the degree (number of connections) of each node.
- Rescales these values to determine node sizes for the plot.

For the future:

We plan to exponentially update and further improve our neural network with model predictions and better visualisation.We can develop a Linear Regression model for predicting the outcome of future matches,based on certain players impact on previous games.

Thank you for your attention!

References:

1)https://github.com/statsbomb/open-data

StatsBomb

GitHub:

Scan me!