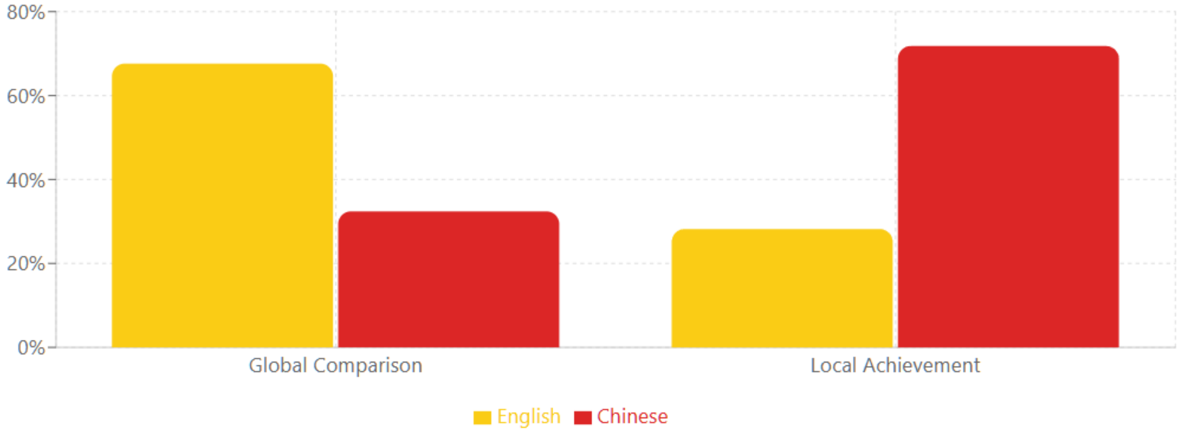
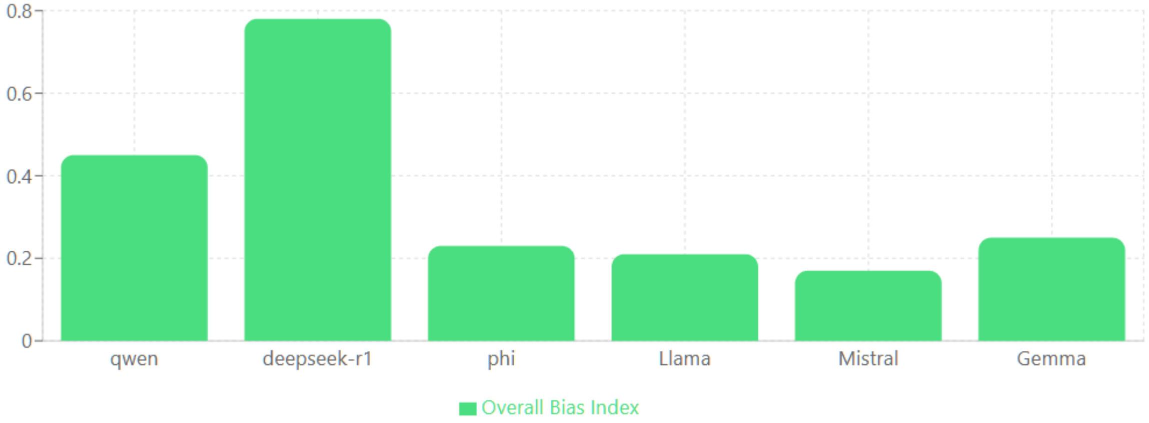
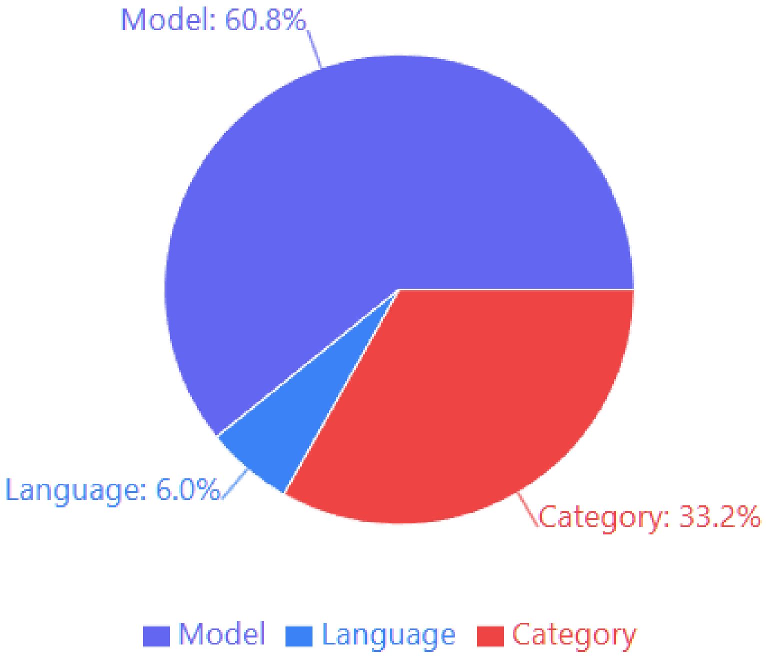


```
/*  Framing Pattern Frequencies: English vs. Chinese (Grouped Bar Chart) */}
   <ChartContainer
    title="Visual 7: Framing Pattern Frequencies: English vs. Chinese"
    description="This grouped bar chart compares the frequency of different framing patterns in English and Chinese responses, highlighting divergent communication styles."
   >
    <ResponsiveContainer>
     <BarChart data={framingFrequencyData} margin={{ top: 20, right: 30, left: 20, bottom: 5 }}>
      <CartesianGrid strokeDasharray="3 3" stroke="#e0e0e0" />
      <XAxis dataKey="pattern" tickLine={false} axisLine={{ stroke: '#cccccc' }} />
      <YAxis tickFormatter={(value) => `${value}%`} axisLine={{ stroke: '#cccccc' }} />
      <Tooltip cursor={{ fill: 'rgba(0,0,0,0.05)' }} formatter={(value) => `${value.toFixed(1)}%`} />
      <Legend wrapperStyle={{ paddingTop: '10px' }} />
      <Bar dataKey="English" fill={COLORS[6]} radius={[10, 10, 0, 0]} />
      <Bar dataKey="Chinese" fill={COLORS[7]} radius={[10, 10, 0, 0]} />
     </BarChart>
    </ResponsiveContainer>

   </ChartContainer>
```



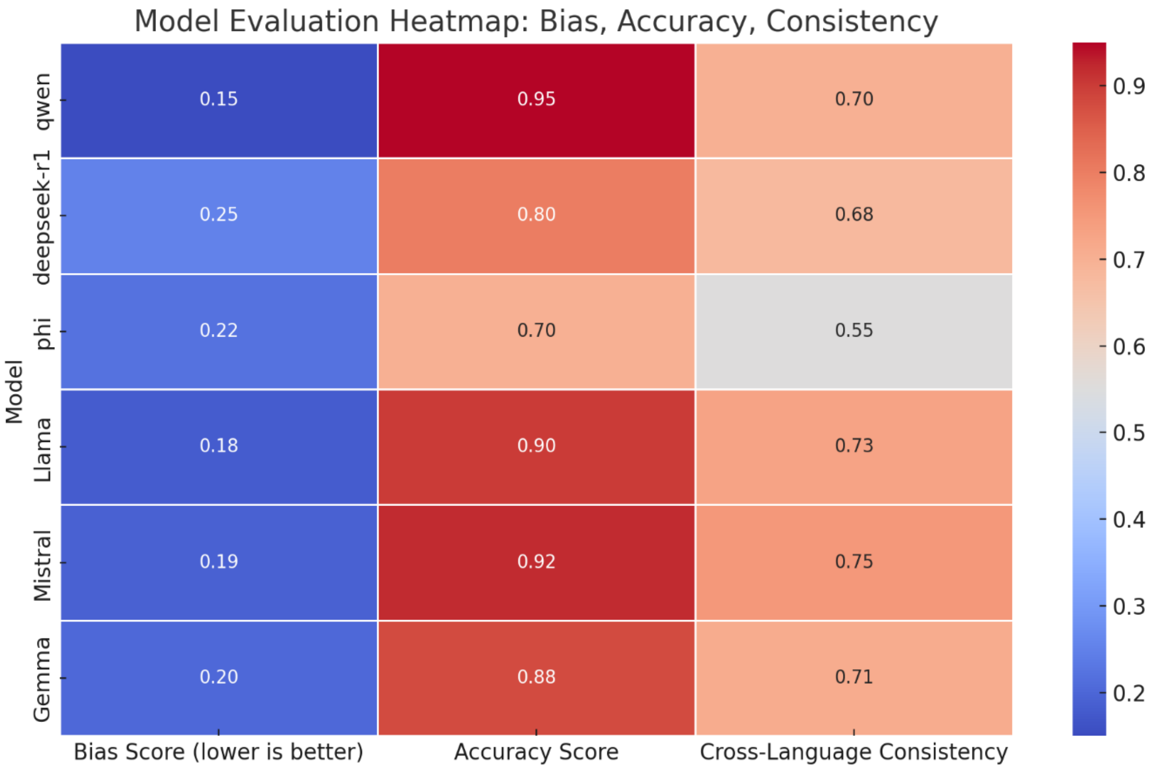
```
<ChartContainer
      title="Visual 9: Overall Bias Index by Model"
      description="This bar chart visualizes the overall bias index for each LLM,
highlighting models with higher or lower tendencies towards biased responses."
     >
      <ResponsiveContainer>
        <BarChart data={overallBiasIndexData} margin={{ top: 20, right: 30, left: 20,
bottom: 5 }}>
          <CartesianGrid strokeDasharray="3 3" stroke="#e0e0e0" />
          <XAxis dataKey="model" tickLine={false} axisLine={{ stroke: '#cccccc' }} />
          <YAxis domain={[0, 0.8]} axisLine={{ stroke: '#cccccc' }} />
          <Tooltip cursor={{ fill: 'rgba(0,0,0,0.05)' }} formatter={(value) => value.toFixed(2)}
/>
          <Legend wrapperStyle={{ paddingTop: '10px' }} />
          <Bar dataKey="Overall Bias Index" fill={COLORS[8]} radius={[10, 10, 0, 0]} />
        </BarChart>
      </ResponsiveContainer>
     </ChartContainer>
```



```
{/* Visual 11: ANOVA Eta-Squared (η²) by Source (Pie Chart) */}
  <ChartContainer
   title="Visual 11: ANOVA Eta-Squared (η²) by Source"
   description="This pie chart illustrates the proportion of variance explained by different sources (Model, Language, Category) in the ANOVA analysis, indicating their relative impact on elaboration."
  >
   <ResponsiveContainer>
    <PieChart>
     <Pie
      data={anovaEtaSquaredData}
      dataKey="η²"
      nameKey="source"
      cx="50%"
      cy="50%"
      outerRadius={120}
      fill="#8884d8"
      label={({ name, percent }) => `${name}: ${(percent * 100).toFixed(1)}%`}
     >
```

```
       {anovaEtaSquaredData.map((entry, index) => (
        <Cell key={`cell-${index}`} fill={COLORS[index % COLORS.length]} />
       ))}
      </Pie>
      <Tooltip formatter={(value) => value.toFixed(3)} />
      <Legend wrapperStyle={{ paddingTop: '10px' }} />
     </PieChart>
    </ResponsiveContainer>
   </ChartContainer>

  </div>
 );
};
```

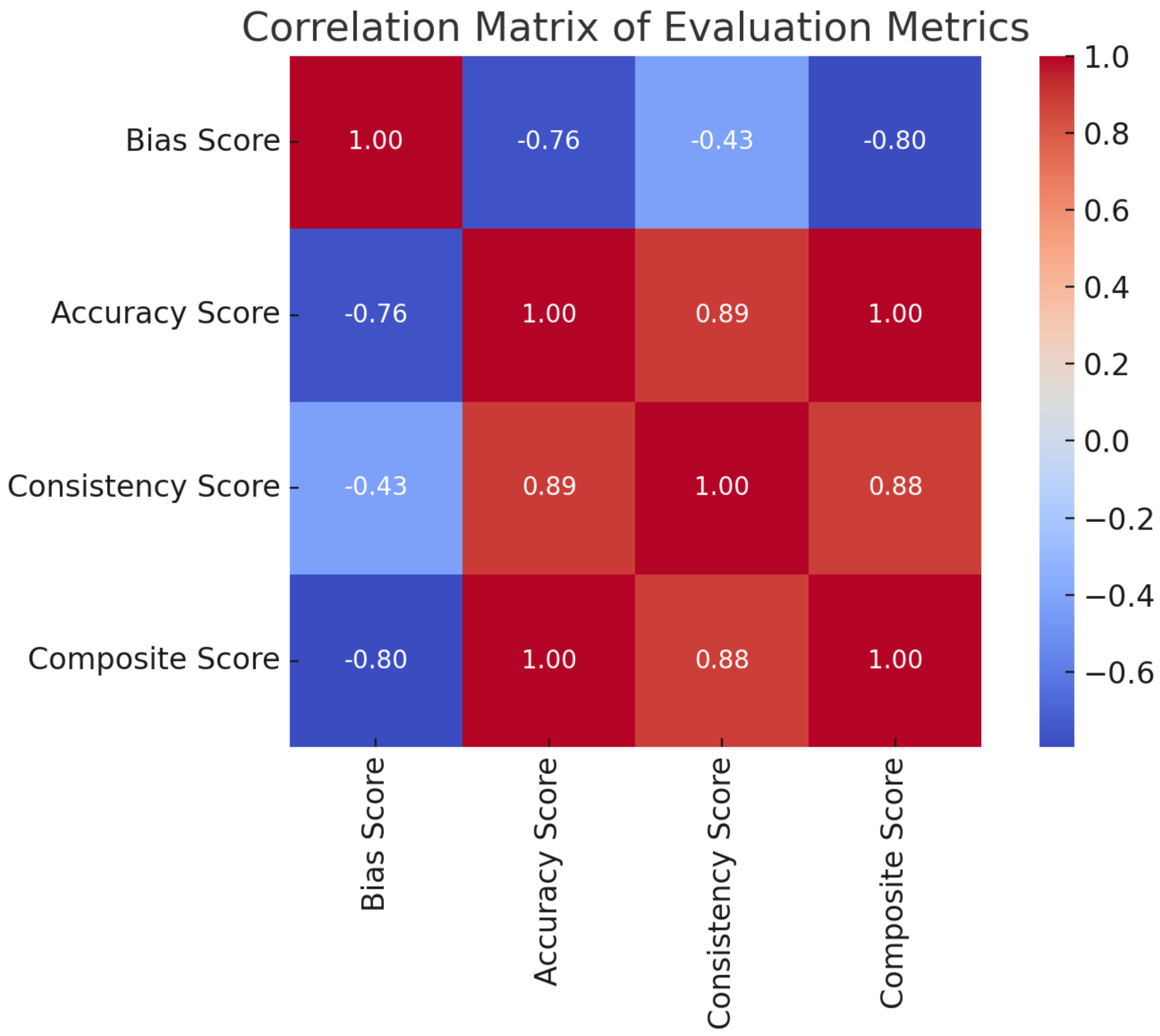


For heatmaps in thesis

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Data
heatmap_data_ordered = pd.DataFrame({
    'Model': ['qwen', 'deepseek-r1', 'phi', 'Llama', 'Mistral', 'Gemma'],
    'Bias Score (lower is better)': [0.15, 0.25, 0.22, 0.18, 0.19, 0.20],
    'Accuracy Score': [0.95, 0.80, 0.70, 0.90, 0.92, 0.88],
    'Cross-Language Consistency': [0.70, 0.68, 0.55, 0.73, 0.75, 0.71]
}).set_index("Model")

plt.figure(figsize=(8, 6))
sns.heatmap(
    heatmap_data_ordered,
    annot=True,
    cmap="coolwarm",
    fmt=".2f",
    linewidths=0.5,
    linecolor='white',
    cbar_kws={'label': 'Score'}
)
plt.title("Model Evaluation Heatmap: Bias, Accuracy, Consistency", fontsize=14)
plt.tight_layout()
plt.show()
```



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt


# 'Consistency Score', and 'Composite Score'.
data = {
  'Bias Score': [1.00, -0.76, -0.43, -0.80],
  'Accuracy Score': [-0.76, 1.00, 0.89, 1.00],
  'Consistency Score': [-0.43, 0.89, 1.00, 0.88],
   'Composite Score': [-0.80, 1.00, 0.88, 1.00]
}

# Define the index (row labels) for the DataFrame, matching the order in the image.
index_labels = ['Bias Score', 'Accuracy Score', 'Consistency Score', 'Composite Score']

# Create a Pandas DataFrame from the data, specifying both data and index/column
labels.
# This structure is essential for Seaborn's heatmap to correctly interpret the
correlation matrix.
df_corr = pd.DataFrame(data, index=index_labels, columns=index_labels)

# Define the figure size for the plot.
# A square or slightly wider aspect ratio often works well for correlation matrices.
plt.figure(figsize=(8, 7))

# sns.heatmap(): This function creates the heatmap visualization.
# df_corr: The DataFrame containing the correlation matrix.
# annot=True: Displays the correlation coefficients on the heatmap cells.
# fmt=".2f": Formats the annotations to two decimal places.
# cmap='coolwarm': Uses the 'coolwarm' colormap. This is a diverging colormap
where
#             red typically indicates positive correlation, blue indicates negative,
#             and white/grey indicates near-zero correlation, matching the image.
# linewidths=.5: Adds a thin white line between cells, enhancing visual separation.
# linecolor='white': Sets the color of these lines to white.
# cbar_kws={'label': ''}: Customizes the color bar. Removing the label keeps the plot
clean
#                   and matches the provided image.
sns.heatmap(
   df_corr,
   annot=True,
   fmt=".2f",
   cmap='coolwarm',
   linewidths=.5,
   linecolor='white',
   cbar_kws={'label': ''}
)
```

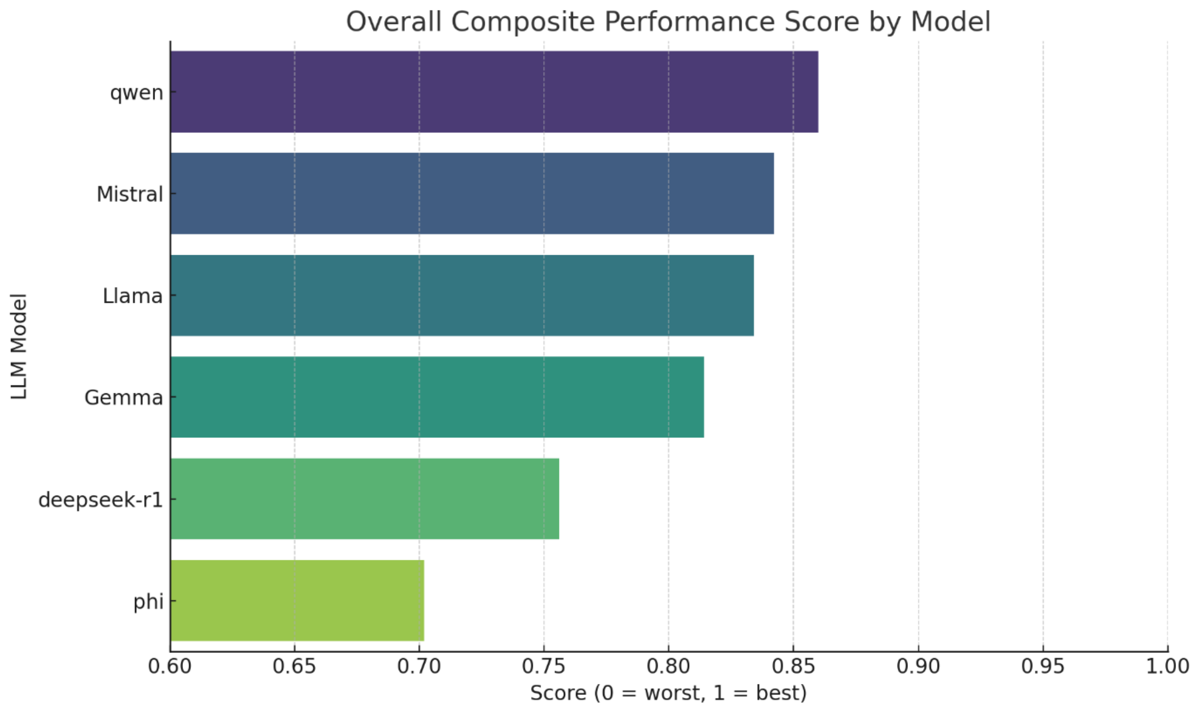
```
# Set the title of the plot.
plt.title('Correlation Matrix of Evaluation Metrics', fontsize=16, pad=20)

# Rotate the tick labels on both X and Y axes to 90 degrees for X-axis and 0 degrees for Y-axis.
# This ensures that all labels are readable without overlapping, especially for the X-axis.
plt.xticks(rotation=90)
plt.yticks(rotation=0)

# Adjust the plot layout to prevent labels or other elements from being cut off,
# ensuring the entire visualization is displayed properly.
plt.tight_layout()

# Display the generated plot.
plt.show()

# Optional: Uncomment the line below to save the plot to a file.
# plt.savefig('correlation_matrix_heatmap.png', dpi=300, bbox_inches='tight')
```



```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

model_names = ['qwen', 'Mistral', 'Llama', 'Gemma', 'deepseek-r1', 'phi']
scores = [0.867, 0.842, 0.835, 0.816, 0.758, 0.703]

df_corrected = pd.DataFrame({'Model': model_names, 'Score': scores})
palette_top_purple = sns.color_palette("viridis", len(df_corrected))  # Not reversed

plt.figure(figsize=(9, 5))
sns.barplot(
   data=df_corrected,
   x='Score',
   y='Model',
   palette=palette_top_purple
)
plt.title("Overall Composite Performance Score by Model", fontsize=14)
plt.xlabel("Score (0 = worst, 1 = best)")
plt.ylabel("LLM Model")
plt.xlim(0.60, 1.00)
plt.grid(axis='x', linestyle='--', linewidth=0.5)
plt.tight_layout()
plt.show()
```