TextAugLLMEdge

June 16, 2025

[1]:
```python
# TextAugLLMEdge

# Supplementary Results

# Partha Pratim Ray, 15/6/2025

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

# Path to your CSV file (change if needed)
csv_path = r"D:\technical writing\textaugmentation\result.csv"

# Load data
df = pd.read_csv(csv_path)
print("Columns:", df.columns.tolist())
print(df.head())
```

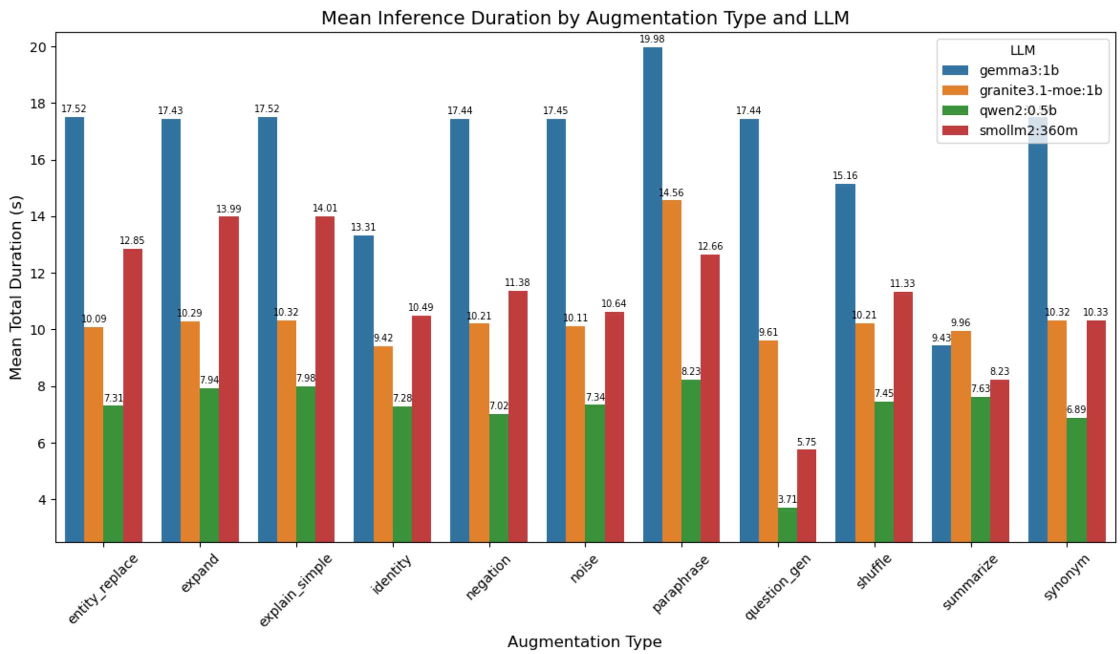
```
Columns: ['timestamp', 'prompt_id', 'prompt', 'augmentation_type', 'model',
'augmented_text', 'total_duration_ns', 'load_duration_ns', 'prompt_eval_count',
'prompt_eval_duration_ns', 'eval_count', 'eval_duration_ns',
'tokens_per_second', 'levenshtein_similarity', 'jaccard_similarity',
'length_ratio', 'bleu', 'cosine_similarity', 'wer', 'char_diversity',
'type_token_ratio', 'bigram_overlap']
                    timestamp  prompt_id  \
0  2025-06-15T09:50:22.829584          1
1  2025-06-15T09:50:33.601336          2
2  2025-06-15T09:50:41.541565          3
3  2025-06-15T09:50:49.370159          4
4  2025-06-15T09:50:57.445704          5

                                              prompt augmentation_type  \
0  Although agriculture remains the backbone of o…        paraphrase
1  The rapid advancement of technology has not on…        paraphrase
2  Despite receiving a prestigious scholarship to…        paraphrase
3  In the aftermath of the devastating earthquake…        paraphrase
4  The increasing adoption of renewable energy so…        paraphrase
```

```
        model                                     augmented_text  \
0  qwen2:0.5b  Despite maintaining a substantial portion of i…
1  qwen2:0.5b  The rapid technological progress over the last…
2  qwen2:0.5b  Despite being offered an impressive scholarshi…
3  qwen2:0.5b  In response to a severe natural disaster that …
4  qwen2:0.5b  Renewable energy sources like solar and wind a…

   total_duration_ns  load_duration_ns  prompt_eval_count  \
0        10743499160        2422096170                 62
1         7918451598          53620750                 58
2         7807772750          51113778                 54
3         8053528241          49385191                 62
4         8027890100          52459502                 61

   prompt_eval_duration_ns  …  tokens_per_second  levenshtein_similarity  \
0               1835488062  …           9.253171                0.008955
1               1415348964  …           9.304752                0.050445
2               1302542405  …           9.300377                0.049261
3               1530142101  …           9.269372                0.213873
4               1501011644  …           9.268947                0.006299

   jaccard_similarity  length_ratio      bleu  cosine_similarity       wer  \
0            0.338710      1.359155  0.216514           0.612094  1.054054
1            0.178082      1.442029  0.023705           0.413481  1.394737
2            0.164384      1.537500  0.018897           0.303488  1.500000
3            0.148649      1.261438  0.024377           0.390257  1.250000
4            0.219178      1.369403  0.084801           0.403823  1.119048

   char_diversity  type_token_ratio  bigram_overlap
0        0.107143          0.924528        0.173333
1        0.038462          0.896552        0.032967
2        0.242424          0.981818        0.011628
3        0.068966          0.945455        0.033333
4        0.193548          0.980769        0.082353

[5 rows x 22 columns]
```

[2]:

```
# BAR PLOT
df['total_duration_s'] = df['total_duration_ns'] / 1e9
agg = df.groupby(['augmentation_type', 'model'])['total_duration_s'].mean().
 ↪reset_index()
augmentation_order = sorted(df['augmentation_type'].unique())

plt.figure(figsize=(12,7))
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
```

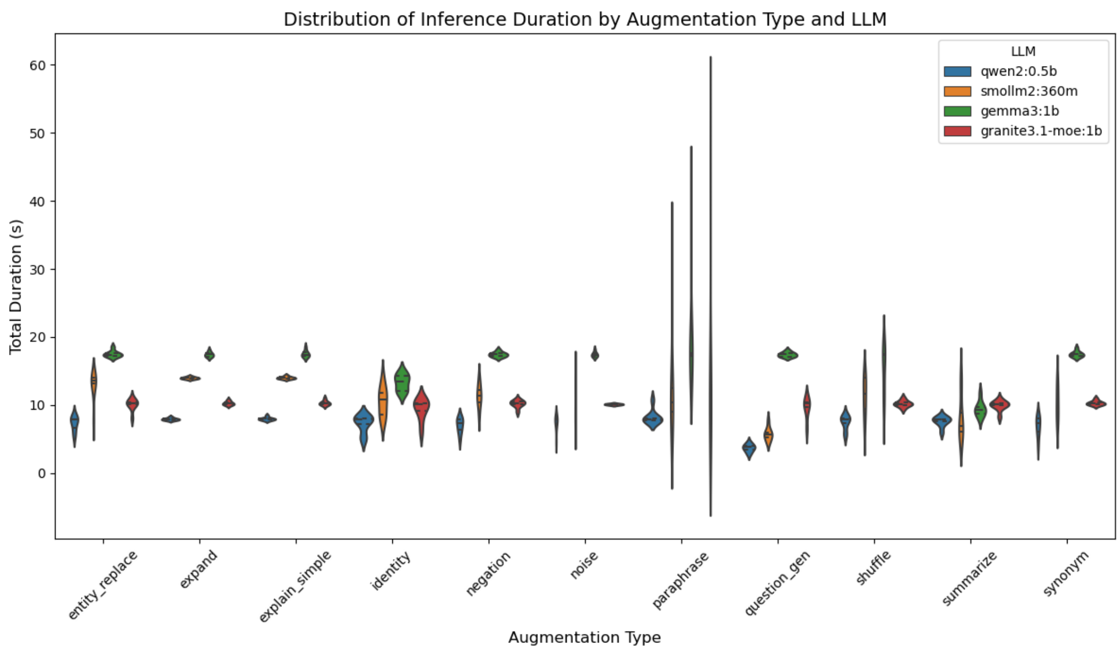
```
    y='total_duration_s',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Total Duration (s)", fontsize=12)
plt.title("Mean Inference Duration by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.ylim(2.5,20.5)
plt.show()

print("Mean Total Duration (s) Table:\n")
print(agg.to_string(index=False))

# VIOLIN PLOT
plt.figure(figsize=(12,7))
violin = sns.violinplot(
    data=df,
    x='augmentation_type',
    y='total_duration_s',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"  # Remove split=True!
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Total Duration (s)", fontsize=12)
plt.title("Distribution of Inference Duration by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Total Duration (s) Table:

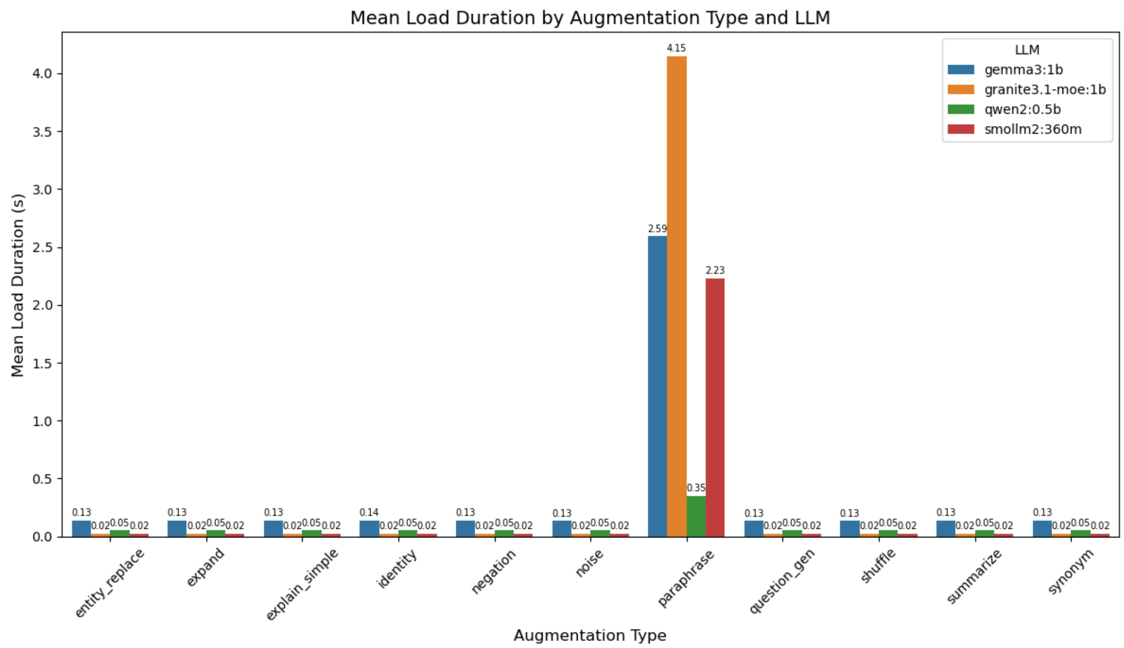
augmentation_type	model	total_duration_s
entity_replace	gemma3:1b	17.519171
entity_replace	granite3.1-moe:1b	10.087887
entity_replace	qwen2:0.5b	7.306076
entity_replace	smollm2:360m	12.848805
expand	gemma3:1b	17.434952
expand	granite3.1-moe:1b	10.289341
expand	qwen2:0.5b	7.936603
expand	smollm2:360m	13.985258
explain_simple	gemma3:1b	17.521006
explain_simple	granite3.1-moe:1b	10.324334
explain_simple	qwen2:0.5b	7.984643
explain_simple	smollm2:360m	14.012370
identity	gemma3:1b	13.313809
identity	granite3.1-moe:1b	9.422094
identity	qwen2:0.5b	7.275286
identity	smollm2:360m	10.487829
negation	gemma3:1b	17.443342
negation	granite3.1-moe:1b	10.205199
negation	qwen2:0.5b	7.019680
negation	smollm2:360m	11.375481
noise	gemma3:1b	17.448303
noise	granite3.1-moe:1b	10.109262
noise	qwen2:0.5b	7.336289

```
       noise       smollm2:360m          10.640014
  paraphrase          gemma3:1b          19.976331
  paraphrase granite3.1-moe:1b          14.560397
  paraphrase         qwen2:0.5b           8.228610
  paraphrase       smollm2:360m          12.661435
question_gen          gemma3:1b          17.435018
question_gen granite3.1-moe:1b           9.610098
question_gen         qwen2:0.5b           3.707023
question_gen       smollm2:360m           5.754444
     shuffle          gemma3:1b          15.157816
     shuffle granite3.1-moe:1b          10.209583
     shuffle         qwen2:0.5b           7.451169
     shuffle       smollm2:360m          11.327010
   summarize          gemma3:1b           9.429700
   summarize granite3.1-moe:1b           9.956269
   summarize         qwen2:0.5b           7.627685
   summarize       smollm2:360m           8.226404
     synonym          gemma3:1b          17.496148
     synonym granite3.1-moe:1b          10.318432
     synonym         qwen2:0.5b           6.892995
     synonym       smollm2:360m          10.327454
```



[3]:
```
# BAR PLOT
df['load_duration_s'] = df['load_duration_ns'] / 1e9
agg = df.groupby(['augmentation_type', 'model'])['load_duration_s'].mean().
 ↪reset_index()
```

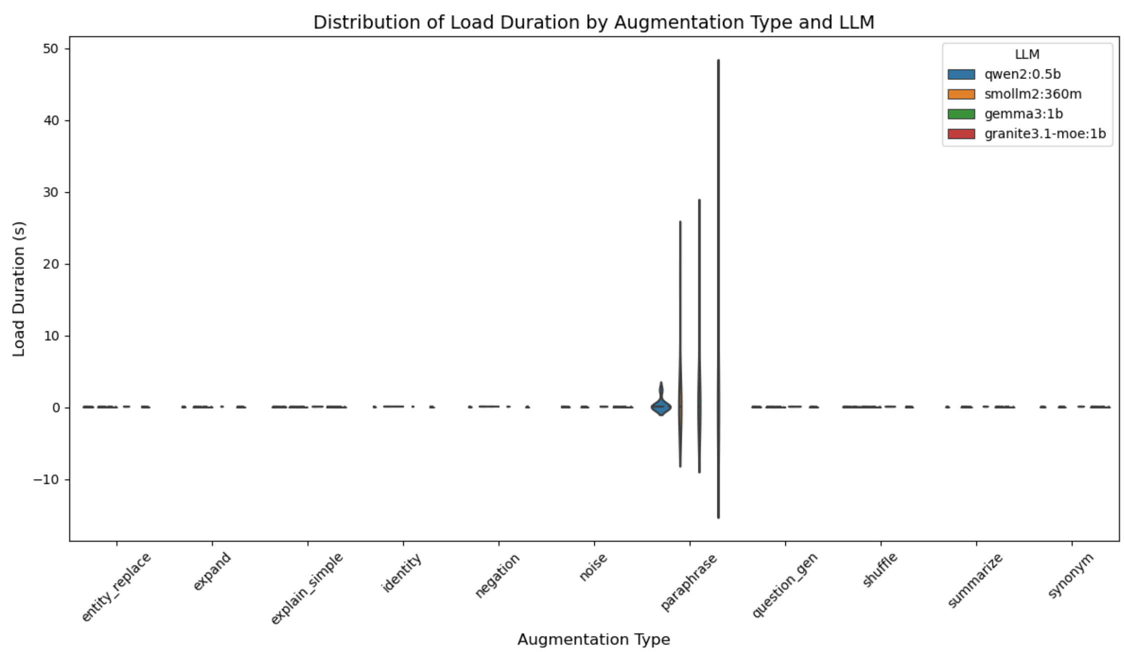
```
plt.figure(figsize=(12,7))
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='load_duration_s',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Load Duration (s)", fontsize=12)
plt.title("Mean Load Duration by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Load Duration (s) Table:\n")
print(agg.to_string(index=False))

# VIOLIN PLOT
plt.figure(figsize=(12,7))
violin = sns.violinplot(
    data=df,
    x='augmentation_type',
    y='load_duration_s',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Load Duration (s)", fontsize=12)
plt.title("Distribution of Load Duration by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Load Duration (s) Table:

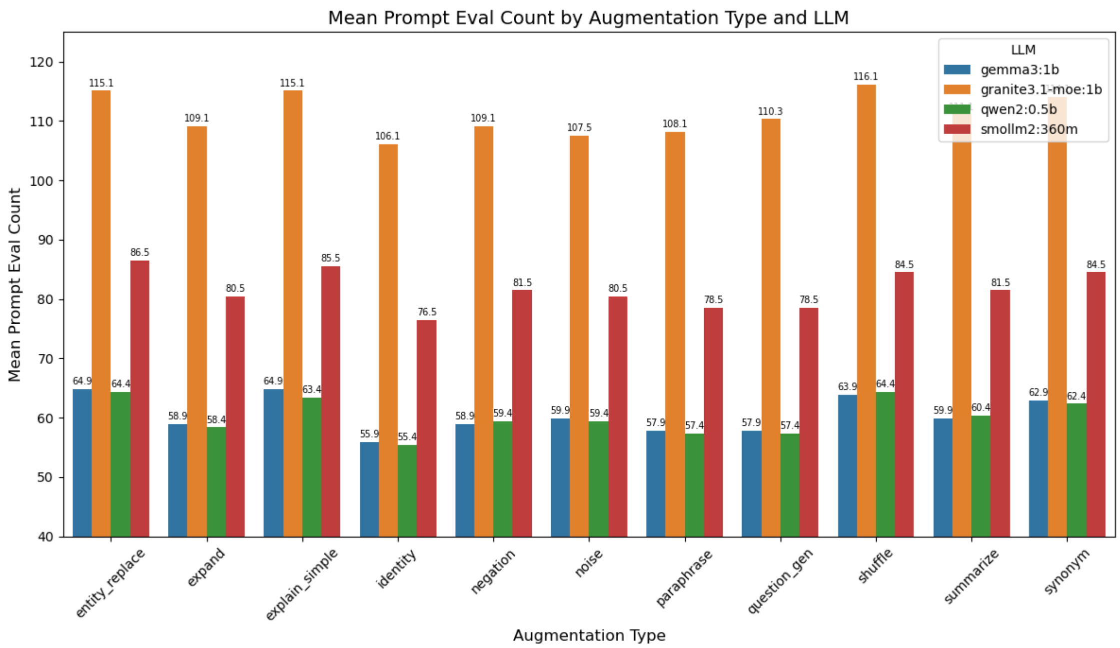
augmentation_type	model	load_duration_s
entity_replace	gemma3:1b	0.134435
entity_replace	granite3.1-moe:1b	0.022135
entity_replace	qwen2:0.5b	0.051694
entity_replace	smollm2:360m	0.020649
expand	gemma3:1b	0.134711
expand	granite3.1-moe:1b	0.020546
expand	qwen2:0.5b	0.050972
expand	smollm2:360m	0.020424
explain_simple	gemma3:1b	0.134660
explain_simple	granite3.1-moe:1b	0.021282
explain_simple	qwen2:0.5b	0.051691
explain_simple	smollm2:360m	0.021615
identity	gemma3:1b	0.135201
identity	granite3.1-moe:1b	0.021608
identity	qwen2:0.5b	0.051934
identity	smollm2:360m	0.020424
negation	gemma3:1b	0.134433
negation	granite3.1-moe:1b	0.021027
negation	qwen2:0.5b	0.052432
negation	smollm2:360m	0.020509
noise	gemma3:1b	0.133162
noise	granite3.1-moe:1b	0.020497
noise	qwen2:0.5b	0.051306

```
       noise      smollm2:360m         0.022082
  paraphrase          gemma3:1b         2.591088
  paraphrase granite3.1-moe:1b         4.148439
  paraphrase         qwen2:0.5b         0.348289
  paraphrase      smollm2:360m         2.230602
question_gen          gemma3:1b         0.133863
question_gen granite3.1-moe:1b         0.021682
question_gen         qwen2:0.5b         0.052184
question_gen      smollm2:360m         0.020965
     shuffle          gemma3:1b         0.132988
     shuffle granite3.1-moe:1b         0.021605
     shuffle         qwen2:0.5b         0.051072
     shuffle      smollm2:360m         0.021318
   summarize          gemma3:1b         0.134930
   summarize granite3.1-moe:1b         0.020477
   summarize         qwen2:0.5b         0.052047
   summarize      smollm2:360m         0.021119
     synonym          gemma3:1b         0.134818
     synonym granite3.1-moe:1b         0.021068
     synonym         qwen2:0.5b         0.052437
     synonym      smollm2:360m         0.022486
```



[4]:
```python
# BAR PLOT
agg = df.groupby(['augmentation_type', 'model'])['prompt_eval_count'].mean().
 ↪reset_index()
```

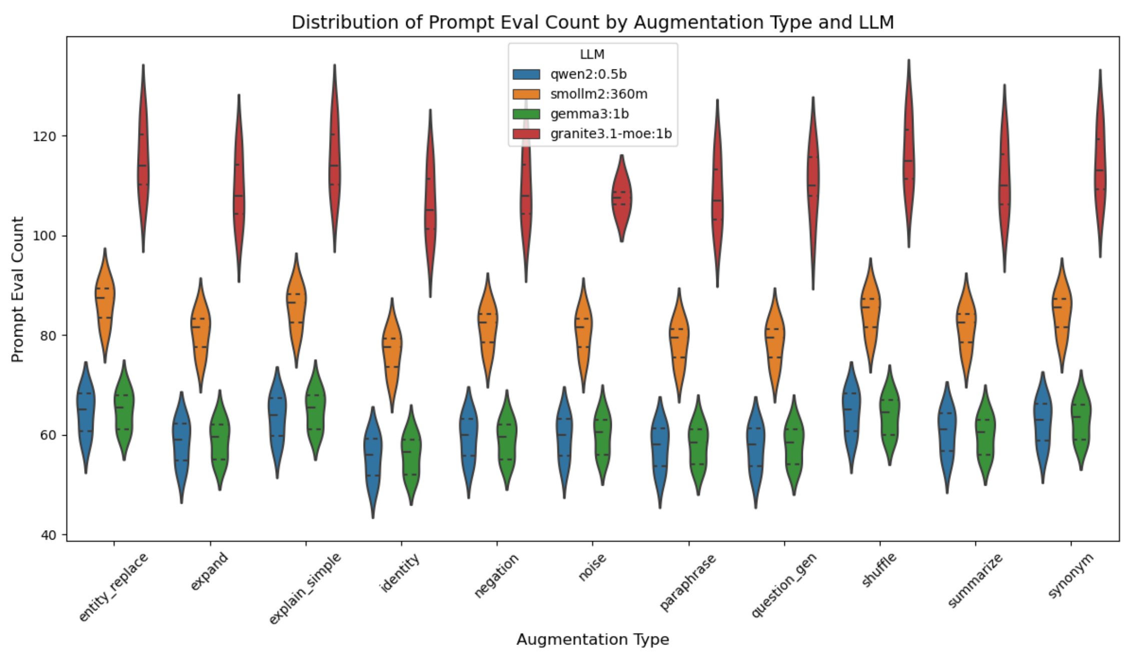
```
plt.figure(figsize=(12,7))
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='prompt_eval_count',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.1f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Prompt Eval Count", fontsize=12)
plt.title("Mean Prompt Eval Count by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.ylim(40,125)
plt.show()

print("Mean Prompt Eval Count Table:\n")
print(agg.to_string(index=False))

# VIOLIN PLOT
plt.figure(figsize=(12,7))
violin = sns.violinplot(
    data=df,
    x='augmentation_type',
    y='prompt_eval_count',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Prompt Eval Count", fontsize=12)
plt.title("Distribution of Prompt Eval Count by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Prompt Eval Count Table:

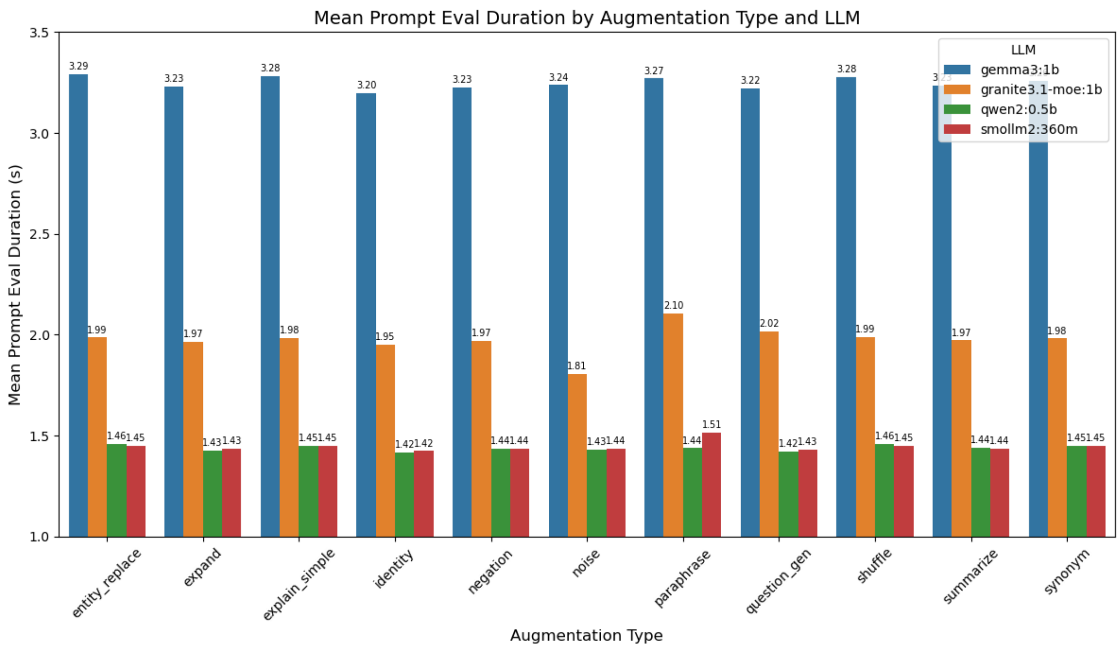
augmentation_type	model	prompt_eval_count
entity_replace	gemma3:1b	64.875000
entity_replace	granite3.1-moe:1b	115.125000
entity_replace	qwen2:0.5b	64.375000
entity_replace	smollm2:360m	86.500000
expand	gemma3:1b	58.875000
expand	granite3.1-moe:1b	109.125000
expand	qwen2:0.5b	58.375000
expand	smollm2:360m	80.500000
explain_simple	gemma3:1b	64.875000
explain_simple	granite3.1-moe:1b	115.125000
explain_simple	qwen2:0.5b	63.375000
explain_simple	smollm2:360m	85.500000
identity	gemma3:1b	55.875000
identity	granite3.1-moe:1b	106.125000
identity	qwen2:0.5b	55.375000
identity	smollm2:360m	76.500000
negation	gemma3:1b	58.875000
negation	granite3.1-moe:1b	109.125000
negation	qwen2:0.5b	59.375000
negation	smollm2:360m	81.500000
noise	gemma3:1b	59.875000
noise	granite3.1-moe:1b	107.500000
noise	qwen2:0.5b	59.375000

```
         noise       smollm2:360m          80.500000
    paraphrase          gemma3:1b          57.875000
    paraphrase granite3.1-moe:1b          108.125000
    paraphrase         qwen2:0.5b          57.375000
    paraphrase       smollm2:360m          78.500000
  question_gen          gemma3:1b          57.875000
  question_gen granite3.1-moe:1b          110.333333
  question_gen         qwen2:0.5b          57.375000
  question_gen       smollm2:360m          78.500000
       shuffle          gemma3:1b          63.875000
       shuffle granite3.1-moe:1b          116.125000
       shuffle         qwen2:0.5b          64.375000
       shuffle       smollm2:360m          84.500000
     summarize          gemma3:1b          59.875000
     summarize granite3.1-moe:1b          111.125000
     summarize         qwen2:0.5b          60.375000
     summarize       smollm2:360m          81.500000
       synonym          gemma3:1b          62.875000
       synonym granite3.1-moe:1b          114.125000
       synonym         qwen2:0.5b          62.375000
       synonym       smollm2:360m          84.500000
```



[5]:
```
# BAR PLOT
df['prompt_eval_duration_s'] = df['prompt_eval_duration_ns'] / 1e9
agg = df.groupby(['augmentation_type', 'model'])['prompt_eval_duration_s'].
 ↪mean().reset_index()
```

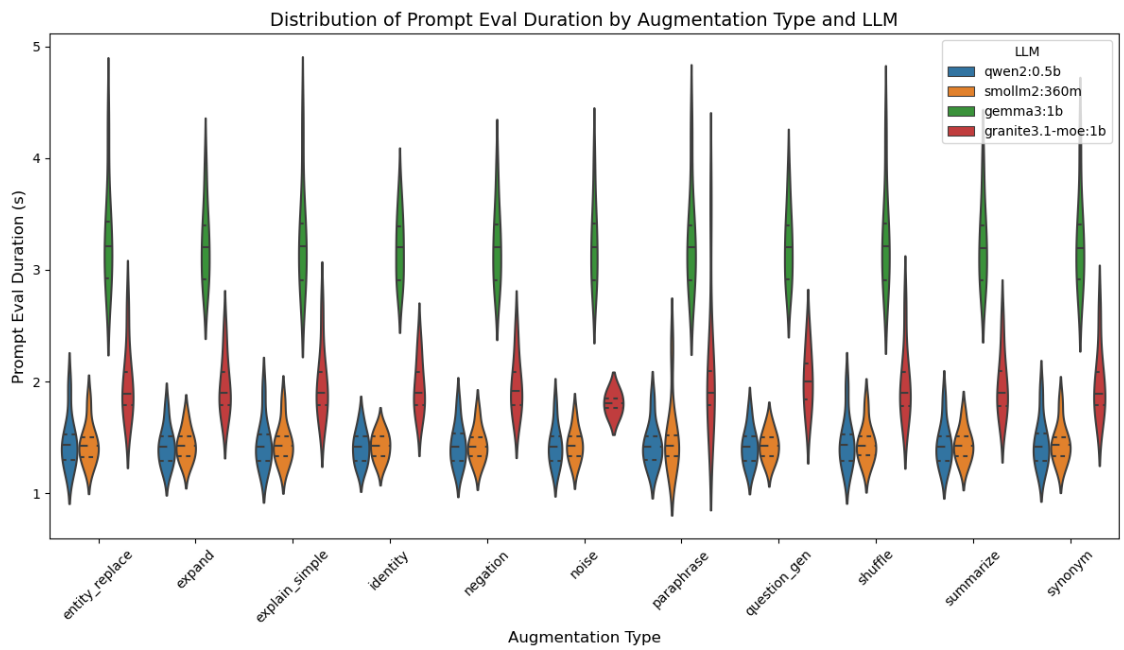
```
plt.figure(figsize=(12,7))
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='prompt_eval_duration_s',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Prompt Eval Duration (s)", fontsize=12)
plt.title("Mean Prompt Eval Duration by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.ylim(1,3.5)
plt.show()

print("Mean Prompt Eval Duration (s) Table:\n")
print(agg.to_string(index=False))

# VIOLIN PLOT
plt.figure(figsize=(12,7))
violin = sns.violinplot(
    data=df,
    x='augmentation_type',
    y='prompt_eval_duration_s',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Prompt Eval Duration (s)", fontsize=12)
plt.title("Distribution of Prompt Eval Duration by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Prompt Eval Duration (s) Table:

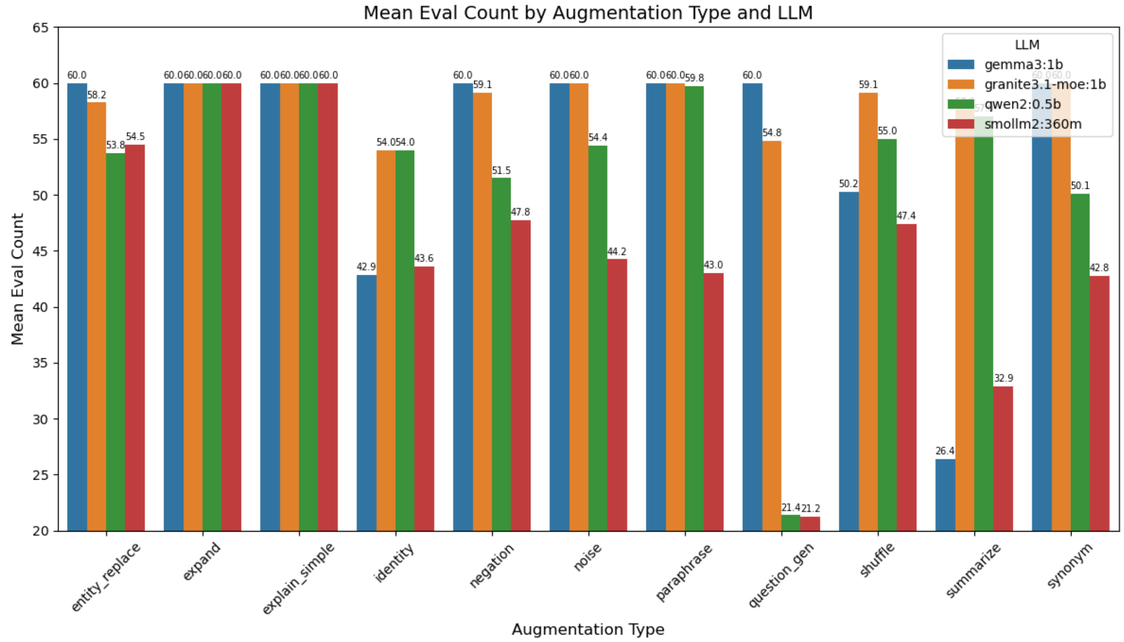
augmentation_type	model	prompt_eval_duration_s
entity_replace	gemma3:1b	3.289920
entity_replace	granite3.1-moe:1b	1.986179
entity_replace	qwen2:0.5b	1.458739
entity_replace	smollm2:360m	1.448993
expand	gemma3:1b	3.230149
expand	granite3.1-moe:1b	1.965222
expand	qwen2:0.5b	1.425312
expand	smollm2:360m	1.434685
explain_simple	gemma3:1b	3.282262
explain_simple	granite3.1-moe:1b	1.984986
explain_simple	qwen2:0.5b	1.450086
explain_simple	smollm2:360m	1.450368
identity	gemma3:1b	3.198626
identity	granite3.1-moe:1b	1.952127
identity	qwen2:0.5b	1.415143
identity	smollm2:360m	1.423411
negation	gemma3:1b	3.226121
negation	granite3.1-moe:1b	1.969167
negation	qwen2:0.5b	1.436152
negation	smollm2:360m	1.435887
noise	gemma3:1b	3.237751
noise	granite3.1-moe:1b	1.806109
noise	qwen2:0.5b	1.429667

```
       noise      smollm2:360m            1.435769
  paraphrase         gemma3:1b            3.269960
  paraphrase granite3.1-moe:1b            2.104556
  paraphrase        qwen2:0.5b            1.437472
  paraphrase      smollm2:360m            1.514472
question_gen         gemma3:1b            3.219902
question_gen granite3.1-moe:1b            2.017061
question_gen        qwen2:0.5b            1.422472
question_gen      smollm2:360m            1.428817
     shuffle         gemma3:1b            3.277141
     shuffle granite3.1-moe:1b            1.988787
     shuffle        qwen2:0.5b            1.459124
     shuffle      smollm2:360m            1.448311
   summarize         gemma3:1b            3.234136
   summarize granite3.1-moe:1b            1.971836
   summarize        qwen2:0.5b            1.438362
   summarize      smollm2:360m            1.436426
     synonym         gemma3:1b            3.260029
     synonym granite3.1-moe:1b            1.981388
     synonym        qwen2:0.5b            1.450563
     synonym      smollm2:360m            1.449356
```



[6]:
```
# BAR PLOT
agg = df.groupby(['augmentation_type', 'model'])['eval_count'].mean().
 ↪reset_index()
```

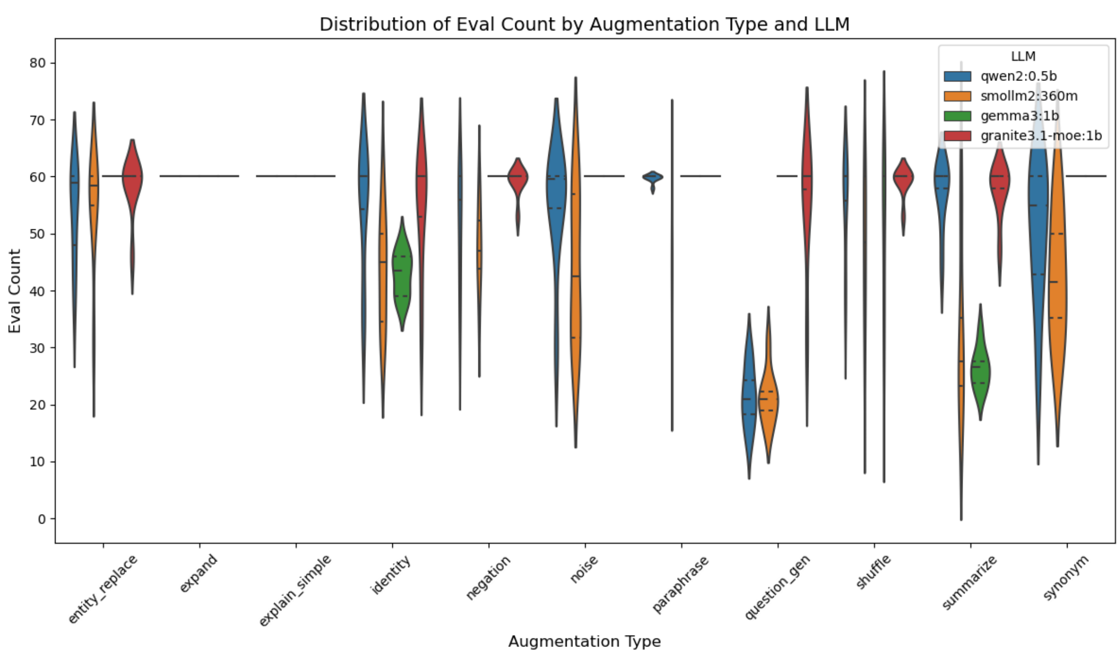
```
plt.figure(figsize=(12,7))
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='eval_count',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.1f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Eval Count", fontsize=12)
plt.title("Mean Eval Count by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.ylim(20,65)
plt.show()

print("Mean Eval Count Table:\n")
print(agg.to_string(index=False))

# VIOLIN PLOT
plt.figure(figsize=(12,7))
violin = sns.violinplot(
    data=df,
    x='augmentation_type',
    y='eval_count',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Eval Count", fontsize=12)
plt.title("Distribution of Eval Count by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Eval Count Table:

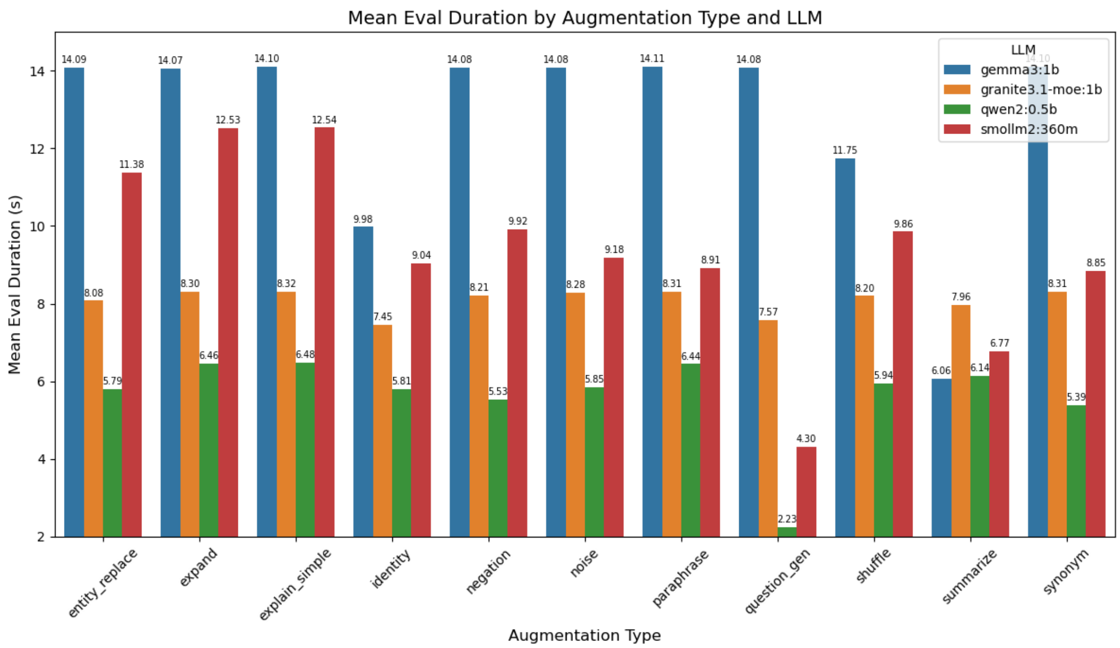
augmentation_type	model	eval_count
entity_replace	gemma3:1b	60.000000
entity_replace	granite3.1-moe:1b	58.250000
entity_replace	qwen2:0.5b	53.750000
entity_replace	smollm2:360m	54.500000
expand	gemma3:1b	60.000000
expand	granite3.1-moe:1b	60.000000
expand	qwen2:0.5b	60.000000
expand	smollm2:360m	60.000000
explain_simple	gemma3:1b	60.000000
explain_simple	granite3.1-moe:1b	60.000000
explain_simple	qwen2:0.5b	60.000000
explain_simple	smollm2:360m	60.000000
identity	gemma3:1b	42.875000
identity	granite3.1-moe:1b	54.000000
identity	qwen2:0.5b	54.000000
identity	smollm2:360m	43.625000
negation	gemma3:1b	60.000000
negation	granite3.1-moe:1b	59.125000
negation	qwen2:0.5b	51.500000
negation	smollm2:360m	47.750000
noise	gemma3:1b	60.000000
noise	granite3.1-moe:1b	60.000000
noise	qwen2:0.5b	54.375000

```
        noise       smollm2:360m   44.250000
   paraphrase          gemma3:1b   60.000000
   paraphrase granite3.1-moe:1b   60.000000
   paraphrase         qwen2:0.5b   59.750000
   paraphrase       smollm2:360m   43.000000
 question_gen          gemma3:1b   60.000000
 question_gen granite3.1-moe:1b   54.833333
 question_gen         qwen2:0.5b   21.375000
 question_gen       smollm2:360m   21.250000
      shuffle          gemma3:1b   50.250000
      shuffle granite3.1-moe:1b   59.125000
      shuffle         qwen2:0.5b   55.000000
      shuffle       smollm2:360m   47.375000
    summarize          gemma3:1b   26.375000
    summarize granite3.1-moe:1b   57.625000
    summarize         qwen2:0.5b   57.000000
    summarize       smollm2:360m   32.875000
      synonym          gemma3:1b   60.000000
      synonym granite3.1-moe:1b   60.000000
      synonym         qwen2:0.5b   50.125000
      synonym       smollm2:360m   42.750000
```



[7]:
```python
# BAR PLOT
df['eval_duration_s'] = df['eval_duration_ns'] / 1e9
agg = df.groupby(['augmentation_type', 'model'])['eval_duration_s'].mean().
 ↪reset_index()
```

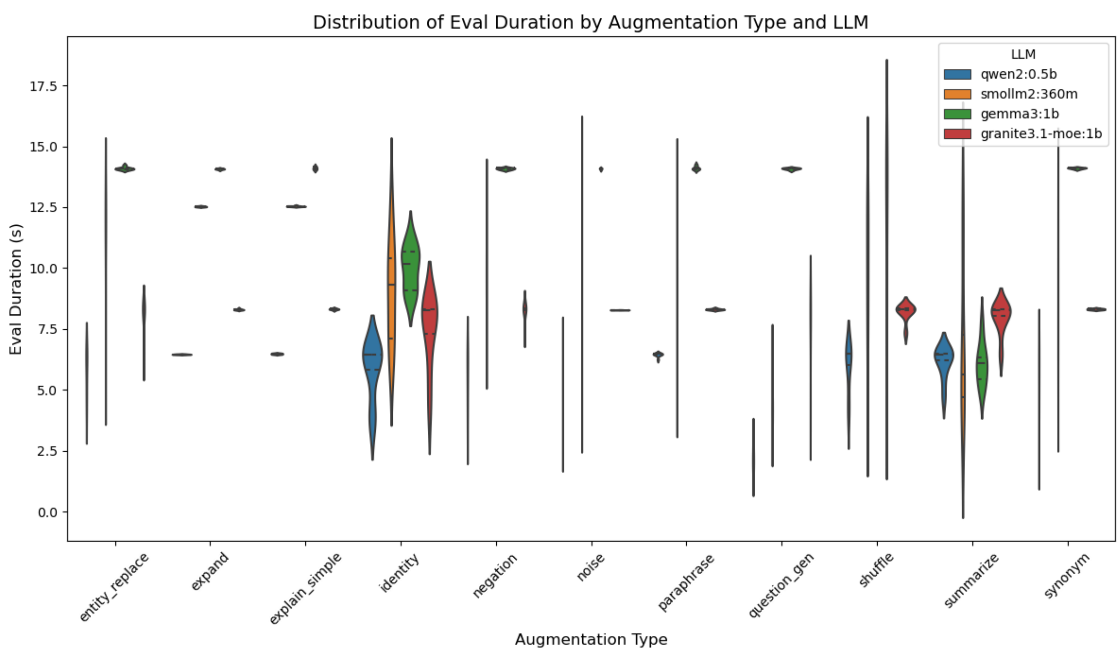
```
plt.figure(figsize=(12,7))
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='eval_duration_s',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Eval Duration (s)", fontsize=12)
plt.title("Mean Eval Duration by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.ylim(2,15)
plt.show()

print("Mean Eval Duration (s) Table:\n")
print(agg.to_string(index=False))

# VIOLIN PLOT
plt.figure(figsize=(12,7))
violin = sns.violinplot(
    data=df,
    x='augmentation_type',
    y='eval_duration_s',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Eval Duration (s)", fontsize=12)
plt.title("Distribution of Eval Duration by Augmentation Type and LLM",␣
↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Eval Duration (s) Table:

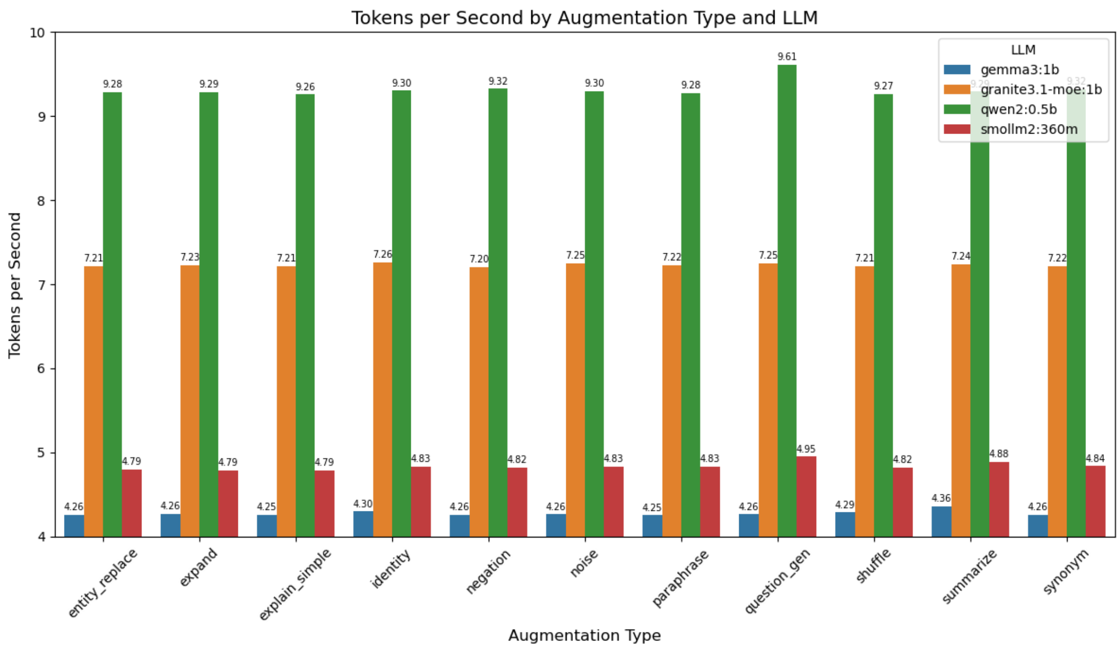
augmentation_type	model	eval_duration_s
entity_replace	gemma3:1b	14.093728
entity_replace	granite3.1-moe:1b	8.077532
entity_replace	qwen2:0.5b	5.794070
entity_replace	smollm2:360m	11.377713
expand	gemma3:1b	14.069005
expand	granite3.1-moe:1b	8.301877
expand	qwen2:0.5b	6.459140
expand	smollm2:360m	12.528693
explain_simple	gemma3:1b	14.102958
explain_simple	granite3.1-moe:1b	8.316195
explain_simple	qwen2:0.5b	6.481418
explain_simple	smollm2:360m	12.538618
identity	gemma3:1b	9.978718
identity	granite3.1-moe:1b	7.446409
identity	qwen2:0.5b	5.806615
identity	smollm2:360m	9.042256
negation	gemma3:1b	14.081699
negation	granite3.1-moe:1b	8.212668
negation	qwen2:0.5b	5.529797
negation	smollm2:360m	9.917460
noise	gemma3:1b	14.076263
noise	granite3.1-moe:1b	8.280945
noise	qwen2:0.5b	5.853783

```
       noise       smollm2:360m          9.180232
  paraphrase           gemma3:1b         14.113944
  paraphrase granite3.1-moe:1b          8.305655
  paraphrase          qwen2:0.5b          6.441398
  paraphrase       smollm2:360m          8.914208
question_gen           gemma3:1b         14.080092
question_gen granite3.1-moe:1b          7.569508
question_gen          qwen2:0.5b          2.231112
question_gen       smollm2:360m          4.302042
     shuffle           gemma3:1b         11.746517
     shuffle granite3.1-moe:1b          8.197225
     shuffle          qwen2:0.5b          5.939719
     shuffle       smollm2:360m          9.855855
   summarize           gemma3:1b          6.059394
   summarize granite3.1-moe:1b          7.962053
   summarize          qwen2:0.5b          6.136115
   summarize       smollm2:360m          6.767067
     synonym           gemma3:1b         14.100174
     synonym granite3.1-moe:1b          8.314038
     synonym          qwen2:0.5b          5.388443
     synonym       smollm2:360m          8.853510
```



[8]:
```python
# BAR PLOT
agg = df.groupby(['augmentation_type', 'model'])['tokens_per_second'].mean().
 ↪reset_index()
```

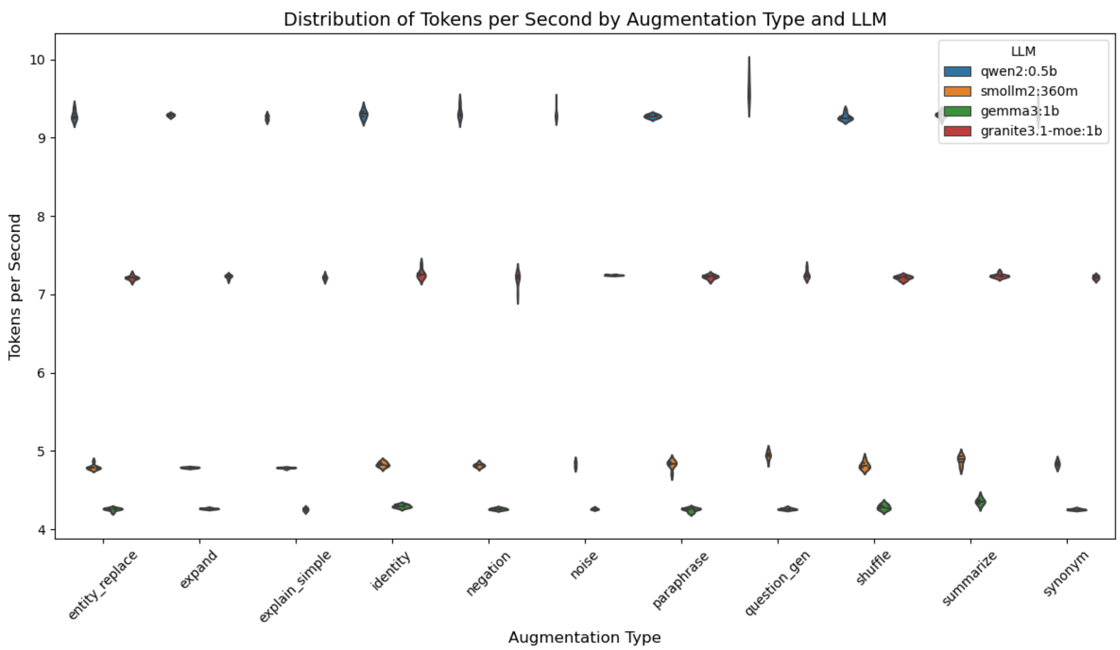
```
plt.figure(figsize=(12,7))
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='tokens_per_second',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Tokens per Second", fontsize=12)
plt.title("Tokens per Second by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.ylim(4,10)
plt.show()

print("Tokens per Second Table:\n")
print(agg.to_string(index=False))

# VIOLIN PLOT
plt.figure(figsize=(12,7))
violin = sns.violinplot(
    data=df,
    x='augmentation_type',
    y='tokens_per_second',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Tokens per Second", fontsize=12)
plt.title("Distribution of Tokens per Second by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Tokens per Second Table:

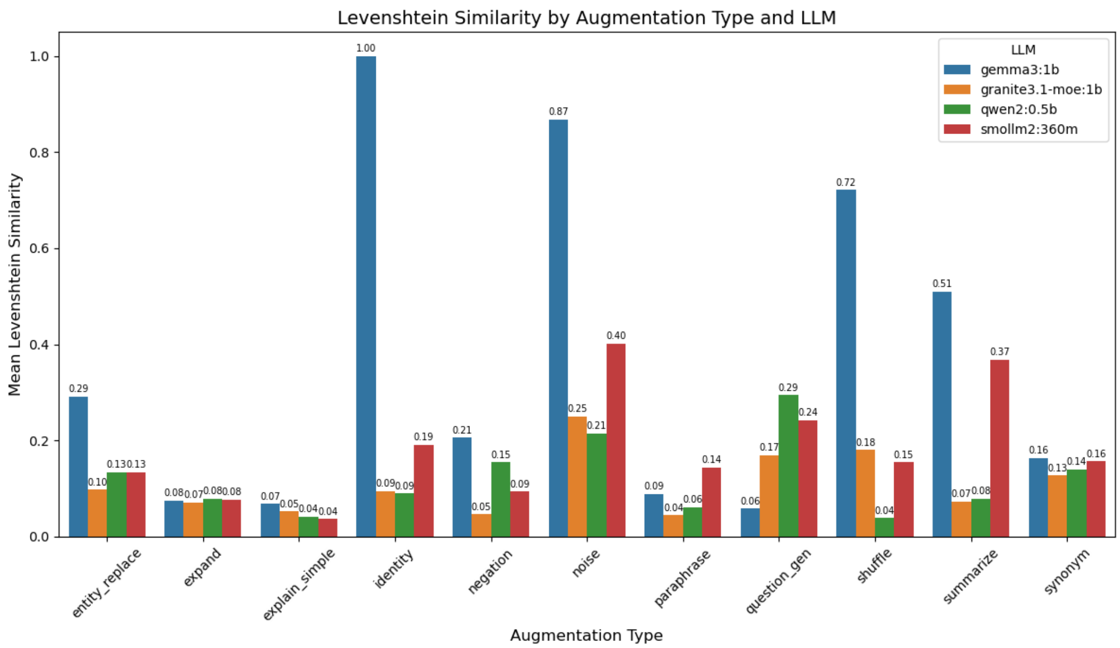
augmentation_type	model	tokens_per_second
entity_replace	gemma3:1b	4.257286
entity_replace	granite3.1-moe:1b	7.212979
entity_replace	qwen2:0.5b	9.283693
entity_replace	smollm2:360m	4.794924
expand	gemma3:1b	4.264705
expand	granite3.1-moe:1b	7.227345
expand	qwen2:0.5b	9.289185
expand	smollm2:360m	4.789015
explain_simple	gemma3:1b	4.254498
explain_simple	granite3.1-moe:1b	7.214922
explain_simple	qwen2:0.5b	9.257323
explain_simple	smollm2:360m	4.785226
identity	gemma3:1b	4.297115
identity	granite3.1-moe:1b	7.261247
identity	qwen2:0.5b	9.301166
identity	smollm2:360m	4.830443
negation	gemma3:1b	4.260880
negation	granite3.1-moe:1b	7.201331
negation	qwen2:0.5b	9.324690
negation	smollm2:360m	4.817815
noise	gemma3:1b	4.262515
noise	granite3.1-moe:1b	7.245551
noise	qwen2:0.5b	9.299473

```
       noise      smollm2:360m          4.829795
  paraphrase          gemma3:1b          4.251231
  paraphrase granite3.1-moe:1b          7.224071
  paraphrase         qwen2:0.5b          9.276019
  paraphrase      smollm2:360m          4.830580
question_gen          gemma3:1b          4.261365
question_gen granite3.1-moe:1b          7.252370
question_gen         qwen2:0.5b          9.610416
question_gen      smollm2:360m          4.948760
     shuffle          gemma3:1b          4.285316
     shuffle granite3.1-moe:1b          7.212863
     shuffle         qwen2:0.5b          9.265605
     shuffle      smollm2:360m          4.817816
   summarize          gemma3:1b          4.356848
   summarize granite3.1-moe:1b          7.238769
   summarize         qwen2:0.5b          9.291481
   summarize      smollm2:360m          4.882819
     synonym          gemma3:1b          4.255283
     synonym granite3.1-moe:1b          7.216768
     synonym         qwen2:0.5b          9.320334
     synonym      smollm2:360m          4.836850
```



[9]:
```python
agg = df.groupby(['augmentation_type', 'model'])['levenshtein_similarity'].
 ↪mean().reset_index()
augmentation_order = sorted(df['augmentation_type'].unique())
```

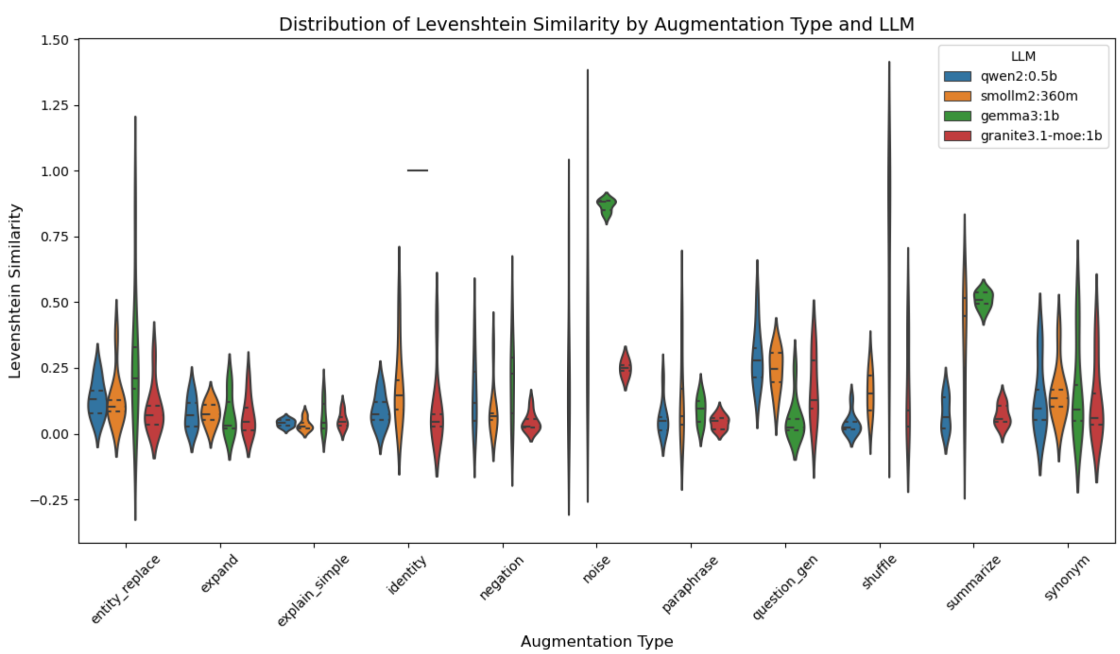
```
plt.figure(figsize=(12,7))
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='levenshtein_similarity',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Levenshtein Similarity", fontsize=12)
plt.title("Levenshtein Similarity by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Levenshtein Similarity Table:\n")
print(agg.to_string(index=False))

# VIOLIN PLOT
plt.figure(figsize=(12,7))
sns.violinplot(
    data=df,
    x='augmentation_type',
    y='levenshtein_similarity',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Levenshtein Similarity", fontsize=12)
plt.title("Distribution of Levenshtein Similarity by Augmentation Type and␣
 ↪LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Levenshtein Similarity Table:

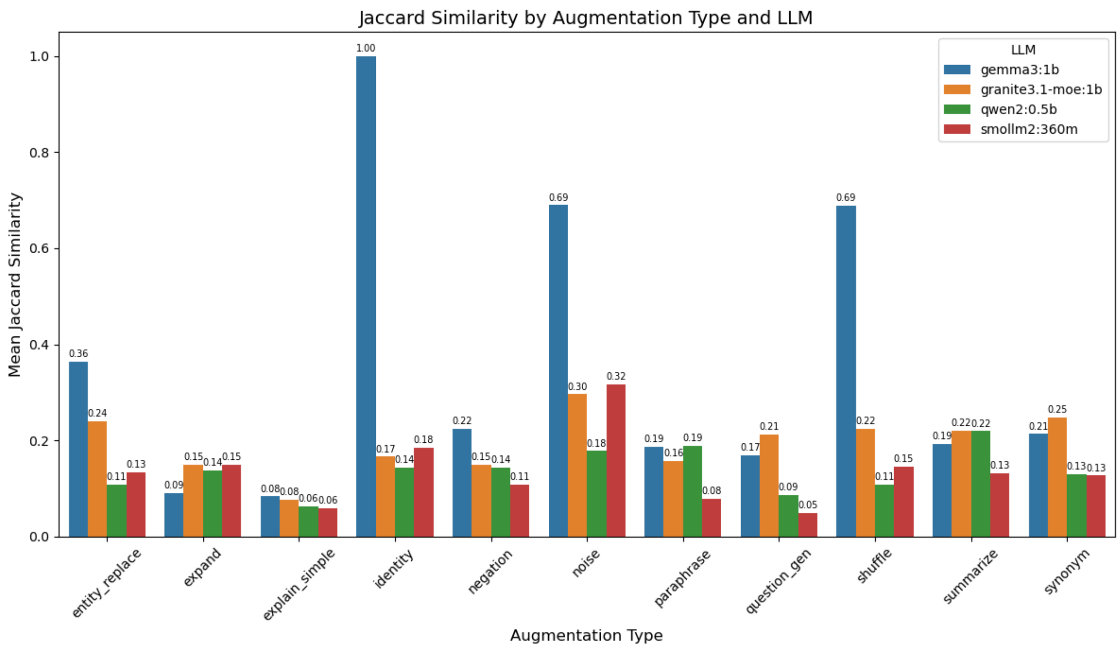
augmentation_type	model	levenshtein_similarity
entity_replace	gemma3:1b	0.290348
entity_replace	granite3.1-moe:1b	0.097073
entity_replace	qwen2:0.5b	0.132908
entity_replace	smollm2:360m	0.132863
expand	gemma3:1b	0.075134
expand	granite3.1-moe:1b	0.069932
expand	qwen2:0.5b	0.078037
expand	smollm2:360m	0.077145
explain_simple	gemma3:1b	0.067784
explain_simple	granite3.1-moe:1b	0.051873
explain_simple	qwen2:0.5b	0.040759
explain_simple	smollm2:360m	0.036005
identity	gemma3:1b	1.000000
identity	granite3.1-moe:1b	0.094511
identity	qwen2:0.5b	0.089047
identity	smollm2:360m	0.190436
negation	gemma3:1b	0.205183
negation	granite3.1-moe:1b	0.045812
negation	qwen2:0.5b	0.154084
negation	smollm2:360m	0.093264
noise	gemma3:1b	0.868107
noise	granite3.1-moe:1b	0.249687
noise	qwen2:0.5b	0.214436

noise	smollm2:360m	0.400869
paraphrase	gemma3:1b	0.088593
paraphrase	granite3.1-moe:1b	0.044057
paraphrase	qwen2:0.5b	0.060148
paraphrase	smollm2:360m	0.144007
question_gen	gemma3:1b	0.057572
question_gen	granite3.1-moe:1b	0.168588
question_gen	qwen2:0.5b	0.293763
question_gen	smollm2:360m	0.241747
shuffle	gemma3:1b	0.721207
shuffle	granite3.1-moe:1b	0.179992
shuffle	qwen2:0.5b	0.038025
shuffle	smollm2:360m	0.154309
summarize	gemma3:1b	0.509782
summarize	granite3.1-moe:1b	0.071765
summarize	qwen2:0.5b	0.078576
summarize	smollm2:360m	0.367185
synonym	gemma3:1b	0.163364
synonym	granite3.1-moe:1b	0.126834
synonym	qwen2:0.5b	0.138962
synonym	smollm2:360m	0.156136



[10]:
```
agg = df.groupby(['augmentation_type', 'model'])['jaccard_similarity'].mean().
  ↪reset_index()

plt.figure(figsize=(12,7))
```

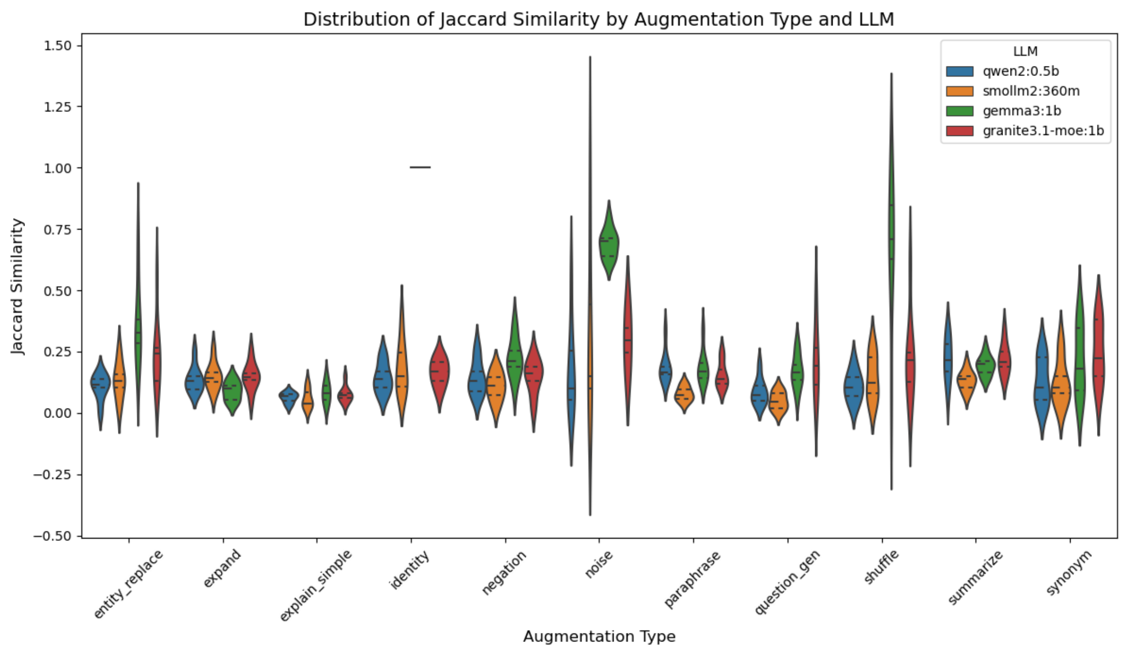
```
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='jaccard_similarity',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Jaccard Similarity", fontsize=12)
plt.title("Jaccard Similarity by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Jaccard Similarity Table:\n")
print(agg.to_string(index=False))


plt.figure(figsize=(12,7))
sns.violinplot(
    data=df,
    x='augmentation_type',
    y='jaccard_similarity',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Jaccard Similarity", fontsize=12)
plt.title("Distribution of Jaccard Similarity by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Jaccard Similarity Table:

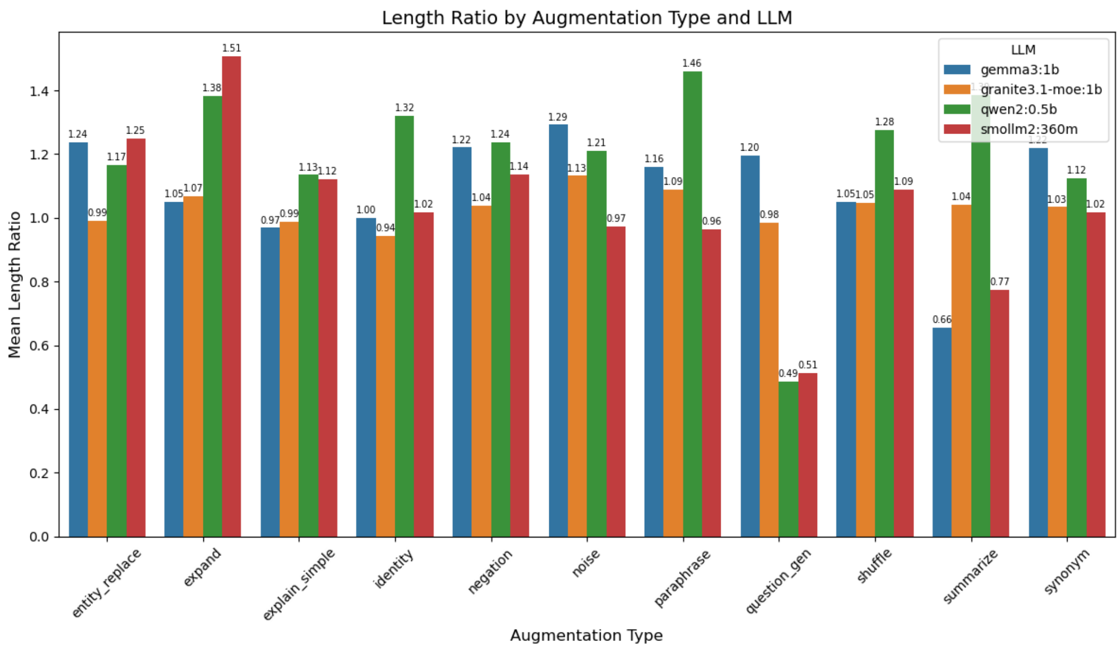
augmentation_type	model	jaccard_similarity
entity_replace	gemma3:1b	0.364388
entity_replace	granite3.1-moe:1b	0.239889
entity_replace	qwen2:0.5b	0.107304
entity_replace	smollm2:360m	0.132787
expand	gemma3:1b	0.090450
expand	granite3.1-moe:1b	0.148763
expand	qwen2:0.5b	0.136878
expand	smollm2:360m	0.149658
explain_simple	gemma3:1b	0.083227
explain_simple	granite3.1-moe:1b	0.076358
explain_simple	qwen2:0.5b	0.062661
explain_simple	smollm2:360m	0.058572
identity	gemma3:1b	1.000000
identity	granite3.1-moe:1b	0.166029
identity	qwen2:0.5b	0.143552
identity	smollm2:360m	0.184490
negation	gemma3:1b	0.223519
negation	granite3.1-moe:1b	0.149267
negation	qwen2:0.5b	0.143467
negation	smollm2:360m	0.108570
noise	gemma3:1b	0.689598
noise	granite3.1-moe:1b	0.295714
noise	qwen2:0.5b	0.177881

```
        noise       smollm2:360m           0.316545
   paraphrase          gemma3:1b           0.186556
   paraphrase granite3.1-moe:1b            0.156475
   paraphrase         qwen2:0.5b           0.188939
   paraphrase       smollm2:360m           0.078176
 question_gen          gemma3:1b           0.169422
 question_gen granite3.1-moe:1b            0.211565
 question_gen         qwen2:0.5b           0.086832
 question_gen       smollm2:360m           0.049405
      shuffle          gemma3:1b           0.688624
      shuffle granite3.1-moe:1b            0.224040
      shuffle         qwen2:0.5b           0.108495
      shuffle       smollm2:360m           0.146016
    summarize          gemma3:1b           0.191970
    summarize granite3.1-moe:1b            0.219977
    summarize         qwen2:0.5b           0.219177
    summarize       smollm2:360m           0.131202
      synonym          gemma3:1b           0.213119
      synonym granite3.1-moe:1b            0.247364
      synonym         qwen2:0.5b           0.129532
      synonym       smollm2:360m           0.126717
```



[11]:
```python
agg = df.groupby(['augmentation_type', 'model'])['length_ratio'].mean().
 ↪reset_index()

plt.figure(figsize=(12,7))
```

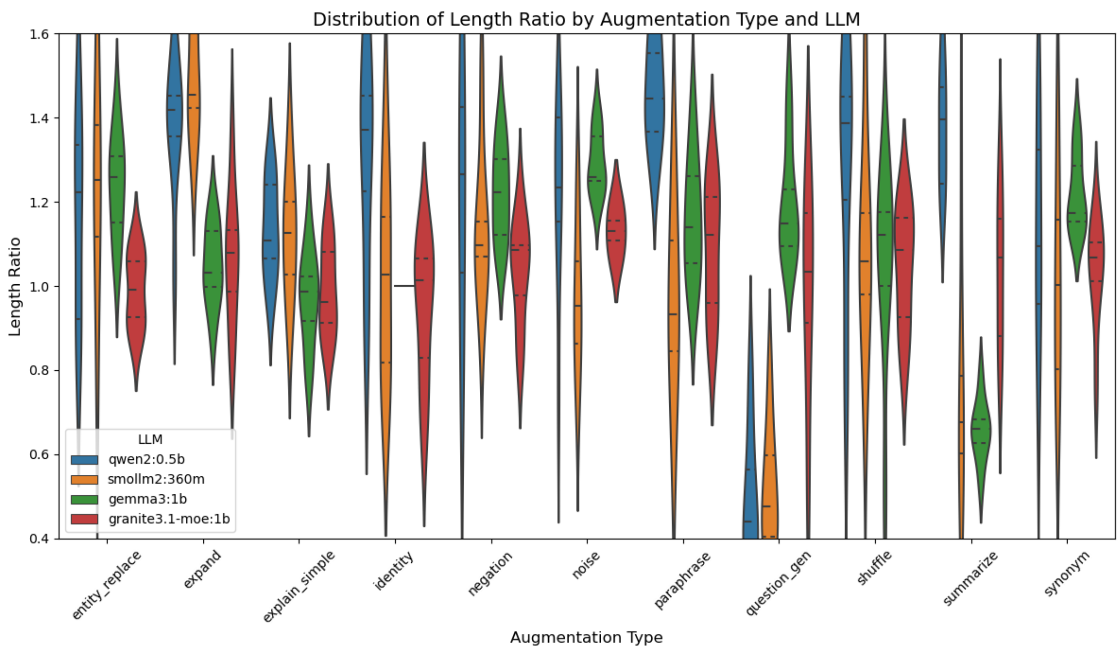
```python
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='length_ratio',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Length Ratio", fontsize=12)
plt.title("Length Ratio by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Length Ratio Table:\n")
print(agg.to_string(index=False))

plt.figure(figsize=(12,7))
sns.violinplot(
    data=df,
    x='augmentation_type',
    y='length_ratio',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Length Ratio", fontsize=12)
plt.title("Distribution of Length Ratio by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.ylim(.4, 1.6)
plt.show()
```



Mean Length Ratio Table:

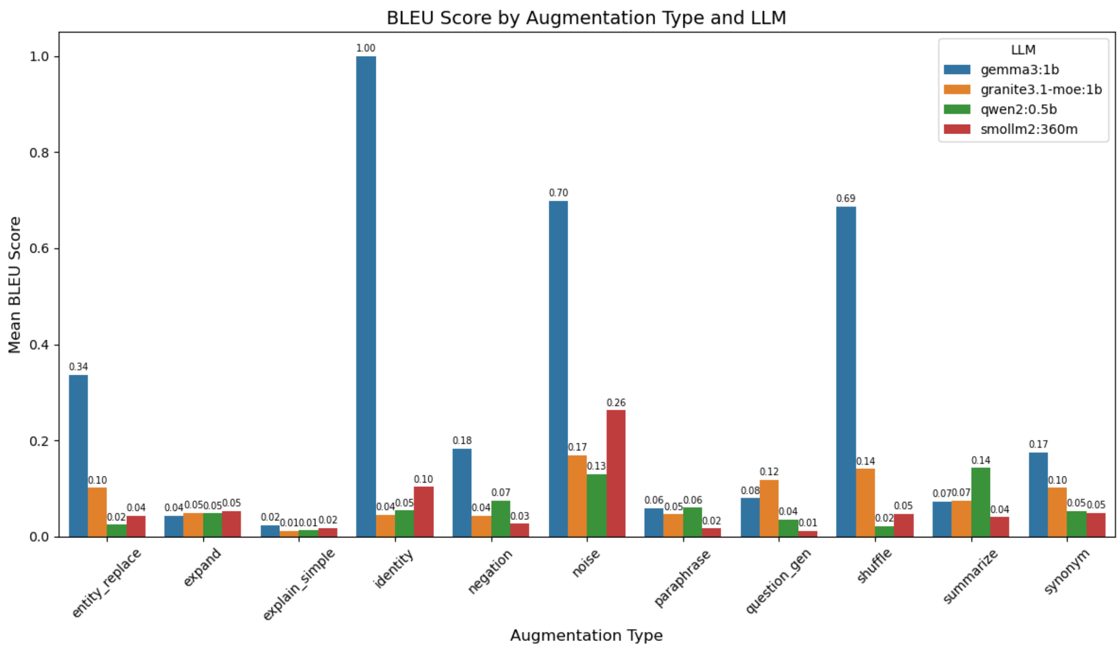
augmentation_type	model	length_ratio
entity_replace	gemma3:1b	1.236984
entity_replace	granite3.1-moe:1b	0.990239
entity_replace	qwen2:0.5b	1.166508
entity_replace	smollm2:360m	1.249348
expand	gemma3:1b	1.049079
expand	granite3.1-moe:1b	1.067523
expand	qwen2:0.5b	1.382795
expand	smollm2:360m	1.507696
explain_simple	gemma3:1b	0.968811
explain_simple	granite3.1-moe:1b	0.988082
explain_simple	qwen2:0.5b	1.134455
explain_simple	smollm2:360m	1.122041
identity	gemma3:1b	1.000000
identity	granite3.1-moe:1b	0.943784
identity	qwen2:0.5b	1.320280
identity	smollm2:360m	1.016836
negation	gemma3:1b	1.220938
negation	granite3.1-moe:1b	1.038055
negation	qwen2:0.5b	1.237072
negation	smollm2:360m	1.137183
noise	gemma3:1b	1.292165
noise	granite3.1-moe:1b	1.131486
noise	qwen2:0.5b	1.211395

```
        noise       smollm2:360m      0.973453
   paraphrase          gemma3:1b      1.159645
   paraphrase granite3.1-moe:1b      1.089286
   paraphrase         qwen2:0.5b      1.458942
   paraphrase       smollm2:360m      0.963935
 question_gen          gemma3:1b      1.195989
 question_gen granite3.1-moe:1b      0.984485
 question_gen         qwen2:0.5b      0.487344
 question_gen       smollm2:360m      0.513460
      shuffle          gemma3:1b      1.050232
      shuffle granite3.1-moe:1b      1.047250
      shuffle         qwen2:0.5b      1.275040
      shuffle       smollm2:360m      1.087972
    summarize          gemma3:1b      0.656102
    summarize granite3.1-moe:1b      1.041472
    summarize         qwen2:0.5b      1.385018
    summarize       smollm2:360m      0.773369
      synonym          gemma3:1b      1.219601
      synonym granite3.1-moe:1b      1.033883
      synonym         qwen2:0.5b      1.123736
      synonym       smollm2:360m      1.018208
```



[12]:
```
agg = df.groupby(['augmentation_type', 'model'])['bleu'].mean().reset_index()

plt.figure(figsize=(12,7))
bar = sns.barplot(
```

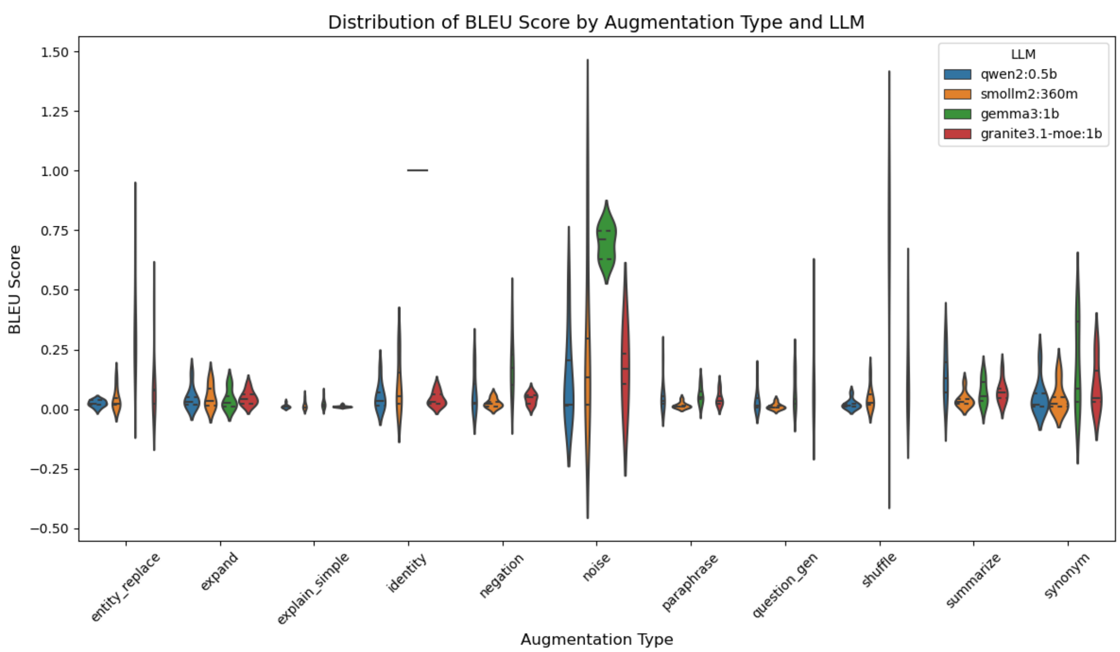
```
    data=agg,
    x='augmentation_type',
    y='bleu',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean BLEU Score", fontsize=12)
plt.title("BLEU Score by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean BLEU Score Table:\n")
print(agg.to_string(index=False))

plt.figure(figsize=(12,7))
sns.violinplot(
    data=df,
    x='augmentation_type',
    y='bleu',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("BLEU Score", fontsize=12)
plt.title("Distribution of BLEU Score by Augmentation Type and LLM",
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean BLEU Score Table:

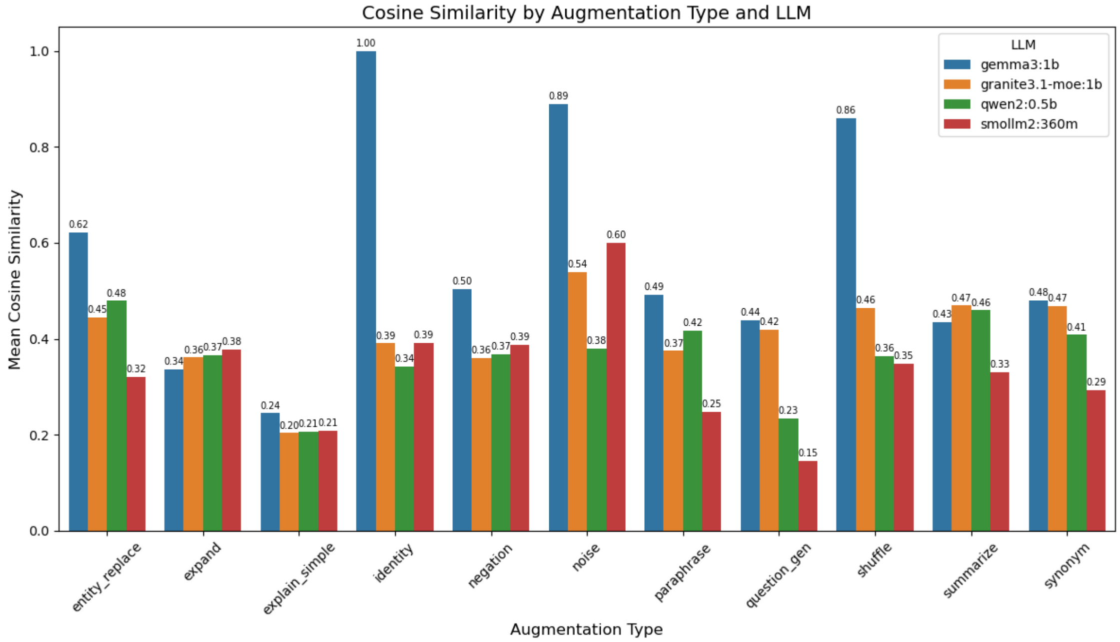
augmentation_type	model	bleu
entity_replace	gemma3:1b	0.336007
entity_replace	granite3.1-moe:1b	0.100904
entity_replace	qwen2:0.5b	0.024408
entity_replace	smollm2:360m	0.042657
expand	gemma3:1b	0.042257
expand	granite3.1-moe:1b	0.048258
expand	qwen2:0.5b	0.047998
expand	smollm2:360m	0.053188
explain_simple	gemma3:1b	0.023838
explain_simple	granite3.1-moe:1b	0.011359
explain_simple	qwen2:0.5b	0.012483
explain_simple	smollm2:360m	0.017801
identity	gemma3:1b	1.000000
identity	granite3.1-moe:1b	0.044345
identity	qwen2:0.5b	0.053984
identity	smollm2:360m	0.102974
negation	gemma3:1b	0.182515
negation	granite3.1-moe:1b	0.043616
negation	qwen2:0.5b	0.074165
negation	smollm2:360m	0.026137
noise	gemma3:1b	0.698746
noise	granite3.1-moe:1b	0.168748
noise	qwen2:0.5b	0.130235

```
        noise      smollm2:360m 0.262345
   paraphrase         gemma3:1b 0.059311
   paraphrase granite3.1-moe:1b 0.046070
   paraphrase        qwen2:0.5b 0.059510
   paraphrase      smollm2:360m 0.016563
 question_gen         gemma3:1b 0.079287
 question_gen granite3.1-moe:1b 0.118030
 question_gen        qwen2:0.5b 0.035539
 question_gen      smollm2:360m 0.012183
      shuffle         gemma3:1b 0.687404
      shuffle granite3.1-moe:1b 0.140264
      shuffle        qwen2:0.5b 0.021556
      shuffle      smollm2:360m 0.047154
    summarize         gemma3:1b 0.072126
    summarize granite3.1-moe:1b 0.074009
    summarize        qwen2:0.5b 0.142341
    summarize      smollm2:360m 0.039963
      synonym         gemma3:1b 0.174645
      synonym granite3.1-moe:1b 0.101196
      synonym        qwen2:0.5b 0.052701
      synonym      smollm2:360m 0.049614
```



[13]:
```
agg = df.groupby(['augmentation_type', 'model'])['cosine_similarity'].mean().
 ↪reset_index()

plt.figure(figsize=(12,7))
```

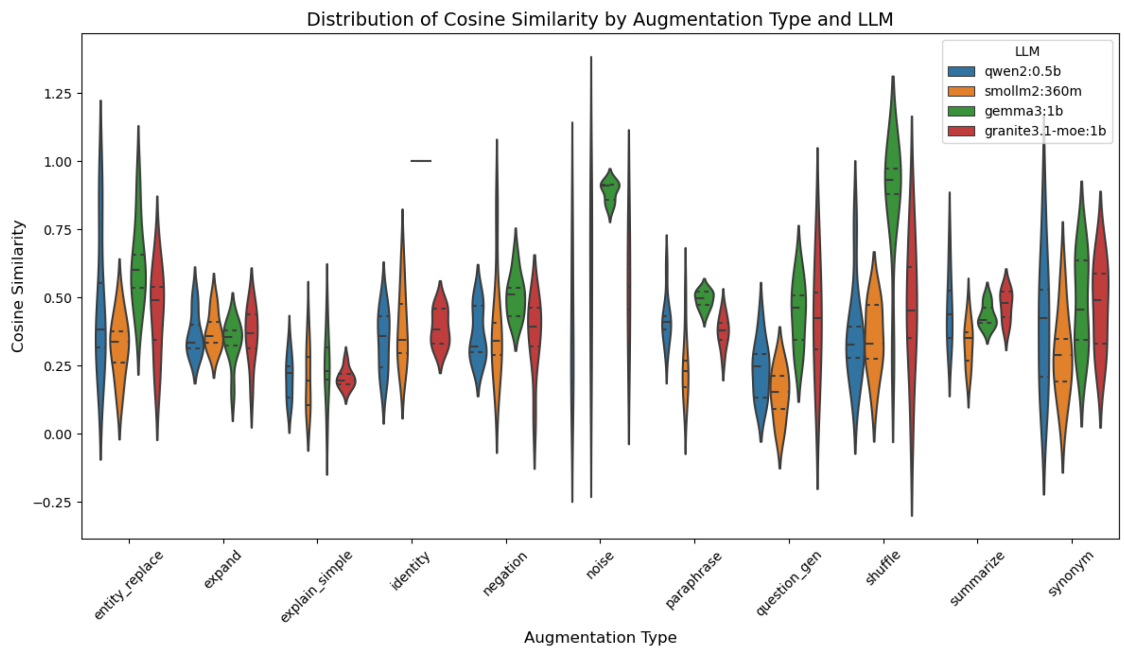
```
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='cosine_similarity',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Cosine Similarity", fontsize=12)
plt.title("Cosine Similarity by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Cosine Similarity Table:\n")
print(agg.to_string(index=False))

plt.figure(figsize=(12,7))
sns.violinplot(
    data=df,
    x='augmentation_type',
    y='cosine_similarity',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Cosine Similarity", fontsize=12)
plt.title("Distribution of Cosine Similarity by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Cosine Similarity Table:

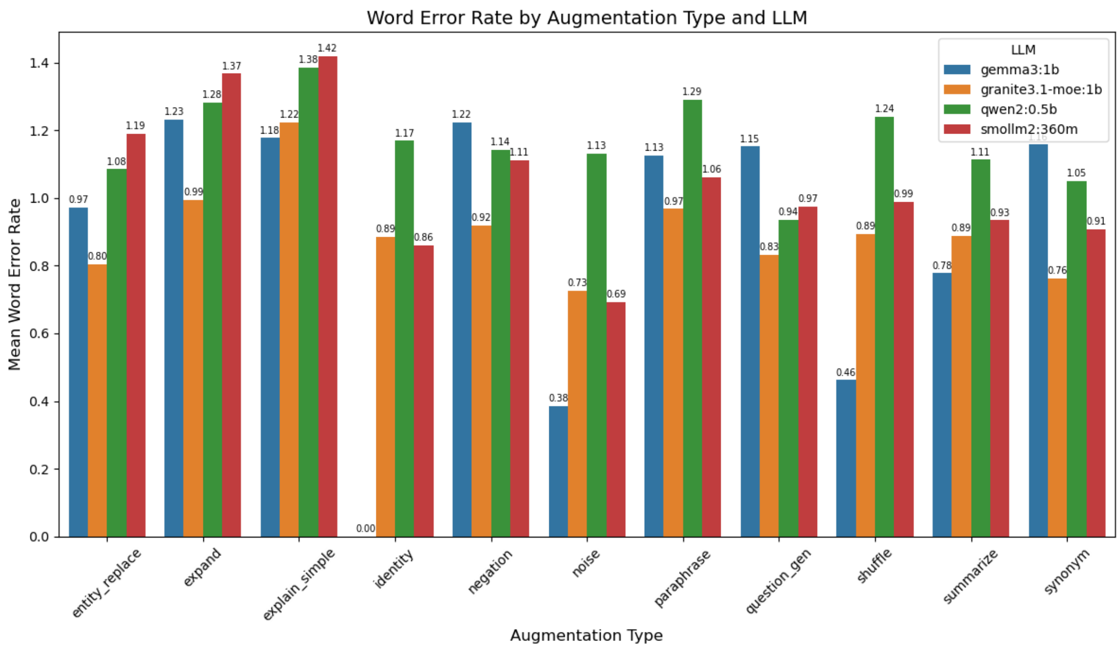
augmentation_type	model	cosine_similarity
entity_replace	gemma3:1b	0.622095
entity_replace	granite3.1-moe:1b	0.445180
entity_replace	qwen2:0.5b	0.479066
entity_replace	smollm2:360m	0.321294
expand	gemma3:1b	0.335939
expand	granite3.1-moe:1b	0.361077
expand	qwen2:0.5b	0.365863
expand	smollm2:360m	0.377856
explain_simple	gemma3:1b	0.244547
explain_simple	granite3.1-moe:1b	0.203341
explain_simple	qwen2:0.5b	0.206513
explain_simple	smollm2:360m	0.208020
identity	gemma3:1b	1.000000
identity	granite3.1-moe:1b	0.390596
identity	qwen2:0.5b	0.342094
identity	smollm2:360m	0.391382
negation	gemma3:1b	0.504206
negation	granite3.1-moe:1b	0.360369
negation	qwen2:0.5b	0.367078
negation	smollm2:360m	0.386901
noise	gemma3:1b	0.889123
noise	granite3.1-moe:1b	0.539366
noise	qwen2:0.5b	0.379859

```
        noise       smollm2:360m           0.600188
   paraphrase           gemma3:1b          0.492292
   paraphrase granite3.1-moe:1b            0.374676
   paraphrase          qwen2:0.5b          0.417718
   paraphrase        smollm2:360m          0.247645
 question_gen           gemma3:1b          0.439280
 question_gen granite3.1-moe:1b            0.419343
 question_gen          qwen2:0.5b          0.234261
 question_gen        smollm2:360m          0.145924
      shuffle           gemma3:1b          0.859389
      shuffle granite3.1-moe:1b            0.464123
      shuffle          qwen2:0.5b          0.364328
      shuffle        smollm2:360m          0.348602
    summarize           gemma3:1b          0.434552
    summarize granite3.1-moe:1b            0.469117
    summarize          qwen2:0.5b          0.459336
    summarize        smollm2:360m          0.330073
      synonym           gemma3:1b          0.480473
      synonym granite3.1-moe:1b            0.467419
      synonym          qwen2:0.5b          0.408312
      synonym        smollm2:360m          0.293056
```



[14]:
```
agg = df.groupby(['augmentation_type', 'model'])['wer'].mean().reset_index()

plt.figure(figsize=(12,7))
bar = sns.barplot(
```

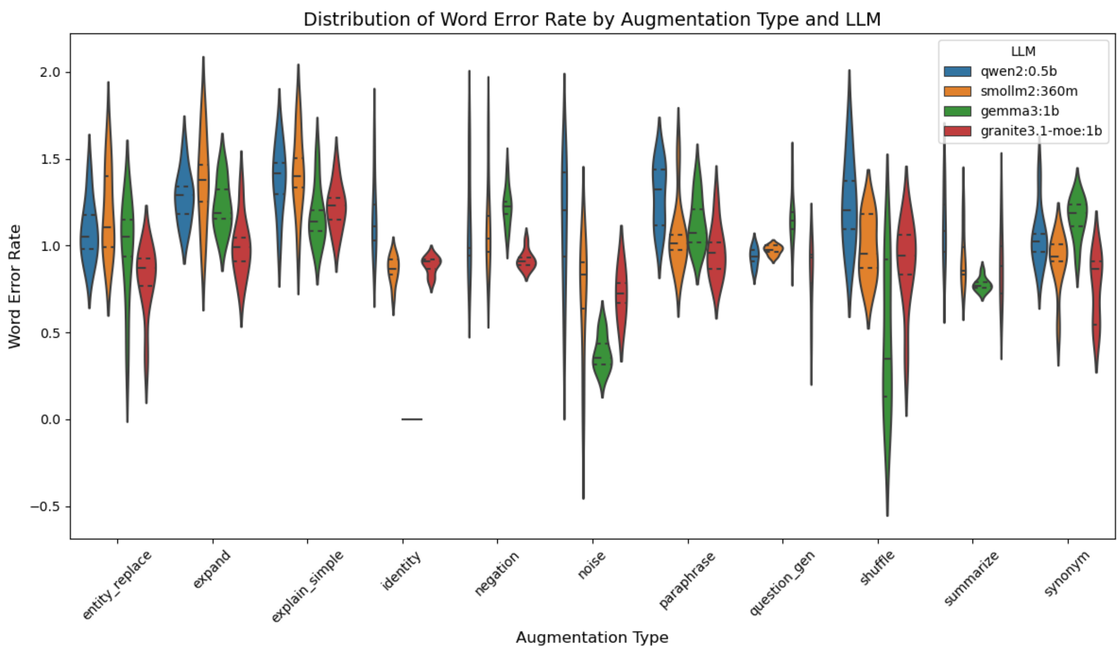
```
    data=agg,
    x='augmentation_type',
    y='wer',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Word Error Rate", fontsize=12)
plt.title("Word Error Rate by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Word Error Rate Table:\n")
print(agg.to_string(index=False))

plt.figure(figsize=(12,7))
sns.violinplot(
    data=df,
    x='augmentation_type',
    y='wer',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Word Error Rate", fontsize=12)
plt.title("Distribution of Word Error Rate by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Word Error Rate Table:

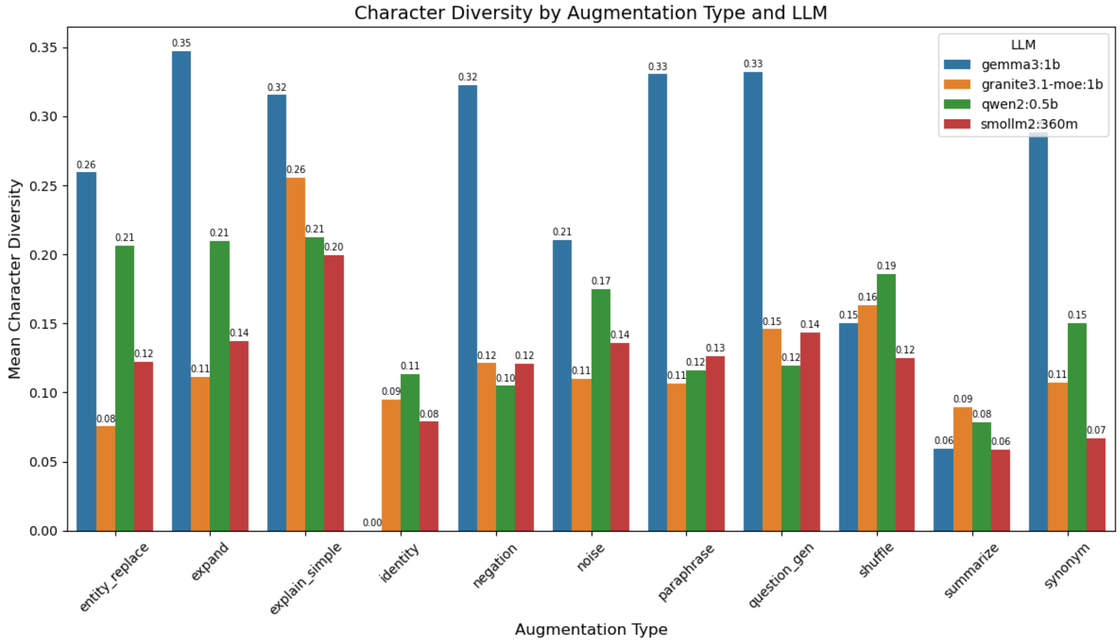
augmentation_type	model	wer
entity_replace	gemma3:1b	0.970939
entity_replace	granite3.1-moe:1b	0.804647
entity_replace	qwen2:0.5b	1.084771
entity_replace	smollm2:360m	1.190478
expand	gemma3:1b	1.230516
expand	granite3.1-moe:1b	0.993683
expand	qwen2:0.5b	1.280821
expand	smollm2:360m	1.367130
explain_simple	gemma3:1b	1.177038
explain_simple	granite3.1-moe:1b	1.223285
explain_simple	qwen2:0.5b	1.384482
explain_simple	smollm2:360m	1.419171
identity	gemma3:1b	0.000000
identity	granite3.1-moe:1b	0.885898
identity	qwen2:0.5b	1.168764
identity	smollm2:360m	0.859945
negation	gemma3:1b	1.222498
negation	granite3.1-moe:1b	0.919054
negation	qwen2:0.5b	1.141896
negation	smollm2:360m	1.111944
noise	gemma3:1b	0.384143
noise	granite3.1-moe:1b	0.725806
noise	qwen2:0.5b	1.131913

```
       noise       smollm2:360m  0.692547
  paraphrase          gemma3:1b  1.125937
  paraphrase granite3.1-moe:1b  0.967314
  paraphrase         qwen2:0.5b  1.290791
  paraphrase       smollm2:360m  1.060419
question_gen          gemma3:1b  1.151979
question_gen granite3.1-moe:1b  0.833109
question_gen         qwen2:0.5b  0.935138
question_gen       smollm2:360m  0.974449
     shuffle          gemma3:1b  0.462177
     shuffle granite3.1-moe:1b  0.893410
     shuffle         qwen2:0.5b  1.240501
     shuffle       smollm2:360m  0.989375
   summarize          gemma3:1b  0.777333
   summarize granite3.1-moe:1b  0.887743
   summarize         qwen2:0.5b  1.113069
   summarize       smollm2:360m  0.933830
     synonym          gemma3:1b  1.157594
     synonym granite3.1-moe:1b  0.763213
     synonym         qwen2:0.5b  1.049095
     synonym       smollm2:360m  0.906838
```



[15]:
```
agg = df.groupby(['augmentation_type', 'model'])['char_diversity'].mean().
 ↪reset_index()

plt.figure(figsize=(12,7))
```

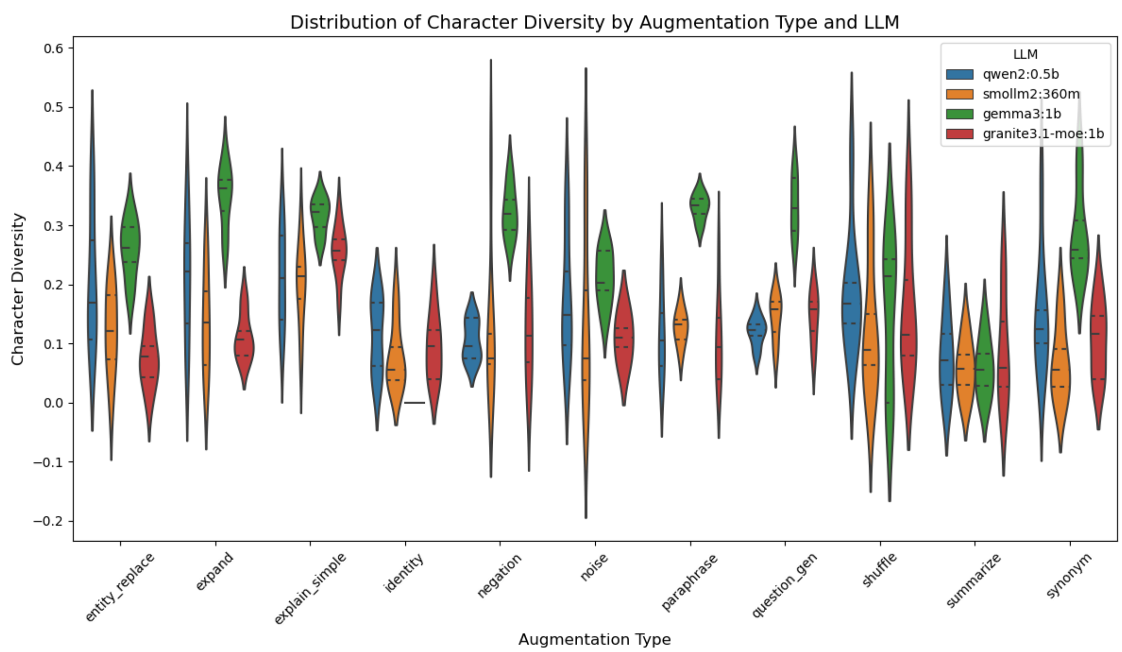
```
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='char_diversity',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Character Diversity", fontsize=12)
plt.title("Character Diversity by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Character Diversity Table:\n")
print(agg.to_string(index=False))

plt.figure(figsize=(12,7))
sns.violinplot(
    data=df,
    x='augmentation_type',
    y='char_diversity',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Character Diversity", fontsize=12)
plt.title("Distribution of Character Diversity by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Character Diversity Table:

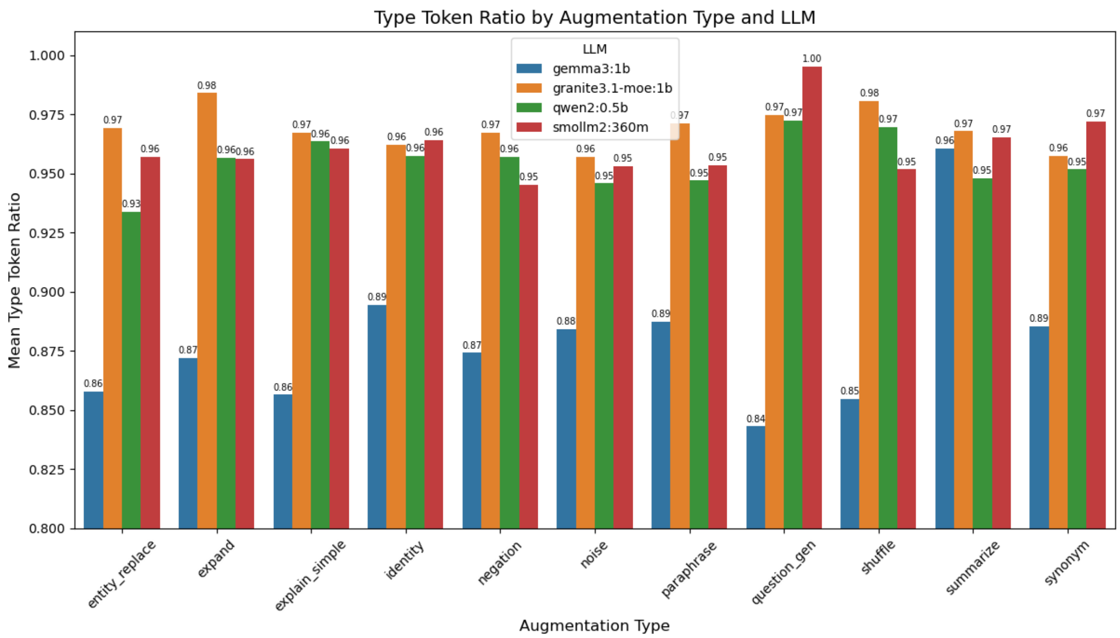
augmentation_type	model	char_diversity
entity_replace	gemma3:1b	0.259408
entity_replace	granite3.1-moe:1b	0.075425
entity_replace	qwen2:0.5b	0.206214
entity_replace	smollm2:360m	0.122106
expand	gemma3:1b	0.347327
expand	granite3.1-moe:1b	0.111536
expand	qwen2:0.5b	0.209976
expand	smollm2:360m	0.137446
explain_simple	gemma3:1b	0.315454
explain_simple	granite3.1-moe:1b	0.255851
explain_simple	qwen2:0.5b	0.212264
explain_simple	smollm2:360m	0.199737
identity	gemma3:1b	0.000000
identity	granite3.1-moe:1b	0.094899
identity	qwen2:0.5b	0.113213
identity	smollm2:360m	0.078906
negation	gemma3:1b	0.322866
negation	granite3.1-moe:1b	0.121212
negation	qwen2:0.5b	0.104828
negation	smollm2:360m	0.120919
noise	gemma3:1b	0.210512
noise	granite3.1-moe:1b	0.109890
noise	qwen2:0.5b	0.174991

```
       noise       smollm2:360m        0.135840
  paraphrase           gemma3:1b        0.330481
  paraphrase granite3.1-moe:1b        0.106477
  paraphrase          qwen2:0.5b        0.116298
  paraphrase        smollm2:360m        0.126579
question_gen           gemma3:1b        0.332157
question_gen granite3.1-moe:1b        0.145877
question_gen          qwen2:0.5b        0.119569
question_gen        smollm2:360m        0.143274
     shuffle           gemma3:1b        0.150104
     shuffle granite3.1-moe:1b        0.163039
     shuffle          qwen2:0.5b        0.186026
     shuffle        smollm2:360m        0.124883
   summarize           gemma3:1b        0.059246
   summarize granite3.1-moe:1b        0.089673
   summarize          qwen2:0.5b        0.078301
   summarize        smollm2:360m        0.058926
     synonym           gemma3:1b        0.288397
     synonym granite3.1-moe:1b        0.107062
     synonym          qwen2:0.5b        0.150192
     synonym        smollm2:360m        0.067023
```



[16]:
```python
agg = df.groupby(['augmentation_type', 'model'])['type_token_ratio'].mean().
↪reset_index()

plt.figure(figsize=(12,7))
```

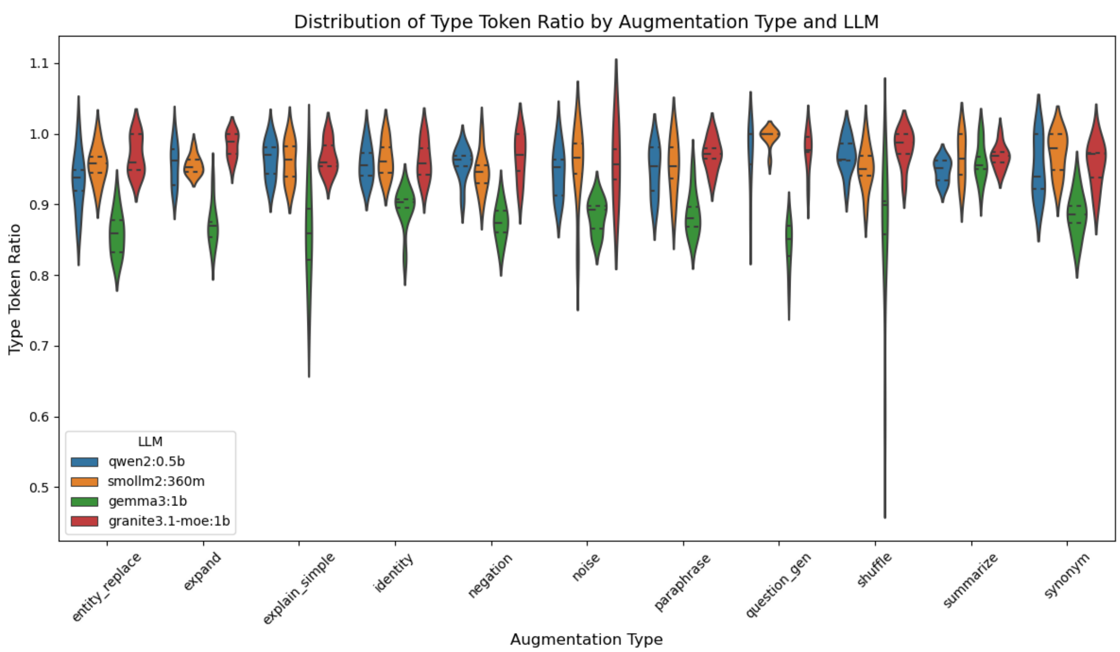
```python
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='type_token_ratio',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Type Token Ratio", fontsize=12)
plt.title("Type Token Ratio by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.ylim(.8,1.01)
plt.show()

print("Mean Type Token Ratio Table:\n")
print(agg.to_string(index=False))

plt.figure(figsize=(12,7))
sns.violinplot(
    data=df,
    x='augmentation_type',
    y='type_token_ratio',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Type Token Ratio", fontsize=12)
plt.title("Distribution of Type Token Ratio by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Type Token Ratio Table:

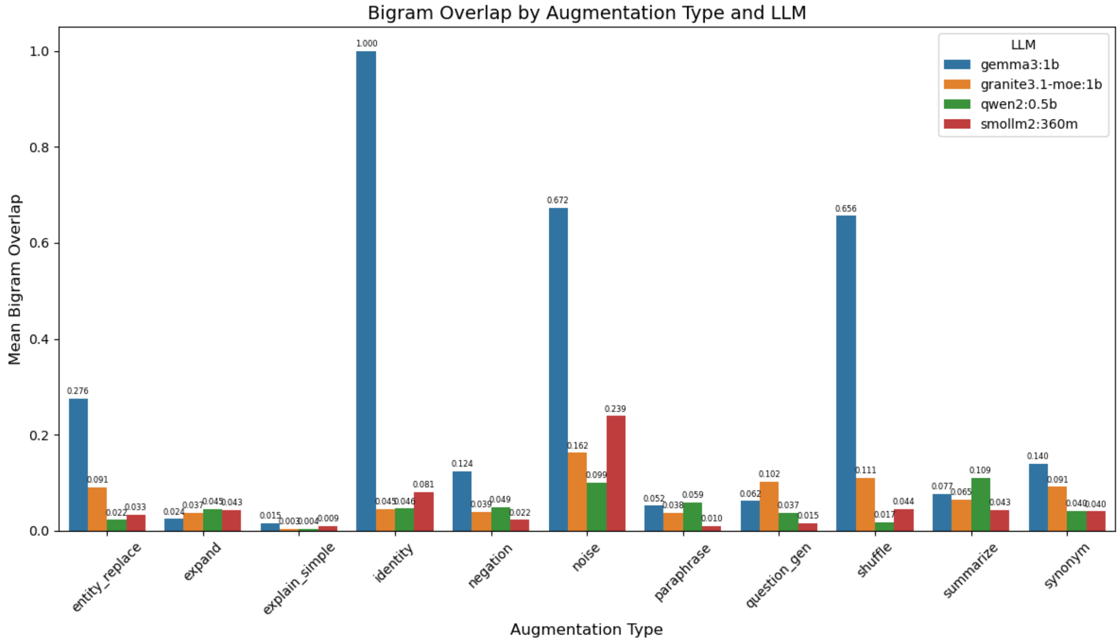
augmentation_type	model	type_token_ratio
entity_replace	gemma3:1b	0.857892
entity_replace	granite3.1-moe:1b	0.969279
entity_replace	qwen2:0.5b	0.933791
entity_replace	smollm2:360m	0.956961
expand	gemma3:1b	0.871875
expand	granite3.1-moe:1b	0.983983
expand	qwen2:0.5b	0.956655
expand	smollm2:360m	0.956069
explain_simple	gemma3:1b	0.856441
explain_simple	granite3.1-moe:1b	0.967255
explain_simple	qwen2:0.5b	0.963499
explain_simple	smollm2:360m	0.960689
identity	gemma3:1b	0.894546
identity	granite3.1-moe:1b	0.962174
identity	qwen2:0.5b	0.957552
identity	smollm2:360m	0.964159
negation	gemma3:1b	0.874110
negation	granite3.1-moe:1b	0.967172
negation	qwen2:0.5b	0.956909
negation	smollm2:360m	0.945347
noise	gemma3:1b	0.884083
noise	granite3.1-moe:1b	0.957143
noise	qwen2:0.5b	0.945767
noise	smollm2:360m	0.953060

```
paraphrase          gemma3:1b          0.887194
paraphrase granite3.1-moe:1b          0.971228
paraphrase         qwen2:0.5b          0.947000
paraphrase       smollm2:360m          0.953424
question_gen          gemma3:1b          0.843012
question_gen granite3.1-moe:1b          0.974845
question_gen         qwen2:0.5b          0.972431
question_gen       smollm2:360m          0.995192
   shuffle          gemma3:1b          0.854640
   shuffle granite3.1-moe:1b          0.980633
   shuffle         qwen2:0.5b          0.969498
   shuffle       smollm2:360m          0.951689
 summarize          gemma3:1b          0.960620
 summarize granite3.1-moe:1b          0.967855
 summarize         qwen2:0.5b          0.947847
 summarize       smollm2:360m          0.965432
   synonym          gemma3:1b          0.885274
   synonym granite3.1-moe:1b          0.957266
   synonym         qwen2:0.5b          0.951701
   synonym       smollm2:360m          0.971937
```



[17]:
```python
agg = df.groupby(['augmentation_type', 'model'])['bigram_overlap'].mean().reset_index()

plt.figure(figsize=(12,7))
bar = sns.barplot(
```

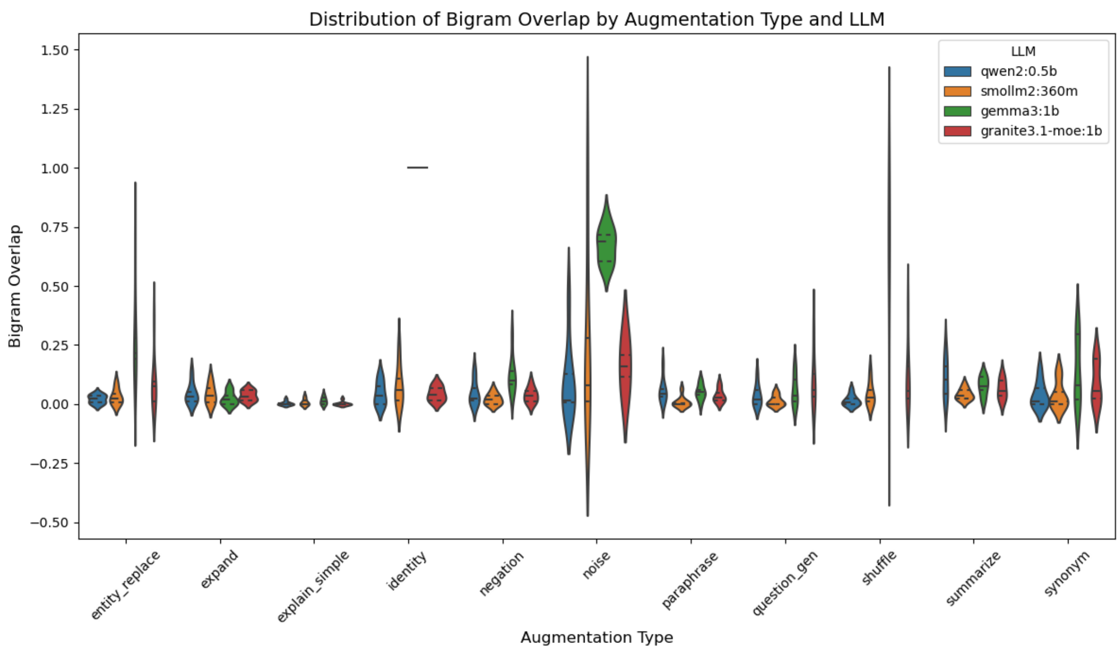
```
    data=agg,
    x='augmentation_type',
    y='bigram_overlap',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.3f', label_type='edge', padding=2,␣
 ↪fontsize=6)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Bigram Overlap", fontsize=12)
plt.title("Bigram Overlap by Augmentation Type and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Bigram Overlap Table:\n")
print(agg.to_string(index=False))

plt.figure(figsize=(12,7))
sns.violinplot(
    data=df,
    x='augmentation_type',
    y='bigram_overlap',
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Bigram Overlap", fontsize=12)
plt.title("Distribution of Bigram Overlap by Augmentation Type and LLM",␣
 ↪fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```



Mean Bigram Overlap Table:

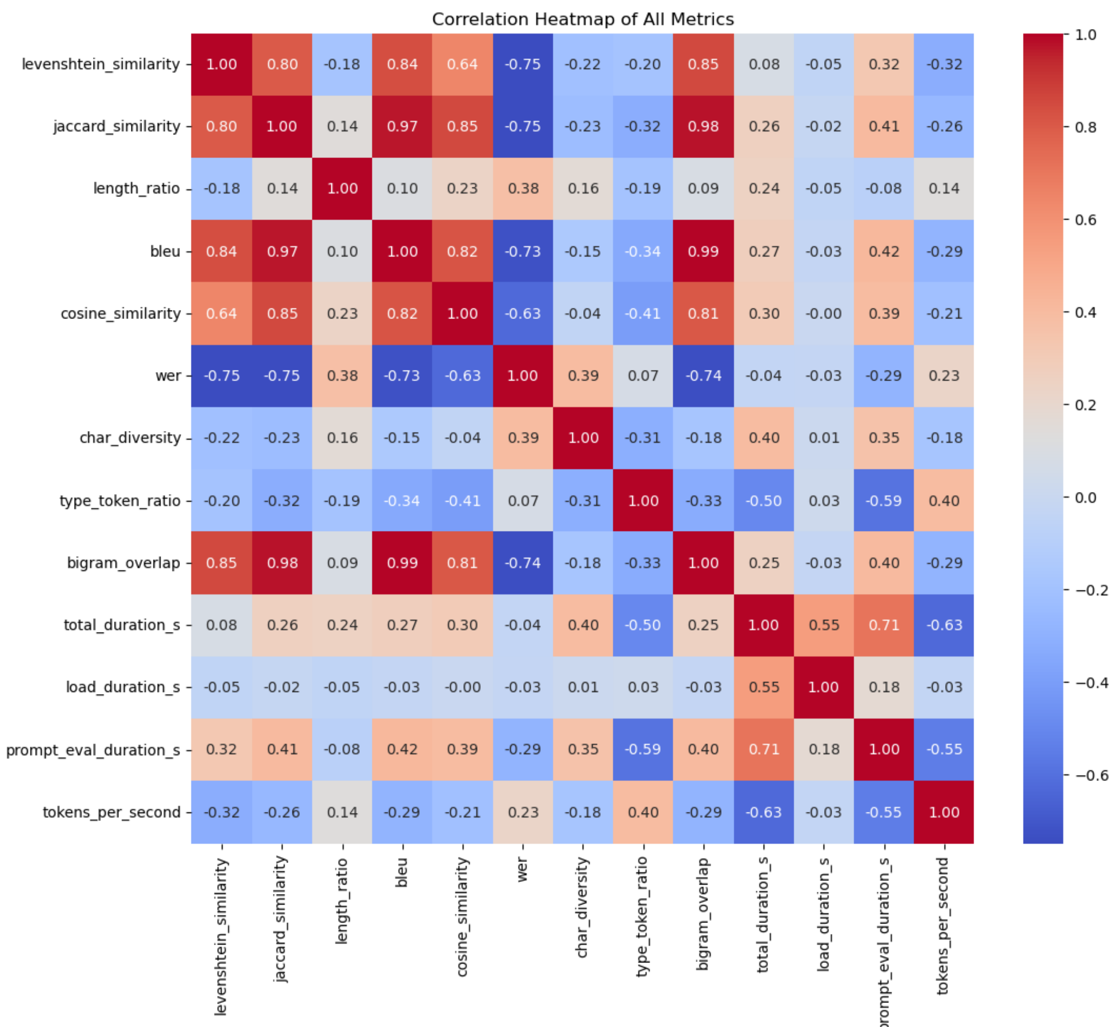
augmentation_type	model	bigram_overlap
entity_replace	gemma3:1b	0.276008
entity_replace	granite3.1-moe:1b	0.090576
entity_replace	qwen2:0.5b	0.022345
entity_replace	smollm2:360m	0.032654
expand	gemma3:1b	0.024316
expand	granite3.1-moe:1b	0.036781
expand	qwen2:0.5b	0.045052
expand	smollm2:360m	0.043334
explain_simple	gemma3:1b	0.015429
explain_simple	granite3.1-moe:1b	0.003049
explain_simple	qwen2:0.5b	0.004320
explain_simple	smollm2:360m	0.008956
identity	gemma3:1b	1.000000
identity	granite3.1-moe:1b	0.044572
identity	qwen2:0.5b	0.046083
identity	smollm2:360m	0.081047
negation	gemma3:1b	0.124283
negation	granite3.1-moe:1b	0.039028
negation	qwen2:0.5b	0.049130
negation	smollm2:360m	0.022086
noise	gemma3:1b	0.672246
noise	granite3.1-moe:1b	0.161934
noise	qwen2:0.5b	0.099316

```
        noise       smollm2:360m        0.238900
   paraphrase          gemma3:1b        0.051615
   paraphrase granite3.1-moe:1b        0.037508
   paraphrase         qwen2:0.5b        0.058800
   paraphrase       smollm2:360m        0.010150
 question_gen          gemma3:1b        0.061532
 question_gen granite3.1-moe:1b        0.102173
 question_gen         qwen2:0.5b        0.036526
 question_gen       smollm2:360m        0.014815
      shuffle          gemma3:1b        0.655958
      shuffle granite3.1-moe:1b        0.110579
      shuffle         qwen2:0.5b        0.016694
      shuffle       smollm2:360m        0.044234
    summarize          gemma3:1b        0.076663
    summarize granite3.1-moe:1b        0.064816
    summarize         qwen2:0.5b        0.109437
    summarize       smollm2:360m        0.042743
      synonym          gemma3:1b        0.139534
      synonym granite3.1-moe:1b        0.091272
      synonym         qwen2:0.5b        0.040477
      synonym       smollm2:360m        0.040125
```



[18]:
```
metrics = [
    'levenshtein_similarity', 'jaccard_similarity', 'length_ratio', 'bleu',
    'cosine_similarity', 'wer', 'char_diversity', 'type_token_ratio',␣
 ↪'bigram_overlap',
```

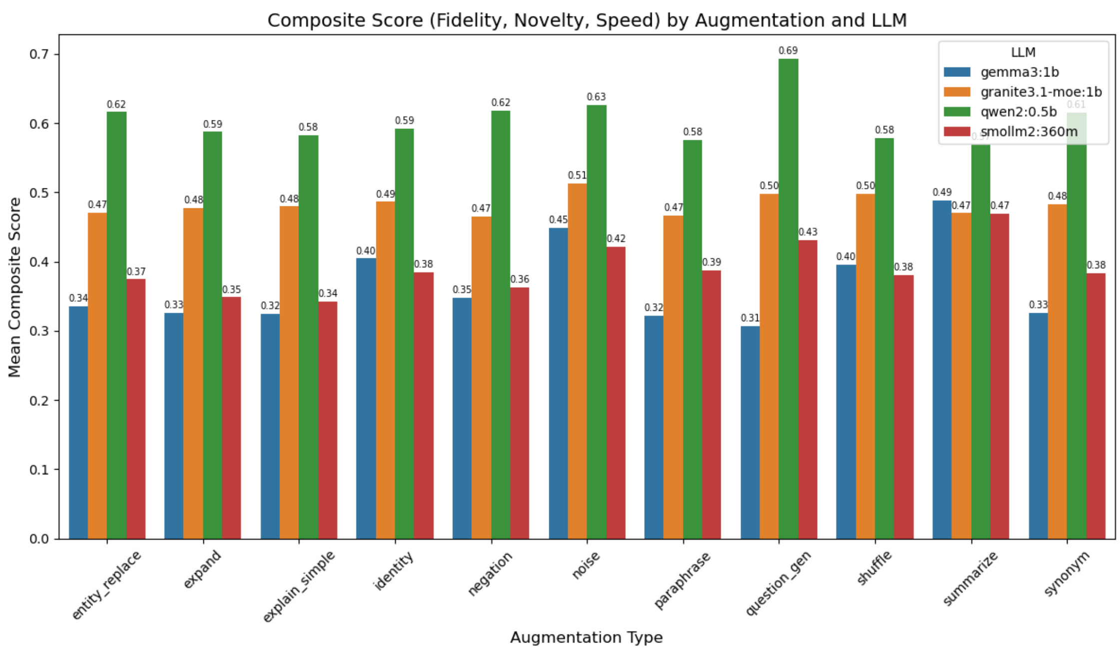
```
    'total_duration_s', 'load_duration_s', 'prompt_eval_duration_s',
    'tokens_per_second'
]
corr = df[metrics].corr()
plt.figure(figsize=(12,10))
sns.heatmap(corr, annot=True, fmt=".2f", cmap="coolwarm")
plt.title("Correlation Heatmap of All Metrics")
plt.show()
```



[19]:
```
import numpy as np

# 1. Normalize tokens_per_second to 0-1 for fair combination
tps_min = df['tokens_per_second'].min()
tps_max = df['tokens_per_second'].max()
```

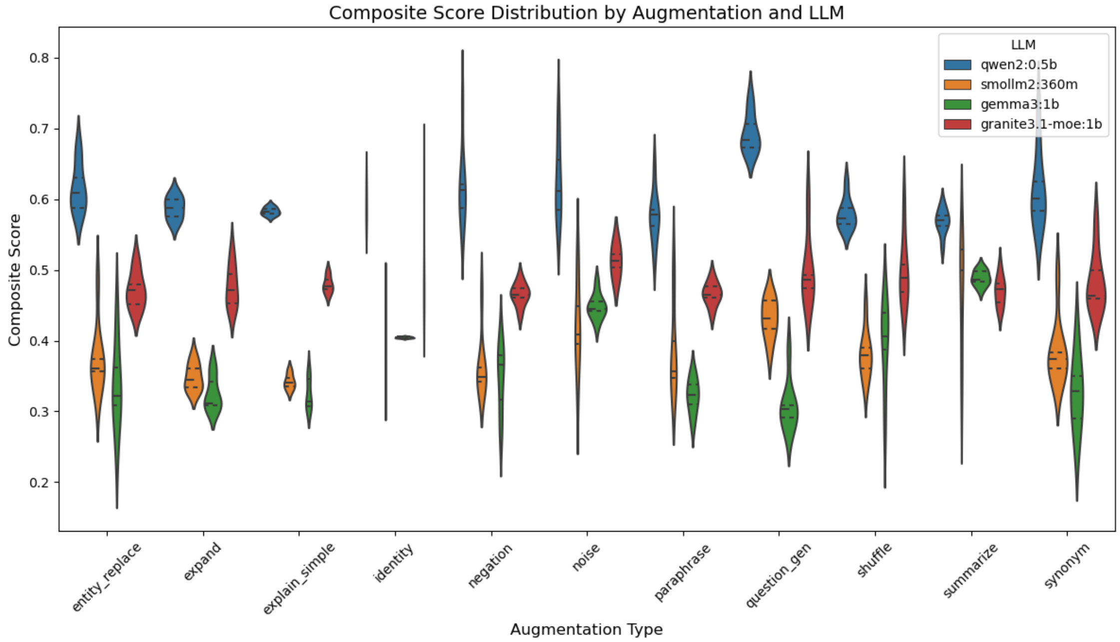
```
df['tokens_per_second_norm'] = (df['tokens_per_second'] - tps_min) / (tps_max -
    tps_min)

# 2. Compute composite score
# You may change these weights (must sum to 1 for interpretation)
w1, w2, w3 = 0.4, 0.3, 0.3
df['composite_score'] = (
    w1 * df['levenshtein_similarity'] +
    w2 * (1 - df['bigram_overlap']) +
    w3 * df['tokens_per_second_norm']
)

# 3. Aggregate by augmentation_type and model
agg = df.groupby(['augmentation_type', 'model'])['composite_score'].mean().
    reset_index()

# 4. Bar Plot
plt.figure(figsize=(12,7))
bar = sns.barplot(
    data=agg,
    x='augmentation_type',
    y='composite_score',
    hue='model',
    order=augmentation_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,
    fontsize=7)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Mean Composite Score", fontsize=12)
plt.title("Composite Score (Fidelity, Novelty, Speed) by Augmentation and LLM",
    fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Composite Score Table:\n")
print(agg.to_string(index=False))

# 5. Violin Plot for distribution
plt.figure(figsize=(12,7))
sns.violinplot(
    data=df,
    x='augmentation_type',
    y='composite_score',
```

```python
    hue='model',
    order=augmentation_order,
    palette='tab10',
    inner="quartile"
)
plt.xlabel("Augmentation Type", fontsize=12)
plt.ylabel("Composite Score", fontsize=12)
plt.title("Composite Score Distribution by Augmentation and LLM", fontsize=14)
plt.xticks(rotation=45)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()
```

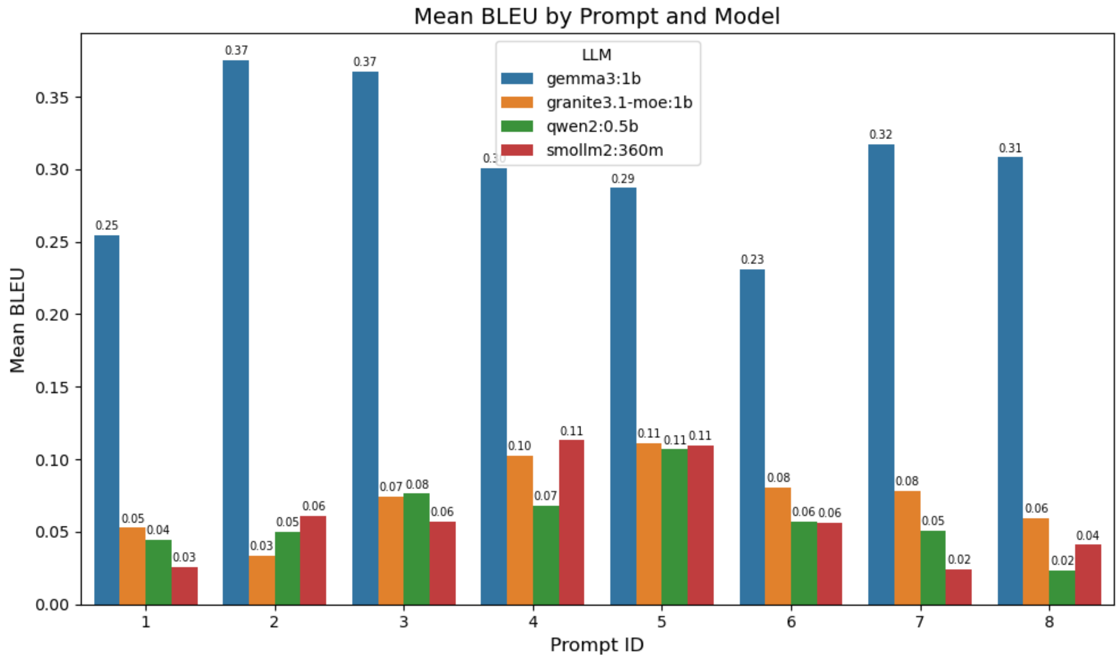



```
Mean Composite Score Table:

augmentation_type             model  composite_score
   entity_replace         gemma3:1b         0.335912
   entity_replace granite3.1-moe:1b         0.471249
   entity_replace        qwen2:0.5b         0.616057
   entity_replace      smollm2:360m         0.374485
           expand         gemma3:1b         0.325728
           expand granite3.1-moe:1b         0.477295
           expand        qwen2:0.5b         0.587588
           expand      smollm2:360m         0.348680
   explain_simple         gemma3:1b         0.324912
   explain_simple granite3.1-moe:1b         0.479531
```

explain_simple	qwen2:0.5b	0.583204
explain_simple	smollm2:360m	0.342336
identity	gemma3:1b	0.404691
identity	granite3.1-moe:1b	0.486590
identity	qwen2:0.5b	0.592319
identity	smollm2:360m	0.384884
negation	gemma3:1b	0.347554
negation	granite3.1-moe:1b	0.465591
negation	qwen2:0.5b	0.618670
negation	smollm2:360m	0.363032
noise	gemma3:1b	0.448422
noise	granite3.1-moe:1b	0.512618
noise	qwen2:0.5b	0.626415
noise	smollm2:360m	0.421667
paraphrase	gemma3:1b	0.322206
paraphrase	granite3.1-moe:1b	0.466552
paraphrase	qwen2:0.5b	0.575609
paraphrase	smollm2:360m	0.387589
question_gen	gemma3:1b	0.307361
question_gen	granite3.1-moe:1b	0.498469
question_gen	qwen2:0.5b	0.693501
question_gen	smollm2:360m	0.431563
shuffle	gemma3:1b	0.395759
shuffle	granite3.1-moe:1b	0.498410
shuffle	qwen2:0.5b	0.578838
shuffle	smollm2:360m	0.380806
summarize	gemma3:1b	0.488778
summarize	granite3.1-moe:1b	0.470224
summarize	qwen2:0.5b	0.568610
summarize	smollm2:360m	0.469857
synonym	gemma3:1b	0.325954
synonym	granite3.1-moe:1b	0.483146
synonym	qwen2:0.5b	0.614986
synonym	smollm2:360m	0.383781



[20]:
```python
metric_col = 'bleu'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean BLEU", fontsize=12)
plt.title("Mean BLEU by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean BLEU Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



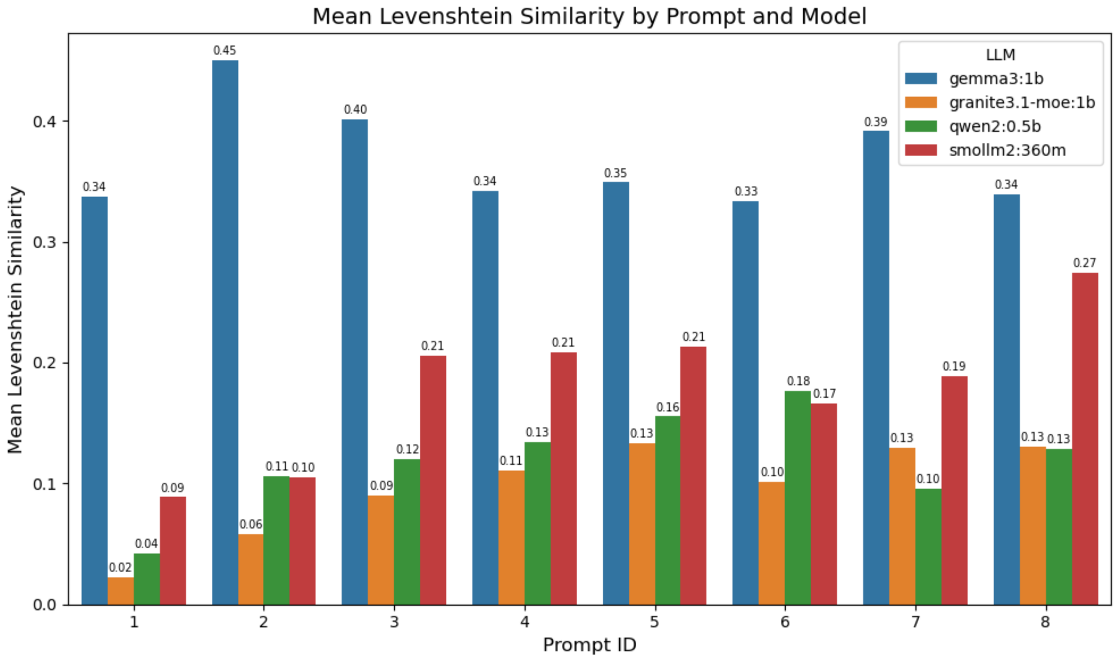
Mean BLEU Table by Prompt and Model:

prompt_id	model	bleu
1	gemma3:1b	0.254414
1	granite3.1-moe:1b	0.052784
1	qwen2:0.5b	0.044227
1	smollm2:360m	0.025847
2	gemma3:1b	0.374889
2	granite3.1-moe:1b	0.033505
2	qwen2:0.5b	0.049927
2	smollm2:360m	0.060754
3	gemma3:1b	0.367525
3	granite3.1-moe:1b	0.074351
3	qwen2:0.5b	0.076296
3	smollm2:360m	0.056945
4	gemma3:1b	0.300483
4	granite3.1-moe:1b	0.102675
4	qwen2:0.5b	0.067952
4	smollm2:360m	0.113136
5	gemma3:1b	0.287023
5	granite3.1-moe:1b	0.110838
5	qwen2:0.5b	0.106843
5	smollm2:360m	0.109319
6	gemma3:1b	0.231133
6	granite3.1-moe:1b	0.080830
6	qwen2:0.5b	0.056787

```
6      smollm2:360m 0.056255
7          gemma3:1b 0.317204
7 granite3.1-moe:1b 0.078031
7         qwen2:0.5b 0.050871
7      smollm2:360m 0.024435
8          gemma3:1b 0.308155
8 granite3.1-moe:1b 0.059570
8         qwen2:0.5b 0.023400
8      smollm2:360m 0.041003
```

[21]:

```
metric_col = 'levenshtein_similarity'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Levenshtein Similarity", fontsize=12)
plt.title("Mean Levenshtein Similarity by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Levenshtein Similarity Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```

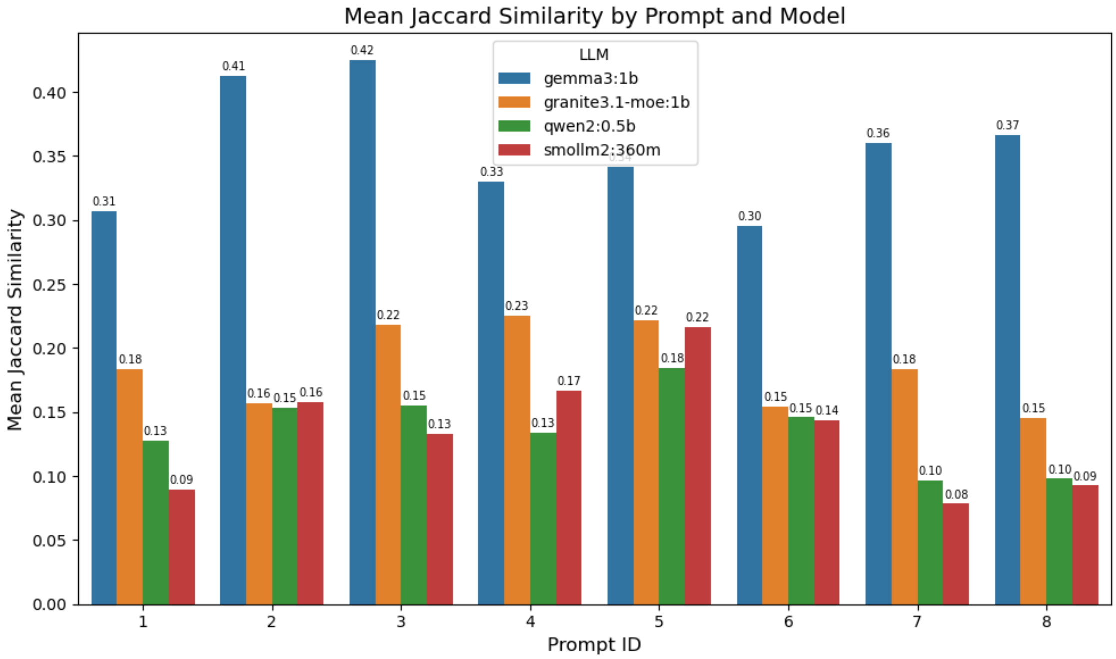


Mean Levenshtein Similarity Table by Prompt and Model:

prompt_id	model	levenshtein_similarity
1	gemma3:1b	0.336949
1	granite3.1-moe:1b	0.022555
1	qwen2:0.5b	0.042446
1	smollm2:360m	0.088772
2	gemma3:1b	0.449745
2	granite3.1-moe:1b	0.058244
2	qwen2:0.5b	0.106182
2	smollm2:360m	0.104981
3	gemma3:1b	0.401415
3	granite3.1-moe:1b	0.090178
3	qwen2:0.5b	0.119815
3	smollm2:360m	0.206051
4	gemma3:1b	0.342165
4	granite3.1-moe:1b	0.110445
4	qwen2:0.5b	0.134457
4	smollm2:360m	0.208253
5	gemma3:1b	0.349078
5	granite3.1-moe:1b	0.133652
5	qwen2:0.5b	0.155324
5	smollm2:360m	0.213184
6	gemma3:1b	0.333702
6	granite3.1-moe:1b	0.101035
6	qwen2:0.5b	0.176270

```
6       smollm2:360m                0.165891
7           gemma3:1b               0.391250
7 granite3.1-moe:1b                 0.129848
7          qwen2:0.5b               0.095614
7       smollm2:360m                0.188834
8           gemma3:1b               0.339023
8 granite3.1-moe:1b                 0.130611
8          qwen2:0.5b               0.128982
8       smollm2:360m                0.274192
```

[22]:
```python
metric_col = 'jaccard_similarity'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Jaccard Similarity", fontsize=12)
plt.title("Mean Jaccard Similarity by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Jaccard Similarity Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```

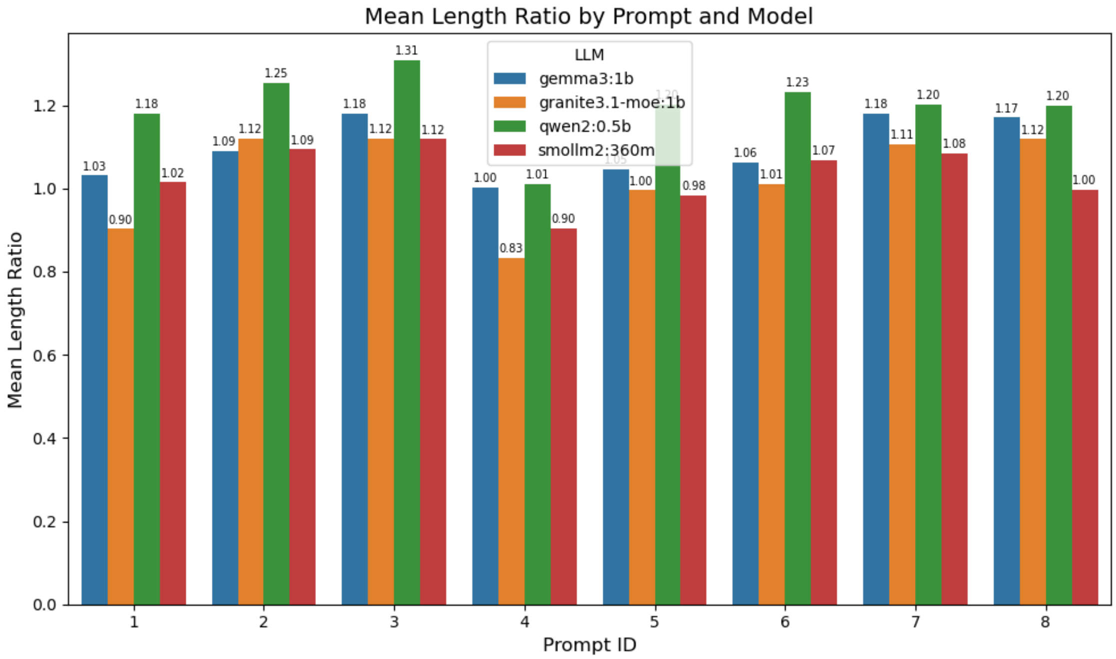


Mean Jaccard Similarity Table by Prompt and Model:

prompt_id	model	jaccard_similarity
1	gemma3:1b	0.307219
1	granite3.1-moe:1b	0.183757
1	qwen2:0.5b	0.127423
1	smollm2:360m	0.089247
2	gemma3:1b	0.412321
2	granite3.1-moe:1b	0.157192
2	qwen2:0.5b	0.153279
2	smollm2:360m	0.157645
3	gemma3:1b	0.424638
3	granite3.1-moe:1b	0.218134
3	qwen2:0.5b	0.154935
3	smollm2:360m	0.133321
4	gemma3:1b	0.329740
4	granite3.1-moe:1b	0.225126
4	qwen2:0.5b	0.133691
4	smollm2:360m	0.166607
5	gemma3:1b	0.341227
5	granite3.1-moe:1b	0.221673
5	qwen2:0.5b	0.184770
5	smollm2:360m	0.216383
6	gemma3:1b	0.295691
6	granite3.1-moe:1b	0.153952
6	qwen2:0.5b	0.145783

```
6      smollm2:360m           0.143711
7          gemma3:1b           0.359963
7 granite3.1-moe:1b           0.183441
7         qwen2:0.5b           0.096504
7      smollm2:360m           0.078461
8          gemma3:1b           0.366199
8 granite3.1-moe:1b           0.145337
8         qwen2:0.5b           0.097956
8      smollm2:360m           0.092543
```

[23]:
```python
metric_col = 'length_ratio'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Length Ratio", fontsize=12)
plt.title("Mean Length Ratio by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Length Ratio Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



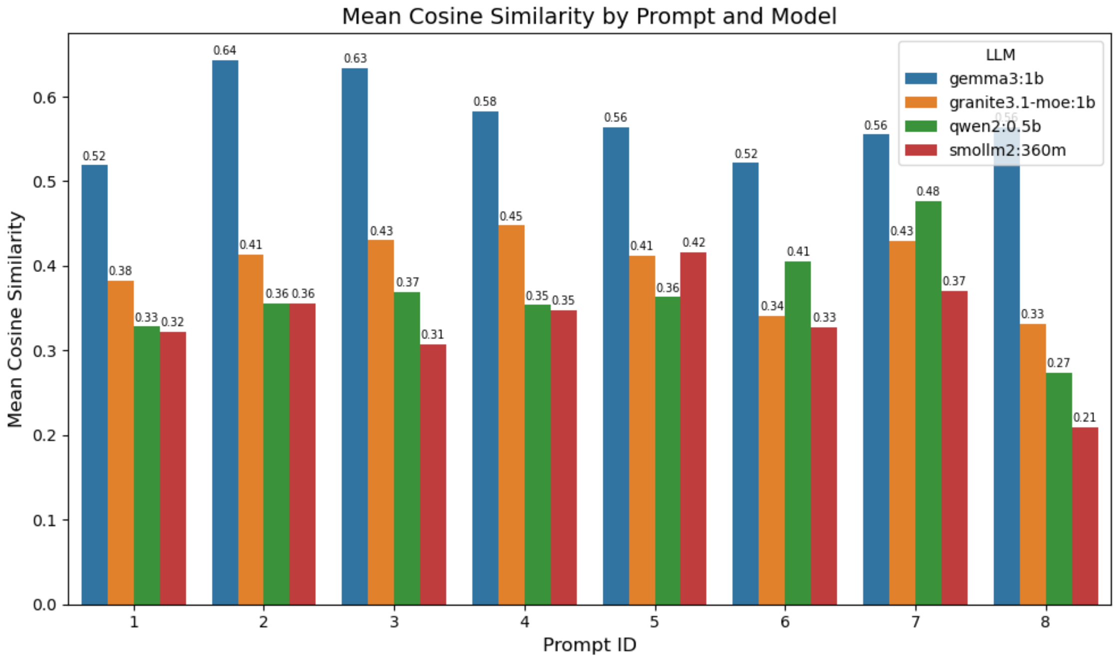
Mean Length Ratio Table by Prompt and Model:

prompt_id	model	length_ratio
1	gemma3:1b	1.031050
1	granite3.1-moe:1b	0.902465
1	qwen2:0.5b	1.180218
1	smollm2:360m	1.015045
2	gemma3:1b	1.090909
2	granite3.1-moe:1b	1.121014
2	qwen2:0.5b	1.253623
2	smollm2:360m	1.094203
3	gemma3:1b	1.181061
3	granite3.1-moe:1b	1.120000
3	qwen2:0.5b	1.307576
3	smollm2:360m	1.118561
4	gemma3:1b	1.002674
4	granite3.1-moe:1b	0.832026
4	qwen2:0.5b	1.011586
4	smollm2:360m	0.903446
5	gemma3:1b	1.045455
5	granite3.1-moe:1b	0.995522
5	qwen2:0.5b	1.201832
5	smollm2:360m	0.983039
6	gemma3:1b	1.062328
6	granite3.1-moe:1b	1.011742
6	qwen2:0.5b	1.231061

```
6      smollm2:360m      1.068526
7          gemma3:1b      1.179189
7 granite3.1-moe:1b      1.107025
7         qwen2:0.5b      1.202855
7      smollm2:360m      1.084899
8          gemma3:1b      1.170641
8 granite3.1-moe:1b      1.118852
8         qwen2:0.5b      1.198584
8      smollm2:360m      0.996647
```

[24]:

```python
metric_col = 'cosine_similarity'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Cosine Similarity", fontsize=12)
plt.title("Mean Cosine Similarity by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Cosine Similarity Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



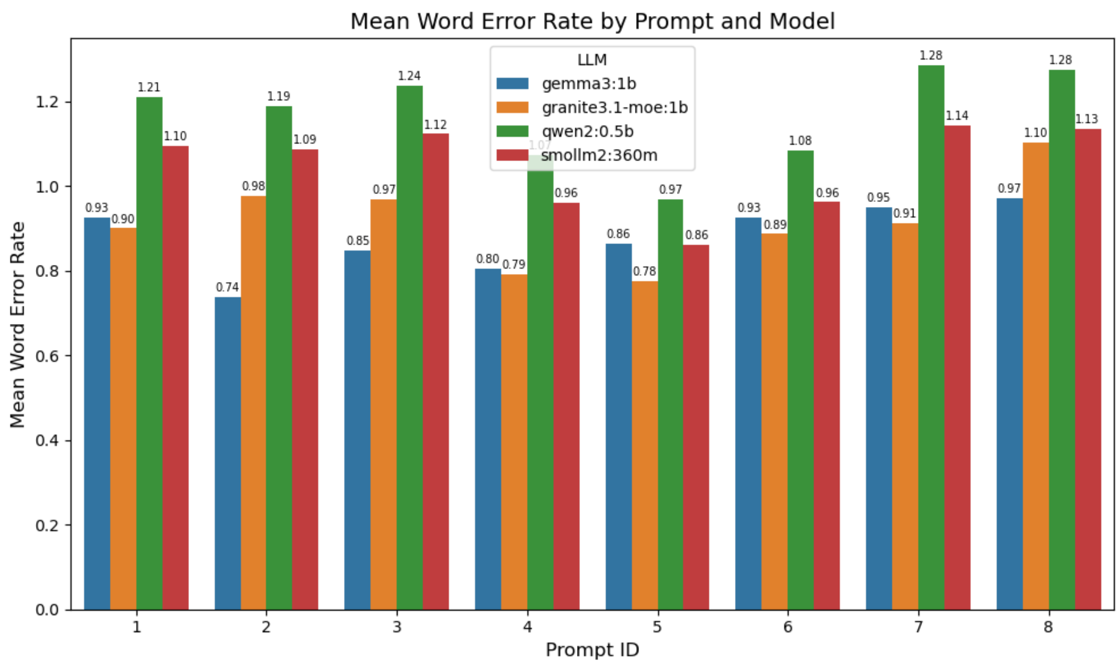

```
Mean Cosine Similarity Table by Prompt and Model:
```

prompt_id	model	cosine_similarity
1	gemma3:1b	0.518970
1	granite3.1-moe:1b	0.382670
1	qwen2:0.5b	0.328296
1	smollm2:360m	0.322072
2	gemma3:1b	0.642691
2	granite3.1-moe:1b	0.413292
2	qwen2:0.5b	0.355949
2	smollm2:360m	0.355266
3	gemma3:1b	0.633511
3	granite3.1-moe:1b	0.430231
3	qwen2:0.5b	0.369297
3	smollm2:360m	0.307824
4	gemma3:1b	0.582984
4	granite3.1-moe:1b	0.447602
4	qwen2:0.5b	0.353659
4	smollm2:360m	0.347627
5	gemma3:1b	0.564296
5	granite3.1-moe:1b	0.412355
5	qwen2:0.5b	0.363213
5	smollm2:360m	0.415689
6	gemma3:1b	0.521634
6	granite3.1-moe:1b	0.341044
6	qwen2:0.5b	0.405859

```
6       smollm2:360m            0.327513
7          gemma3:1b            0.555280
7 granite3.1-moe:1b            0.429507
7         qwen2:0.5b            0.477016
7       smollm2:360m            0.370274
8          gemma3:1b            0.563831
8 granite3.1-moe:1b            0.331496
8         qwen2:0.5b            0.273568
8       smollm2:360m            0.208965
```

[25]:

```
metric_col = 'wer'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Word Error Rate", fontsize=12)
plt.title("Mean Word Error Rate by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Word Error Rate Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



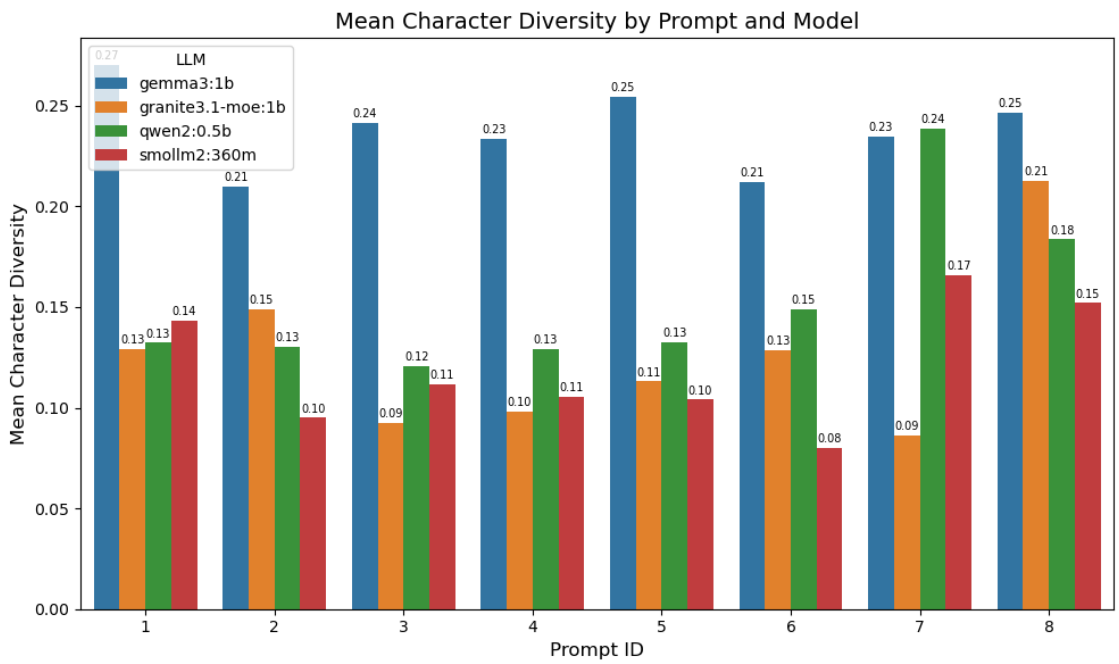
Mean Word Error Rate Table by Prompt and Model:

prompt_id	model	wer
1	gemma3:1b	0.926290
1	granite3.1-moe:1b	0.900000
1	qwen2:0.5b	1.211302
1	smollm2:360m	1.095823
2	gemma3:1b	0.736842
2	granite3.1-moe:1b	0.976316
2	qwen2:0.5b	1.188995
2	smollm2:360m	1.086124
3	gemma3:1b	0.847594
3	granite3.1-moe:1b	0.967647
3	qwen2:0.5b	1.237968
3	smollm2:360m	1.122995
4	gemma3:1b	0.804545
4	granite3.1-moe:1b	0.792500
4	qwen2:0.5b	1.072727
4	smollm2:360m	0.961364
5	gemma3:1b	0.863636
5	granite3.1-moe:1b	0.776190
5	qwen2:0.5b	0.969697
5	smollm2:360m	0.861472
6	gemma3:1b	0.925837
6	granite3.1-moe:1b	0.886842
6	qwen2:0.5b	1.083732

```
6      smollm2:360m 0.961722
7          gemma3:1b 0.950147
7 granite3.1-moe:1b 0.912903
7         qwen2:0.5b 1.284457
7      smollm2:360m 1.143695
8          gemma3:1b 0.970674
8 granite3.1-moe:1b 1.103226
8         qwen2:0.5b 1.275660
8      smollm2:360m 1.134897
```

[26]:

```
metric_col = 'char_diversity'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Character Diversity", fontsize=12)
plt.title("Mean Character Diversity by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Character Diversity Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



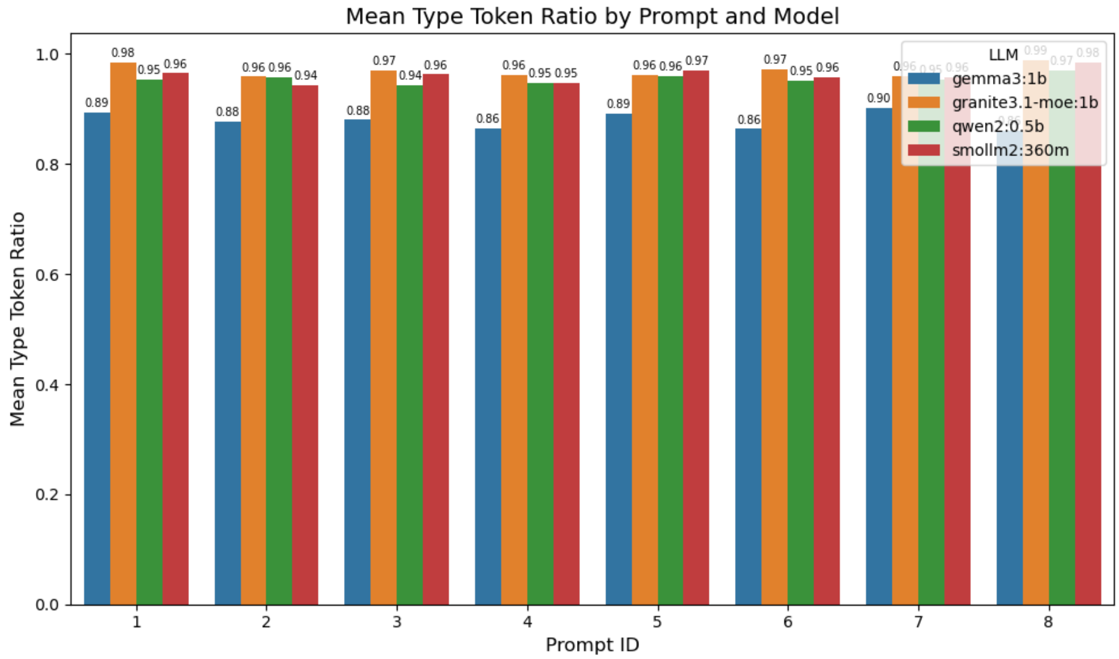
Mean Character Diversity Table by Prompt and Model:

prompt_id	model	char_diversity
1	gemma3:1b	0.270020
1	granite3.1-moe:1b	0.129324
1	qwen2:0.5b	0.132288
1	smollm2:360m	0.143191
2	gemma3:1b	0.209836
2	granite3.1-moe:1b	0.148893
2	qwen2:0.5b	0.130082
2	smollm2:360m	0.095010
3	gemma3:1b	0.241285
3	granite3.1-moe:1b	0.092397
3	qwen2:0.5b	0.120813
3	smollm2:360m	0.111742
4	gemma3:1b	0.233601
4	granite3.1-moe:1b	0.098295
4	qwen2:0.5b	0.128973
4	smollm2:360m	0.105373
5	gemma3:1b	0.254620
5	granite3.1-moe:1b	0.113102
5	qwen2:0.5b	0.132592
5	smollm2:360m	0.103992
6	gemma3:1b	0.212050
6	granite3.1-moe:1b	0.128421
6	qwen2:0.5b	0.149011

```
6      smollm2:360m        0.079962
7          gemma3:1b        0.234534
7 granite3.1-moe:1b        0.086330
7         qwen2:0.5b        0.238583
7      smollm2:360m        0.165616
8          gemma3:1b        0.246564
8 granite3.1-moe:1b        0.212882
8         qwen2:0.5b        0.183565
8      smollm2:360m        0.151941
```

[27]:

```
metric_col = 'type_token_ratio'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Type Token Ratio", fontsize=12)
plt.title("Mean Type Token Ratio by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Type Token Ratio Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



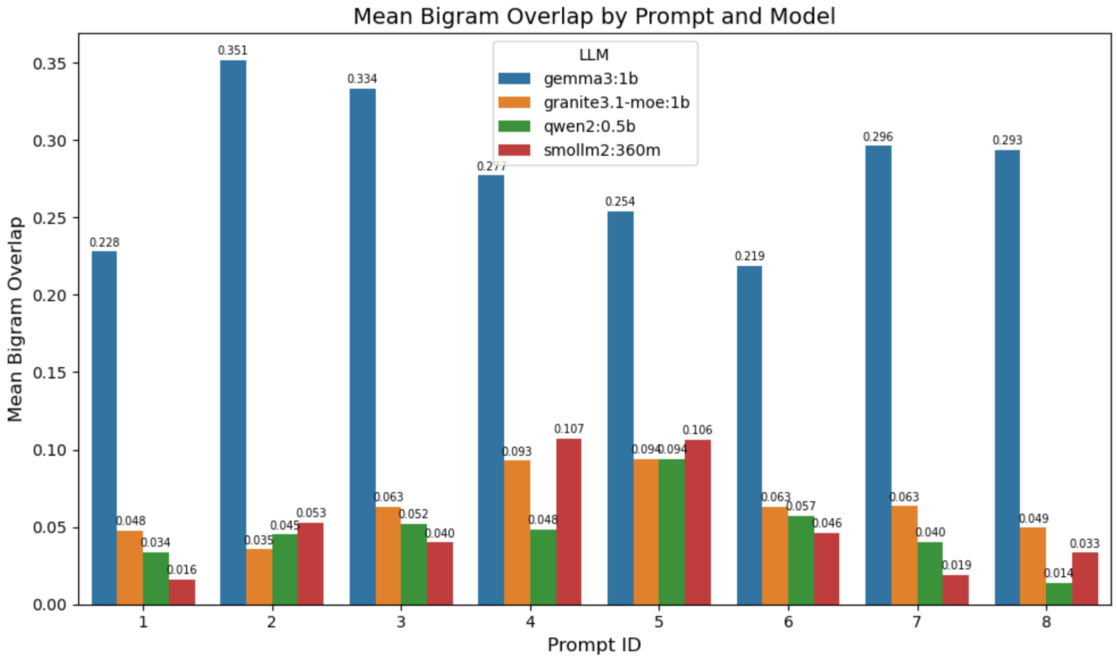
Mean Type Token Ratio Table by Prompt and Model:

prompt_id	model	type_token_ratio
1	gemma3:1b	0.893512
1	granite3.1-moe:1b	0.984078
1	qwen2:0.5b	0.954470
1	smollm2:360m	0.964942
2	gemma3:1b	0.877512
2	granite3.1-moe:1b	0.958874
2	qwen2:0.5b	0.957332
2	smollm2:360m	0.943816
3	gemma3:1b	0.880468
3	granite3.1-moe:1b	0.969680
3	qwen2:0.5b	0.942743
3	smollm2:360m	0.963048
4	gemma3:1b	0.864202
4	granite3.1-moe:1b	0.962176
4	qwen2:0.5b	0.948129
4	smollm2:360m	0.948189
5	gemma3:1b	0.890984
5	granite3.1-moe:1b	0.962562
5	qwen2:0.5b	0.959796
5	smollm2:360m	0.970213
6	gemma3:1b	0.863806
6	granite3.1-moe:1b	0.972466
6	qwen2:0.5b	0.952460

```
6      smollm2:360m           0.958267
7          gemma3:1b          0.901537
7 granite3.1-moe:1b           0.959901
7         qwen2:0.5b          0.953417
7      smollm2:360m           0.957546
8          gemma3:1b          0.860478
8 granite3.1-moe:1b           0.988074
8         qwen2:0.5b          0.969944
8      smollm2:360m           0.984129
```

[28]:

```
metric_col = 'bigram_overlap'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.3f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Bigram Overlap", fontsize=12)
plt.title("Mean Bigram Overlap by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Bigram Overlap Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



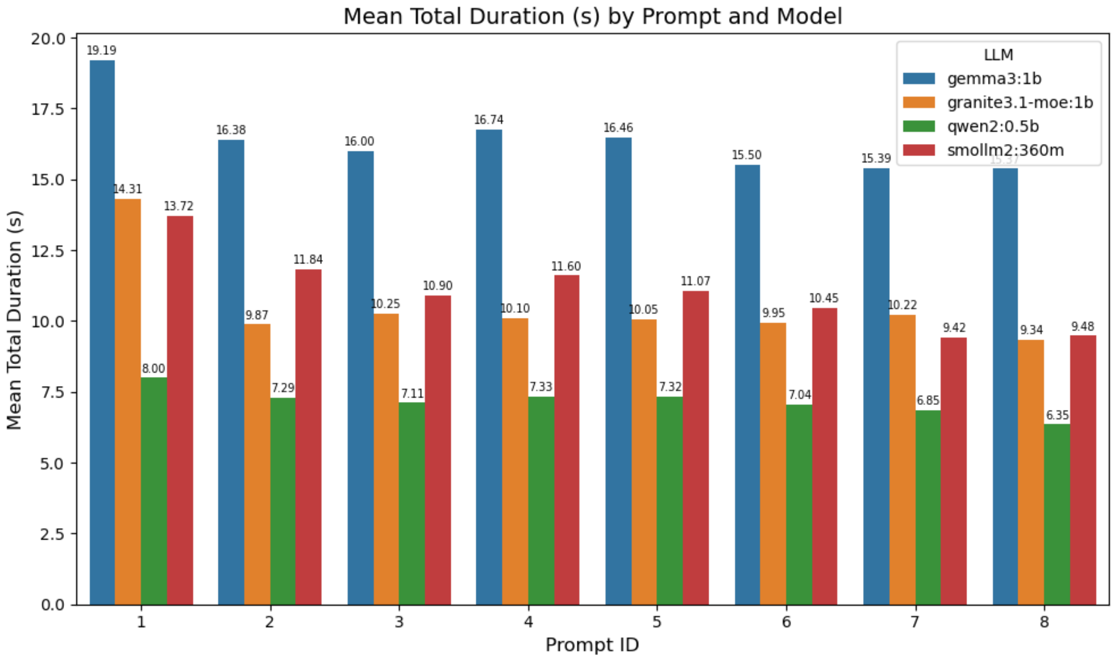
Mean Bigram Overlap Table by Prompt and Model:

prompt_id	model	bigram_overlap
1	gemma3:1b	0.227948
1	granite3.1-moe:1b	0.047773
1	qwen2:0.5b	0.033781
1	smollm2:360m	0.016246
2	gemma3:1b	0.351405
2	granite3.1-moe:1b	0.035467
2	qwen2:0.5b	0.045038
2	smollm2:360m	0.052751
3	gemma3:1b	0.333533
3	granite3.1-moe:1b	0.062834
3	qwen2:0.5b	0.052062
3	smollm2:360m	0.039926
4	gemma3:1b	0.277089
4	granite3.1-moe:1b	0.092721
4	qwen2:0.5b	0.048092
4	smollm2:360m	0.107319
5	gemma3:1b	0.254168
5	granite3.1-moe:1b	0.093665
5	qwen2:0.5b	0.094014
5	smollm2:360m	0.106278
6	gemma3:1b	0.218936
6	granite3.1-moe:1b	0.063029
6	qwen2:0.5b	0.057376

```
6      smollm2:360m        0.046410
7          gemma3:1b        0.296233
7 granite3.1-moe:1b        0.063329
7         qwen2:0.5b        0.040025
7      smollm2:360m        0.018833
8          gemma3:1b        0.293477
8 granite3.1-moe:1b        0.049417
8         qwen2:0.5b        0.013744
8      smollm2:360m        0.033360
```

[29]:

```
metric_col = 'total_duration_s'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Total Duration (s)", fontsize=12)
plt.title("Mean Total Duration (s) by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Total Duration (s) Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



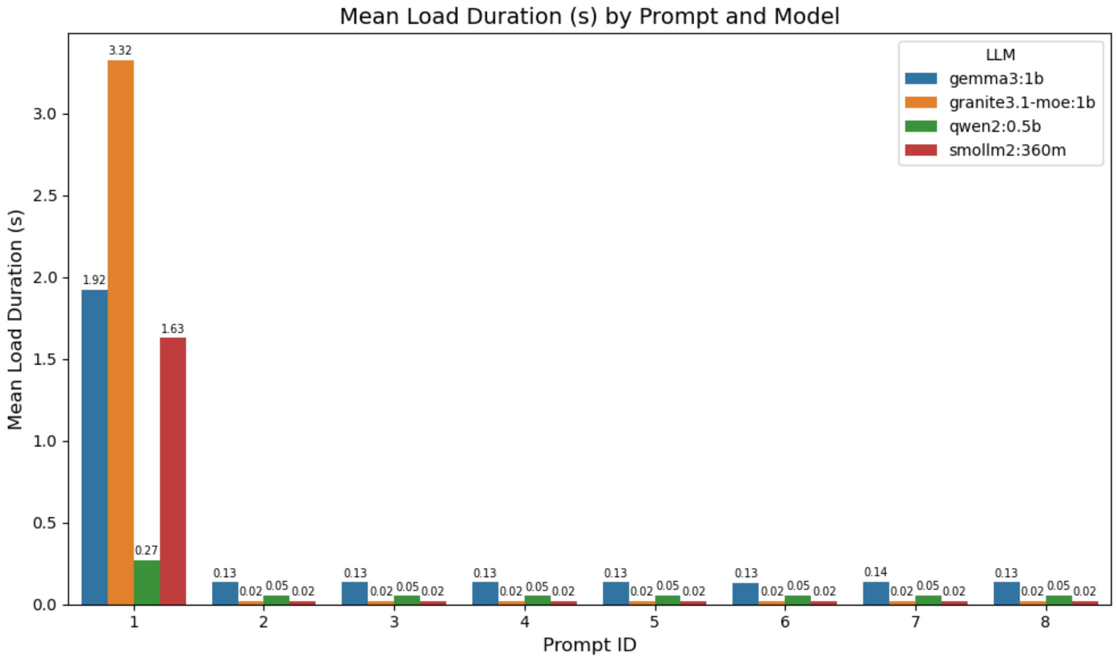
Mean Total Duration (s) Table by Prompt and Model:

prompt_id	model	total_duration_s
1	gemma3:1b	19.188144
1	granite3.1-moe:1b	14.308235
1	qwen2:0.5b	7.998199
1	smollm2:360m	13.717496
2	gemma3:1b	16.380431
2	granite3.1-moe:1b	9.874609
2	qwen2:0.5b	7.288003
2	smollm2:360m	11.835116
3	gemma3:1b	16.000983
3	granite3.1-moe:1b	10.246667
3	qwen2:0.5b	7.114733
3	smollm2:360m	10.895771
4	gemma3:1b	16.743796
4	granite3.1-moe:1b	10.095100
4	qwen2:0.5b	7.329973
4	smollm2:360m	11.602065
5	gemma3:1b	16.463278
5	granite3.1-moe:1b	10.051996
5	qwen2:0.5b	7.319335
5	smollm2:360m	11.066141
6	gemma3:1b	15.499161
6	granite3.1-moe:1b	9.947697
6	qwen2:0.5b	7.036435

```
6      smollm2:360m         10.453246
7          gemma3:1b        15.390212
7 granite3.1-moe:1b         10.223190
7         qwen2:0.5b         6.846961
7      smollm2:360m          9.418833
8          gemma3:1b        15.370792
8 granite3.1-moe:1b          9.339246
8         qwen2:0.5b         6.350768
8      smollm2:360m          9.481517
```

[30]:

```python
df['load_duration_s'] = df['load_duration_ns'] / 1e9
metric_col = 'load_duration_s'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Load Duration (s)", fontsize=12)
plt.title("Mean Load Duration (s) by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Load Duration (s) Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



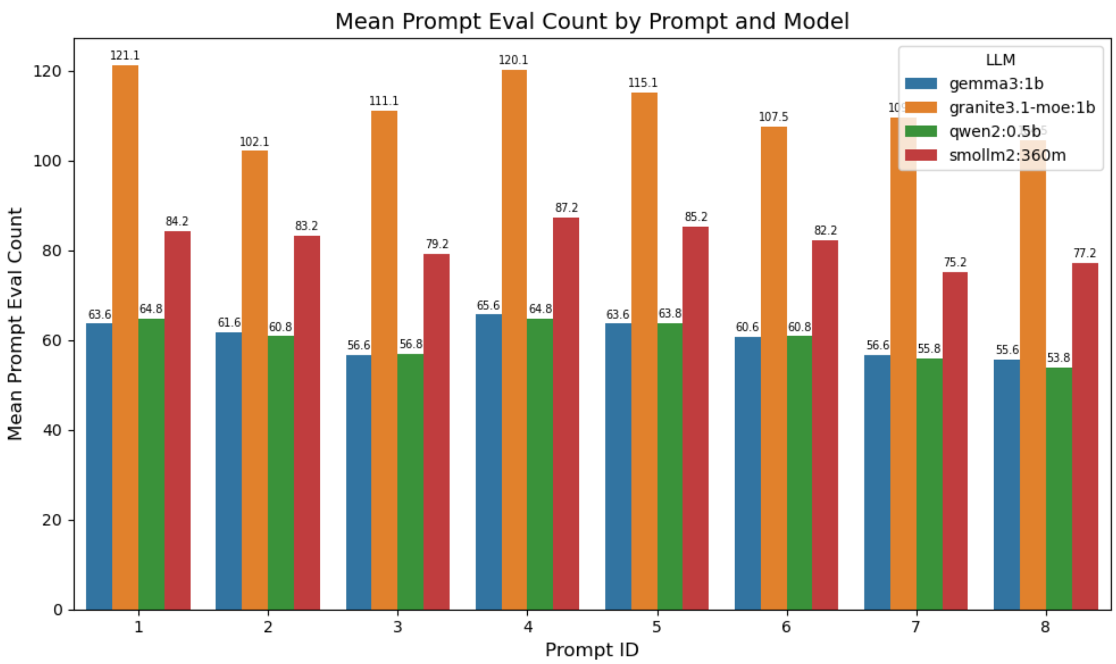
Mean Load Duration (s) Table by Prompt and Model:

prompt_id	model	load_duration_s
1	gemma3:1b	1.920152
1	granite3.1-moe:1b	3.322415
1	qwen2:0.5b	0.267021
1	smollm2:360m	1.627052
2	gemma3:1b	0.134766
2	granite3.1-moe:1b	0.021508
2	qwen2:0.5b	0.051730
2	smollm2:360m	0.021827
3	gemma3:1b	0.134548
3	granite3.1-moe:1b	0.021360
3	qwen2:0.5b	0.051673
3	smollm2:360m	0.021094
4	gemma3:1b	0.134743
4	granite3.1-moe:1b	0.020843
4	qwen2:0.5b	0.051226
4	smollm2:360m	0.021520
5	gemma3:1b	0.134060
5	granite3.1-moe:1b	0.021788
5	qwen2:0.5b	0.052652
5	smollm2:360m	0.021343
6	gemma3:1b	0.133452
6	granite3.1-moe:1b	0.021591
6	qwen2:0.5b	0.051767

```
6      smollm2:360m         0.021137
7          gemma3:1b        0.135561
7 granite3.1-moe:1b         0.021237
7         qwen2:0.5b        0.052055
7      smollm2:360m         0.020865
8          gemma3:1b        0.134019
8 granite3.1-moe:1b         0.020916
8         qwen2:0.5b        0.051736
8      smollm2:360m         0.021303
```

[31]:

```
metric_col = 'prompt_eval_count'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.1f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Prompt Eval Count", fontsize=12)
plt.title("Mean Prompt Eval Count by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Prompt Eval Count Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



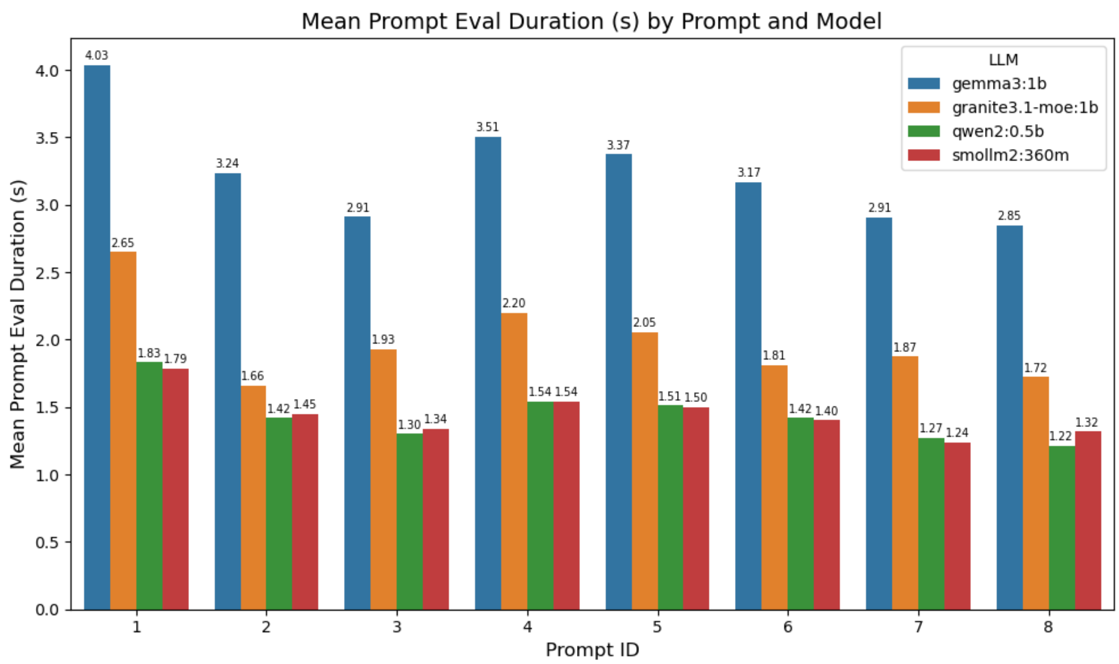
Mean Prompt Eval Count Table by Prompt and Model:

prompt_id	model	prompt_eval_count
1	gemma3:1b	63.636364
1	granite3.1-moe:1b	121.100000
1	qwen2:0.5b	64.818182
1	smollm2:360m	84.181818
2	gemma3:1b	61.636364
2	granite3.1-moe:1b	102.100000
2	qwen2:0.5b	60.818182
2	smollm2:360m	83.181818
3	gemma3:1b	56.636364
3	granite3.1-moe:1b	111.100000
3	qwen2:0.5b	56.818182
3	smollm2:360m	79.181818
4	gemma3:1b	65.636364
4	granite3.1-moe:1b	120.100000
4	qwen2:0.5b	64.818182
4	smollm2:360m	87.181818
5	gemma3:1b	63.636364
5	granite3.1-moe:1b	115.100000
5	qwen2:0.5b	63.818182
5	smollm2:360m	85.181818
6	gemma3:1b	60.636364
6	granite3.1-moe:1b	107.500000
6	qwen2:0.5b	60.818182

```
6      smollm2:360m          82.181818
7          gemma3:1b          56.636364
7 granite3.1-moe:1b         109.500000
7         qwen2:0.5b          55.818182
7      smollm2:360m          75.181818
8          gemma3:1b          55.636364
8 granite3.1-moe:1b         104.500000
8         qwen2:0.5b          53.818182
8      smollm2:360m          77.181818
```

[32]:

```
df['prompt_eval_duration_s'] = df['prompt_eval_duration_ns'] / 1e9
metric_col = 'prompt_eval_duration_s'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Prompt Eval Duration (s)", fontsize=12)
plt.title("Mean Prompt Eval Duration (s) by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Prompt Eval Duration (s) Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



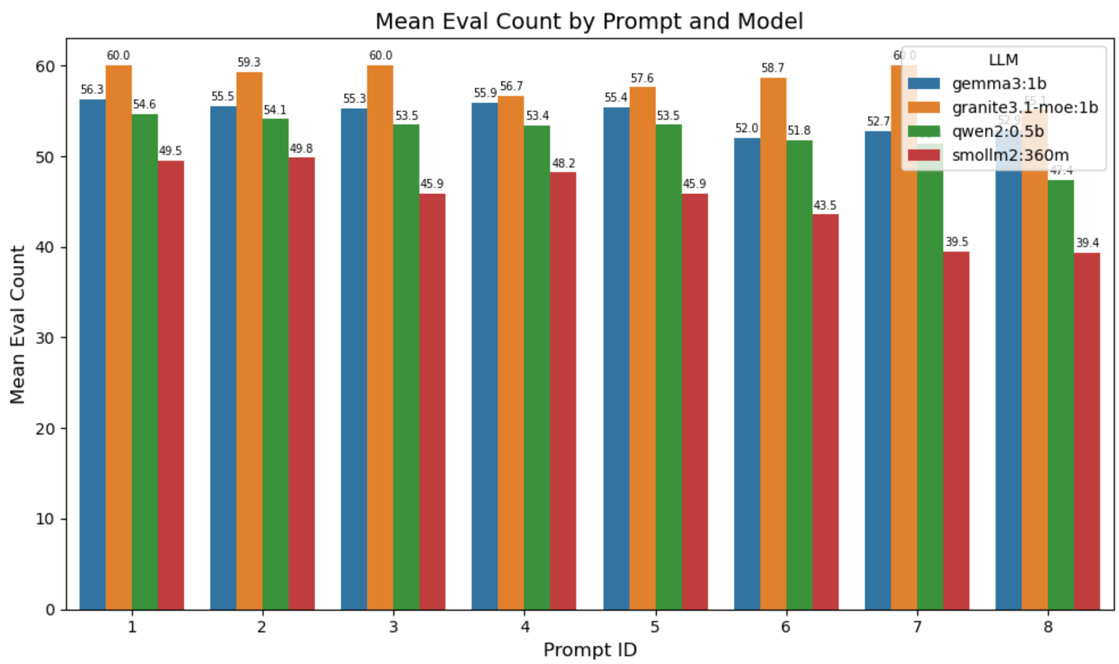
Mean Prompt Eval Duration (s) Table by Prompt and Model:

prompt_id	model	prompt_eval_duration_s
1	gemma3:1b	4.032869
1	granite3.1-moe:1b	2.649561
1	qwen2:0.5b	1.831723
1	smollm2:360m	1.786276
2	gemma3:1b	3.236039
2	granite3.1-moe:1b	1.662311
2	qwen2:0.5b	1.418149
2	smollm2:360m	1.450954
3	gemma3:1b	2.910115
3	granite3.1-moe:1b	1.926919
3	qwen2:0.5b	1.301053
3	smollm2:360m	1.338266
4	gemma3:1b	3.505857
4	granite3.1-moe:1b	2.198182
4	qwen2:0.5b	1.537797
4	smollm2:360m	1.537981
5	gemma3:1b	3.373812
5	granite3.1-moe:1b	2.052317
5	qwen2:0.5b	1.512005
5	smollm2:360m	1.497412
6	gemma3:1b	3.169544
6	granite3.1-moe:1b	1.810106
6	qwen2:0.5b	1.418276

```
6      smollm2:360m              1.402899
7          gemma3:1b             2.909144
7 granite3.1-moe:1b              1.873770
7         qwen2:0.5b             1.273654
7      smollm2:360m              1.237074
8          gemma3:1b             2.845163
8 granite3.1-moe:1b              1.721691
8         qwen2:0.5b             1.215047
8      smollm2:360m              1.317496
```

[33]:

```
metric_col = 'eval_count'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.1f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Eval Count", fontsize=12)
plt.title("Mean Eval Count by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Eval Count Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



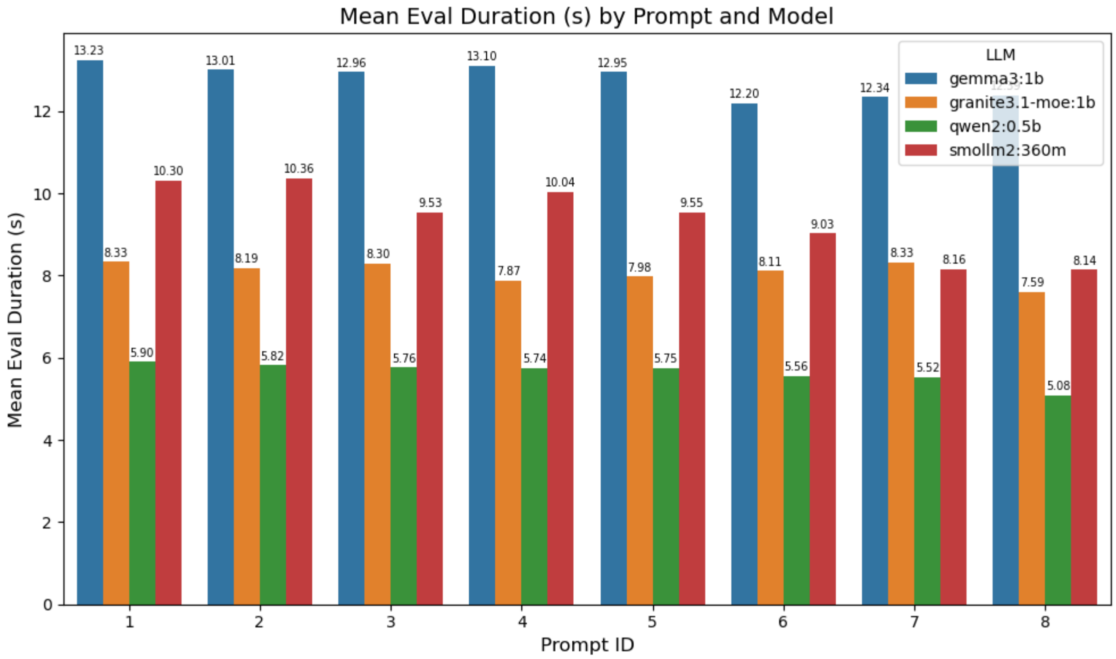
Mean Eval Count Table by Prompt and Model:

prompt_id	model	eval_count
1	gemma3:1b	56.272727
1	granite3.1-moe:1b	60.000000
1	qwen2:0.5b	54.636364
1	smollm2:360m	49.545455
2	gemma3:1b	55.545455
2	granite3.1-moe:1b	59.300000
2	qwen2:0.5b	54.090909
2	smollm2:360m	49.818182
3	gemma3:1b	55.272727
3	granite3.1-moe:1b	60.000000
3	qwen2:0.5b	53.454545
3	smollm2:360m	45.909091
4	gemma3:1b	55.909091
4	granite3.1-moe:1b	56.700000
4	qwen2:0.5b	53.363636
4	smollm2:360m	48.181818
5	gemma3:1b	55.363636
5	granite3.1-moe:1b	57.600000
5	qwen2:0.5b	53.454545
5	smollm2:360m	45.909091
6	gemma3:1b	52.000000
6	granite3.1-moe:1b	58.700000
6	qwen2:0.5b	51.818182

```
6      smollm2:360m   43.545455
7         gemma3:1b   52.727273
7 granite3.1-moe:1b   60.000000
7        qwen2:0.5b   51.363636
7      smollm2:360m   39.454545
8         gemma3:1b   52.909091
8 granite3.1-moe:1b   55.100000
8        qwen2:0.5b   47.363636
8      smollm2:360m   39.363636
```

[34]:

```
df['eval_duration_s'] = df['eval_duration_ns'] / 1e9
metric_col = 'eval_duration_s'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Eval Duration (s)", fontsize=12)
plt.title("Mean Eval Duration (s) by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Eval Duration (s) Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



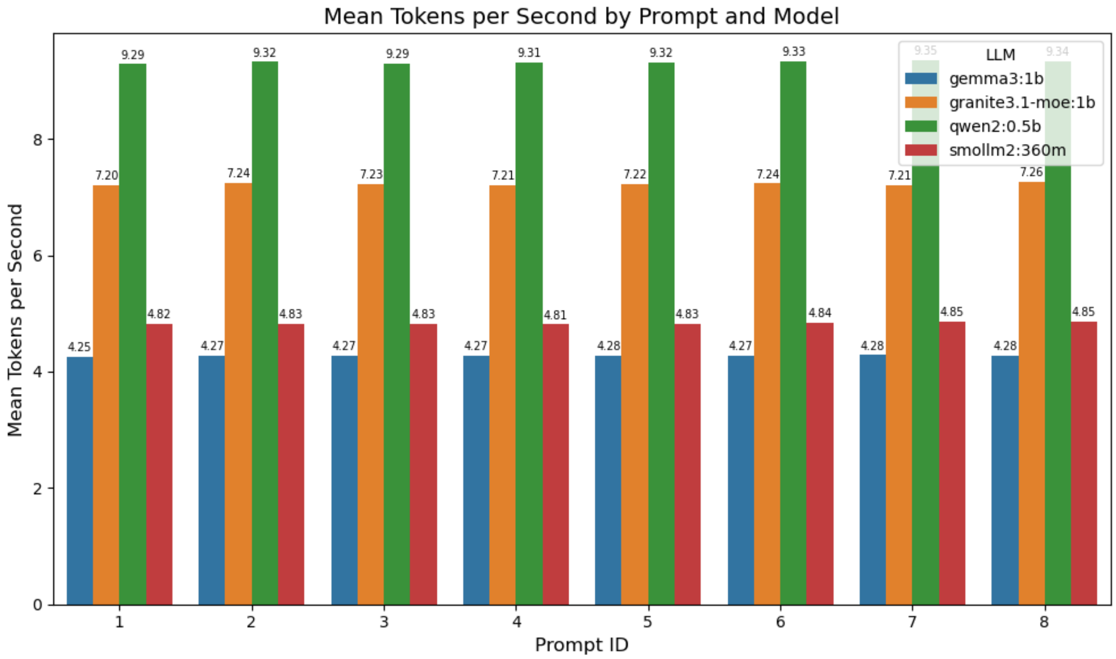
Mean Eval Duration (s) Table by Prompt and Model:

prompt_id	model	eval_duration_s
1	gemma3:1b	13.233804
1	granite3.1-moe:1b	8.333593
1	qwen2:0.5b	5.898187
1	smollm2:360m	10.302506
2	gemma3:1b	13.008460
2	granite3.1-moe:1b	8.188722
2	qwen2:0.5b	5.816410
2	smollm2:360m	10.360541
3	gemma3:1b	12.955185
3	granite3.1-moe:1b	8.296848
3	qwen2:0.5b	5.760703
3	smollm2:360m	9.534744
4	gemma3:1b	13.102083
4	granite3.1-moe:1b	7.874251
4	qwen2:0.5b	5.739488
4	smollm2:360m	10.040729
5	gemma3:1b	12.954279
5	granite3.1-moe:1b	7.976062
5	qwen2:0.5b	5.753469
5	smollm2:360m	9.545345
6	gemma3:1b	12.195011
6	granite3.1-moe:1b	8.114146
6	qwen2:0.5b	5.564658

```
  6       smollm2:360m         9.027609
  7          gemma3:1b        12.344349
  7 granite3.1-moe:1b          8.326613
  7         qwen2:0.5b         5.520027
  7       smollm2:360m         8.158739
  8          gemma3:1b        12.390459
  8 granite3.1-moe:1b          7.594581
  8         qwen2:0.5b         5.082774
  8       smollm2:360m         8.140808
```

[35]:

```
metric_col = 'tokens_per_second'
agg = df.groupby(['prompt_id', 'model'])[metric_col].mean().reset_index()
prompt_order = sorted(df['prompt_id'].unique())

plt.figure(figsize=(10,6))
bar = sns.barplot(
    data=agg,
    x='prompt_id',
    y=metric_col,
    hue='model',
    order=prompt_order,
    palette='tab10'
)
for container in bar.containers:
    bar.bar_label(container, fmt='%.2f', label_type='edge', padding=2,␣
 ↪fontsize=7)
plt.xlabel("Prompt ID", fontsize=12)
plt.ylabel("Mean Tokens per Second", fontsize=12)
plt.title("Mean Tokens per Second by Prompt and Model", fontsize=14)
plt.legend(title="LLM")
plt.tight_layout()
plt.show()

print("Mean Tokens per Second Table by Prompt and Model:\n")
print(agg.to_string(index=False))
```



Mean Tokens per Second Table by Prompt and Model:

prompt_id	model	tokens_per_second
1	gemma3:1b	4.254656
1	granite3.1-moe:1b	7.199806
1	qwen2:0.5b	9.286185
1	smollm2:360m	4.818912
2	gemma3:1b	4.274260
2	granite3.1-moe:1b	7.241827
2	qwen2:0.5b	9.324803
2	smollm2:360m	4.825393
3	gemma3:1b	4.272150
3	granite3.1-moe:1b	7.231682
3	qwen2:0.5b	9.291864
3	smollm2:360m	4.827106
4	gemma3:1b	4.271214
4	granite3.1-moe:1b	7.207941
4	qwen2:0.5b	9.314725
4	smollm2:360m	4.813005
5	gemma3:1b	4.280101
5	granite3.1-moe:1b	7.223503
5	qwen2:0.5b	9.322892
5	smollm2:360m	4.829543
6	gemma3:1b	4.274892
6	granite3.1-moe:1b	7.235559
6	qwen2:0.5b	9.331980

```
6       smollm2:360m          4.842847
7          gemma3:1b          4.283511
7 granite3.1-moe:1b          7.206529
7         qwen2:0.5b          9.347458
7       smollm2:360m          4.854049
8          gemma3:1b          4.276154
8 granite3.1-moe:1b          7.261920
8         qwen2:0.5b          9.339646
8       smollm2:360m          4.853902
```

[36]:
```python
from scipy.stats import f_oneway
import pandas as pd

metric_col = 'bleu'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (BLEU):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean BLEU across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean BLEU across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

	augmentation_type	F-statistic	p-value	Significant
0	entity_replace	12.031344	3.074649e-05	True
1	expand	0.088237	9.659122e-01	False
2	explain_simple	1.439620	2.522929e-01	False

```
3           identity   418.946728  2.337463e-23        True
4           negation     7.310758  9.090059e-04        True
5              noise     9.195276  3.905748e-04        True
6         paraphrase     1.888940  1.543782e-01       False
7       question_gen     2.234300  1.080536e-01       False
8            shuffle    24.313615  5.957550e-08        True
9          summarize     3.417430  3.085063e-02        True
10           synonym     2.062183  1.279123e-01       False


Significance Interpretation (BLEU):
Augmentation 'entity_replace': Significant difference in mean BLEU across LLMs
(p=3.075e-05).
Augmentation 'expand': No significant difference in mean BLEU across LLMs
(p=0.9659).
Augmentation 'explain_simple': No significant difference in mean BLEU across
LLMs (p=0.2523).
Augmentation 'identity': Significant difference in mean BLEU across LLMs
(p=2.337e-23).
Augmentation 'negation': Significant difference in mean BLEU across LLMs
(p=0.000909).
Augmentation 'noise': Significant difference in mean BLEU across LLMs
(p=0.0003906).
Augmentation 'paraphrase': No significant difference in mean BLEU across LLMs
(p=0.1544).
Augmentation 'question_gen': No significant difference in mean BLEU across LLMs
(p=0.1081).
Augmentation 'shuffle': Significant difference in mean BLEU across LLMs
(p=5.958e-08).
Augmentation 'summarize': Significant difference in mean BLEU across LLMs
(p=0.03085).
Augmentation 'synonym': No significant difference in mean BLEU across LLMs
(p=0.1279).
```

[37]:

```
metric_col = 'levenshtein_similarity'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
 ↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
 ↪float('nan'), 'p-value': float('nan')})
```

```python
anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Levenshtein Similarity):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean Levenshtein similarity across LLMs (p={row['p-value']:.
↪4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean Levenshtein similarity across LLMs (p={row['p-value']:.
↪4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

	augmentation_type	F-statistic	p-value	Significant
0	entity_replace	2.625844	6.997952e-02	False
1	expand	0.024014	9.948452e-01	False
2	explain_simple	1.190140	3.314228e-01	False
3	identity	132.481680	1.200935e-16	True
4	negation	2.854926	5.501458e-02	False
5	noise	14.193399	2.299913e-05	True
6	paraphrase	1.649674	2.004514e-01	False
7	question_gen	8.240009	5.109847e-04	True
8	shuffle	20.934390	2.570053e-07	True
9	summarize	31.819413	3.665627e-09	True
10	synonym	0.103508	9.573280e-01	False

```
Significance Interpretation (Levenshtein Similarity):
Augmentation 'entity_replace': No significant difference in mean Levenshtein
similarity across LLMs (p=0.06998).
Augmentation 'expand': No significant difference in mean Levenshtein similarity
across LLMs (p=0.9948).
Augmentation 'explain_simple': No significant difference in mean Levenshtein
similarity across LLMs (p=0.3314).
Augmentation 'identity': Significant difference in mean Levenshtein similarity
across LLMs (p=1.201e-16).
Augmentation 'negation': No significant difference in mean Levenshtein
similarity across LLMs (p=0.05501).
Augmentation 'noise': Significant difference in mean Levenshtein similarity
across LLMs (p=2.3e-05).
Augmentation 'paraphrase': No significant difference in mean Levenshtein
similarity across LLMs (p=0.2005).
```

```
Augmentation 'question_gen': Significant difference in mean Levenshtein
similarity across LLMs (p=0.000511).
Augmentation 'shuffle': Significant difference in mean Levenshtein similarity
across LLMs (p=2.57e-07).
Augmentation 'summarize': Significant difference in mean Levenshtein similarity
across LLMs (p=3.666e-09).
Augmentation 'synonym': No significant difference in mean Levenshtein similarity
across LLMs (p=0.9573).
```

[38]:

```python
metric_col = 'jaccard_similarity'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Jaccard Similarity):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean Jaccard similarity across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean Jaccard similarity across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

	augmentation_type	F-statistic	p-value	Significant
0	entity_replace	7.304946	9.132850e-04	True
1	expand	2.343734	9.446474e-02	False
2	explain_simple	0.829942	4.886380e-01	False
3	identity	316.158954	1.086267e-21	True
4	negation	3.746211	2.218612e-02	True
5	noise	7.561195	1.180207e-03	True
6	paraphrase	6.759171	1.429236e-03	True

```
7       question_gen     5.763377  3.678599e-03          True
8            shuffle    17.924767  1.088418e-06          True
9          summarize     3.590329  2.591959e-02          True
10           synonym     2.171286  1.136909e-01         False

Significance Interpretation (Jaccard Similarity):
Augmentation 'entity_replace': Significant difference in mean Jaccard similarity
across LLMs (p=0.0009133).
Augmentation 'expand': No significant difference in mean Jaccard similarity
across LLMs (p=0.09446).
Augmentation 'explain_simple': No significant difference in mean Jaccard
similarity across LLMs (p=0.4886).
Augmentation 'identity': Significant difference in mean Jaccard similarity
across LLMs (p=1.086e-21).
Augmentation 'negation': Significant difference in mean Jaccard similarity
across LLMs (p=0.02219).
Augmentation 'noise': Significant difference in mean Jaccard similarity across
LLMs (p=0.00118).
Augmentation 'paraphrase': Significant difference in mean Jaccard similarity
across LLMs (p=0.001429).
Augmentation 'question_gen': Significant difference in mean Jaccard similarity
across LLMs (p=0.003679).
Augmentation 'shuffle': Significant difference in mean Jaccard similarity across
LLMs (p=1.088e-06).
Augmentation 'summarize': Significant difference in mean Jaccard similarity
across LLMs (p=0.02592).
Augmentation 'synonym': No significant difference in mean Jaccard similarity
across LLMs (p=0.1137).
```

[39]:

```
metric_col = 'length_ratio'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
 ↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
 ↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)
```

```python
print("\nSignificance Interpretation (Length Ratio):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean Length Ratio across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean Length Ratio across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

```
   augmentation_type  F-statistic       p-value  Significant
0     entity_replace     2.566639  7.450275e-02        False
1             expand    19.571886  4.850757e-07         True
2     explain_simple     3.906114  1.894207e-02         True
3           identity     5.717529  3.494327e-03         True
4           negation     1.480470  2.412523e-01        False
5              noise     4.947161  8.949647e-03         True
6         paraphrase    11.201796  5.277643e-05         True
7       question_gen    30.531595  1.125467e-08         True
8            shuffle     1.587574  2.145492e-01        False
9          summarize    20.727578  2.825033e-07         True
10           synonym     1.586254  2.148594e-01        False

Significance Interpretation (Length Ratio):
Augmentation 'entity_replace': No significant difference in mean Length Ratio
across LLMs (p=0.0745).
Augmentation 'expand': Significant difference in mean Length Ratio across LLMs
(p=4.851e-07).
Augmentation 'explain_simple': Significant difference in mean Length Ratio
across LLMs (p=0.01894).
Augmentation 'identity': Significant difference in mean Length Ratio across LLMs
(p=0.003494).
Augmentation 'negation': No significant difference in mean Length Ratio across
LLMs (p=0.2413).
Augmentation 'noise': Significant difference in mean Length Ratio across LLMs
(p=0.00895).
Augmentation 'paraphrase': Significant difference in mean Length Ratio across
LLMs (p=5.278e-05).
Augmentation 'question_gen': Significant difference in mean Length Ratio across
LLMs (p=1.125e-08).
Augmentation 'shuffle': No significant difference in mean Length Ratio across
LLMs (p=0.2145).
Augmentation 'summarize': Significant difference in mean Length Ratio across
LLMs (p=2.825e-07).
```

```
Augmentation 'synonym': No significant difference in mean Length Ratio across
LLMs (p=0.2149).
```

[40]:
```python
metric_col = 'cosine_similarity'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Cosine Similarity):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean Cosine Similarity across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean Cosine Similarity across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

	augmentation_type	F-statistic	p-value	Significant
0	entity_replace	3.811782	2.078983e-02	True
1	expand	0.350688	7.889392e-01	False
2	explain_simple	0.315831	8.137917e-01	False
3	identity	88.400634	2.170128e-14	True
4	negation	1.984077	1.392127e-01	False
5	noise	6.981303	1.795858e-03	True
6	paraphrase	11.692212	3.825179e-05	True
7	question_gen	8.245998	5.087193e-04	True
8	shuffle	10.959296	6.205005e-05	True
9	summarize	4.278674	1.318119e-02	True
10	synonym	1.635780	2.035221e-01	False

```
Significance Interpretation (Cosine Similarity):
Augmentation 'entity_replace': Significant difference in mean Cosine Similarity
across LLMs (p=0.02079).
Augmentation 'expand': No significant difference in mean Cosine Similarity
across LLMs (p=0.7889).
Augmentation 'explain_simple': No significant difference in mean Cosine
Similarity across LLMs (p=0.8138).
Augmentation 'identity': Significant difference in mean Cosine Similarity across
LLMs (p=2.17e-14).
Augmentation 'negation': No significant difference in mean Cosine Similarity
across LLMs (p=0.1392).
Augmentation 'noise': Significant difference in mean Cosine Similarity across
LLMs (p=0.001796).
Augmentation 'paraphrase': Significant difference in mean Cosine Similarity
across LLMs (p=3.825e-05).
Augmentation 'question_gen': Significant difference in mean Cosine Similarity
across LLMs (p=0.0005087).
Augmentation 'shuffle': Significant difference in mean Cosine Similarity across
LLMs (p=6.205e-05).
Augmentation 'summarize': Significant difference in mean Cosine Similarity
across LLMs (p=0.01318).
Augmentation 'synonym': No significant difference in mean Cosine Similarity
across LLMs (p=0.2035).
```

[41]:
```
metric_col = 'wer'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Word Error Rate):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
```

```python
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean WER across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean WER across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

```
   augmentation_type  F-statistic       p-value  Significant
0     entity_replace     4.048903  1.646887e-02         True
1             expand     5.770317  3.335208e-03         True
2     explain_simple     3.311658  3.434822e-02         True
3           identity   141.124096  5.255988e-17         True
4           negation     3.193739  3.874627e-02         True
5              noise     9.000695  4.430880e-04         True
6         paraphrase     4.997235  6.697015e-03         True
7       question_gen     9.699730  1.803215e-04         True
8            shuffle    10.068562  1.142896e-04         True
9          summarize     5.380230  4.722528e-03         True
10           synonym     8.927697  2.600684e-04         True


Significance Interpretation (Word Error Rate):
Augmentation 'entity_replace': Significant difference in mean WER across LLMs
(p=0.01647).
Augmentation 'expand': Significant difference in mean WER across LLMs
(p=0.003335).
Augmentation 'explain_simple': Significant difference in mean WER across LLMs
(p=0.03435).
Augmentation 'identity': Significant difference in mean WER across LLMs
(p=5.256e-17).
Augmentation 'negation': Significant difference in mean WER across LLMs
(p=0.03875).
Augmentation 'noise': Significant difference in mean WER across LLMs
(p=0.0004431).
Augmentation 'paraphrase': Significant difference in mean WER across LLMs
(p=0.006697).
Augmentation 'question_gen': Significant difference in mean WER across LLMs
(p=0.0001803).
Augmentation 'shuffle': Significant difference in mean WER across LLMs
(p=0.0001143).
Augmentation 'summarize': Significant difference in mean WER across LLMs
(p=0.004723).
Augmentation 'synonym': Significant difference in mean WER across LLMs
(p=0.0002601).
```

[42]:

```
metric_col = 'char_diversity'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Character Diversity):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean Character Diversity across LLMs (p={row['p-value']:.4g}).
↪")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean Character Diversity across LLMs (p={row['p-value']:.4g}).
↪")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

	augmentation_type	F-statistic	p-value	Significant
0	entity_replace	9.366203	1.885479e-04	True
1	expand	15.506217	3.904275e-06	True
2	explain_simple	6.145506	2.404549e-03	True
3	identity	7.637759	6.995811e-04	True
4	negation	13.222636	1.464390e-05	True
5	noise	0.882166	4.655683e-01	False
6	paraphrase	29.943815	6.997905e-09	True
7	question_gen	46.338710	1.425922e-10	True
8	shuffle	0.392716	7.591815e-01	False
9	summarize	0.401780	7.528100e-01	False
10	synonym	11.899835	3.345085e-05	True

```
Significance Interpretation (Character Diversity):
```

```
Augmentation 'entity_replace': Significant difference in mean Character
Diversity across LLMs (p=0.0001885).
Augmentation 'expand': Significant difference in mean Character Diversity across
LLMs (p=3.904e-06).
Augmentation 'explain_simple': Significant difference in mean Character
Diversity across LLMs (p=0.002405).
Augmentation 'identity': Significant difference in mean Character Diversity
across LLMs (p=0.0006996).
Augmentation 'negation': Significant difference in mean Character Diversity
across LLMs (p=1.464e-05).
Augmentation 'noise': No significant difference in mean Character Diversity
across LLMs (p=0.4656).
Augmentation 'paraphrase': Significant difference in mean Character Diversity
across LLMs (p=6.998e-09).
Augmentation 'question_gen': Significant difference in mean Character Diversity
across LLMs (p=1.426e-10).
Augmentation 'shuffle': No significant difference in mean Character Diversity
across LLMs (p=0.7592).
Augmentation 'summarize': No significant difference in mean Character Diversity
across LLMs (p=0.7528).
Augmentation 'synonym': Significant difference in mean Character Diversity
across LLMs (p=3.345e-05).
```

[43]:
```
metric_col = 'type_token_ratio'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Type Token Ratio):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean Type Token Ratio across LLMs (p={row['p-value']:.4g}).")
```

```
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean Type Token Ratio across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

```
   augmentation_type  F-statistic       p-value  Significant
0     entity_replace    20.346446  3.368696e-07         True
1             expand    33.260904  2.273976e-09         True
2     explain_simple    14.251822  7.948207e-06         True
3           identity    12.132673  2.882260e-05         True
4           negation    18.305278  8.993081e-07         True
5              noise     4.669504  1.134595e-02         True
6         paraphrase     9.965862  1.228372e-04         True
7       question_gen    36.764313  1.681870e-09         True
8            shuffle     7.630062  7.038690e-04         True
9          summarize     1.061596  3.811208e-01        False
10           synonym    10.262494  9.983533e-05         True


Significance Interpretation (Type Token Ratio):
Augmentation 'entity_replace': Significant difference in mean Type Token Ratio
across LLMs (p=3.369e-07).
Augmentation 'expand': Significant difference in mean Type Token Ratio across
LLMs (p=2.274e-09).
Augmentation 'explain_simple': Significant difference in mean Type Token Ratio
across LLMs (p=7.948e-06).
Augmentation 'identity': Significant difference in mean Type Token Ratio across
LLMs (p=2.882e-05).
Augmentation 'negation': Significant difference in mean Type Token Ratio across
LLMs (p=8.993e-07).
Augmentation 'noise': Significant difference in mean Type Token Ratio across
LLMs (p=0.01135).
Augmentation 'paraphrase': Significant difference in mean Type Token Ratio
across LLMs (p=0.0001228).
Augmentation 'question_gen': Significant difference in mean Type Token Ratio
across LLMs (p=1.682e-09).
Augmentation 'shuffle': Significant difference in mean Type Token Ratio across
LLMs (p=0.0007039).
Augmentation 'summarize': No significant difference in mean Type Token Ratio
across LLMs (p=0.3811).
Augmentation 'synonym': Significant difference in mean Type Token Ratio across
LLMs (p=9.984e-05).
```

[44]:
```
metric_col = 'bigram_overlap'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
```

```
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
 ↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
 ↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Bigram Overlap):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
 ↪difference in mean Bigram Overlap across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
 ↪difference in mean Bigram Overlap across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
 ↪for ANOVA.")
```

	augmentation_type	F-statistic	p-value	Significant
0	entity_replace	8.484103	3.627449e-04	True
1	expand	0.573622	6.370908e-01	False
2	explain_simple	1.664026	1.973288e-01	False
3	identity	645.730192	6.116771e-26	True
4	negation	6.273466	2.154121e-03	True
5	noise	9.714903	2.807608e-04	True
6	paraphrase	2.946594	5.000476e-02	False
7	question_gen	2.092720	1.256118e-01	False
8	shuffle	23.042056	1.014598e-07	True
9	summarize	2.147089	1.166971e-01	False
10	synonym	2.112927	1.210820e-01	False

```
Significance Interpretation (Bigram Overlap):
Augmentation 'entity_replace': Significant difference in mean Bigram Overlap
across LLMs (p=0.0003627).
Augmentation 'expand': No significant difference in mean Bigram Overlap across
LLMs (p=0.6371).
Augmentation 'explain_simple': No significant difference in mean Bigram Overlap
```

```
across LLMs (p=0.1973).
Augmentation 'identity': Significant difference in mean Bigram Overlap across
LLMs (p=6.117e-26).
Augmentation 'negation': Significant difference in mean Bigram Overlap across
LLMs (p=0.002154).
Augmentation 'noise': Significant difference in mean Bigram Overlap across LLMs
(p=0.0002808).
Augmentation 'paraphrase': No significant difference in mean Bigram Overlap
across LLMs (p=0.05).
Augmentation 'question_gen': No significant difference in mean Bigram Overlap
across LLMs (p=0.1256).
Augmentation 'shuffle': Significant difference in mean Bigram Overlap across
LLMs (p=1.015e-07).
Augmentation 'summarize': No significant difference in mean Bigram Overlap
across LLMs (p=0.1167).
Augmentation 'synonym': No significant difference in mean Bigram Overlap across
LLMs (p=0.1211).
```

[45]:

```
metric_col = 'total_duration_s'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Total Duration (s)):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean total duration (s) across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean total duration (s) across LLMs (p={row['p-value']:.4g}).")
    else:
```

```
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

```
   augmentation_type  F-statistic       p-value  Significant
0     entity_replace    88.881725  2.026440e-14         True
1             expand  2069.061530  5.865327e-33         True
2     explain_simple  1272.918072  5.061652e-30         True
3           identity    19.903110  4.145660e-07         True
4           negation   136.157393  8.400300e-17         True
5              noise    44.356838  1.690792e-09         True
6         paraphrase     2.900621  5.245446e-02        False
7       question_gen   349.253167  4.069736e-21         True
8            shuffle    14.522880  6.795688e-06         True
9          summarize     2.706015  6.430778e-02        False
10           synonym    70.628896  3.564170e-13         True


Significance Interpretation (Total Duration (s)):
Augmentation 'entity_replace': Significant difference in mean total duration (s)
across LLMs (p=2.026e-14).
Augmentation 'expand': Significant difference in mean total duration (s) across
LLMs (p=5.865e-33).
Augmentation 'explain_simple': Significant difference in mean total duration (s)
across LLMs (p=5.062e-30).
Augmentation 'identity': Significant difference in mean total duration (s)
across LLMs (p=4.146e-07).
Augmentation 'negation': Significant difference in mean total duration (s)
across LLMs (p=8.4e-17).
Augmentation 'noise': Significant difference in mean total duration (s) across
LLMs (p=1.691e-09).
Augmentation 'paraphrase': No significant difference in mean total duration (s)
across LLMs (p=0.05245).
Augmentation 'question_gen': Significant difference in mean total duration (s)
across LLMs (p=4.07e-21).
Augmentation 'shuffle': Significant difference in mean total duration (s) across
LLMs (p=6.796e-06).
Augmentation 'summarize': No significant difference in mean total duration (s)
across LLMs (p=0.06431).
Augmentation 'synonym': Significant difference in mean total duration (s) across
LLMs (p=3.564e-13).
```

[46]:
```
metric_col = 'prompt_eval_count'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
```

```python
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Prompt Eval Count):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean Prompt Eval Count across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean Prompt Eval Count across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

	augmentation_type	F-statistic	p-value	Significant
0	entity_replace	184.551363	1.519865e-18	True
1	expand	184.551363	1.519865e-18	True
2	explain_simple	187.784906	1.206164e-18	True
3	identity	183.820126	1.602289e-18	True
4	negation	181.532495	1.892597e-18	True
5	noise	109.941443	2.146687e-13	True
6	paraphrase	183.820126	1.602289e-18	True
7	question_gen	180.483303	1.606188e-17	True
8	shuffle	194.251992	7.681570e-19	True
9	summarize	187.677568	1.215382e-18	True
10	synonym	191.588679	9.233930e-19	True

```
Significance Interpretation (Prompt Eval Count):
Augmentation 'entity_replace': Significant difference in mean Prompt Eval Count
across LLMs (p=1.52e-18).
Augmentation 'expand': Significant difference in mean Prompt Eval Count across
LLMs (p=1.52e-18).
Augmentation 'explain_simple': Significant difference in mean Prompt Eval Count
across LLMs (p=1.206e-18).
Augmentation 'identity': Significant difference in mean Prompt Eval Count across
LLMs (p=1.602e-18).
Augmentation 'negation': Significant difference in mean Prompt Eval Count across
```

```
LLMs (p=1.893e-18).
Augmentation 'noise': Significant difference in mean Prompt Eval Count across
LLMs (p=2.147e-13).
Augmentation 'paraphrase': Significant difference in mean Prompt Eval Count
across LLMs (p=1.602e-18).
Augmentation 'question_gen': Significant difference in mean Prompt Eval Count
across LLMs (p=1.606e-17).
Augmentation 'shuffle': Significant difference in mean Prompt Eval Count across
LLMs (p=7.682e-19).
Augmentation 'summarize': Significant difference in mean Prompt Eval Count
across LLMs (p=1.215e-18).
Augmentation 'synonym': Significant difference in mean Prompt Eval Count across
LLMs (p=9.234e-19).
```

[47]:
```python
metric_col = 'prompt_eval_duration_ns'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Prompt Eval Duration [ns]):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean Prompt Eval Duration (ns) across LLMs (p={row['p-value']:.
↪4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean Prompt Eval Duration (ns) across LLMs (p={row['p-value']:.
↪4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

augmentation_type	F-statistic	p-value	Significant

```
0     entity_replace    58.697083  3.386839e-12        True
1             expand    93.546078  1.060680e-14        True
2     explain_simple    58.958322  3.210801e-12        True
3           identity   117.871546  5.481998e-16        True
4           negation    90.102476  1.705638e-14        True
5              noise    92.650919  1.234457e-12        True
6         paraphrase    30.775348  5.234439e-09        True
7       question_gen    99.581542  2.236007e-14        True
8            shuffle    59.311762  2.988070e-12        True
9          summarize    81.265498  6.247786e-14        True
10           synonym    64.419395  1.099528e-12        True

Significance Interpretation (Prompt Eval Duration [ns]):
Augmentation 'entity_replace': Significant difference in mean Prompt Eval
Duration (ns) across LLMs (p=3.387e-12).
Augmentation 'expand': Significant difference in mean Prompt Eval Duration (ns)
across LLMs (p=1.061e-14).
Augmentation 'explain_simple': Significant difference in mean Prompt Eval
Duration (ns) across LLMs (p=3.211e-12).
Augmentation 'identity': Significant difference in mean Prompt Eval Duration
(ns) across LLMs (p=5.482e-16).
Augmentation 'negation': Significant difference in mean Prompt Eval Duration
(ns) across LLMs (p=1.706e-14).
Augmentation 'noise': Significant difference in mean Prompt Eval Duration (ns)
across LLMs (p=1.234e-12).
Augmentation 'paraphrase': Significant difference in mean Prompt Eval Duration
(ns) across LLMs (p=5.234e-09).
Augmentation 'question_gen': Significant difference in mean Prompt Eval Duration
(ns) across LLMs (p=2.236e-14).
Augmentation 'shuffle': Significant difference in mean Prompt Eval Duration (ns)
across LLMs (p=2.988e-12).
Augmentation 'summarize': Significant difference in mean Prompt Eval Duration
(ns) across LLMs (p=6.248e-14).
Augmentation 'synonym': Significant difference in mean Prompt Eval Duration (ns)
across LLMs (p=1.1e-12).
```

[48]:
```
metric_col = 'eval_count'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
```

```
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Eval Count):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean Eval Count across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean Eval Count across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

```
C:\Users\parth\anaconda3\Lib\site-packages\scipy\stats\_stats_py.py:4167:
ConstantInputWarning: Each of the input arrays is constant;the F statistic is
not defined or infinite
  warnings.warn(stats.ConstantInputWarning(msg))
```

	augmentation_type	F-statistic	p-value	Significant
0	entity_replace	1.449674	2.495296e-01	False
1	expand	NaN	NaN	False
2	explain_simple	NaN	NaN	False
3	identity	3.540955	2.723647e-02	True
4	negation	6.485087	1.799039e-03	True
5	noise	4.094457	1.883038e-02	True
6	paraphrase	21.558619	1.939062e-07	True
7	question_gen	88.272611	9.341158e-14	True
8	shuffle	1.874725	1.567859e-01	False
9	summarize	27.612183	1.633497e-08	True
10	synonym	7.764514	6.328400e-04	True

```
Significance Interpretation (Eval Count):
Augmentation 'entity_replace': No significant difference in mean Eval Count
across LLMs (p=0.2495).
Augmentation 'expand': Insufficient data for ANOVA.
Augmentation 'explain_simple': Insufficient data for ANOVA.
Augmentation 'identity': Significant difference in mean Eval Count across LLMs
(p=0.02724).
Augmentation 'negation': Significant difference in mean Eval Count across LLMs
(p=0.001799).
```

```
Augmentation 'noise': Significant difference in mean Eval Count across LLMs
(p=0.01883).
Augmentation 'paraphrase': Significant difference in mean Eval Count across LLMs
(p=1.939e-07).
Augmentation 'question_gen': Significant difference in mean Eval Count across
LLMs (p=9.341e-14).
Augmentation 'shuffle': No significant difference in mean Eval Count across LLMs
(p=0.1568).
Augmentation 'summarize': Significant difference in mean Eval Count across LLMs
(p=1.633e-08).
Augmentation 'synonym': Significant difference in mean Eval Count across LLMs
(p=0.0006328).
```

[49]:
```python
metric_col = 'eval_duration_ns'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
 ↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
 ↪float('nan'), 'p-value': float('nan')})

anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Eval Duration [ns]):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
 ↪difference in mean Eval Duration (ns) across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
 ↪difference in mean Eval Duration (ns) across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
 ↪for ANOVA.")
```

	augmentation_type	F-statistic	p-value	Significant
0	entity_replace	72.447491	2.605049e-13	True
1	expand	239890.205769	7.888191e-62	True
2	explain_simple	73165.378112	1.307125e-54	True

```
3          identity      11.974073  3.189430e-05         True
4          negation      99.381020  4.911820e-15         True
5             noise      31.464285  3.869458e-08         True
6        paraphrase      72.540123  2.564261e-13         True
7      question_gen     254.463979  2.211799e-19         True
8           shuffle       9.738025  1.443391e-04         True
9         summarize       2.031106  1.322907e-01        False
10          synonym      52.952672  1.156670e-11         True

Significance Interpretation (Eval Duration [ns]):
Augmentation 'entity_replace': Significant difference in mean Eval Duration (ns)
across LLMs (p=2.605e-13).
Augmentation 'expand': Significant difference in mean Eval Duration (ns) across
LLMs (p=7.888e-62).
Augmentation 'explain_simple': Significant difference in mean Eval Duration (ns)
across LLMs (p=1.307e-54).
Augmentation 'identity': Significant difference in mean Eval Duration (ns)
across LLMs (p=3.189e-05).
Augmentation 'negation': Significant difference in mean Eval Duration (ns)
across LLMs (p=4.912e-15).
Augmentation 'noise': Significant difference in mean Eval Duration (ns) across
LLMs (p=3.869e-08).
Augmentation 'paraphrase': Significant difference in mean Eval Duration (ns)
across LLMs (p=2.564e-13).
Augmentation 'question_gen': Significant difference in mean Eval Duration (ns)
across LLMs (p=2.212e-19).
Augmentation 'shuffle': Significant difference in mean Eval Duration (ns) across
LLMs (p=0.0001443).
Augmentation 'summarize': No significant difference in mean Eval Duration (ns)
across LLMs (p=0.1323).
Augmentation 'synonym': Significant difference in mean Eval Duration (ns) across
LLMs (p=1.157e-11).
```

[50]:
```
metric_col = 'tokens_per_second'
anova_results = []
for aug in sorted(df['augmentation_type'].unique()):
    subset = df[df['augmentation_type'] == aug]
    groups = [subset[subset['model'] == m][metric_col].dropna().values
              for m in subset['model'].unique()]
    if sum([len(g) > 1 for g in groups]) >= 2:
        stat, pval = f_oneway(*groups)
        anova_results.append({'augmentation_type': aug, 'F-statistic': stat,␣
↪'p-value': pval})
    else:
        anova_results.append({'augmentation_type': aug, 'F-statistic':␣
↪float('nan'), 'p-value': float('nan')})
```

```
anova_df = pd.DataFrame(anova_results)
anova_df['Significant'] = anova_df['p-value'] < 0.05
display(anova_df)

print("\nSignificance Interpretation (Tokens per Second):")
for idx, row in anova_df.iterrows():
    if pd.notna(row['p-value']):
        if row['Significant']:
            print(f"Augmentation '{row['augmentation_type']}': Significant␣
↪difference in mean Tokens per Second across LLMs (p={row['p-value']:.4g}).")
        else:
            print(f"Augmentation '{row['augmentation_type']}': No significant␣
↪difference in mean Tokens per Second across LLMs (p={row['p-value']:.4g}).")
    else:
        print(f"Augmentation '{row['augmentation_type']}': Insufficient data␣
↪for ANOVA.")
```

```
   augmentation_type    F-statistic       p-value  Significant
0     entity_replace   30649.552539  2.544171e-49         True
1             expand  198606.280785  1.109686e-60         True
2     explain_simple   84203.953624  1.828294e-55         True
3           identity   24097.032617  7.370114e-48         True
4           negation   12476.274974  7.370716e-44         True
5              noise   21347.177609  3.029594e-38         True
6         paraphrase   37493.323921  1.515228e-50         True
7       question_gen    7603.696943  2.258063e-38         True
8            shuffle   31266.724459  1.924724e-49         True
9          summarize   23577.888763  9.996357e-48         True
10           synonym   19457.175842  1.469829e-46         True

Significance Interpretation (Tokens per Second):
Augmentation 'entity_replace': Significant difference in mean Tokens per Second
across LLMs (p=2.544e-49).
Augmentation 'expand': Significant difference in mean Tokens per Second across
LLMs (p=1.11e-60).
Augmentation 'explain_simple': Significant difference in mean Tokens per Second
across LLMs (p=1.828e-55).
Augmentation 'identity': Significant difference in mean Tokens per Second across
LLMs (p=7.37e-48).
Augmentation 'negation': Significant difference in mean Tokens per Second across
LLMs (p=7.371e-44).
Augmentation 'noise': Significant difference in mean Tokens per Second across
LLMs (p=3.03e-38).
Augmentation 'paraphrase': Significant difference in mean Tokens per Second
across LLMs (p=1.515e-50).
Augmentation 'question_gen': Significant difference in mean Tokens per Second
across LLMs (p=2.258e-38).
```

```
Augmentation 'shuffle': Significant difference in mean Tokens per Second across
LLMs (p=1.925e-49).
Augmentation 'summarize': Significant difference in mean Tokens per Second
across LLMs (p=9.996e-48).
Augmentation 'synonym': Significant difference in mean Tokens per Second across
LLMs (p=1.47e-46).
```

[51]:
```python
# This code runs the Kruskal-Wallis test for each metric, grouped by␣
↪augmentation_type, and does pairwise Mann-Whitney U tests if significant.
# It prints results for each metric and group.

# Interpretation:
# If p < 0.05 in Kruskal-Wallis, at least one model is different for that␣
↪metric/augmentation.
# Mann-Whitney U will tell you which LLM pairs differ significantly.

from scipy.stats import kruskal, mannwhitneyu
from itertools import combinations
import numpy as np

metrics = [
    'levenshtein_similarity', 'jaccard_similarity', 'length_ratio', 'bleu',
    'cosine_similarity', 'wer', 'char_diversity', 'type_token_ratio',␣
↪'bigram_overlap',
    'total_duration_ns', 'load_duration_ns', 'prompt_eval_count',␣
↪'prompt_eval_duration_ns',
    'eval_count', 'eval_duration_ns', 'tokens_per_second'
]

group_var = 'augmentation_type'  # Or 'prompt_id'

for metric_col in metrics:
    print(f"\n\n==== Kruskal-Wallis and Mann-Whitney U for {metric_col} (by␣
↪{group_var}) ====")
    for group_val in sorted(df[group_var].unique()):
        sub = df[df[group_var] == group_val]
        models = sub['model'].unique()
        data = [sub[sub['model'] == m][metric_col].dropna().values for m in␣
↪models]
        if len(models) > 1:
            # Flatten all values for all groups, ignore nan
            all_values = np.concatenate([d for d in data if len(d) > 0])
            if len(all_values) == 0:
                print(f"  {group_var.capitalize()} '{group_val}': No data␣
↪available.")
                continue
            # Check if all values are identical
```

```
            if np.all(all_values == all_values[0]):
                print(f"  {group_var.capitalize()} '{group_val}': All values identical; cannot perform Kruskal-Wallis test.")
                continue
            stat, p = kruskal(*data)
            print(f"{group_var.capitalize()} '{group_val}': Kruskal-Wallis H={stat:.3f}, p={p:.4g}")
            if p < 0.05:
                print("  Significant: Pairwise comparisons:")
                for m1, m2 in combinations(models, 2):
                    d1 = sub[sub['model'] == m1][metric_col].dropna()
                    d2 = sub[sub['model'] == m2][metric_col].dropna()
                    # Mann-Whitney also needs at least one unique value
                    if len(d1) > 0 and len(d2) > 0 and (not np.all(d1 == d1.iloc[0]) or not np.all(d2 == d2.iloc[0])):
                        try:
                            u, p_mw = mannwhitneyu(d1, d2, alternative='two-sided')
                            print(f"    {m1} vs {m2}: U={u:.1f}, p={p_mw:.4g}")
                        except ValueError as e:
                            print(f"    {m1} vs {m2}: Mann-Whitney not possible ({e})")
            else:
                print("  No significant difference among LLMs.")
        else:
            print(f"{group_var.capitalize()} '{group_val}': Only one model present.")
```

```
==== Kruskal-Wallis and Mann-Whitney U for levenshtein_similarity (by
augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=6.463, p=0.09113
  No significant difference among LLMs.
Augmentation_type 'expand': Kruskal-Wallis H=0.509, p=0.917
  No significant difference among LLMs.
Augmentation_type 'explain_simple': Kruskal-Wallis H=1.656, p=0.6467
  No significant difference among LLMs.
Augmentation_type 'identity': Kruskal-Wallis H=21.072, p=0.0001017
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=17.0, p=0.1304
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0004099
    qwen2:0.5b vs granite3.1-moe:1b: U=40.0, p=0.4418
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0004099
    smollm2:360m vs granite3.1-moe:1b: U=54.0, p=0.02067
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0004099
Augmentation_type 'negation': Kruskal-Wallis H=5.327, p=0.1494
```

```
  No significant difference among LLMs.
Augmentation_type 'noise': Kruskal-Wallis H=14.105, p=0.002766
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=19.0, p=0.1949
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=6.0, p=0.7111
    smollm2:360m vs gemma3:1b: U=8.0, p=0.01041
    smollm2:360m vs granite3.1-moe:1b: U=9.0, p=0.8889
    gemma3:1b vs granite3.1-moe:1b: U=16.0, p=0.04444
Augmentation_type 'paraphrase': Kruskal-Wallis H=3.991, p=0.2624
  No significant difference among LLMs.
Augmentation_type 'question_gen': Kruskal-Wallis H=13.592, p=0.003516
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=37.0, p=0.6454
    qwen2:0.5b vs gemma3:1b: U=61.0, p=0.001088
    qwen2:0.5b vs granite3.1-moe:1b: U=36.0, p=0.1419
    smollm2:360m vs gemma3:1b: U=60.0, p=0.001865
    smollm2:360m vs granite3.1-moe:1b: U=32.0, p=0.345
    gemma3:1b vs granite3.1-moe:1b: U=8.0, p=0.04262
Augmentation_type 'shuffle': Kruskal-Wallis H=20.262, p=0.0001498
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=5.0, p=0.002953
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.000931
    qwen2:0.5b vs granite3.1-moe:1b: U=12.0, p=0.03792
    smollm2:360m vs gemma3:1b: U=1.0, p=0.001348
    smollm2:360m vs granite3.1-moe:1b: U=36.0, p=0.7209
    gemma3:1b vs granite3.1-moe:1b: U=58.0, p=0.007362
Augmentation_type 'summarize': Kruskal-Wallis H=18.702, p=0.0003151
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=9.0, p=0.01476
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=30.0, p=0.8785
    smollm2:360m vs gemma3:1b: U=18.0, p=0.1605
    smollm2:360m vs granite3.1-moe:1b: U=55.0, p=0.01476
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'synonym': Kruskal-Wallis H=2.645, p=0.4497
  No significant difference among LLMs.


==== Kruskal-Wallis and Mann-Whitney U for jaccard_similarity (by
augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=15.628, p=0.001352
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=24.0, p=0.4306
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.000931
    qwen2:0.5b vs granite3.1-moe:1b: U=14.0, p=0.06588
    smollm2:360m vs gemma3:1b: U=3.0, p=0.001088
    smollm2:360m vs granite3.1-moe:1b: U=17.0, p=0.1304
```

```
    gemma3:1b vs granite3.1-moe:1b: U=49.0, p=0.08298
Augmentation_type 'expand': Kruskal-Wallis H=6.976, p=0.07267
  No significant difference among LLMs.
Augmentation_type 'explain_simple': Kruskal-Wallis H=2.332, p=0.5064
  No significant difference among LLMs.
Augmentation_type 'identity': Kruskal-Wallis H=18.281, p=0.0003848
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=25.0, p=0.5054
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0004099
    qwen2:0.5b vs granite3.1-moe:1b: U=23.0, p=0.3823
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0004099
    smollm2:360m vs granite3.1-moe:1b: U=33.0, p=0.9591
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0004099
Augmentation_type 'negation': Kruskal-Wallis H=8.592, p=0.03524
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=41.0, p=0.3823
    qwen2:0.5b vs gemma3:1b: U=14.0, p=0.06496
    qwen2:0.5b vs granite3.1-moe:1b: U=26.5, p=0.5992
    smollm2:360m vs gemma3:1b: U=7.0, p=0.006993
    smollm2:360m vs granite3.1-moe:1b: U=20.0, p=0.2345
    gemma3:1b vs granite3.1-moe:1b: U=50.0, p=0.06496
Augmentation_type 'noise': Kruskal-Wallis H=12.904, p=0.004849
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=22.0, p=0.3282
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.000931
    qwen2:0.5b vs granite3.1-moe:1b: U=4.0, p=0.4
    smollm2:360m vs gemma3:1b: U=11.0, p=0.0312
    smollm2:360m vs granite3.1-moe:1b: U=5.0, p=0.5333
    gemma3:1b vs granite3.1-moe:1b: U=16.0, p=0.04949
Augmentation_type 'paraphrase': Kruskal-Wallis H=17.713, p=0.000504
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.000931
    qwen2:0.5b vs gemma3:1b: U=35.0, p=0.7927
    qwen2:0.5b vs granite3.1-moe:1b: U=46.0, p=0.1556
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=4.5, p=0.004427
    gemma3:1b vs granite3.1-moe:1b: U=44.0, p=0.2268
Augmentation_type 'question_gen': Kruskal-Wallis H=13.291, p=0.004047
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=46.0, p=0.1559
    qwen2:0.5b vs gemma3:1b: U=9.0, p=0.01796
    qwen2:0.5b vs granite3.1-moe:1b: U=12.0, p=0.1372
    smollm2:360m vs gemma3:1b: U=3.0, p=0.002742
    smollm2:360m vs granite3.1-moe:1b: U=5.0, p=0.01265
    gemma3:1b vs granite3.1-moe:1b: U=22.0, p=0.8463
Augmentation_type 'shuffle': Kruskal-Wallis H=12.812, p=0.005061
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=26.0, p=0.5737
```

```
    qwen2:0.5b vs gemma3:1b: U=5.0, p=0.005351
    qwen2:0.5b vs granite3.1-moe:1b: U=17.0, p=0.1304
    smollm2:360m vs gemma3:1b: U=6.0, p=0.007362
    smollm2:360m vs granite3.1-moe:1b: U=24.0, p=0.4418
    gemma3:1b vs granite3.1-moe:1b: U=56.0, p=0.01352
Augmentation_type 'summarize': Kruskal-Wallis H=10.275, p=0.01637
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=54.0, p=0.02385
    qwen2:0.5b vs gemma3:1b: U=38.0, p=0.5632
    qwen2:0.5b vs granite3.1-moe:1b: U=32.0, p=1
    smollm2:360m vs gemma3:1b: U=9.5, p=0.02058
    smollm2:360m vs granite3.1-moe:1b: U=5.0, p=0.005284
    gemma3:1b vs granite3.1-moe:1b: U=23.5, p=0.3991
Augmentation_type 'synonym': Kruskal-Wallis H=5.595, p=0.133
  No significant difference among LLMs.


==== Kruskal-Wallis and Mann-Whitney U for length_ratio (by augmentation_type)
====
Augmentation_type 'entity_replace': Kruskal-Wallis H=7.583, p=0.05546
  No significant difference among LLMs.
Augmentation_type 'expand': Kruskal-Wallis H=20.349, p=0.0001437
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=19.0, p=0.1949
    qwen2:0.5b vs gemma3:1b: U=60.0, p=0.001865
    qwen2:0.5b vs granite3.1-moe:1b: U=58.0, p=0.004662
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=63.0, p=0.0003108
    gemma3:1b vs granite3.1-moe:1b: U=30.0, p=0.8785
Augmentation_type 'explain_simple': Kruskal-Wallis H=8.957, p=0.02986
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=34.0, p=0.8785
    qwen2:0.5b vs gemma3:1b: U=54.0, p=0.02067
    qwen2:0.5b vs granite3.1-moe:1b: U=52.0, p=0.03792
    smollm2:360m vs gemma3:1b: U=51.0, p=0.04988
    smollm2:360m vs granite3.1-moe:1b: U=50.0, p=0.06496
    gemma3:1b vs granite3.1-moe:1b: U=30.0, p=0.8785
Augmentation_type 'identity': Kruskal-Wallis H=6.772, p=0.07954
  No significant difference among LLMs.
Augmentation_type 'negation': Kruskal-Wallis H=5.442, p=0.1421
  No significant difference among LLMs.
Augmentation_type 'noise': Kruskal-Wallis H=9.853, p=0.01986
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=51.5, p=0.04584
    qwen2:0.5b vs gemma3:1b: U=28.0, p=0.7209
    qwen2:0.5b vs granite3.1-moe:1b: U=11.0, p=0.5333
    smollm2:360m vs gemma3:1b: U=5.0, p=0.002953
    smollm2:360m vs granite3.1-moe:1b: U=4.0, p=0.4
```

```
    gemma3:1b vs granite3.1-moe:1b: U=16.0, p=0.04444
Augmentation_type 'paraphrase': Kruskal-Wallis H=17.026, p=0.0006982
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=62.0, p=0.0006216
    qwen2:0.5b vs gemma3:1b: U=60.0, p=0.001865
    qwen2:0.5b vs granite3.1-moe:1b: U=63.0, p=0.0003108
    smollm2:360m vs gemma3:1b: U=16.0, p=0.1049
    smollm2:360m vs granite3.1-moe:1b: U=20.0, p=0.2345
    gemma3:1b vs granite3.1-moe:1b: U=39.0, p=0.5054
Augmentation_type 'question_gen': Kruskal-Wallis H=20.380, p=0.0001416
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=28.0, p=0.7209
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=2.0, p=0.002664
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=3.0, p=0.004662
    gemma3:1b vs granite3.1-moe:1b: U=33.0, p=0.2824
Augmentation_type 'shuffle': Kruskal-Wallis H=4.444, p=0.2173
  No significant difference among LLMs.
Augmentation_type 'summarize': Kruskal-Wallis H=20.957, p=0.0001074
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=60.0, p=0.001865
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=59.0, p=0.002953
    smollm2:360m vs gemma3:1b: U=36.0, p=0.7209
    smollm2:360m vs granite3.1-moe:1b: U=12.0, p=0.03792
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0001554
Augmentation_type 'synonym': Kruskal-Wallis H=6.611, p=0.08539
  No significant difference among LLMs.


==== Kruskal-Wallis and Mann-Whitney U for bleu (by augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=16.699, p=0.000815
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=28.0, p=0.7209
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=17.0, p=0.1304
    smollm2:360m vs gemma3:1b: U=1.0, p=0.0003108
    smollm2:360m vs granite3.1-moe:1b: U=24.0, p=0.4418
    gemma3:1b vs granite3.1-moe:1b: U=58.0, p=0.004662
Augmentation_type 'expand': Kruskal-Wallis H=1.363, p=0.7142
  No significant difference among LLMs.
Augmentation_type 'explain_simple': Kruskal-Wallis H=4.384, p=0.2228
  No significant difference among LLMs.
Augmentation_type 'identity': Kruskal-Wallis H=18.515, p=0.0003443
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=22.0, p=0.3282
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0004099
```

```
    qwen2:0.5b vs granite3.1-moe:1b: U=26.0, p=0.5737
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0004099
    smollm2:360m vs granite3.1-moe:1b: U=39.0, p=0.5054
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0004099
Augmentation_type 'negation': Kruskal-Wallis H=12.764, p=0.005175
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=41.0, p=0.3823
    qwen2:0.5b vs gemma3:1b: U=13.0, p=0.04988
    qwen2:0.5b vs granite3.1-moe:1b: U=34.0, p=0.8785
    smollm2:360m vs gemma3:1b: U=1.0, p=0.0003108
    smollm2:360m vs granite3.1-moe:1b: U=20.0, p=0.2345
    gemma3:1b vs granite3.1-moe:1b: U=59.0, p=0.002953
Augmentation_type 'noise': Kruskal-Wallis H=13.708, p=0.003331
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=23.0, p=0.3823
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.000931
    qwen2:0.5b vs granite3.1-moe:1b: U=4.0, p=0.4
    smollm2:360m vs gemma3:1b: U=8.0, p=0.01352
    smollm2:360m vs granite3.1-moe:1b: U=7.0, p=0.8889
    gemma3:1b vs granite3.1-moe:1b: U=16.0, p=0.04949
Augmentation_type 'paraphrase': Kruskal-Wallis H=11.199, p=0.0107
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=54.0, p=0.02067
    qwen2:0.5b vs gemma3:1b: U=23.0, p=0.3823
    qwen2:0.5b vs granite3.1-moe:1b: U=30.0, p=0.8785
    smollm2:360m vs gemma3:1b: U=5.0, p=0.002953
    smollm2:360m vs granite3.1-moe:1b: U=8.0, p=0.01041
    gemma3:1b vs granite3.1-moe:1b: U=43.0, p=0.2786
Augmentation_type 'question_gen': Kruskal-Wallis H=9.025, p=0.02896
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=44.0, p=0.2345
    qwen2:0.5b vs gemma3:1b: U=18.0, p=0.1605
    qwen2:0.5b vs granite3.1-moe:1b: U=12.0, p=0.1419
    smollm2:360m vs gemma3:1b: U=8.0, p=0.01041
    smollm2:360m vs granite3.1-moe:1b: U=7.0, p=0.0293
    gemma3:1b vs granite3.1-moe:1b: U=21.0, p=0.7546
Augmentation_type 'shuffle': Kruskal-Wallis H=12.465, p=0.005948
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=20.0, p=0.2345
    qwen2:0.5b vs gemma3:1b: U=8.0, p=0.01352
    qwen2:0.5b vs granite3.1-moe:1b: U=13.0, p=0.04988
    smollm2:360m vs gemma3:1b: U=8.0, p=0.01352
    smollm2:360m vs granite3.1-moe:1b: U=21.0, p=0.2786
    gemma3:1b vs granite3.1-moe:1b: U=56.0, p=0.01352
Augmentation_type 'summarize': Kruskal-Wallis H=6.739, p=0.08071
  No significant difference among LLMs.
Augmentation_type 'synonym': Kruskal-Wallis H=5.062, p=0.1673
  No significant difference among LLMs.
```

```
==== Kruskal-Wallis and Mann-Whitney U for cosine_similarity (by
augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=10.707, p=0.01342
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=43.0, p=0.2786
    qwen2:0.5b vs gemma3:1b: U=15.0, p=0.08298
    qwen2:0.5b vs granite3.1-moe:1b: U=30.0, p=0.8785
    smollm2:360m vs gemma3:1b: U=2.0, p=0.0006216
    smollm2:360m vs granite3.1-moe:1b: U=16.0, p=0.1049
    gemma3:1b vs granite3.1-moe:1b: U=50.0, p=0.06496
Augmentation_type 'expand': Kruskal-Wallis H=0.474, p=0.9245
  No significant difference among LLMs.
Augmentation_type 'explain_simple': Kruskal-Wallis H=1.310, p=0.7268
  No significant difference among LLMs.
Augmentation_type 'identity': Kruskal-Wallis H=18.302, p=0.0003811
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=25.0, p=0.5054
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0004099
    qwen2:0.5b vs granite3.1-moe:1b: U=24.0, p=0.4418
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0004099
    smollm2:360m vs granite3.1-moe:1b: U=27.0, p=0.6454
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0004099
Augmentation_type 'negation': Kruskal-Wallis H=8.043, p=0.04514
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=35.0, p=0.7984
    qwen2:0.5b vs gemma3:1b: U=9.0, p=0.01476
    qwen2:0.5b vs granite3.1-moe:1b: U=30.0, p=0.8785
    smollm2:360m vs gemma3:1b: U=11.0, p=0.02813
    smollm2:360m vs granite3.1-moe:1b: U=25.0, p=0.5054
    gemma3:1b vs granite3.1-moe:1b: U=52.0, p=0.03792
Augmentation_type 'noise': Kruskal-Wallis H=14.062, p=0.002822
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=17.0, p=0.1304
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=4.0, p=0.4
    smollm2:360m vs gemma3:1b: U=10.0, p=0.02067
    smollm2:360m vs granite3.1-moe:1b: U=9.0, p=0.8889
    gemma3:1b vs granite3.1-moe:1b: U=16.0, p=0.04444
Augmentation_type 'paraphrase': Kruskal-Wallis H=18.014, p=0.0004369
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=56.0, p=0.01041
    qwen2:0.5b vs gemma3:1b: U=8.0, p=0.01041
    qwen2:0.5b vs granite3.1-moe:1b: U=43.0, p=0.2786
    smollm2:360m vs gemma3:1b: U=5.0, p=0.002953
    smollm2:360m vs granite3.1-moe:1b: U=9.0, p=0.01476
    gemma3:1b vs granite3.1-moe:1b: U=63.0, p=0.0003108
```

```
Augmentation_type 'question_gen': Kruskal-Wallis H=14.934, p=0.001874
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=45.0, p=0.1949
    qwen2:0.5b vs gemma3:1b: U=7.0, p=0.006993
    qwen2:0.5b vs granite3.1-moe:1b: U=10.0, p=0.08125
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=6.0, p=0.01998
    gemma3:1b vs granite3.1-moe:1b: U=24.0, p=1
Augmentation_type 'shuffle': Kruskal-Wallis H=12.857, p=0.004955
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=31.0, p=0.9591
    qwen2:0.5b vs gemma3:1b: U=6.0, p=0.007362
    qwen2:0.5b vs granite3.1-moe:1b: U=21.0, p=0.2786
    smollm2:360m vs gemma3:1b: U=5.0, p=0.005351
    smollm2:360m vs granite3.1-moe:1b: U=19.0, p=0.1949
    gemma3:1b vs granite3.1-moe:1b: U=57.0, p=0.01003
Augmentation_type 'summarize': Kruskal-Wallis H=10.281, p=0.01632
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=48.0, p=0.1049
    qwen2:0.5b vs gemma3:1b: U=34.0, p=0.8785
    qwen2:0.5b vs granite3.1-moe:1b: U=26.0, p=0.5737
    smollm2:360m vs gemma3:1b: U=6.0, p=0.004662
    smollm2:360m vs granite3.1-moe:1b: U=4.0, p=0.001865
    gemma3:1b vs granite3.1-moe:1b: U=21.0, p=0.2786
Augmentation_type 'synonym': Kruskal-Wallis H=5.770, p=0.1234
  No significant difference among LLMs.


==== Kruskal-Wallis and Mann-Whitney U for wer (by augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=11.207, p=0.01066
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=25.0, p=0.4942
    qwen2:0.5b vs gemma3:1b: U=35.0, p=0.7923
    qwen2:0.5b vs granite3.1-moe:1b: U=58.0, p=0.004662
    smollm2:360m vs gemma3:1b: U=41.0, p=0.3706
    smollm2:360m vs granite3.1-moe:1b: U=61.0, p=0.001088
    gemma3:1b vs granite3.1-moe:1b: U=50.0, p=0.06588
Augmentation_type 'expand': Kruskal-Wallis H=12.001, p=0.00738
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=24.5, p=0.4619
    qwen2:0.5b vs gemma3:1b: U=40.0, p=0.4306
    qwen2:0.5b vs granite3.1-moe:1b: U=59.5, p=0.004545
    smollm2:360m vs gemma3:1b: U=41.5, p=0.3439
    smollm2:360m vs granite3.1-moe:1b: U=57.0, p=0.006993
    gemma3:1b vs granite3.1-moe:1b: U=56.0, p=0.01352
Augmentation_type 'explain_simple': Kruskal-Wallis H=7.563, p=0.05596
  No significant difference among LLMs.
Augmentation_type 'identity': Kruskal-Wallis H=26.338, p=8.104e-06
```

```
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=63.0, p=0.001348
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0004099
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0004054
    smollm2:360m vs granite3.1-moe:1b: U=29.5, p=0.8334
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0004099
Augmentation_type 'negation': Kruskal-Wallis H=12.225, p=0.006649
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=29.5, p=0.8335
    qwen2:0.5b vs gemma3:1b: U=20.0, p=0.2345
    qwen2:0.5b vs granite3.1-moe:1b: U=50.5, p=0.05852
    smollm2:360m vs gemma3:1b: U=17.0, p=0.1304
    smollm2:360m vs granite3.1-moe:1b: U=52.5, p=0.03556
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'noise': Kruskal-Wallis H=13.909, p=0.003032
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=53.0, p=0.0312
    qwen2:0.5b vs gemma3:1b: U=62.0, p=0.001933
    qwen2:0.5b vs granite3.1-moe:1b: U=13.0, p=0.2667
    smollm2:360m vs gemma3:1b: U=52.0, p=0.04027
    smollm2:360m vs granite3.1-moe:1b: U=10.0, p=0.6944
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.04949
Augmentation_type 'paraphrase': Kruskal-Wallis H=12.062, p=0.007174
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=54.5, p=0.02077
    qwen2:0.5b vs gemma3:1b: U=51.0, p=0.04988
    qwen2:0.5b vs granite3.1-moe:1b: U=59.0, p=0.005351
    smollm2:360m vs gemma3:1b: U=20.5, p=0.2476
    smollm2:360m vs granite3.1-moe:1b: U=41.0, p=0.3706
    gemma3:1b vs granite3.1-moe:1b: U=51.0, p=0.05134
Augmentation_type 'question_gen': Kruskal-Wallis H=16.093, p=0.001085
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=17.5, p=0.1353
    qwen2:0.5b vs gemma3:1b: U=3.0, p=0.002742
    qwen2:0.5b vs granite3.1-moe:1b: U=30.0, p=0.4772
    smollm2:360m vs gemma3:1b: U=6.0, p=0.007232
    smollm2:360m vs granite3.1-moe:1b: U=41.0, p=0.03239
    gemma3:1b vs granite3.1-moe:1b: U=45.5, p=0.006646
Augmentation_type 'shuffle': Kruskal-Wallis H=14.024, p=0.002872
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=50.0, p=0.06496
    qwen2:0.5b vs gemma3:1b: U=61.0, p=0.002742
    qwen2:0.5b vs granite3.1-moe:1b: U=56.0, p=0.01041
    smollm2:360m vs gemma3:1b: U=52.0, p=0.04042
    smollm2:360m vs granite3.1-moe:1b: U=38.5, p=0.5283
    gemma3:1b vs granite3.1-moe:1b: U=13.0, p=0.05186
Augmentation_type 'summarize': Kruskal-Wallis H=12.826, p=0.005029
```

```
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=48.5, p=0.09265
    qwen2:0.5b vs gemma3:1b: U=63.0, p=0.001348
    qwen2:0.5b vs granite3.1-moe:1b: U=50.0, p=0.06496
    smollm2:360m vs gemma3:1b: U=59.0, p=0.005351
    smollm2:360m vs granite3.1-moe:1b: U=35.5, p=0.7525
    gemma3:1b vs granite3.1-moe:1b: U=24.0, p=0.4306
Augmentation_type 'synonym': Kruskal-Wallis H=17.847, p=0.000473
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=47.5, p=0.1146
    qwen2:0.5b vs gemma3:1b: U=15.0, p=0.0829
    qwen2:0.5b vs granite3.1-moe:1b: U=59.0, p=0.005351
    smollm2:360m vs gemma3:1b: U=4.5, p=0.004545
    smollm2:360m vs granite3.1-moe:1b: U=49.5, p=0.07399
    gemma3:1b vs granite3.1-moe:1b: U=62.5, p=0.001616


==== Kruskal-Wallis and Mann-Whitney U for char_diversity (by augmentation_type)
====
Augmentation_type 'entity_replace': Kruskal-Wallis H=16.813, p=0.0007721
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=44.0, p=0.2258
    qwen2:0.5b vs gemma3:1b: U=18.0, p=0.1559
    qwen2:0.5b vs granite3.1-moe:1b: U=58.0, p=0.007319
    smollm2:360m vs gemma3:1b: U=3.0, p=0.001088
    smollm2:360m vs granite3.1-moe:1b: U=42.5, p=0.2929
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.000931
Augmentation_type 'expand': Kruskal-Wallis H=17.293, p=0.000615
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=46.0, p=0.1559
    qwen2:0.5b vs gemma3:1b: U=7.0, p=0.01003
    qwen2:0.5b vs granite3.1-moe:1b: U=47.0, p=0.1256
    smollm2:360m vs gemma3:1b: U=0.5, p=0.001122
    smollm2:360m vs granite3.1-moe:1b: U=35.0, p=0.7923
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0009069
Augmentation_type 'explain_simple': Kruskal-Wallis H=13.851, p=0.003116
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=34.5, p=0.8335
    qwen2:0.5b vs gemma3:1b: U=8.0, p=0.01345
    qwen2:0.5b vs granite3.1-moe:1b: U=22.5, p=0.3442
    smollm2:360m vs gemma3:1b: U=2.5, p=0.002271
    smollm2:360m vs granite3.1-moe:1b: U=13.0, p=0.04988
    gemma3:1b vs granite3.1-moe:1b: U=57.0, p=0.009808
Augmentation_type 'identity': Kruskal-Wallis H=18.517, p=0.0003441
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=41.0, p=0.3696
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0004054
    qwen2:0.5b vs granite3.1-moe:1b: U=35.0, p=0.7921
```

```
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0004054
    smollm2:360m vs granite3.1-moe:1b: U=23.0, p=0.3696
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0004054
Augmentation_type 'negation': Kruskal-Wallis H=14.712, p=0.00208
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=39.0, p=0.4847
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.000891
    qwen2:0.5b vs granite3.1-moe:1b: U=27.0, p=0.6335
    smollm2:360m vs gemma3:1b: U=8.0, p=0.01324
    smollm2:360m vs granite3.1-moe:1b: U=25.0, p=0.4929
    gemma3:1b vs granite3.1-moe:1b: U=63.0, p=0.001348
Augmentation_type 'noise': Kruskal-Wallis H=4.849, p=0.1832
  No significant difference among LLMs.
Augmentation_type 'paraphrase': Kruskal-Wallis H=18.096, p=0.0004202
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=24.0, p=0.4278
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0009229
    qwen2:0.5b vs granite3.1-moe:1b: U=30.0, p=0.8743
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0009148
    smollm2:360m vs granite3.1-moe:1b: U=40.5, p=0.398
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.000891
Augmentation_type 'question_gen': Kruskal-Wallis H=18.394, p=0.0003648
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=18.0, p=0.1553
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.000931
    qwen2:0.5b vs granite3.1-moe:1b: U=15.0, p=0.272
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=23.0, p=0.9484
    gemma3:1b vs granite3.1-moe:1b: U=48.0, p=0.000666
Augmentation_type 'shuffle': Kruskal-Wallis H=1.506, p=0.6808
  No significant difference among LLMs.
Augmentation_type 'summarize': Kruskal-Wallis H=0.333, p=0.9536
  No significant difference among LLMs.
Augmentation_type 'synonym': Kruskal-Wallis H=18.256, p=0.0003895
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=51.0, p=0.05117
    qwen2:0.5b vs gemma3:1b: U=6.0, p=0.007319
    qwen2:0.5b vs granite3.1-moe:1b: U=39.0, p=0.492
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=20.0, p=0.2227
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.000891


==== Kruskal-Wallis and Mann-Whitney U for type_token_ratio (by
augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=19.190, p=0.0002498
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=18.0, p=0.1556
```

```
    qwen2:0.5b vs gemma3:1b: U=60.0, p=0.001865
    qwen2:0.5b vs granite3.1-moe:1b: U=12.5, p=0.04441
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=24.5, p=0.4589
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009069
Augmentation_type 'expand': Kruskal-Wallis H=20.286, p=0.0001481
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=32.5, p=1
    qwen2:0.5b vs gemma3:1b: U=61.0, p=0.002624
    qwen2:0.5b vs granite3.1-moe:1b: U=16.0, p=0.09796
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0008832
    smollm2:360m vs granite3.1-moe:1b: U=7.0, p=0.009431
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.000822
Augmentation_type 'explain_simple': Kruskal-Wallis H=14.494, p=0.002304
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=30.5, p=0.9161
    qwen2:0.5b vs gemma3:1b: U=61.0, p=0.001088
    qwen2:0.5b vs granite3.1-moe:1b: U=30.5, p=0.9161
    smollm2:360m vs gemma3:1b: U=60.0, p=0.003798
    smollm2:360m vs granite3.1-moe:1b: U=27.0, p=0.6348
    gemma3:1b vs granite3.1-moe:1b: U=2.0, p=0.001933
Augmentation_type 'identity': Kruskal-Wallis H=17.688, p=0.0005101
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=26.0, p=0.5624
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0009229
    qwen2:0.5b vs granite3.1-moe:1b: U=30.0, p=0.8745
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009148
    smollm2:360m vs granite3.1-moe:1b: U=33.0, p=0.9578
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009148
Augmentation_type 'negation': Kruskal-Wallis H=18.646, p=0.0003235
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=47.0, p=0.1304
    qwen2:0.5b vs gemma3:1b: U=63.0, p=0.001348
    qwen2:0.5b vs granite3.1-moe:1b: U=24.0, p=0.4295
    smollm2:360m vs gemma3:1b: U=63.0, p=0.001348
    smollm2:360m vs granite3.1-moe:1b: U=19.0, p=0.1857
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0008989
Augmentation_type 'noise': Kruskal-Wallis H=11.094, p=0.01123
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=24.0, p=0.4295
    qwen2:0.5b vs gemma3:1b: U=60.0, p=0.00385
    qwen2:0.5b vs granite3.1-moe:1b: U=5.5, p=0.6004
    smollm2:360m vs gemma3:1b: U=56.0, p=0.01345
    smollm2:360m vs granite3.1-moe:1b: U=8.0, p=1
    gemma3:1b vs granite3.1-moe:1b: U=1.0, p=0.08868
Augmentation_type 'paraphrase': Kruskal-Wallis H=13.738, p=0.003285
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=29.5, p=0.8333
```

```
    qwen2:0.5b vs gemma3:1b: U=58.0, p=0.007276
    qwen2:0.5b vs granite3.1-moe:1b: U=21.0, p=0.2694
    smollm2:360m vs gemma3:1b: U=58.0, p=0.007276
    smollm2:360m vs granite3.1-moe:1b: U=25.0, p=0.4916
    gemma3:1b vs granite3.1-moe:1b: U=2.0, p=0.001903
Augmentation_type 'question_gen': Kruskal-Wallis H=20.201, p=0.0001542
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=22.5, p=0.2143
    qwen2:0.5b vs gemma3:1b: U=63.5, p=0.0009342
    qwen2:0.5b vs granite3.1-moe:1b: U=27.0, p=0.7304
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0005479
    smollm2:360m vs granite3.1-moe:1b: U=36.0, p=0.08359
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.002362
Augmentation_type 'shuffle': Kruskal-Wallis H=18.362, p=0.0003704
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=43.0, p=0.268
    qwen2:0.5b vs gemma3:1b: U=63.0, p=0.001326
    qwen2:0.5b vs granite3.1-moe:1b: U=20.0, p=0.2146
    smollm2:360m vs gemma3:1b: U=60.0, p=0.003824
    smollm2:360m vs granite3.1-moe:1b: U=13.5, p=0.05464
    gemma3:1b vs granite3.1-moe:1b: U=1.0, p=0.001241
Augmentation_type 'summarize': Kruskal-Wallis H=3.078, p=0.3798
  No significant difference among LLMs.
Augmentation_type 'synonym': Kruskal-Wallis H=15.644, p=0.001342
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=22.5, p=0.3314
    qwen2:0.5b vs gemma3:1b: U=59.0, p=0.005183
    qwen2:0.5b vs granite3.1-moe:1b: U=30.5, p=0.9155
    smollm2:360m vs gemma3:1b: U=63.0, p=0.001305
    smollm2:360m vs granite3.1-moe:1b: U=43.5, p=0.2441
    gemma3:1b vs granite3.1-moe:1b: U=3.0, p=0.002722


==== Kruskal-Wallis and Mann-Whitney U for bigram_overlap (by augmentation_type)
====
Augmentation_type 'entity_replace': Kruskal-Wallis H=16.003, p=0.001132
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=28.0, p=0.7112
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.000931
    qwen2:0.5b vs granite3.1-moe:1b: U=18.0, p=0.1529
    smollm2:360m vs gemma3:1b: U=1.0, p=0.001348
    smollm2:360m vs granite3.1-moe:1b: U=23.0, p=0.3681
    gemma3:1b vs granite3.1-moe:1b: U=55.0, p=0.01796
Augmentation_type 'expand': Kruskal-Wallis H=1.631, p=0.6524
  No significant difference among LLMs.
Augmentation_type 'explain_simple': Kruskal-Wallis H=5.506, p=0.1383
  No significant difference among LLMs.
Augmentation_type 'identity': Kruskal-Wallis H=18.329, p=0.0003762
```

```
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=22.5, p=0.341
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0003918
    qwen2:0.5b vs granite3.1-moe:1b: U=27.0, p=0.6355
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0004099
    smollm2:360m vs granite3.1-moe:1b: U=36.0, p=0.7209
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0004099
Augmentation_type 'negation': Kruskal-Wallis H=12.423, p=0.006065
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=41.5, p=0.341
    qwen2:0.5b vs gemma3:1b: U=12.0, p=0.03792
    qwen2:0.5b vs granite3.1-moe:1b: U=33.5, p=0.9163
    smollm2:360m vs gemma3:1b: U=1.0, p=0.001315
    smollm2:360m vs granite3.1-moe:1b: U=21.5, p=0.29
    gemma3:1b vs granite3.1-moe:1b: U=57.0, p=0.006993
Augmentation_type 'noise': Kruskal-Wallis H=13.667, p=0.003396
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=24.0, p=0.4275
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0009229
    qwen2:0.5b vs granite3.1-moe:1b: U=4.0, p=0.3593
    smollm2:360m vs gemma3:1b: U=8.0, p=0.01345
    smollm2:360m vs granite3.1-moe:1b: U=6.0, p=0.6944
    gemma3:1b vs granite3.1-moe:1b: U=16.0, p=0.04949
Augmentation_type 'paraphrase': Kruskal-Wallis H=10.377, p=0.01562
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=57.0, p=0.008244
    qwen2:0.5b vs gemma3:1b: U=32.0, p=1
    qwen2:0.5b vs granite3.1-moe:1b: U=40.0, p=0.4418
    smollm2:360m vs gemma3:1b: U=10.0, p=0.01833
    smollm2:360m vs granite3.1-moe:1b: U=8.5, p=0.01306
    gemma3:1b vs granite3.1-moe:1b: U=44.0, p=0.2268
Augmentation_type 'question_gen': Kruskal-Wallis H=5.139, p=0.1619
  No significant difference among LLMs.
Augmentation_type 'shuffle': Kruskal-Wallis H=13.706, p=0.003334
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=20.0, p=0.2149
    qwen2:0.5b vs gemma3:1b: U=6.0, p=0.006521
    qwen2:0.5b vs granite3.1-moe:1b: U=13.0, p=0.0486
    smollm2:360m vs gemma3:1b: U=7.0, p=0.009808
    smollm2:360m vs granite3.1-moe:1b: U=22.0, p=0.317
    gemma3:1b vs granite3.1-moe:1b: U=56.5, p=0.0116
Augmentation_type 'summarize': Kruskal-Wallis H=3.779, p=0.2863
  No significant difference among LLMs.
Augmentation_type 'synonym': Kruskal-Wallis H=4.229, p=0.2377
  No significant difference among LLMs.


==== Kruskal-Wallis and Mann-Whitney U for total_duration_ns (by
```

```
augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=25.997, p=9.55e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=5.0, p=0.002953
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=1.0, p=0.0003108
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=56.0, p=0.01041
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'expand': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=64.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'explain_simple': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=64.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'identity': Kruskal-Wallis H=21.702, p=7.525e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=6.0, p=0.004662
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=7.0, p=0.006993
    smollm2:360m vs gemma3:1b: U=7.0, p=0.006993
    smollm2:360m vs granite3.1-moe:1b: U=43.0, p=0.2786
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'negation': Kruskal-Wallis H=27.003, p=5.879e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=49.0, p=0.08298
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'noise': Kruskal-Wallis H=18.876, p=0.00029
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=13.0, p=0.04988
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.04444
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=8.0, p=1
```

```
    gemma3:1b vs granite3.1-moe:1b: U=16.0, p=0.04444
Augmentation_type 'paraphrase': Kruskal-Wallis H=16.690, p=0.0008183
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=18.0, p=0.1605
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=7.0, p=0.006993
    smollm2:360m vs gemma3:1b: U=7.0, p=0.006993
    smollm2:360m vs granite3.1-moe:1b: U=31.0, p=0.9591
    gemma3:1b vs granite3.1-moe:1b: U=56.0, p=0.01041
Augmentation_type 'question_gen': Kruskal-Wallis H=26.972, p=5.969e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.000666
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=1.0, p=0.001332
    gemma3:1b vs granite3.1-moe:1b: U=48.0, p=0.000666
Augmentation_type 'shuffle': Kruskal-Wallis H=19.486, p=0.0002169
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=6.0, p=0.004662
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=15.0, p=0.08298
    smollm2:360m vs granite3.1-moe:1b: U=40.0, p=0.4418
    gemma3:1b vs granite3.1-moe:1b: U=56.0, p=0.01041
Augmentation_type 'summarize': Kruskal-Wallis H=13.213, p=0.004198
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=42.0, p=0.3282
    qwen2:0.5b vs gemma3:1b: U=2.0, p=0.0006216
    qwen2:0.5b vs granite3.1-moe:1b: U=1.0, p=0.0003108
    smollm2:360m vs gemma3:1b: U=16.0, p=0.1049
    smollm2:360m vs granite3.1-moe:1b: U=16.0, p=0.1049
    gemma3:1b vs granite3.1-moe:1b: U=15.0, p=0.08298
Augmentation_type 'synonym': Kruskal-Wallis H=24.207, p=2.261e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=8.0, p=0.01041
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=31.0, p=0.9591
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554


==== Kruskal-Wallis and Mann-Whitney U for load_duration_ns (by
augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=27.318, p=5.049e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
```

```
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=12.0, p=0.03792
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'expand': Kruskal-Wallis H=26.227, p=8.547e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=28.0, p=0.7209
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'explain_simple': Kruskal-Wallis H=26.364, p=8.003e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=40.0, p=0.4418
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'identity': Kruskal-Wallis H=28.253, p=3.214e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=5.0, p=0.002953
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'negation': Kruskal-Wallis H=26.207, p=8.63e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=29.0, p=0.7984
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'noise': Kruskal-Wallis H=22.538, p=5.039e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=16.0, p=0.04444
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=14.0, p=0.1778
    gemma3:1b vs granite3.1-moe:1b: U=16.0, p=0.04444
Augmentation_type 'paraphrase': Kruskal-Wallis H=15.372, p=0.001525
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=56.0, p=0.01041
```

```
    qwen2:0.5b vs gemma3:1b: U=7.0, p=0.006993
    qwen2:0.5b vs granite3.1-moe:1b: U=56.0, p=0.01041
    smollm2:360m vs gemma3:1b: U=7.0, p=0.006993
    smollm2:360m vs granite3.1-moe:1b: U=40.0, p=0.4418
    gemma3:1b vs granite3.1-moe:1b: U=56.0, p=0.01041
Augmentation_type 'question_gen': Kruskal-Wallis H=25.015, p=1.533e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=48.0, p=0.000666
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=21.0, p=0.7546
    gemma3:1b vs granite3.1-moe:1b: U=48.0, p=0.000666
Augmentation_type 'shuffle': Kruskal-Wallis H=26.466, p=7.618e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=42.0, p=0.3282
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'summarize': Kruskal-Wallis H=27.003, p=5.879e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=49.0, p=0.08298
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'synonym': Kruskal-Wallis H=26.412, p=7.819e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=41.0, p=0.3823
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554


==== Kruskal-Wallis and Mann-Whitney U for prompt_eval_count (by
augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=26.267, p=8.385e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=29.5, p=0.8319
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0009229
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
```

```
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009229
Augmentation_type 'expand': Kruskal-Wallis H=26.267, p=8.385e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=29.5, p=0.8319
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0009229
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009229
Augmentation_type 'explain_simple': Kruskal-Wallis H=26.450, p=7.677e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=23.5, p=0.3963
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0009229
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009229
Augmentation_type 'identity': Kruskal-Wallis H=26.267, p=8.385e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=29.5, p=0.8319
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0009229
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009229
Augmentation_type 'negation': Kruskal-Wallis H=26.290, p=8.293e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=36.0, p=0.7105
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0009229
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009229
Augmentation_type 'noise': Kruskal-Wallis H=18.577, p=0.0003343
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=29.5, p=0.8319
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0488
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.04444
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0488
Augmentation_type 'paraphrase': Kruskal-Wallis H=26.267, p=8.385e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=29.5, p=0.8319
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0009229
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
```

```
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009229
Augmentation_type 'question_gen': Kruskal-Wallis H=23.945, p=2.565e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=29.5, p=0.8319
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.002335
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.002388
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.002335
Augmentation_type 'shuffle': Kruskal-Wallis H=26.290, p=8.293e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=36.0, p=0.7105
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0009229
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009229
Augmentation_type 'summarize': Kruskal-Wallis H=26.290, p=8.293e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=36.0, p=0.7105
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0009229
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009229
Augmentation_type 'synonym': Kruskal-Wallis H=26.267, p=8.385e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0009229
    qwen2:0.5b vs gemma3:1b: U=29.5, p=0.8319
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0009229
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0009229
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0009229


==== Kruskal-Wallis and Mann-Whitney U for prompt_eval_duration_ns (by
augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=24.136, p=2.339e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=31.0, p=0.9591
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=5.0, p=0.002953
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=3.0, p=0.001088
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'expand': Kruskal-Wallis H=25.389, p=1.28e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=30.0, p=0.8785
```

```
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=2.0, p=0.0006216
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=1.0, p=0.0003108
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'explain_simple': Kruskal-Wallis H=24.136, p=2.339e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=31.0, p=0.9591
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=5.0, p=0.002953
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=3.0, p=0.001088
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'identity': Kruskal-Wallis H=25.918, p=9.924e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=30.0, p=0.8785
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=1.0, p=0.0003108
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'negation': Kruskal-Wallis H=25.128, p=1.452e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=31.0, p=0.9591
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=2.0, p=0.0006216
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=2.0, p=0.0006216
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'noise': Kruskal-Wallis H=18.173, p=0.0004051
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=30.0, p=0.8785
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=1.0, p=0.08889
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.04444
    gemma3:1b vs granite3.1-moe:1b: U=16.0, p=0.04444
Augmentation_type 'paraphrase': Kruskal-Wallis H=22.438, p=5.289e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=28.0, p=0.7209
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=3.0, p=0.001088
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=7.0, p=0.006993
    gemma3:1b vs granite3.1-moe:1b: U=57.0, p=0.006993
Augmentation_type 'question_gen': Kruskal-Wallis H=23.572, p=3.069e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=30.0, p=0.8785
```

```
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=1.0, p=0.001332
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.000666
    gemma3:1b vs granite3.1-moe:1b: U=48.0, p=0.000666
Augmentation_type 'shuffle': Kruskal-Wallis H=24.378, p=2.083e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=31.0, p=0.9591
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=5.0, p=0.002953
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=2.0, p=0.0006216
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'summarize': Kruskal-Wallis H=25.125, p=1.454e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=31.0, p=0.9591
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=3.0, p=0.001088
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=1.0, p=0.0003108
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'synonym': Kruskal-Wallis H=24.622, p=1.852e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=31.0, p=0.9591
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=4.0, p=0.001865
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=2.0, p=0.0006216
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554


==== Kruskal-Wallis and Mann-Whitney U for eval_count (by augmentation_type)
====
Augmentation_type 'entity_replace': Kruskal-Wallis H=7.167, p=0.06675
  No significant difference among LLMs.
  Augmentation_type 'expand': All values identical; cannot perform
Kruskal-Wallis test.
  Augmentation_type 'explain_simple': All values identical; cannot perform
Kruskal-Wallis test.
Augmentation_type 'identity': Kruskal-Wallis H=7.916, p=0.04778
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=50.5, p=0.04809
    qwen2:0.5b vs gemma3:1b: U=48.0, p=0.09413
    qwen2:0.5b vs granite3.1-moe:1b: U=35.0, p=0.7496
    smollm2:360m vs gemma3:1b: U=34.5, p=0.8332
    smollm2:360m vs granite3.1-moe:1b: U=14.5, p=0.06657
    gemma3:1b vs granite3.1-moe:1b: U=11.5, p=0.03261
Augmentation_type 'negation': Kruskal-Wallis H=17.854, p=0.0004715
```

```
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=42.5, p=0.2897
    qwen2:0.5b vs gemma3:1b: U=16.0, p=0.03247
    qwen2:0.5b vs granite3.1-moe:1b: U=18.0, p=0.08474
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0004099
    smollm2:360m vs granite3.1-moe:1b: U=2.0, p=0.00122
    gemma3:1b vs granite3.1-moe:1b: U=36.0, p=0.3816
Augmentation_type 'noise': Kruskal-Wallis H=10.862, p=0.0125
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=45.0, p=0.177
    qwen2:0.5b vs gemma3:1b: U=16.0, p=0.03247
    qwen2:0.5b vs granite3.1-moe:1b: U=4.0, p=0.3032
    smollm2:360m vs gemma3:1b: U=8.0, p=0.004569
    smollm2:360m vs granite3.1-moe:1b: U=2.0, p=0.1384
Augmentation_type 'paraphrase': Kruskal-Wallis H=23.092, p=3.865e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=59.5, p=0.002435
    qwen2:0.5b vs gemma3:1b: U=28.0, p=0.3816
    qwen2:0.5b vs granite3.1-moe:1b: U=28.0, p=0.3816
    smollm2:360m vs gemma3:1b: U=4.0, p=0.001446
    smollm2:360m vs granite3.1-moe:1b: U=4.0, p=0.001446
Augmentation_type 'question_gen': Kruskal-Wallis H=23.453, p=3.249e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=31.5, p=1
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0004099
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.002157
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0004008
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.002107
    gemma3:1b vs granite3.1-moe:1b: U=32.0, p=0.1121
Augmentation_type 'shuffle': Kruskal-Wallis H=4.933, p=0.1768
  No significant difference among LLMs.
Augmentation_type 'summarize': Kruskal-Wallis H=19.974, p=0.0001719
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=57.0, p=0.007185
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0006754
    qwen2:0.5b vs granite3.1-moe:1b: U=34.0, p=0.8481
    smollm2:360m vs gemma3:1b: U=35.0, p=0.7918
    smollm2:360m vs granite3.1-moe:1b: U=6.5, p=0.006978
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0007781
Augmentation_type 'synonym': Kruskal-Wallis H=17.606, p=0.0005304
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=44.0, p=0.2202
    qwen2:0.5b vs gemma3:1b: U=16.0, p=0.03247
    qwen2:0.5b vs granite3.1-moe:1b: U=16.0, p=0.03247
    smollm2:360m vs gemma3:1b: U=4.0, p=0.00146
    smollm2:360m vs granite3.1-moe:1b: U=4.0, p=0.00146
```

```
==== Kruskal-Wallis and Mann-Whitney U for eval_duration_ns (by
augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=25.912, p=9.951e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=4.0, p=0.001865
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=4.0, p=0.001865
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=57.0, p=0.006993
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'expand': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=64.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'explain_simple': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=64.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'identity': Kruskal-Wallis H=18.659, p=0.0003216
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=6.0, p=0.004662
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=12.0, p=0.03792
    smollm2:360m vs gemma3:1b: U=22.0, p=0.3282
    smollm2:360m vs granite3.1-moe:1b: U=46.0, p=0.1605
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'negation': Kruskal-Wallis H=27.818, p=3.966e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=56.0, p=0.01041
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'noise': Kruskal-Wallis H=19.731, p=0.000193
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=9.0, p=0.01476
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.04444
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
```

```
    smollm2:360m vs granite3.1-moe:1b: U=8.0, p=1
    gemma3:1b vs granite3.1-moe:1b: U=16.0, p=0.04444
Augmentation_type 'paraphrase': Kruskal-Wallis H=22.366, p=5.472e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=16.0, p=0.1049
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=41.0, p=0.3823
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554
Augmentation_type 'question_gen': Kruskal-Wallis H=26.640, p=7.003e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=0.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.000666
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=3.0, p=0.004662
    gemma3:1b vs granite3.1-moe:1b: U=48.0, p=0.000666
Augmentation_type 'shuffle': Kruskal-Wallis H=16.656, p=0.0008316
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=6.0, p=0.004662
    qwen2:0.5b vs gemma3:1b: U=6.0, p=0.004662
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=17.0, p=0.1304
    smollm2:360m vs granite3.1-moe:1b: U=42.0, p=0.3282
    gemma3:1b vs granite3.1-moe:1b: U=56.0, p=0.01041
Augmentation_type 'summarize': Kruskal-Wallis H=10.261, p=0.01647
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=42.0, p=0.3282
    qwen2:0.5b vs gemma3:1b: U=39.0, p=0.5054
    qwen2:0.5b vs granite3.1-moe:1b: U=5.0, p=0.002953
    smollm2:360m vs gemma3:1b: U=24.0, p=0.4418
    smollm2:360m vs granite3.1-moe:1b: U=16.0, p=0.1049
    gemma3:1b vs granite3.1-moe:1b: U=3.0, p=0.001088
Augmentation_type 'synonym': Kruskal-Wallis H=24.182, p=2.289e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=8.0, p=0.01041
    qwen2:0.5b vs gemma3:1b: U=0.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=0.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=0.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=32.0, p=1
    gemma3:1b vs granite3.1-moe:1b: U=64.0, p=0.0001554


==== Kruskal-Wallis and Mann-Whitney U for tokens_per_second (by
augmentation_type) ====
Augmentation_type 'entity_replace': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
```

```
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0001554
Augmentation_type 'expand': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0001554
Augmentation_type 'explain_simple': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0001554
Augmentation_type 'identity': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0001554
Augmentation_type 'negation': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0001554
Augmentation_type 'noise': Kruskal-Wallis H=22.838, p=4.366e-05
  Significant: Pairwise comparisons:
    qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
    qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
    qwen2:0.5b vs granite3.1-moe:1b: U=16.0, p=0.04444
    smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
    smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.04444
    gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.04444
Augmentation_type 'paraphrase': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
```

```
      qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
      qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
      qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
      smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
      smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
      gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0001554
Augmentation_type 'question_gen': Kruskal-Wallis H=27.148, p=5.48e-06
  Significant: Pairwise comparisons:
      qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
      qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
      qwen2:0.5b vs granite3.1-moe:1b: U=48.0, p=0.000666
      smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
      smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.000666
      gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.000666
Augmentation_type 'shuffle': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
      qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
      qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
      qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
      smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
      smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
      gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0001554
Augmentation_type 'summarize': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
      qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
      qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
      qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
      smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
      smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
      gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0001554
Augmentation_type 'synonym': Kruskal-Wallis H=29.091, p=2.143e-06
  Significant: Pairwise comparisons:
      qwen2:0.5b vs smollm2:360m: U=64.0, p=0.0001554
      qwen2:0.5b vs gemma3:1b: U=64.0, p=0.0001554
      qwen2:0.5b vs granite3.1-moe:1b: U=64.0, p=0.0001554
      smollm2:360m vs gemma3:1b: U=64.0, p=0.0001554
      smollm2:360m vs granite3.1-moe:1b: U=0.0, p=0.0001554
      gemma3:1b vs granite3.1-moe:1b: U=0.0, p=0.0001554
```

[52]:
```python
# Friedman's Test (by augmentation_type)

from scipy.stats import friedmanchisquare

metrics = [
    'levenshtein_similarity', 'jaccard_similarity', 'length_ratio', 'bleu',
    'cosine_similarity', 'wer', 'char_diversity', 'type_token_ratio',␣
 ↪'bigram_overlap',
```

```
    'total_duration_ns', 'load_duration_ns', 'prompt_eval_count',␣
↪'prompt_eval_duration_ns',
    'eval_count', 'eval_duration_ns', 'tokens_per_second'
]

group_var = 'augmentation_type'   # Or use 'prompt_id' if you want

for metric_col in metrics:
    print(f"\n=== Friedman's Test for {metric_col} (by {group_var}) ===")
    for group_val in sorted(df[group_var].unique()):
        sub = df[df[group_var] == group_val]
        pivot = sub.pivot(index='prompt_id', columns='model', values=metric_col)
        # Only run if all models have data for all prompt_id
        if pivot.notnull().all(axis=1).any() and pivot.shape[1] > 1:
            # Only keep rows (prompts) where all models present
            data = [pivot[m].dropna().values for m in pivot.columns]
            if all(len(x) == len(data[0]) for x in data):
                stat, p = friedmanchisquare(*data)
                print(f"{group_var.capitalize()} '{group_val}': Friedman's␣
↪chi2={stat:.3f}, p={p:.4g}")
            else:
                print(f"{group_var.capitalize()} '{group_val}': Not all models␣
↪have data for all prompts.")
        else:
            print(f"{group_var.capitalize()} '{group_val}': Not enough data for␣
↪Friedman's test.")
#
```

```
=== Friedman's Test for levenshtein_similarity (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=6.450, p=0.09166
Augmentation_type 'expand': Friedman's chi2=1.350, p=0.7173
Augmentation_type 'explain_simple': Friedman's chi2=2.550, p=0.4663
Augmentation_type 'identity': Friedman's chi2=19.950, p=0.0001738
Augmentation_type 'negation': Friedman's chi2=4.200, p=0.2407
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=1.050, p=0.7892
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=20.250, p=0.0001506
Augmentation_type 'summarize': Friedman's chi2=17.250, p=0.0006278
Augmentation_type 'synonym': Friedman's chi2=1.950, p=0.5828

=== Friedman's Test for jaccard_similarity (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=12.300, p=0.006423
Augmentation_type 'expand': Friedman's chi2=4.443, p=0.2174
Augmentation_type 'explain_simple': Friedman's chi2=2.544, p=0.4673
Augmentation_type 'identity': Friedman's chi2=14.550, p=0.002245
```

```
Augmentation_type 'negation': Friedman's chi2=8.250, p=0.04112
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=14.700, p=0.002092
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=12.450, p=0.00599
Augmentation_type 'summarize': Friedman's chi2=4.950, p=0.1755
Augmentation_type 'synonym': Friedman's chi2=4.650, p=0.1993

=== Friedman's Test for length_ratio (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=8.089, p=0.04422
Augmentation_type 'expand': Friedman's chi2=20.550, p=0.0001305
Augmentation_type 'explain_simple': Friedman's chi2=14.850, p=0.001949
Augmentation_type 'identity': Friedman's chi2=4.950, p=0.1755
Augmentation_type 'negation': Friedman's chi2=5.430, p=0.1429
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=15.450, p=0.00147
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=3.300, p=0.3476
Augmentation_type 'summarize': Friedman's chi2=17.100, p=0.000674
Augmentation_type 'synonym': Friedman's chi2=5.550, p=0.1357

=== Friedman's Test for bleu (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=14.700, p=0.002092
Augmentation_type 'expand': Friedman's chi2=1.650, p=0.6481
Augmentation_type 'explain_simple': Friedman's chi2=4.050, p=0.2561
Augmentation_type 'identity': Friedman's chi2=15.000, p=0.001817
Augmentation_type 'negation': Friedman's chi2=12.450, p=0.00599
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=11.400, p=0.009748
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=9.150, p=0.02736
Augmentation_type 'summarize': Friedman's chi2=4.200, p=0.2407
Augmentation_type 'synonym': Friedman's chi2=4.050, p=0.2561

=== Friedman's Test for cosine_similarity (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=11.250, p=0.01045
Augmentation_type 'expand': Friedman's chi2=0.750, p=0.8614
Augmentation_type 'explain_simple': Friedman's chi2=1.800, p=0.6149
Augmentation_type 'identity': Friedman's chi2=15.000, p=0.001817
Augmentation_type 'negation': Friedman's chi2=6.600, p=0.0858
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=13.950, p=0.002974
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=13.050, p=0.00453
Augmentation_type 'summarize': Friedman's chi2=5.850, p=0.1191
Augmentation_type 'synonym': Friedman's chi2=3.450, p=0.3273

=== Friedman's Test for wer (by augmentation_type) ===
```

```
Augmentation_type 'entity_replace': Friedman's chi2=8.423, p=0.03803
Augmentation_type 'expand': Friedman's chi2=17.416, p=0.0005804
Augmentation_type 'explain_simple': Friedman's chi2=13.720, p=0.003312
Augmentation_type 'identity': Friedman's chi2=21.911, p=6.806e-05
Augmentation_type 'negation': Friedman's chi2=12.797, p=0.005096
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=12.038, p=0.007253
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=12.600, p=0.005587
Augmentation_type 'summarize': Friedman's chi2=11.962, p=0.007514
Augmentation_type 'synonym': Friedman's chi2=16.269, p=0.0009986

=== Friedman’s Test for char_diversity (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=13.709, p=0.003329
Augmentation_type 'expand': Friedman's chi2=14.316, p=0.002505
Augmentation_type 'explain_simple': Friedman's chi2=8.250, p=0.04112
Augmentation_type 'identity': Friedman's chi2=15.450, p=0.00147
Augmentation_type 'negation': Friedman's chi2=14.760, p=0.002034
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=14.400, p=0.002408
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=0.797, p=0.8501
Augmentation_type 'summarize': Friedman's chi2=0.375, p=0.9454
Augmentation_type 'synonym': Friedman's chi2=14.392, p=0.002417

=== Friedman’s Test for type_token_ratio (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=19.105, p=0.00026
Augmentation_type 'expand': Friedman's chi2=16.350, p=0.0009612
Augmentation_type 'explain_simple': Friedman's chi2=14.924, p=0.001883
Augmentation_type 'identity': Friedman's chi2=15.759, p=0.00127
Augmentation_type 'negation': Friedman's chi2=16.063, p=0.001101
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=12.342, p=0.0063
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=18.974, p=0.0002768
Augmentation_type 'summarize': Friedman's chi2=4.378, p=0.2234
Augmentation_type 'synonym': Friedman's chi2=13.937, p=0.002993

=== Friedman’s Test for bigram_overlap (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=15.000, p=0.001817
Augmentation_type 'expand': Friedman's chi2=1.481, p=0.6867
Augmentation_type 'explain_simple': Friedman's chi2=5.388, p=0.1455
Augmentation_type 'identity': Friedman's chi2=14.550, p=0.002245
Augmentation_type 'negation': Friedman's chi2=8.924, p=0.03032
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=9.115, p=0.0278
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=11.960, p=0.007521
```

```
Augmentation_type 'summarize': Friedman's chi2=1.481, p=0.6867
Augmentation_type 'synonym': Friedman's chi2=4.038, p=0.2573

=== Friedman’s Test for total_duration_ns (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=22.950, p=4.136e-05
Augmentation_type 'expand': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'explain_simple': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'identity': Friedman's chi2=18.300, p=0.0003814
Augmentation_type 'negation': Friedman's chi2=22.950, p=4.136e-05
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=16.950, p=0.0007237
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=16.650, p=0.0008341
Augmentation_type 'summarize': Friedman's chi2=11.100, p=0.0112
Augmentation_type 'synonym': Friedman's chi2=19.950, p=0.0001738

=== Friedman’s Test for load_duration_ns (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=22.950, p=4.136e-05
Augmentation_type 'expand': Friedman's chi2=21.600, p=7.9e-05
Augmentation_type 'explain_simple': Friedman's chi2=22.200, p=5.927e-05
Augmentation_type 'identity': Friedman's chi2=22.950, p=4.136e-05
Augmentation_type 'negation': Friedman's chi2=21.600, p=7.9e-05
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=15.750, p=0.001276
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=22.200, p=5.927e-05
Augmentation_type 'summarize': Friedman's chi2=22.200, p=5.927e-05
Augmentation_type 'synonym': Friedman's chi2=21.750, p=7.353e-05

=== Friedman’s Test for prompt_eval_count (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=22.792, p=4.462e-05
Augmentation_type 'expand': Friedman's chi2=22.792, p=4.462e-05
Augmentation_type 'explain_simple': Friedman's chi2=23.734, p=2.838e-05
Augmentation_type 'identity': Friedman's chi2=22.792, p=4.462e-05
Augmentation_type 'negation': Friedman's chi2=22.792, p=4.462e-05
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=22.792, p=4.462e-05
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=22.792, p=4.462e-05
Augmentation_type 'summarize': Friedman's chi2=22.792, p=4.462e-05
Augmentation_type 'synonym': Friedman's chi2=22.792, p=4.462e-05

=== Friedman’s Test for prompt_eval_duration_ns (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=21.750, p=7.353e-05
Augmentation_type 'expand': Friedman's chi2=21.600, p=7.9e-05
Augmentation_type 'explain_simple': Friedman's chi2=21.750, p=7.353e-05
Augmentation_type 'identity': Friedman's chi2=21.600, p=7.9e-05
Augmentation_type 'negation': Friedman's chi2=21.600, p=7.9e-05
```

```
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=22.200, p=5.927e-05
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=21.750, p=7.353e-05
Augmentation_type 'summarize': Friedman's chi2=21.750, p=7.353e-05
Augmentation_type 'synonym': Friedman's chi2=21.750, p=7.353e-05

=== Friedman's Test for eval_count (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=9.279, p=0.0258
Augmentation_type 'expand': Friedman's chi2=nan, p=nan
Augmentation_type 'explain_simple': Friedman's chi2=nan, p=nan
Augmentation_type 'identity': Friedman's chi2=5.211, p=0.157
Augmentation_type 'negation': Friedman's chi2=13.476, p=0.003712
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=19.800, p=0.0001867
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=7.163, p=0.06687
Augmentation_type 'summarize': Friedman's chi2=18.917, p=0.0002845
Augmentation_type 'synonym': Friedman's chi2=14.053, p=0.002834

=== Friedman's Test for eval_duration_ns (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'expand': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'explain_simple': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'identity': Friedman's chi2=15.600, p=0.001369
Augmentation_type 'negation': Friedman's chi2=22.950, p=4.136e-05
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=18.450, p=0.0003552
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=13.650, p=0.003422
Augmentation_type 'summarize': Friedman's chi2=8.700, p=0.03356
Augmentation_type 'synonym': Friedman's chi2=19.950, p=0.0001738

=== Friedman's Test for tokens_per_second (by augmentation_type) ===
Augmentation_type 'entity_replace': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'expand': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'explain_simple': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'identity': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'negation': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'noise': Not all models have data for all prompts.
Augmentation_type 'paraphrase': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'question_gen': Not all models have data for all prompts.
Augmentation_type 'shuffle': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'summarize': Friedman's chi2=24.000, p=2.498e-05
Augmentation_type 'synonym': Friedman's chi2=24.000, p=2.498e-05

C:\Users\parth\anaconda3\Lib\site-packages\scipy\stats\_stats_py.py:9427:
RuntimeWarning: invalid value encountered in scalar divide
```

```
  chisq = (12.0 / (k*n*(k+1)) * ssbn - 3*n*(k+1)) / c
C:\Users\parth\anaconda3\Lib\site-packages\scipy\stats\_stats_py.py:9427:
RuntimeWarning: invalid value encountered in scalar divide
  chisq = (12.0 / (k*n*(k+1)) * ssbn - 3*n*(k+1)) / c
```

[53]:

```python
# Kolmogorov-Smirnov (K-S) Test (for all pairs of models)

from scipy.stats import ks_2samp
from itertools import combinations

for metric_col in metrics:
    print(f"\n=== Kolmogorov-Smirnov Test for {metric_col} (All Model Pairs)␣
 ↪===")
    models = df['model'].unique()
    for m1, m2 in combinations(models, 2):
        data1 = df[df['model'] == m1][metric_col].dropna()
        data2 = df[df['model'] == m2][metric_col].dropna()
        if len(data1) > 0 and len(data2) > 0:
            ks_stat, ks_p = ks_2samp(data1, data2)
            print(f"  {m1} vs {m2}: KS stat={ks_stat:.3f}, p={ks_p:.4g}")
        else:
            print(f"  {m1} vs {m2}: Not enough data.")
```

```
=== Kolmogorov-Smirnov Test for levenshtein_similarity (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.239, p=0.01307
  qwen2:0.5b vs gemma3:1b: KS stat=0.364, p=1.447e-05
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.175, p=0.1334
  smollm2:360m vs gemma3:1b: KS stat=0.273, p=0.002746
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.328, p=0.0001615
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.430, p=1.772e-07

=== Kolmogorov-Smirnov Test for jaccard_similarity (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.125, p=0.4999
  qwen2:0.5b vs gemma3:1b: KS stat=0.386, p=3.018e-06
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.261, p=0.005102
  smollm2:360m vs gemma3:1b: KS stat=0.432, p=9.537e-08
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.350, p=4.452e-05
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.309, p=0.0004753

=== Kolmogorov-Smirnov Test for length_ratio (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.352, p=3.046e-05
  qwen2:0.5b vs gemma3:1b: KS stat=0.352, p=3.046e-05
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.552, p=2.635e-12
  smollm2:360m vs gemma3:1b: KS stat=0.250, p=0.007959
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.234, p=0.01648
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.263, p=0.004845
```

```
=== Kolmogorov-Smirnov Test for bleu (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.114, p=0.6237
  qwen2:0.5b vs gemma3:1b: KS stat=0.409, p=5.664e-07
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.266, p=0.004142
  smollm2:360m vs gemma3:1b: KS stat=0.432, p=9.537e-08
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.250, p=0.00845
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.419, p=3.87e-07

=== Kolmogorov-Smirnov Test for cosine_similarity (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.159, p=0.2161
  qwen2:0.5b vs gemma3:1b: KS stat=0.398, p=1.325e-06
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.188, p=0.09014
  smollm2:360m vs gemma3:1b: KS stat=0.489, p=6.79e-10
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.289, p=0.001383
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.307, p=0.0005372

=== Kolmogorov-Smirnov Test for wer (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.284, p=0.001555
  qwen2:0.5b vs gemma3:1b: KS stat=0.330, p=0.000125
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.474, p=4.679e-09
  smollm2:360m vs gemma3:1b: KS stat=0.261, p=0.004732
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.266, p=0.004142
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.317, p=0.0003074

=== Kolmogorov-Smirnov Test for char_diversity (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.216, p=0.03278
  qwen2:0.5b vs gemma3:1b: KS stat=0.523, p=2.457e-11
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.165, p=0.1801
  smollm2:360m vs gemma3:1b: KS stat=0.602, p=3.508e-15
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.170, p=0.1528
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.562, p=8.939e-13

=== Kolmogorov-Smirnov Test for type_token_ratio (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.125, p=0.4999
  qwen2:0.5b vs gemma3:1b: KS stat=0.716, p=5.142e-22
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.309, p=0.0004753
  smollm2:360m vs gemma3:1b: KS stat=0.773, p=3.786e-26
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.230, p=0.01977
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.815, p=1.146e-28

=== Kolmogorov-Smirnov Test for bigram_overlap (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.068, p=0.9876
  qwen2:0.5b vs gemma3:1b: KS stat=0.409, p=5.664e-07
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.211, p=0.03958
  smollm2:360m vs gemma3:1b: KS stat=0.432, p=9.537e-08
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.223, p=0.02583
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.339, p=8.87e-05
```

```
=== Kolmogorov-Smirnov Test for total_duration_ns (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.693, p=1.659e-20
  qwen2:0.5b vs gemma3:1b: KS stat=0.977, p=5.354e-48
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.926, p=2.924e-39
  smollm2:360m vs gemma3:1b: KS stat=0.795, p=5.799e-28
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.533, p=1.9e-11
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.897, p=1.893e-36

=== Kolmogorov-Smirnov Test for load_duration_ns (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.989, p=6.119e-50
  qwen2:0.5b vs gemma3:1b: KS stat=0.989, p=6.119e-50
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.988, p=1.766e-47
  smollm2:360m vs gemma3:1b: KS stat=0.989, p=6.119e-50
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.116, p=0.578
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.988, p=1.766e-47

=== Kolmogorov-Smirnov Test for prompt_eval_count (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=1.000, p=3.477e-52
  qwen2:0.5b vs gemma3:1b: KS stat=0.068, p=0.9876
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=1.000, p=1.051e-49
  smollm2:360m vs gemma3:1b: KS stat=0.989, p=6.119e-50
  smollm2:360m vs granite3.1-moe:1b: KS stat=1.000, p=1.051e-49
  gemma3:1b vs granite3.1-moe:1b: KS stat=1.000, p=1.051e-49

=== Kolmogorov-Smirnov Test for prompt_eval_duration_ns (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.250, p=0.007959
  qwen2:0.5b vs gemma3:1b: KS stat=1.000, p=3.477e-52
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.875, p=3.991e-34
  smollm2:360m vs gemma3:1b: KS stat=1.000, p=3.477e-52
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.898, p=1.456e-36
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.988, p=1.766e-47

=== Kolmogorov-Smirnov Test for eval_count (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.341, p=6.249e-05
  qwen2:0.5b vs gemma3:1b: KS stat=0.136, p=0.3883
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.244, p=0.01078
  smollm2:360m vs gemma3:1b: KS stat=0.455, p=1.437e-08
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.535, p=1.513e-11
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.155, p=0.2388

=== Kolmogorov-Smirnov Test for eval_duration_ns (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=0.739, p=1.327e-23
  qwen2:0.5b vs gemma3:1b: KS stat=0.920, p=3.198e-40
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=0.938, p=1.104e-40
  smollm2:360m vs gemma3:1b: KS stat=0.784, p=4.822e-27
  smollm2:360m vs granite3.1-moe:1b: KS stat=0.625, p=6.761e-16
  gemma3:1b vs granite3.1-moe:1b: KS stat=0.898, p=1.456e-36
```

```
=== Kolmogorov-Smirnov Test for tokens_per_second (All Model Pairs) ===
  qwen2:0.5b vs smollm2:360m: KS stat=1.000, p=3.477e-52
  qwen2:0.5b vs gemma3:1b: KS stat=1.000, p=3.477e-52
  qwen2:0.5b vs granite3.1-moe:1b: KS stat=1.000, p=1.051e-49
  smollm2:360m vs gemma3:1b: KS stat=1.000, p=3.477e-52
  smollm2:360m vs granite3.1-moe:1b: KS stat=1.000, p=1.051e-49
  gemma3:1b vs granite3.1-moe:1b: KS stat=1.000, p=1.051e-49
```

[54]:
```python
# Spearman's Rank Correlation for All Metrics

# How to interpret:

# Correlation coefficient in [-1, 1]:
# Closer to 1 = strong positive monotonic relation
# Closer to -1 = strong negative
# Close to 0 = no monotonic association

# P-value:
# p < 0.05 means significant correlation
# p > 0.05 means correlation is not statistically significant

import pandas as pd
from scipy.stats import spearmanr

# List of your numeric metrics
metrics = [
    'levenshtein_similarity', 'jaccard_similarity', 'length_ratio', 'bleu',
    'cosine_similarity', 'wer', 'char_diversity', 'type_token_ratio',␣
 ↪'bigram_overlap',
    'total_duration_ns', 'load_duration_ns', 'prompt_eval_count',␣
 ↪'prompt_eval_duration_ns',
    'eval_count', 'eval_duration_ns', 'tokens_per_second'
]

# Subset dataframe to just these columns, drop rows with any NaN
metrics_df = df[metrics].dropna()

# Compute Spearman correlation (returns correlation matrix and p-value matrix)
corr, pval = spearmanr(metrics_df, axis=0)

# To DataFrame for easier reading:
corr_df = pd.DataFrame(corr, index=metrics, columns=metrics)
pval_df = pd.DataFrame(pval, index=metrics, columns=metrics)

# Print or display the correlation matrix
print("Spearman Correlation Matrix:")
print(corr_df.round(3))
```

```python
print("\nSpearman Correlation P-value Matrix:")
print(pval_df.round(3))
```

```
Spearman Correlation Matrix:
                         levenshtein_similarity  jaccard_similarity  \
levenshtein_similarity                    1.000               0.329
jaccard_similarity                        0.329               1.000
length_ratio                             -0.240               0.301
bleu                                      0.403               0.921
cosine_similarity                         0.325               0.843
wer                                      -0.538              -0.454
char_diversity                           -0.226              -0.262
type_token_ratio                         -0.060              -0.352
bigram_overlap                            0.447               0.911
total_duration_ns                        -0.079               0.340
load_duration_ns                          0.131               0.198
prompt_eval_count                        -0.183              -0.053
prompt_eval_duration_ns                   0.041               0.369
eval_count                               -0.440               0.245
eval_duration_ns                         -0.115               0.305
tokens_per_second                        -0.172              -0.227

                         length_ratio   bleu  cosine_similarity    wer  \
levenshtein_similarity         -0.240  0.403              0.325 -0.538
jaccard_similarity              0.301  0.921              0.843 -0.454
length_ratio                    1.000  0.253              0.279  0.494
bleu                            0.253  1.000              0.816 -0.448
cosine_similarity               0.279  0.816              1.000 -0.399
wer                             0.494 -0.448             -0.399  1.000
char_diversity                  0.168 -0.173             -0.087  0.456
type_token_ratio               -0.270 -0.379             -0.507 -0.069
bigram_overlap                  0.218  0.972              0.794 -0.462
total_duration_ns               0.218  0.365              0.368  0.031
load_duration_ns                0.127  0.253              0.273  0.082
prompt_eval_count              -0.277 -0.101             -0.137 -0.257
prompt_eval_duration_ns        -0.142  0.373              0.360 -0.238
eval_count                      0.625  0.203              0.238  0.413
eval_duration_ns                0.298  0.335              0.341  0.112
tokens_per_second               0.023 -0.273             -0.263  0.072

                         char_diversity  type_token_ratio  bigram_overlap  \
levenshtein_similarity           -0.226            -0.060           0.447
jaccard_similarity               -0.262            -0.352           0.911
length_ratio                      0.168            -0.270           0.218
bleu                             -0.173            -0.379           0.972
cosine_similarity                -0.087            -0.507           0.794
wer                               0.456            -0.069          -0.462
```

```
char_diversity                  1.000           -0.238          -0.213
type_token_ratio               -0.238            1.000          -0.342
bigram_overlap                 -0.213           -0.342           1.000
total_duration_ns               0.324           -0.561           0.302
load_duration_ns                0.307           -0.512           0.247
prompt_eval_count              -0.242            0.394          -0.111
prompt_eval_duration_ns         0.223           -0.445           0.345
eval_count                      0.397           -0.294           0.145
eval_duration_ns                0.372           -0.568           0.268
tokens_per_second              -0.266            0.511          -0.246

                         total_duration_ns  load_duration_ns  \
levenshtein_similarity              -0.079             0.131
jaccard_similarity                   0.340             0.198
length_ratio                         0.218             0.127
bleu                                 0.365             0.253
cosine_similarity                    0.368             0.273
wer                                  0.031             0.082
char_diversity                       0.324             0.307
type_token_ratio                    -0.561            -0.512
bigram_overlap                       0.302             0.247
total_duration_ns                    1.000             0.285
load_duration_ns                     0.285             1.000
prompt_eval_count                    0.078            -0.728
prompt_eval_duration_ns              0.703             0.457
eval_count                           0.505             0.117
eval_duration_ns                     0.971             0.239
tokens_per_second                   -0.841            -0.292

                         prompt_eval_count  prompt_eval_duration_ns  \
levenshtein_similarity              -0.183                    0.041
jaccard_similarity                  -0.053                    0.369
length_ratio                        -0.277                   -0.142
bleu                                -0.101                    0.373
cosine_similarity                   -0.137                    0.360
wer                                 -0.257                   -0.238
char_diversity                      -0.242                    0.223
type_token_ratio                     0.394                   -0.445
bigram_overlap                      -0.111                    0.345
total_duration_ns                    0.078                    0.703
load_duration_ns                    -0.728                    0.457
prompt_eval_count                    1.000                    0.061
prompt_eval_duration_ns              0.061                    1.000
eval_count                           0.091                    0.314
eval_duration_ns                     0.052                    0.604
tokens_per_second                    0.043                   -0.608

                         eval_count  eval_duration_ns  tokens_per_second
```

```
levenshtein_similarity      -0.440          -0.115          -0.172
jaccard_similarity           0.245           0.305          -0.227
length_ratio                 0.625           0.298           0.023
bleu                         0.203           0.335          -0.273
cosine_similarity            0.238           0.341          -0.263
wer                          0.413           0.112           0.072
char_diversity               0.397           0.372          -0.266
type_token_ratio            -0.294          -0.568           0.511
bigram_overlap               0.145           0.268          -0.246
total_duration_ns            0.505           0.971          -0.841
load_duration_ns             0.117           0.239          -0.292
prompt_eval_count            0.091           0.052           0.043
prompt_eval_duration_ns      0.314           0.604          -0.608
eval_count                   1.000           0.548          -0.119
eval_duration_ns             0.548           1.000          -0.830
tokens_per_second           -0.119          -0.830           1.000

Spearman Correlation P-value Matrix:
                         levenshtein_similarity  jaccard_similarity  \
levenshtein_similarity                    0.000               0.000
jaccard_similarity                        0.000               0.000
length_ratio                              0.000               0.000
bleu                                      0.000               0.000
cosine_similarity                         0.000               0.000
wer                                       0.000               0.000
char_diversity                            0.000               0.000
type_token_ratio                          0.263               0.000
bigram_overlap                            0.000               0.000
total_duration_ns                         0.145               0.000
load_duration_ns                          0.015               0.000
prompt_eval_count                         0.001               0.326
prompt_eval_duration_ns                   0.450               0.000
eval_count                                0.000               0.000
eval_duration_ns                          0.033               0.000
tokens_per_second                         0.001               0.000

                         length_ratio   bleu  cosine_similarity    wer  \
levenshtein_similarity          0.000  0.000              0.000  0.000
jaccard_similarity              0.000  0.000              0.000  0.000
length_ratio                    0.000  0.000              0.000  0.000
bleu                            0.000  0.000              0.000  0.000
cosine_similarity               0.000  0.000              0.000  0.000
wer                             0.000  0.000              0.000  0.000
char_diversity                  0.002  0.001              0.106  0.000
type_token_ratio                0.000  0.000              0.000  0.201
bigram_overlap                  0.000  0.000              0.000  0.000
total_duration_ns               0.000  0.000              0.000  0.565
load_duration_ns                0.018  0.000              0.000  0.128
```

prompt_eval_count	0.000	0.061	0.011	0.000
prompt_eval_duration_ns	0.008	0.000	0.000	0.000
eval_count	0.000	0.000	0.000	0.000
eval_duration_ns	0.000	0.000	0.000	0.039
tokens_per_second	0.674	0.000	0.000	0.184

	char_diversity	type_token_ratio	bigram_overlap	\
levenshtein_similarity	0.000	0.263	0.000	
jaccard_similarity	0.000	0.000	0.000	
length_ratio	0.002	0.000	0.000	
bleu	0.001	0.000	0.000	
cosine_similarity	0.106	0.000	0.000	
wer	0.000	0.201	0.000	
char_diversity	0.000	0.000	0.000	
type_token_ratio	0.000	0.000	0.000	
bigram_overlap	0.000	0.000	0.000	
total_duration_ns	0.000	0.000	0.000	
load_duration_ns	0.000	0.000	0.000	
prompt_eval_count	0.000	0.000	0.040	
prompt_eval_duration_ns	0.000	0.000	0.000	
eval_count	0.000	0.000	0.007	
eval_duration_ns	0.000	0.000	0.000	
tokens_per_second	0.000	0.000	0.000	

	total_duration_ns	load_duration_ns	\
levenshtein_similarity	0.145	0.015	
jaccard_similarity	0.000	0.000	
length_ratio	0.000	0.018	
bleu	0.000	0.000	
cosine_similarity	0.000	0.000	
wer	0.565	0.128	
char_diversity	0.000	0.000	
type_token_ratio	0.000	0.000	
bigram_overlap	0.000	0.000	
total_duration_ns	0.000	0.000	
load_duration_ns	0.000	0.000	
prompt_eval_count	0.150	0.000	
prompt_eval_duration_ns	0.000	0.000	
eval_count	0.000	0.031	
eval_duration_ns	0.000	0.000	
tokens_per_second	0.000	0.000	

	prompt_eval_count	prompt_eval_duration_ns	\
levenshtein_similarity	0.001	0.450	
jaccard_similarity	0.326	0.000	
length_ratio	0.000	0.008	
bleu	0.061	0.000	
cosine_similarity	0.011	0.000	

```
wer                                  0.000                    0.000
char_diversity                       0.000                    0.000
type_token_ratio                     0.000                    0.000
bigram_overlap                       0.040                    0.000
total_duration_ns                    0.150                    0.000
load_duration_ns                     0.000                    0.000
prompt_eval_count                    0.000                    0.261
prompt_eval_duration_ns              0.261                    0.000
eval_count                           0.091                    0.000
eval_duration_ns                     0.338                    0.000
tokens_per_second                    0.425                    0.000

                         eval_count  eval_duration_ns  tokens_per_second
levenshtein_similarity        0.000             0.033              0.001
jaccard_similarity            0.000             0.000              0.000
length_ratio                  0.000             0.000              0.674
bleu                          0.000             0.000              0.000
cosine_similarity             0.000             0.000              0.000
wer                           0.000             0.039              0.184
char_diversity                0.000             0.000              0.000
type_token_ratio              0.000             0.000              0.000
bigram_overlap                0.007             0.000              0.000
total_duration_ns             0.000             0.000              0.000
load_duration_ns              0.031             0.000              0.000
prompt_eval_count             0.091             0.338              0.425
prompt_eval_duration_ns       0.000             0.000              0.000
eval_count                    0.000             0.000              0.027
eval_duration_ns              0.000             0.000              0.000
tokens_per_second             0.027             0.000              0.000
```

[55]:
```python
import pandas as pd

metrics = [
    'levenshtein_similarity', 'jaccard_similarity', 'length_ratio', 'bleu',
    'cosine_similarity', 'wer', 'char_diversity', 'type_token_ratio',␣
↪'bigram_overlap',
    'total_duration_ns', 'load_duration_ns', 'prompt_eval_count',␣
↪'prompt_eval_duration_ns',
    'eval_count', 'eval_duration_ns', 'tokens_per_second'
]

# Filter to relevant columns, drop rows with any missing value in those columns
df_manova = df[['model', 'augmentation_type'] + metrics].dropna()
print(df_manova.shape)


## MANOVA by Model Only
```

```
# Interpretation
# Look at the line for model in Wilks' lambda, Pillai's trace, Hotelling-Lawley
↪trace, and Roy's largest root.
# The Pr > F column gives the p-value for the global null: "All LLMs have the
↪same mean vector of metrics."
# p < 0.05 means at least one LLM is different on at least one combination of
↪metrics.

from statsmodels.multivariate.manova import MANOVA

formula = ' + '.join(metrics) + ' ~ model'
maov = MANOVA.from_formula(formula, data=df_manova)
print(maov.mv_test())
```

```
(344, 18)
                     Multivariate linear model
======================================================================

----------------------------------------------------------------------
         Intercept          Value   Num DF  Den DF    F Value    Pr > F
----------------------------------------------------------------------
            Wilks' lambda    0.0002 4.0000 337.0000 397563.2644 0.0000
           Pillai's trace    0.9998 4.0000 337.0000 397563.2644 0.0000
   Hotelling-Lawley trace 4718.8518 4.0000 337.0000 397563.2644 0.0000
      Roy's greatest root 4718.8518 4.0000 337.0000 397563.2644 0.0000
----------------------------------------------------------------------

----------------------------------------------------------------------
         model             Value    Num DF   Den DF    F Value    Pr > F
----------------------------------------------------------------------
          Wilks' lambda    0.0000 12.0000  891.9097  14012.8080 0.0000
         Pillai's trace    2.7918 12.0000 1017.0000   1136.1569 0.0000
 Hotelling-Lawley trace 4114.5409 12.0000  585.4521 115258.0057 0.0000
    Roy's greatest root 4062.1679  4.0000  339.0000 344268.7302 0.0000
======================================================================
```

[56]:
```
#MANOVA by Model and Augmentation Type (Factorial MANOVA)

formula = ' + '.join(metrics) + ' ~ model + augmentation_type'
maov2 = MANOVA.from_formula(formula, data=df_manova)
print(maov2.mv_test())
```

```
                     Multivariate linear model
=====================================================================

---------------------------------------------------------------------
```

```
        Intercept          Value   Num DF  Den DF    F Value   Pr > F
---------------------------------------------------------------------
           Wilks' lambda    0.0007 4.0000 327.0000 118474.4430 0.0000
          Pillai's trace    0.9993 4.0000 327.0000 118474.4430 0.0000
  Hotelling-Lawley trace 1449.2287 4.0000 327.0000 118474.4430 0.0000
     Roy's greatest root 1449.2287 4.0000 327.0000 118474.4430 0.0000
---------------------------------------------------------------------

---------------------------------------------------------------------
         model            Value    Num DF  Den DF    F Value   Pr > F
---------------------------------------------------------------------
          Wilks' lambda    0.0000 12.0000 865.4522  18710.9331 0.0000
         Pillai's trace    2.8161 12.0000 987.0000   1259.7218 0.0000
 Hotelling-Lawley trace 4490.8920 12.0000 567.9526 122058.1435 0.0000
    Roy's greatest root 4389.9069  4.0000 329.0000 361069.8449 0.0000
---------------------------------------------------------------------

----------------------------------------------------------------------
     augmentation_type     Value    Num DF    Den DF   F Value  Pr > F
----------------------------------------------------------------------
            Wilks' lambda  0.0624  40.0000  1241.8012  33.4889  0.0000
           Pillai's trace  2.1361  40.0000  1320.0000  37.8209  0.0000
   Hotelling-Lawley trace  3.8299  40.0000   903.4338  31.1871  0.0000
      Roy's greatest root  1.6174  10.0000   330.0000  53.3743  0.0000
=====================================================================
```

[57]:

```python
# MANOVA With Interaction Term

formula = ' + '.join(metrics) + ' ~ model * augmentation_type'
maov3 = MANOVA.from_formula(formula, data=df_manova)
print(maov3.mv_test())
```

```
                    Multivariate linear model
===================================================================

-------------------------------------------------------------------
        Intercept         Value   Num DF  Den DF   F Value   Pr > F
-------------------------------------------------------------------
           Wilks' lambda   0.0017 4.0000 297.0000 42399.6329 0.0000
          Pillai's trace   0.9983 4.0000 297.0000 42399.6329 0.0000
  Hotelling-Lawley trace 571.0388 4.0000 297.0000 42399.6329 0.0000
     Roy's greatest root 571.0388 4.0000 297.0000 42399.6329 0.0000
-------------------------------------------------------------------

-------------------------------------------------------------------
         model           Value   Num DF  Den DF   F Value   Pr > F
-------------------------------------------------------------------
```

```
          Wilks' lambda   0.0001 12.0000 786.0796  1896.7015 0.0000
         Pillai's trace   2.2510 12.0000 897.0000   224.6345 0.0000
 Hotelling-Lawley trace 475.3388 12.0000 515.4541 11730.9296 0.0000
    Roy's greatest root 464.4037  4.0000 299.0000 34714.1753 0.0000
-------------------------------------------------------------------

-------------------------------------------------------------------
      augmentation_type    Value   Num DF   Den DF  F Value  Pr > F
-------------------------------------------------------------------
             Wilks' lambda 0.0221 40.0000 1128.0447  48.8563 0.0000
            Pillai's trace 2.4002 40.0000 1200.0000  45.0087 0.0000
    Hotelling-Lawley trace 7.5527 40.0000  819.4412  55.8374 0.0000
       Roy's greatest root 4.4835 10.0000  300.0000 134.5065 0.0000
-------------------------------------------------------------------

-------------------------------------------------------------------
   model:augmentation_type Value   Num DF    Den DF  F Value Pr > F
-------------------------------------------------------------------
             Wilks' lambda 0.0071 120.0000 1183.2579 24.4081 0.0000
            Pillai's trace 3.4426 120.0000 1200.0000 61.7649 0.0000
    Hotelling-Lawley trace 8.2202 120.0000 1016.8090 20.2479 0.0000
       Roy's greatest root 3.2635  30.0000  300.0000 32.6351 0.0000
===================================================================
```

[58]:

```python
#Post-hoc Exploration: Which Metrics Are Responsible?

#If MANOVA is significant, run univariate ANOVA for each metric:


import statsmodels.api as sm
from statsmodels.formula.api import ols

for metric in metrics:
    model = ols(f'{metric} ~ model', data=df_manova).fit()
    anova_table = sm.stats.anova_lm(model, typ=2)
    print(f"ANOVA for {metric}:\n", anova_table, "\n")
```

```
ANOVA for levenshtein_similarity:
              sum_sq     df          F        PR(>F)
model      3.910936    3.0  26.939258  1.177641e-15
Residual  16.453291  340.0        NaN           NaN

ANOVA for jaccard_similarity:
              sum_sq     df          F        PR(>F)
model      2.841742    3.0  27.623562  5.199714e-16
Residual  11.659036  340.0        NaN           NaN
```

```
ANOVA for length_ratio:
             sum_sq     df         F    PR(>F)
model      1.649220    3.0  7.683006  0.000056
Residual  24.327932  340.0       NaN       NaN

ANOVA for bleu:
             sum_sq     df          F        PR(>F)
model      3.799517    3.0  32.509959  1.687200e-18
Residual  13.245540  340.0        NaN           NaN

ANOVA for cosine_similarity:
             sum_sq     df          F        PR(>F)
model      3.033348    3.0  26.071969  3.336757e-15
Residual  13.185790  340.0        NaN           NaN

ANOVA for wer:
             sum_sq     df          F        PR(>F)
model      4.486417    3.0  15.780305  1.249106e-09
Residual  32.221214  340.0        NaN           NaN

ANOVA for char_diversity:
            sum_sq     df        F        PR(>F)
model     0.773419    3.0  25.9462  3.882705e-15
Residual  3.378306  340.0      NaN           NaN

ANOVA for type_token_ratio:
            sum_sq     df           F        PR(>F)
model     0.456416    3.0  107.263221  6.956917e-49
Residual  0.482245  340.0         NaN           NaN

ANOVA for bigram_overlap:
             sum_sq     df          F        PR(>F)
model      3.389387    3.0  30.396176  1.967544e-17
Residual  12.637461  340.0        NaN           NaN

ANOVA for total_duration_ns:
                sum_sq     df          F        PR(>F)
model     3.833423e+21    3.0  114.62624  2.664217e-51
Residual  3.790185e+21  340.0        NaN           NaN

ANOVA for load_duration_ns:
                sum_sq     df         F    PR(>F)
model     6.248601e+18    3.0  0.399618  0.753364
Residual  1.772131e+21  340.0       NaN       NaN

ANOVA for prompt_eval_count:
                 sum_sq     df            F         PR(>F)
model     145085.989429    3.0  1572.665804  6.740158e-199
```

```
Residual   10455.545455  340.0          NaN            NaN

ANOVA for prompt_eval_duration_ns:
                sum_sq     df           F         PR(>F)
model     1.914607e+20    3.0  825.644883  1.072120e-155
Residual  2.628112e+19  340.0         NaN            NaN

ANOVA for eval_count:
                sum_sq     df          F        PR(>F)
model      7829.789852    3.0  19.476164  1.124172e-11
Residual  45562.163636  340.0        NaN           NaN

ANOVA for eval_duration_ns:
                sum_sq     df           F        PR(>F)
model     2.324837e+21    3.0  158.921245  2.226701e-64
Residual  1.657938e+21  340.0         NaN           NaN

ANOVA for tokens_per_second:
               sum_sq     df            F  PR(>F)
model     1418.920496    3.0  97736.33144     0.0
Residual     1.645355  340.0          NaN     NaN
```

[59]:

```
# Interpreting Model-wise Effect Sizes
#1. Mean Difference (mean_diff):
#This is simply the difference in mean value of a metric between two models.

#Positive value: model_1 has a higher mean than model_2 for that metric.

#Negative value: model_2 has a higher mean.

#2. Cohen’s d (cohens_d):
#Cohen’s d quantifies the magnitude of the difference (in standard deviations) between two distributions.

#Conventional thresholds for interpretation:

#    |d| < 0.2: Negligible difference (practically equivalent)

#    0.2   |d| < 0.5: Small effect (slight difference)

#     0.5   |d| < 0.8: Medium effect (moderate difference)

#   |d|   0.8: Large effect (substantial/practically meaningful difference)

# Sign of d:
```

```
# Positive: model_1 performs better (higher mean for positively oriented␣
↪metrics, e.g., similarity, tokens/sec).

# Negative: model_2 performs better.

# 3. Metric Orientation
# For similarity/accuracy metrics (bleu, levenshtein_similarity,␣
↪cosine_similarity, etc.): Higher is better.

# For error or duration metrics (wer, total_duration_ns, etc.): Lower is better.

# If Cohen's d is positive and the metric is an error/duration, model_1 has␣
↪worse performance.
```

[60]:
```
import pandas as pd
import numpy as np
from itertools import combinations
from scipy.stats import ttest_ind

def cohens_d(x, y):
    nx, ny = len(x), len(y)
    pooled_std = np.sqrt(((nx-1)*np.var(x, ddof=1) + (ny-1)*np.var(y, ddof=1)) /
↪ (nx + ny - 2))
    return (np.mean(x) - np.mean(y)) / pooled_std

metrics = [
    "total_duration_ns", "load_duration_ns", "prompt_eval_count",␣
↪"prompt_eval_duration_ns", "eval_count",
    "eval_duration_ns", "tokens_per_second", "levenshtein_similarity",␣
↪"jaccard_similarity", "length_ratio",
    "bleu", "cosine_similarity", "wer", "char_diversity", "type_token_ratio",␣
↪"bigram_overlap"
]

model_list = sorted(df['model'].unique())
results = []


for metric in metrics:
    for m1, m2 in combinations(model_list, 2):
        x = df[df['model'] == m1][metric].dropna()
        y = df[df['model'] == m2][metric].dropna()
        if len(x) > 1 and len(y) > 1:
            d = cohens_d(x, y)
            mean_diff = x.mean() - y.mean()
            t_stat, p_val = ttest_ind(x, y, equal_var=False)
            results.append({
```

```python
                'metric': metric,
                'model_1': m1,
                'model_2': m2,
                'mean_model_1': x.mean(),
                'mean_model_2': y.mean(),
                'std_model_1': x.std(ddof=1),
                'std_model_2': y.std(ddof=1),
                'n_model_1': len(x),
                'n_model_2': len(y),
                'mean_diff': mean_diff,
                'cohens_d': d,
                'ttest_pval': p_val
            })

effectsize_df = pd.DataFrame(results)

pd.set_option('display.max_rows', None)
print("Table: Pairwise Model Comparisons Across All Metrics-Means, Standard␣
↪Deviations, Sample Sizes, Mean Differences, Cohen’s d, and t-Test p-values")
display(effectsize_df)
```

```
Table: Pairwise Model Comparisons Across All Metrics-Means, Standard Deviations,
Sample Sizes, Mean Differences, Cohen’s d, and t-Test p-values

                  metric            model_1            model_2  \
0      total_duration_ns         gemma3:1b  granite3.1-moe:1b
1      total_duration_ns         gemma3:1b        qwen2:0.5b
2      total_duration_ns         gemma3:1b       smollm2:360m
3      total_duration_ns  granite3.1-moe:1b        qwen2:0.5b
4      total_duration_ns  granite3.1-moe:1b       smollm2:360m
5      total_duration_ns        qwen2:0.5b       smollm2:360m
6       load_duration_ns         gemma3:1b  granite3.1-moe:1b
7       load_duration_ns         gemma3:1b        qwen2:0.5b
8       load_duration_ns         gemma3:1b       smollm2:360m
9       load_duration_ns  granite3.1-moe:1b        qwen2:0.5b
10      load_duration_ns  granite3.1-moe:1b       smollm2:360m
11      load_duration_ns        qwen2:0.5b       smollm2:360m
12     prompt_eval_count         gemma3:1b  granite3.1-moe:1b
13     prompt_eval_count         gemma3:1b        qwen2:0.5b
14     prompt_eval_count         gemma3:1b       smollm2:360m
15     prompt_eval_count  granite3.1-moe:1b        qwen2:0.5b
16     prompt_eval_count  granite3.1-moe:1b       smollm2:360m
17     prompt_eval_count        qwen2:0.5b       smollm2:360m
18  prompt_eval_duration_ns         gemma3:1b  granite3.1-moe:1b
19  prompt_eval_duration_ns         gemma3:1b        qwen2:0.5b
20  prompt_eval_duration_ns         gemma3:1b       smollm2:360m
21  prompt_eval_duration_ns  granite3.1-moe:1b        qwen2:0.5b
22  prompt_eval_duration_ns  granite3.1-moe:1b       smollm2:360m
```

23	prompt_eval_duration_ns	qwen2:0.5b	smollm2:360m
24	eval_count	gemma3:1b	granite3.1-moe:1b
25	eval_count	gemma3:1b	qwen2:0.5b
26	eval_count	gemma3:1b	smollm2:360m
27	eval_count	granite3.1-moe:1b	qwen2:0.5b
28	eval_count	granite3.1-moe:1b	smollm2:360m
29	eval_count	qwen2:0.5b	smollm2:360m
30	eval_duration_ns	gemma3:1b	granite3.1-moe:1b
31	eval_duration_ns	gemma3:1b	qwen2:0.5b
32	eval_duration_ns	gemma3:1b	smollm2:360m
33	eval_duration_ns	granite3.1-moe:1b	qwen2:0.5b
34	eval_duration_ns	granite3.1-moe:1b	smollm2:360m
35	eval_duration_ns	qwen2:0.5b	smollm2:360m
36	tokens_per_second	gemma3:1b	granite3.1-moe:1b
37	tokens_per_second	gemma3:1b	qwen2:0.5b
38	tokens_per_second	gemma3:1b	smollm2:360m
39	tokens_per_second	granite3.1-moe:1b	qwen2:0.5b
40	tokens_per_second	granite3.1-moe:1b	smollm2:360m
41	tokens_per_second	qwen2:0.5b	smollm2:360m
42	levenshtein_similarity	gemma3:1b	granite3.1-moe:1b
43	levenshtein_similarity	gemma3:1b	qwen2:0.5b
44	levenshtein_similarity	gemma3:1b	smollm2:360m
45	levenshtein_similarity	granite3.1-moe:1b	qwen2:0.5b
46	levenshtein_similarity	granite3.1-moe:1b	smollm2:360m
47	levenshtein_similarity	qwen2:0.5b	smollm2:360m
48	jaccard_similarity	gemma3:1b	granite3.1-moe:1b
49	jaccard_similarity	gemma3:1b	qwen2:0.5b
50	jaccard_similarity	gemma3:1b	smollm2:360m
51	jaccard_similarity	granite3.1-moe:1b	qwen2:0.5b
52	jaccard_similarity	granite3.1-moe:1b	smollm2:360m
53	jaccard_similarity	qwen2:0.5b	smollm2:360m
54	length_ratio	gemma3:1b	granite3.1-moe:1b
55	length_ratio	gemma3:1b	qwen2:0.5b
56	length_ratio	gemma3:1b	smollm2:360m
57	length_ratio	granite3.1-moe:1b	qwen2:0.5b
58	length_ratio	granite3.1-moe:1b	smollm2:360m
59	length_ratio	qwen2:0.5b	smollm2:360m
60	bleu	gemma3:1b	granite3.1-moe:1b
61	bleu	gemma3:1b	qwen2:0.5b
62	bleu	gemma3:1b	smollm2:360m
63	bleu	granite3.1-moe:1b	qwen2:0.5b
64	bleu	granite3.1-moe:1b	smollm2:360m
65	bleu	qwen2:0.5b	smollm2:360m
66	cosine_similarity	gemma3:1b	granite3.1-moe:1b
67	cosine_similarity	gemma3:1b	qwen2:0.5b
68	cosine_similarity	gemma3:1b	smollm2:360m
69	cosine_similarity	granite3.1-moe:1b	qwen2:0.5b
70	cosine_similarity	granite3.1-moe:1b	smollm2:360m

```
71        cosine_similarity         qwen2:0.5b       smollm2:360m
72                      wer          gemma3:1b  granite3.1-moe:1b
73                      wer          gemma3:1b         qwen2:0.5b
74                      wer          gemma3:1b       smollm2:360m
75                      wer  granite3.1-moe:1b         qwen2:0.5b
76                      wer  granite3.1-moe:1b       smollm2:360m
77                      wer         qwen2:0.5b       smollm2:360m
78           char_diversity          gemma3:1b  granite3.1-moe:1b
79           char_diversity          gemma3:1b         qwen2:0.5b
80           char_diversity          gemma3:1b       smollm2:360m
81           char_diversity  granite3.1-moe:1b         qwen2:0.5b
82           char_diversity  granite3.1-moe:1b       smollm2:360m
83           char_diversity         qwen2:0.5b       smollm2:360m
84         type_token_ratio          gemma3:1b  granite3.1-moe:1b
85         type_token_ratio          gemma3:1b         qwen2:0.5b
86         type_token_ratio          gemma3:1b       smollm2:360m
87         type_token_ratio  granite3.1-moe:1b         qwen2:0.5b
88         type_token_ratio  granite3.1-moe:1b       smollm2:360m
89         type_token_ratio         qwen2:0.5b       smollm2:360m
90           bigram_overlap          gemma3:1b  granite3.1-moe:1b
91           bigram_overlap          gemma3:1b         qwen2:0.5b
92           bigram_overlap          gemma3:1b       smollm2:360m
93           bigram_overlap  granite3.1-moe:1b         qwen2:0.5b
94           bigram_overlap  granite3.1-moe:1b       smollm2:360m
95           bigram_overlap         qwen2:0.5b       smollm2:360m

    mean_model_1  mean_model_2   std_model_1   std_model_2  n_model_1  \
0   1.637960e+10  1.051084e+10  3.626593e+09  3.985375e+09         88
1   1.637960e+10  7.160551e+09  3.626593e+09  1.498693e+09         88
2   1.637960e+10  1.105877e+10  3.626593e+09  3.707344e+09         88
3   1.051084e+10  7.160551e+09  3.985375e+09  1.498693e+09         80
4   1.051084e+10  1.105877e+10  3.985375e+09  3.707344e+09         80
5   7.160551e+09  1.105877e+10  1.498693e+09  3.707344e+09         88
6   3.576625e+08  4.339573e+08  2.094949e+09  3.690988e+09         88
7   3.576625e+08  7.873249e+07  2.094949e+09  2.526810e+08         88
8   3.576625e+08  2.220176e+08  2.094949e+09  1.883081e+09         88
9   4.339573e+08  7.873249e+07  3.690988e+09  2.526810e+08         80
10  4.339573e+08  2.220176e+08  3.690988e+09  1.883081e+09         80
11  7.873249e+07  2.220176e+08  2.526810e+08  1.883081e+09         88
12  6.051136e+01  1.113750e+02  4.651027e+00  7.350837e+00         88
13  6.051136e+01  6.019318e+01  4.651027e+00  4.977785e+00         88
14  6.051136e+01  8.168182e+01  4.651027e+00  4.970130e+00         88
15  1.113750e+02  6.019318e+01  7.350837e+00  4.977785e+00         80
16  1.113750e+02  8.168182e+01  7.350837e+00  4.970130e+00         80
17  6.019318e+01  8.168182e+01  4.977785e+00  4.970130e+00         88
18  3.247818e+09  1.986857e+09  3.790890e+08  3.220527e+08         88
19  3.247818e+09  1.438463e+09  3.790890e+08  1.865101e+08         88
20  3.247818e+09  1.446045e+09  3.790890e+08  1.714840e+08         88
```

21	1.986857e+09	1.438463e+09	3.220527e+08	1.865101e+08	80
22	1.986857e+09	1.446045e+09	3.220527e+08	1.714840e+08	80
23	1.438463e+09	1.446045e+09	1.865101e+08	1.714840e+08	88
24	5.450000e+01	5.842500e+01	1.127911e+01	5.204124e+00	88
25	5.450000e+01	5.244318e+01	1.127911e+01	1.289735e+01	88
26	5.450000e+01	4.521591e+01	1.127911e+01	1.433703e+01	88
27	5.842500e+01	5.244318e+01	5.204124e+00	1.289735e+01	80
28	5.842500e+01	4.521591e+01	5.204124e+00	1.433703e+01	80
29	5.244318e+01	4.521591e+01	1.289735e+01	1.433703e+01	88
30	1.277295e+10	8.088102e+09	2.695345e+09	7.398549e+08	88
31	1.277295e+10	5.641965e+09	2.695345e+09	1.415511e+09	88
32	1.277295e+10	9.388878e+09	2.695345e+09	3.048139e+09	88
33	8.088102e+09	5.641965e+09	7.398549e+08	1.415511e+09	80
34	8.088102e+09	9.388878e+09	7.398549e+08	3.048139e+09	80
35	5.641965e+09	9.388878e+09	1.415511e+09	3.048139e+09	88
36	4.273367e+00	7.226096e+00	3.561606e-02	4.246905e-02	88
37	4.273367e+00	9.319944e+00	3.561606e-02	1.127637e-01	88
38	4.273367e+00	4.833095e+00	3.561606e-02	5.736037e-02	88
39	7.226096e+00	9.319944e+00	4.246905e-02	1.127637e-01	80
40	7.226096e+00	4.833095e+00	4.246905e-02	5.736037e-02	80
41	9.319944e+00	4.833095e+00	1.127637e-01	5.736037e-02	88
42	3.679158e-01	9.707118e-02	3.630858e-01	1.097015e-01	88
43	3.679158e-01	1.198860e-01	3.630858e-01	1.267990e-01	88
44	3.679158e-01	1.812696e-01	3.630858e-01	1.740149e-01	88
45	9.707118e-02	1.198860e-01	1.097015e-01	1.267990e-01	80
46	9.707118e-02	1.812696e-01	1.097015e-01	1.740149e-01	80
47	1.198860e-01	1.812696e-01	1.267990e-01	1.740149e-01	88
48	3.546248e-01	1.860765e-01	3.107514e-01	1.117166e-01	88
49	3.546248e-01	1.367926e-01	3.107514e-01	8.982222e-02	88
50	3.546248e-01	1.347398e-01	3.107514e-01	1.343297e-01	88
51	1.860765e-01	1.367926e-01	1.117166e-01	8.982222e-02	80
52	1.860765e-01	1.347398e-01	1.117166e-01	1.343297e-01	80
53	1.367926e-01	1.347398e-01	8.982222e-02	1.343297e-01	88
54	1.095413e+00	1.026081e+00	2.090215e-01	1.511895e-01	88
55	1.095413e+00	1.198417e+00	2.090215e-01	3.287786e-01	88
56	1.095413e+00	1.033046e+00	2.090215e-01	3.272457e-01	88
57	1.026081e+00	1.198417e+00	1.511895e-01	3.287786e-01	80
58	1.026081e+00	1.033046e+00	1.511895e-01	3.272457e-01	80
59	1.198417e+00	1.033046e+00	3.287786e-01	3.272457e-01	88
60	3.051032e-01	7.407307e-02	3.461640e-01	9.534950e-02	88
61	3.051032e-01	5.953811e-02	3.461640e-01	8.590007e-02	88
62	3.051032e-01	6.096194e-02	3.461640e-01	1.295520e-01	88
63	7.407307e-02	5.953811e-02	9.534950e-02	8.590007e-02	80
64	7.407307e-02	6.096194e-02	9.534950e-02	1.295520e-01	80
65	5.953811e-02	6.096194e-02	8.590007e-02	1.295520e-01	88
66	5.728996e-01	3.985247e-01	2.586751e-01	1.549181e-01	88
67	5.728996e-01	3.658572e-01	2.586751e-01	1.736955e-01	88
68	5.728996e-01	3.319037e-01	2.586751e-01	1.807902e-01	88

69	3.985247e-01	3.658572e-01	1.549181e-01	1.736955e-01	80
70	3.985247e-01	3.319037e-01	1.549181e-01	1.807902e-01	80
71	3.658572e-01	3.319037e-01	1.736955e-01	1.807902e-01	88
72	8.781957e-01	9.144531e-01	4.386598e-01	2.030163e-01	88
73	8.781957e-01	1.165567e+00	4.386598e-01	2.397264e-01	88
74	8.781957e-01	1.046012e+00	4.386598e-01	2.881699e-01	88
75	9.144531e-01	1.165567e+00	2.030163e-01	2.397264e-01	80
76	9.144531e-01	1.046012e+00	2.030163e-01	2.881699e-01	80
77	1.165567e+00	1.046012e+00	2.397264e-01	2.881699e-01	88
78	2.378138e-01	1.262054e-01	1.270707e-01	8.308520e-02	88
79	2.378138e-01	1.519883e-01	1.270707e-01	9.223367e-02	88
80	2.378138e-01	1.196035e-01	1.270707e-01	8.893092e-02	88
81	1.262054e-01	1.519883e-01	8.308520e-02	9.223367e-02	80
82	1.262054e-01	1.196035e-01	8.308520e-02	8.893092e-02	80
83	1.519883e-01	1.196035e-01	9.223367e-02	8.893092e-02	88
84	8.790624e-01	9.697264e-01	5.318228e-02	2.606863e-02	88
85	8.790624e-01	9.547864e-01	5.318228e-02	3.235066e-02	88
86	8.790624e-01	9.612689e-01	5.318228e-02	3.241982e-02	88
87	9.697264e-01	9.547864e-01	2.606863e-02	3.235066e-02	80
88	9.697264e-01	9.612689e-01	2.606863e-02	3.241982e-02	80
89	9.547864e-01	9.612689e-01	3.235066e-02	3.241982e-02	88
90	2.815986e-01	6.352941e-02	3.447873e-01	8.072161e-02	88
91	2.815986e-01	4.801648e-02	3.447873e-01	6.967926e-02	88
92	2.815986e-01	5.264021e-02	3.447873e-01	1.249316e-01	88
93	6.352941e-02	4.801648e-02	8.072161e-02	6.967926e-02	80
94	6.352941e-02	5.264021e-02	8.072161e-02	1.249316e-01	80
95	4.801648e-02	5.264021e-02	6.967926e-02	1.249316e-01	88

	n_model_2	mean_diff	cohens_d	ttest_pval
0	80	5.868757e+09	1.543774	1.851807e-18
1	88	9.219049e+09	3.322503	2.961777e-43
2	88	5.320826e+09	1.450927	7.612526e-18
3	88	3.350292e+09	1.133510	2.123763e-10
4	88	-5.479307e+08	-0.142610	3.589552e-01
5	88	-3.898222e+09	-1.378640	2.599088e-15
6	80	-7.629477e+07	-0.025743	8.711013e-01
7	88	2.789301e+08	0.186939	2.182100e-01
8	88	1.356449e+08	0.068101	6.520340e-01
9	88	3.552248e+08	0.139150	3.929292e-01
10	88	2.119397e+08	0.073381	6.450656e-01
11	88	-1.432851e+08	-0.106653	4.811095e-01
12	80	-5.086364e+01	-8.355988	1.886816e-90
13	88	3.181818e-01	0.066052	6.618335e-01
14	88	-2.117045e+01	-4.398394	9.215735e-69
15	88	5.118182e+01	8.227185	2.036954e-92
16	88	2.969318e+01	4.775472	1.143564e-62
17	88	-2.148864e+01	-4.320228	8.524955e-68
18	80	1.260961e+09	3.571155	4.332445e-54

19	88	1.809355e+09	6.056565	5.553946e-74
20	88	1.801773e+09	6.124173	1.976495e-72
21	88	5.483942e+08	2.109349	1.038791e-25
22	88	5.408122e+08	2.124975	1.991987e-25
23	88	-7.582020e+06	-0.042321	7.792561e-01
24	80	-3.925000e+00	-0.440030	3.929526e-03
25	88	2.056818e+00	0.169771	2.616884e-01
26	88	9.284091e+00	0.719752	3.956313e-06
27	88	5.981818e+00	0.597979	1.086813e-04
28	88	1.320909e+01	1.202731	8.598580e-13
29	88	7.227273e+00	0.530006	5.608076e-04
30	80	4.684852e+09	2.322765	8.042706e-29
31	88	7.130989e+09	3.312522	7.247881e-46
32	88	3.384076e+09	1.176189	5.715397e-13
33	88	2.446137e+09	2.136695	1.594519e-28
34	88	-1.300776e+09	-0.574309	1.895015e-04
35	88	-3.746913e+09	-1.576699	1.068502e-18
36	80	-2.952729e+00	-75.657094	2.351837e-248
37	88	-5.046577e+00	-60.352309	7.143902e-168
38	88	-5.597273e-01	-11.723850	2.343432e-120
39	88	-2.093848e+00	-24.141375	7.374049e-136
40	88	2.393001e+00	47.087189	1.213217e-223
41	88	4.486849e+00	50.155324	2.266405e-191
42	80	2.708446e-01	0.990177	1.254957e-09
43	88	2.480298e-01	0.912055	2.128155e-08
44	88	1.866462e-01	0.655580	2.813334e-05
45	88	-2.281481e-02	-0.191771	2.130662e-01
46	88	-8.419844e-02	-0.572931	2.215139e-04
47	88	-6.138362e-02	-0.403181	8.268438e-03
48	80	1.685483e-01	0.708777	5.856547e-06
49	88	2.178322e-01	0.952357	7.156117e-09
50	88	2.198850e-01	0.918539	1.420898e-08
51	88	4.928390e-02	0.488751	2.087465e-03
52	88	5.133669e-02	0.413728	7.620339e-03
53	88	2.052788e-03	0.017965	9.052994e-01
54	80	6.933222e-02	0.377251	1.422422e-02
55	88	-1.030037e-01	-0.373898	1.425727e-02
56	88	6.236747e-02	0.227144	1.340200e-01
57	88	-1.723359e-01	-0.663170	2.045843e-05
58	88	-6.964743e-03	-0.026906	8.577026e-01
59	88	1.653711e-01	0.504160	1.010546e-03
60	80	2.310301e-01	0.891690	2.877416e-08
61	88	2.455651e-01	0.973697	4.123433e-09
62	88	2.441413e-01	0.934135	1.008213e-08
63	88	1.453496e-02	0.160571	3.025711e-01
64	88	1.311113e-02	0.114452	4.534489e-01
65	88	-1.423834e-03	-0.012954	9.316383e-01
66	80	1.743749e-01	0.808729	3.295645e-07

```
67          88  2.070424e-01   0.939730   4.274439e-09
68          88  2.409959e-01   1.079940   2.947308e-11
69          88  3.266754e-02   0.197954   1.993847e-01
70          88  6.662102e-02   0.394272   1.101874e-02
71          88  3.395348e-02   0.191526   2.056260e-01
72          80 -3.625732e-02  -0.104465   4.867596e-01
73          88 -2.873716e-01  -0.812987   3.021430e-07
74          88 -1.678158e-01  -0.452184   3.166699e-03
75          88 -2.511143e-01  -1.126018   8.851207e-12
76          88 -1.315585e-01  -0.523574   7.346167e-04
77          88  1.195558e-01   0.451057   3.188480e-03
78          80  1.116084e-01   1.029719   2.334366e-10
79          88  8.582543e-02   0.773014   8.445384e-07
80          88  1.182103e-01   1.077858   3.172278e-11
81          88 -2.578296e-02  -0.292993   5.836048e-02
82          88  6.601907e-03   0.076590   6.195587e-01
83          88  3.238487e-02   0.357459   1.882925e-02
84          80 -9.066401e-02  -2.133571   3.319137e-28
85          88 -7.572400e-02  -1.720351   7.128313e-22
86          88 -8.220651e-02  -1.866547   2.014489e-24
87          88  1.494001e-02   0.505958   1.152241e-03
88          88  8.457504e-03   0.286037   6.316345e-02
89          88 -6.482510e-03  -0.200169   1.859940e-01
90          80  2.180692e-01   0.852688   9.730077e-08
91          88  2.335821e-01   0.939098   1.306125e-08
92          88  2.289584e-01   0.882943   5.028529e-08
93          88  1.551292e-02   0.206462   1.863744e-01
94          88  1.088920e-02   0.102523   4.995303e-01
95          88 -4.623723e-03  -0.045711   7.621899e-01
```

[61]:
```
#What does Cliff's Delta calculate?
#It measures the probability that a randomly selected value from group A will
 ↪be larger than a randomly selected value from group B, minus the reverse
 ↪probability.

#Values range from -1 (all values in A are less than all in B) to +1 (all
 ↪values in A are greater than all in B).

#0 means the two groups completely overlap (no effect).

#Positive values: A tends to be greater than B.

#Negative values: A tends to be less than B.



# Interpretation
```

```
# Large effect (|delta|  0.474): Models are substantially different for this␣
↪metric

# Medium/small effect: Moderate/small practical differences

# Negligible: Distributions largely overlap
```

[62]:
```
import pandas as pd
import numpy as np
from itertools import combinations

# --- Pure Python Cliff's Delta implementation ---
def cliffs_delta(x, y):
    """
    Computes Cliff's delta and magnitude for two arrays/lists x, y.
    Returns delta (float) and magnitude (str: 'negligible', 'small', 'medium',␣
↪'large')
    """
    x, y = np.asarray(x), np.asarray(y)
    n_x, n_y = len(x), len(y)
    more = sum(xi > yj for xi in x for yj in y)
    less = sum(xi < yj for xi in x for yj in y)
    delta = (more - less) / (n_x * n_y)
    # Magnitude thresholds (Romano et al., 2006)
    adelta = abs(delta)
    if adelta < 0.147:
        magnitude = "negligible"
    elif adelta < 0.33:
        magnitude = "small"
    elif adelta < 0.474:
        magnitude = "medium"
    else:
        magnitude = "large"
    return delta, magnitude



# --- Specify metrics for which to compute Cliff's delta ---
metrics = [
    "total_duration_ns", "load_duration_ns", "prompt_eval_count",␣
↪"prompt_eval_duration_ns", "eval_count",
    "eval_duration_ns", "tokens_per_second", "levenshtein_similarity",␣
↪"jaccard_similarity", "length_ratio",
    "bleu", "cosine_similarity", "wer", "char_diversity", "type_token_ratio",␣
↪"bigram_overlap"
]
```

```python
# --- Get unique models ---
model_list = sorted(df['model'].unique())

# --- Compute Cliff's delta for all pairs and all metrics ---
results = []
for metric in metrics:
    for m1, m2 in combinations(model_list, 2):
        x = df[df['model'] == m1][metric].dropna()
        y = df[df['model'] == m2][metric].dropna()
        if len(x) > 1 and len(y) > 1:
            delta, magnitude = cliffs_delta(x, y)
            results.append({
                'metric': metric,
                'model_1': m1,
                'model_2': m2,
                "cliffs_delta": delta,
                "magnitude": magnitude,
                "mean_model_1": x.mean(),
                "mean_model_2": y.mean(),
                "n_model_1": len(x),
                "n_model_2": len(y)
            })

cliffs_df = pd.DataFrame(results)

# Show all rows (optional, for Jupyter)
pd.set_option("display.max_rows", None)
print("Table: Cliff’s Delta Effect Sizes for Pairwise Model Comparisons Across All Metrics")
display(cliffs_df)
```

```
Table: Cliff’s Delta Effect Sizes for Pairwise Model Comparisons Across All
Metrics

                metric            model_1            model_2  \
0    total_duration_ns          gemma3:1b  granite3.1-moe:1b
1    total_duration_ns          gemma3:1b         qwen2:0.5b
2    total_duration_ns          gemma3:1b       smollm2:360m
3    total_duration_ns  granite3.1-moe:1b         qwen2:0.5b
4    total_duration_ns  granite3.1-moe:1b       smollm2:360m
5    total_duration_ns         qwen2:0.5b       smollm2:360m
6     load_duration_ns          gemma3:1b  granite3.1-moe:1b
7     load_duration_ns          gemma3:1b         qwen2:0.5b
8     load_duration_ns          gemma3:1b       smollm2:360m
9     load_duration_ns  granite3.1-moe:1b         qwen2:0.5b
10    load_duration_ns  granite3.1-moe:1b       smollm2:360m
11    load_duration_ns         qwen2:0.5b       smollm2:360m
12   prompt_eval_count          gemma3:1b  granite3.1-moe:1b
```

13	prompt_eval_count	gemma3:1b	qwen2:0.5b
14	prompt_eval_count	gemma3:1b	smollm2:360m
15	prompt_eval_count	granite3.1-moe:1b	qwen2:0.5b
16	prompt_eval_count	granite3.1-moe:1b	smollm2:360m
17	prompt_eval_count	qwen2:0.5b	smollm2:360m
18	prompt_eval_duration_ns	gemma3:1b	granite3.1-moe:1b
19	prompt_eval_duration_ns	gemma3:1b	qwen2:0.5b
20	prompt_eval_duration_ns	gemma3:1b	smollm2:360m
21	prompt_eval_duration_ns	granite3.1-moe:1b	qwen2:0.5b
22	prompt_eval_duration_ns	granite3.1-moe:1b	smollm2:360m
23	prompt_eval_duration_ns	qwen2:0.5b	smollm2:360m
24	eval_count	gemma3:1b	granite3.1-moe:1b
25	eval_count	gemma3:1b	qwen2:0.5b
26	eval_count	gemma3:1b	smollm2:360m
27	eval_count	granite3.1-moe:1b	qwen2:0.5b
28	eval_count	granite3.1-moe:1b	smollm2:360m
29	eval_count	qwen2:0.5b	smollm2:360m
30	eval_duration_ns	gemma3:1b	granite3.1-moe:1b
31	eval_duration_ns	gemma3:1b	qwen2:0.5b
32	eval_duration_ns	gemma3:1b	smollm2:360m
33	eval_duration_ns	granite3.1-moe:1b	qwen2:0.5b
34	eval_duration_ns	granite3.1-moe:1b	smollm2:360m
35	eval_duration_ns	qwen2:0.5b	smollm2:360m
36	tokens_per_second	gemma3:1b	granite3.1-moe:1b
37	tokens_per_second	gemma3:1b	qwen2:0.5b
38	tokens_per_second	gemma3:1b	smollm2:360m
39	tokens_per_second	granite3.1-moe:1b	qwen2:0.5b
40	tokens_per_second	granite3.1-moe:1b	smollm2:360m
41	tokens_per_second	qwen2:0.5b	smollm2:360m
42	levenshtein_similarity	gemma3:1b	granite3.1-moe:1b
43	levenshtein_similarity	gemma3:1b	qwen2:0.5b
44	levenshtein_similarity	gemma3:1b	smollm2:360m
45	levenshtein_similarity	granite3.1-moe:1b	qwen2:0.5b
46	levenshtein_similarity	granite3.1-moe:1b	smollm2:360m
47	levenshtein_similarity	qwen2:0.5b	smollm2:360m
48	jaccard_similarity	gemma3:1b	granite3.1-moe:1b
49	jaccard_similarity	gemma3:1b	qwen2:0.5b
50	jaccard_similarity	gemma3:1b	smollm2:360m
51	jaccard_similarity	granite3.1-moe:1b	qwen2:0.5b
52	jaccard_similarity	granite3.1-moe:1b	smollm2:360m
53	jaccard_similarity	qwen2:0.5b	smollm2:360m
54	length_ratio	gemma3:1b	granite3.1-moe:1b
55	length_ratio	gemma3:1b	qwen2:0.5b
56	length_ratio	gemma3:1b	smollm2:360m
57	length_ratio	granite3.1-moe:1b	qwen2:0.5b
58	length_ratio	granite3.1-moe:1b	smollm2:360m
59	length_ratio	qwen2:0.5b	smollm2:360m
60	bleu	gemma3:1b	granite3.1-moe:1b

61	bleu	gemma3:1b	qwen2:0.5b
62	bleu	gemma3:1b	smollm2:360m
63	bleu	granite3.1-moe:1b	qwen2:0.5b
64	bleu	granite3.1-moe:1b	smollm2:360m
65	bleu	qwen2:0.5b	smollm2:360m
66	cosine_similarity	gemma3:1b	granite3.1-moe:1b
67	cosine_similarity	gemma3:1b	qwen2:0.5b
68	cosine_similarity	gemma3:1b	smollm2:360m
69	cosine_similarity	granite3.1-moe:1b	qwen2:0.5b
70	cosine_similarity	granite3.1-moe:1b	smollm2:360m
71	cosine_similarity	qwen2:0.5b	smollm2:360m
72	wer	gemma3:1b	granite3.1-moe:1b
73	wer	gemma3:1b	qwen2:0.5b
74	wer	gemma3:1b	smollm2:360m
75	wer	granite3.1-moe:1b	qwen2:0.5b
76	wer	granite3.1-moe:1b	smollm2:360m
77	wer	qwen2:0.5b	smollm2:360m
78	char_diversity	gemma3:1b	granite3.1-moe:1b
79	char_diversity	gemma3:1b	qwen2:0.5b
80	char_diversity	gemma3:1b	smollm2:360m
81	char_diversity	granite3.1-moe:1b	qwen2:0.5b
82	char_diversity	granite3.1-moe:1b	smollm2:360m
83	char_diversity	qwen2:0.5b	smollm2:360m
84	type_token_ratio	gemma3:1b	granite3.1-moe:1b
85	type_token_ratio	gemma3:1b	qwen2:0.5b
86	type_token_ratio	gemma3:1b	smollm2:360m
87	type_token_ratio	granite3.1-moe:1b	qwen2:0.5b
88	type_token_ratio	granite3.1-moe:1b	smollm2:360m
89	type_token_ratio	qwen2:0.5b	smollm2:360m
90	bigram_overlap	gemma3:1b	granite3.1-moe:1b
91	bigram_overlap	gemma3:1b	qwen2:0.5b
92	bigram_overlap	gemma3:1b	smollm2:360m
93	bigram_overlap	granite3.1-moe:1b	qwen2:0.5b
94	bigram_overlap	granite3.1-moe:1b	smollm2:360m
95	bigram_overlap	qwen2:0.5b	smollm2:360m

	cliffs_delta	magnitude	mean_model_1	mean_model_2	n_model_1	n_model_2
0	0.809943	large	1.637960e+10	1.051084e+10	88	80
1	0.993027	large	1.637960e+10	7.160551e+09	88	88
2	0.778151	large	1.637960e+10	1.105877e+10	88	88
3	0.934943	large	1.051084e+10	7.160551e+09	80	88
4	-0.219034	small	1.051084e+10	1.105877e+10	80	88
5	-0.594525	large	7.160551e+09	1.105877e+10	88	88
6	0.975000	large	3.576625e+08	4.339573e+08	88	80
7	0.977531	large	3.576625e+08	7.873249e+07	88	88
8	0.977531	large	3.576625e+08	2.220176e+08	88	88
9	-0.975000	large	4.339573e+08	7.873249e+07	80	88
10	-0.015909	negligible	4.339573e+08	2.220176e+08	80	88

11	0.977273	large	7.873249e+07	2.220176e+08	88	88
12	-1.000000	large	6.051136e+01	1.113750e+02	88	80
13	0.030088	negligible	6.051136e+01	6.019318e+01	88	88
14	-0.999742	large	6.051136e+01	8.168182e+01	88	88
15	1.000000	large	1.113750e+02	6.019318e+01	80	88
16	1.000000	large	1.113750e+02	8.168182e+01	80	88
17	-1.000000	large	6.019318e+01	8.168182e+01	88	88
18	0.978125	large	3.247818e+09	1.986857e+09	88	80
19	1.000000	large	3.247818e+09	1.438463e+09	88	88
20	1.000000	large	3.247818e+09	1.446045e+09	88	88
21	0.906250	large	1.986857e+09	1.438463e+09	80	88
22	0.935227	large	1.986857e+09	1.446045e+09	80	88
23	-0.059143	negligible	1.438463e+09	1.446045e+09	88	88
24	-0.100000	negligible	5.450000e+01	5.842500e+01	88	80
25	0.121772	negligible	5.450000e+01	5.244318e+01	88	88
26	0.428848	medium	5.450000e+01	4.521591e+01	88	88
27	0.237358	small	5.842500e+01	5.244318e+01	80	88
28	0.580966	large	5.842500e+01	4.521591e+01	80	88
29	0.324768	small	5.244318e+01	4.521591e+01	88	88
30	0.803693	large	1.277295e+10	8.088102e+09	88	80
31	0.887138	large	1.277295e+10	5.641965e+09	88	88
32	0.715909	large	1.277295e+10	9.388878e+09	88	88
33	0.909943	large	8.088102e+09	5.641965e+09	80	88
34	-0.284091	small	8.088102e+09	9.388878e+09	80	88
35	-0.608471	large	5.641965e+09	9.388878e+09	88	88
36	-1.000000	large	4.273367e+00	7.226096e+00	88	80
37	-1.000000	large	4.273367e+00	9.319944e+00	88	88
38	-1.000000	large	4.273367e+00	4.833095e+00	88	88
39	-1.000000	large	7.226096e+00	9.319944e+00	80	88
40	1.000000	large	7.226096e+00	4.833095e+00	80	88
41	1.000000	large	9.319944e+00	4.833095e+00	88	88
42	0.417898	medium	3.679158e-01	9.707118e-02	88	80
43	0.362087	medium	3.679158e-01	1.198860e-01	88	88
44	0.196152	small	3.679158e-01	1.812696e-01	88	88
45	-0.105256	negligible	9.707118e-02	1.198860e-01	80	88
46	-0.385085	medium	9.707118e-02	1.812696e-01	80	88
47	-0.251033	small	1.198860e-01	1.812696e-01	88	88
48	0.249148	small	3.546248e-01	1.860765e-01	88	80
49	0.462939	medium	3.546248e-01	1.367926e-01	88	88
50	0.522082	large	3.546248e-01	1.347398e-01	88	88
51	0.292330	small	1.860765e-01	1.367926e-01	80	88
52	0.377273	medium	1.860765e-01	1.347398e-01	80	88
53	0.098011	negligible	1.367926e-01	1.347398e-01	88	88
54	0.263494	small	1.095413e+00	1.026081e+00	88	80
55	-0.307076	small	1.095413e+00	1.198417e+00	88	88
56	0.133135	negligible	1.095413e+00	1.033046e+00	88	88
57	-0.453409	medium	1.026081e+00	1.198417e+00	80	88
58	-0.008381	negligible	1.026081e+00	1.033046e+00	80	88

```
59       0.319473       small  1.198417e+00  1.033046e+00          88          88
60       0.381960      medium  3.051032e-01  7.407307e-02          88          80
61       0.506586       large  3.051032e-01  5.953811e-02          88          88
62       0.537319       large  3.051032e-01  6.096194e-02          88          88
63       0.231534       small  7.407307e-02  5.953811e-02          80          88
64       0.279403       small  7.407307e-02  6.096194e-02          80          88
65       0.048425  negligible  5.953811e-02  6.096194e-02          88          88
66       0.388068      medium  5.728996e-01  3.985247e-01          88          80
67       0.510072       large  5.728996e-01  3.658572e-01          88          88
68       0.584065       large  5.728996e-01  3.319037e-01          88          88
69       0.177273       small  3.985247e-01  3.658572e-01          80          88
70       0.297301       small  3.985247e-01  3.319037e-01          80          88
71       0.131973  negligible  3.658572e-01  3.319037e-01          88          88
72       0.149290       small  8.781957e-01  9.144531e-01          88          80
73      -0.323735       small  8.781957e-01  1.165567e+00          88          88
74      -0.070377  negligible  8.781957e-01  1.046012e+00          88          88
75      -0.598580       large  9.144531e-01  1.165567e+00          80          88
76      -0.296733       small  9.144531e-01  1.046012e+00          80          88
77       0.299458       small  1.165567e+00  1.046012e+00          88          88
78       0.514773       large  2.378138e-01  1.262054e-01          88          80
79       0.427686      medium  2.378138e-01  1.519883e-01          88          88
80       0.525956       large  2.378138e-01  1.196035e-01          88          88
81      -0.140341  negligible  1.262054e-01  1.519883e-01          80          88
82       0.097869  negligible  1.262054e-01  1.196035e-01          80          88
83       0.228564       small  1.519883e-01  1.196035e-01          88          88
84      -0.903835       large  8.790624e-01  9.697264e-01          88          80
85      -0.819473       large  8.790624e-01  9.547864e-01          88          88
86      -0.845558       large  8.790624e-01  9.612689e-01          88          88
87       0.266051       small  9.697264e-01  9.547864e-01          80          88
88       0.148438       small  9.697264e-01  9.612689e-01          80          88
89      -0.113895  negligible  9.547864e-01  9.612689e-01          88          88
90       0.371165      medium  2.815986e-01  6.352941e-02          88          80
91       0.477402       large  2.815986e-01  4.801648e-02          88          88
92       0.509168       large  2.815986e-01  5.264021e-02          88          88
93       0.175142       small  6.352941e-02  4.801648e-02          80          88
94       0.225426       small  6.352941e-02  5.264021e-02          80          88
95       0.051911  negligible  4.801648e-02  5.264021e-02          88          88
```

[63]:
```python
#Association between Text Augmentation with Prompts

import pandas as pd
import numpy as np
from scipy.stats import chi2_contingency
from sklearn.metrics import mutual_info_score

# Optional: If you don’t have `theils_u` function, define it:
def theils_u(x, y):
```

```python
    """Theil's U (uncertainty coefficient U(x|y)), asymmetric"""
    s_xy = mutual_info_score(x, y)
    s_x = entropy_from_series(x)
    return 0 if s_x == 0 else s_xy / s_x

def entropy_from_series(s):
    p = s.value_counts(normalize=True)
    return -np.sum(p * np.log2(p + 1e-12))

# Build contingency table
contingency = pd.crosstab(df['augmentation_type'], df['prompt_id'])

# 1. Chi-square Test
chi2_stat, p_chi2, dof, expected = chi2_contingency(contingency)

# 2. Cramér's V
n = contingency.values.sum()
cramers_v = np.sqrt(chi2_stat / (n * (min(contingency.shape) - 1)))

# 3. Mutual Information Score
mi = mutual_info_score(df['augmentation_type'], df['prompt_id'])

# 4. Theil's U (both directions, asymmetric)
u_aug_given_prompt = theils_u(df['augmentation_type'], df['prompt_id'])
u_prompt_given_aug = theils_u(df['prompt_id'], df['augmentation_type'])

# 5. Summary Table
summary = pd.DataFrame({
    'Test': [
        "Chi-square Test of Independence",
        "Cramér's V (Effect Size)",
        "Mutual Information Score",
        "Theil's U (Augmentation|Prompt)",
        "Theil's U (Prompt|Augmentation)"
    ],
    'Statistic/Value': [
        f"Chi2={chi2_stat:.5g}, dof={dof}, p={p_chi2:.5g}",
        f"{cramers_v:.5f}",
        f"{mi:.5f}",
        f"{u_aug_given_prompt:.5f}",
        f"{u_prompt_given_aug:.5f}"
    ]
})

print("Association between Text Augmentation with Prompts")
display(summary)
```

```
Association between Text Augmentation with Prompts
```

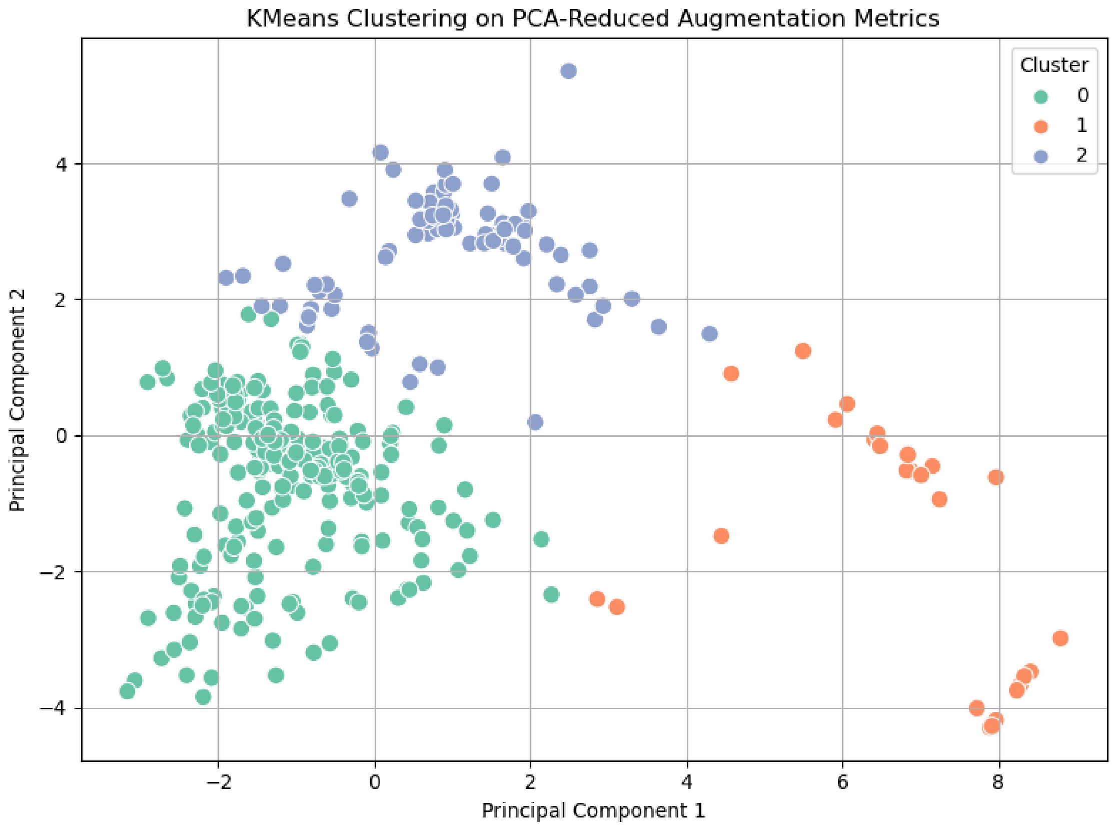
```
                                 Test          Statistic/Value
0  Chi-square Test of Independence  Chi2=0.86154, dof=70, p=1
1          Cramér's V (Effect Size)                   0.01892
2          Mutual Information Score                   0.00125
3  Theil's U (Augmentation|Prompt)                    0.00036
4  Theil's U (Prompt|Augmentation)                    0.00042
```

[69]:

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import seaborn as sns

# 2. Select Numeric Columns for PCA
numeric_cols = [
    'total_duration_ns', 'load_duration_ns', 'prompt_eval_count',
    'prompt_eval_duration_ns', 'eval_count', 'eval_duration_ns',
    'tokens_per_second', 'levenshtein_similarity', 'jaccard_similarity',
    'length_ratio', 'bleu', 'cosine_similarity', 'wer', 'char_diversity',
    'type_token_ratio', 'bigram_overlap'
]
df_numeric = df[numeric_cols].dropna()

# 3. Standardize
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df_numeric)

# 4. PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
df_pca = pd.DataFrame(X_pca, columns=['PC1', 'PC2'])

# 5. KMeans
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
clusters = kmeans.fit_predict(X_pca)
df_pca['Cluster'] = clusters

# 6. Plotting
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df_pca, x='PC1', y='PC2', hue='Cluster', palette='Set2',␣
 ↪s=80)
plt.title('KMeans Clustering on PCA-Reduced Augmentation Metrics')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.grid(True)
plt.tight_layout()
```

```python
plt.show()



# Print the cluster assignments for the first few rows and the centroids
print("Cluster assignments (first 10):")
print(df_pca['Cluster'].head(10))

print("\nKMeans cluster centers in PCA space:")
print(kmeans.cluster_centers_)

# Count the number of points in each cluster
print("\nNumber of points in each cluster:")
print(df_pca['Cluster'].value_counts())

loadings = pd.DataFrame(
    pca.components_.T,  # shape: (n_features, n_components)
    index=numeric_cols,
    columns=['PC1', 'PC2']
)
print(loadings)
```

```
C:\Users\parth\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=2.
  warnings.warn(
```




```
Cluster assignments (first 10):
0    0
1    0
2    0
3    0
4    0
5    0
6    0
7    0
8    0
9    0
Name: Cluster, dtype: int32

KMeans cluster centers in PCA space:
[[-1.10796476 -0.66622526]
 [ 6.82928688 -1.79858054]
 [ 0.9291741   2.67848547]]

Number of points in each cluster:
Cluster
0    238
```

```
2     78
1     28
Name: count, dtype: int64
                              PC1       PC2
total_duration_ns        0.226049  0.368875
load_duration_ns         0.028109  0.105715
prompt_eval_count       -0.091955 -0.058772
prompt_eval_duration_ns  0.271370  0.210033
eval_count               0.062268  0.354597
eval_duration_ns         0.230526  0.390617
tokens_per_second       -0.216960 -0.162517
levenshtein_similarity   0.313847 -0.260757
jaccard_similarity       0.370734 -0.155544
length_ratio             0.048719  0.257775
bleu                     0.374915 -0.149463
cosine_similarity        0.342834 -0.069804
wer                     -0.272953  0.314114
char_diversity           0.010643  0.364538
type_token_ratio        -0.228997 -0.236165
bigram_overlap           0.372093 -0.165159
```

[ ]:

```

```