param_exp_results-with_clustering_metrics-50train50run

June 22, 2023

[ ]:
```
import sys
sys.path.insert(1, '/Users/madisonthantu/Desktop/DREAM/t-recs')

import trecs

import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
from scipy.ndimage import gaussian_filter1d
from collections import defaultdict
from chaney_utils import (
    load_sim_results,
    graph_relative_to_ideal,
    merge_results,
    graph_metrics,
    graph_metrics_by_axis,
    graph_relative_to_global_by_axis,
    transform_relative_to_global,
    graph_histogram_metric_by_axis
)
import warnings
warnings.simplefilter("ignore")

import itertools
```

```
/Users/madisonthantu/Desktop/DREAM/T-RECS-RS-
research/prelim_experiments/param_experiments/chaney_utils.py:61: SyntaxWarning:
"is" with a literal. Did you mean "=="?
  if model_key is "ideal" and not absolute_measure:
```

[ ]:
```
DEBUG = True # don't save figures
```

[ ]:
```
# results = merge_results(["param_exp_results/repeated_training",␣
  ↪"param_exp_results/single_training"])

results_path = "param_exp_results/with_clustering_metrics/10train90run"
```

```
results = merge_results([results_path])
print(results.keys())

model_keys = ['0.0drift_0attention_0retraining', '0.
  ↪0drift_0attention_1retraining', '0.0drift_-0.8attention_0retraining', '0.
  ↪0drift_-0.8attention_1retraining', '0.05drift_0attention_0retraining', '0.
  ↪05drift_0attention_1retraining', '0.05drift_-0.8attention_0retraining', '0.
  ↪05drift_-0.8attention_1retraining', '0.1drift_0attention_0retraining', '0.
  ↪1drift_0attention_1retraining', '0.1drift_-0.8attention_0retraining', '0.
  ↪1drift_-0.8attention_1retraining']
print(model_keys)

id_to_readable = dict(zip(model_keys, model_keys))
print(id_to_readable)
```

```
dict_keys(['mse', 'interaction_spread', 'global_interaction_similarity',
'inter_cluster_interaction_similarity', 'intra_cluster_interaction_similarity',
'mean_global_cosine_sim', 'mean_intra_cluster_cosine_sim',
'mean_inter_cluster_cosine_sim', 'mean_cosine_sim_per_cluster',
'mean_cluster_distance_from_centroid', 'mean_global_distance_from_centroid',
'mean_distance_from_centroid_per_cluster'])
['0.0drift_0attention_0retraining', '0.0drift_0attention_1retraining',
'0.0drift_-0.8attention_0retraining', '0.0drift_-0.8attention_1retraining',
'0.05drift_0attention_0retraining', '0.05drift_0attention_1retraining',
'0.05drift_-0.8attention_0retraining', '0.05drift_-0.8attention_1retraining',
'0.1drift_0attention_0retraining', '0.1drift_0attention_1retraining',
'0.1drift_-0.8attention_0retraining', '0.1drift_-0.8attention_1retraining']
{'0.0drift_0attention_0retraining': '0.0drift_0attention_0retraining',
'0.0drift_0attention_1retraining': '0.0drift_0attention_1retraining',
'0.0drift_-0.8attention_0retraining': '0.0drift_-0.8attention_0retraining',
'0.0drift_-0.8attention_1retraining': '0.0drift_-0.8attention_1retraining',
'0.05drift_0attention_0retraining': '0.05drift_0attention_0retraining',
'0.05drift_0attention_1retraining': '0.05drift_0attention_1retraining',
'0.05drift_-0.8attention_0retraining': '0.05drift_-0.8attention_0retraining',
'0.05drift_-0.8attention_1retraining': '0.05drift_-0.8attention_1retraining',
'0.1drift_0attention_0retraining': '0.1drift_0attention_0retraining',
'0.1drift_0attention_1retraining': '0.1drift_0attention_1retraining',
'0.1drift_-0.8attention_0retraining': '0.1drift_-0.8attention_0retraining',
'0.1drift_-0.8attention_1retraining': '0.1drift_-0.8attention_1retraining'}
```

[ ]:
```
hyper_params = {"drift":[0.0, 0.05, 0.1], "attention_exp":[0, -0.8],␣
  ↪"repeated_training":[0,1]}
models = dict([(f"{p[0]}drift_{p[1]}attention_{p[2]}retraining", p) for p in␣
  ↪itertools.product(*hyper_params.values())])

results_df = pd.DataFrame(columns=["drift", "attention_exp",␣
  ↪"repeated_training"] + list(results.keys()))
```

```
results_df

for params in model_keys:
    df = pd.DataFrame(columns=["drift", "attention_exp", "repeated_training"] +␣
 ↪list(results.keys()))
    for metric in results:
        metric_results = [vals for sim_trial in results[metric][params] for␣
 ↪vals in sim_trial]
        df[metric] = metric_results
    df["drift"] = models[params][0]
    df["attention_exp"] = models[params][1]
    df["repeated_training"] = models[params][2]
    results_df = pd.concat([results_df, df])
```

[ ]:
```
print(results_df.shape)
results_df.head()
# len(models)
```

```
(3600, 15)
```

[ ]:

	drift	attention_exp	repeated_training	mse	interaction_spread	\
0	0.0	0	0	0.090065	-938.5	
1	0.0	0	0	0.090065	0.0	
2	0.0	0	0	0.090065	-1.0	
3	0.0	0	0	0.090065	1.0	
4	0.0	0	0	0.090065	0.0	

	global_interaction_similarity	inter_cluster_interaction_similarity	\
0	0.001601	0.001386	
1	0.002274	0.002032	
2	0.003058	0.002794	
3	0.003888	0.003562	
4	0.004737	0.004383	

	intra_cluster_interaction_similarity	mean_global_cosine_sim	\
0	0.001954	0.244844	
1	0.002672	0.244844	
2	0.003493	0.244844	
3	0.004424	0.244844	
4	0.005318	0.244844	

	mean_intra_cluster_cosine_sim	mean_inter_cluster_cosine_sim	\
0	0.318086	0.200335	
1	0.318086	0.200335	
2	0.318086	0.200335	
3	0.318086	0.200335	
4	0.318086	0.200335	

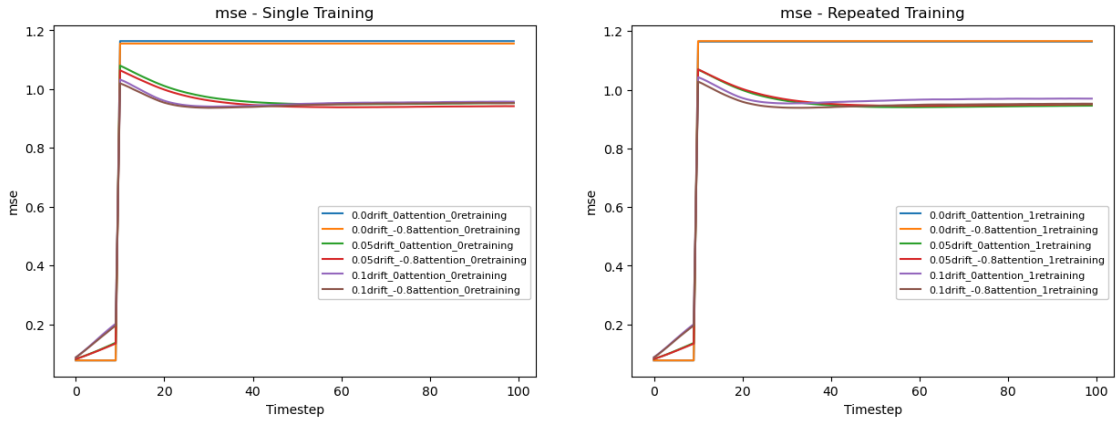
```
                          mean_cosine_sim_per_cluster  \
0  [0.5116548734034113, 0.9312028843954681, 0.727…
1  [0.5116548734034113, 0.9312028843954681, 0.727…
2  [0.5116548734034113, 0.9312028843954681, 0.727…
3  [0.5116548734034113, 0.9312028843954681, 0.727…
4  [0.5116548734034113, 0.9312028843954681, 0.727…

   mean_cluster_distance_from_centroid  mean_global_distance_from_centroid  \
0                              0.43184                            0.687343
1                              0.43184                            0.687343
2                              0.43184                            0.687343
3                              0.43184                            0.687343
4                              0.43184                            0.687343

            mean_distance_from_centroid_per_cluster
0  [0.46841372717852575, 0.777901889986511, 0.688…
1  [0.46841372717852575, 0.777901889986511, 0.688…
2  [0.46841372717852575, 0.777901889986511, 0.688…
3  [0.46841372717852575, 0.777901889986511, 0.688…
4  [0.46841372717852575, 0.777901889986511, 0.688…
```

[ ]:
```python
y_labels = dict([
    ("mse", "mse"),
    ("interaction_spread", "interaction spread"),
    ("global_interaction_similarity", "average jacard similarity"),
    ("inter_cluster_interaction_similarity", "average jacard similarity"),
    ("intra_cluster_interaction_similarity", "average jacard similarity"),
    ("mean_global_cosine_sim", "average cosine sim"),
    ("mean_intra_cluster_cosine_sim", "average cosine sim"),
    ("mean_inter_cluster_cosine_sim", "average cosine sim"),
    ("mean_cosine_sim_per_cluster", "average cosine sim"),
    ("mean_cluster_distance_from_centroid", "distance"),
    ("mean_global_distance_from_centroid", "distance"),
    ("mean_distance_from_centroid_per_cluster", "distance"),
])

model_key_pairs = [(model_keys[i], model_keys[i+1]) for i in range(0,␣
 ↪len(model_keys), 2)]

single_training_keys = [key[0] for key in model_key_pairs]
repeated_training_keys = [key[1] for key in model_key_pairs]
```

1 Graphing MSE

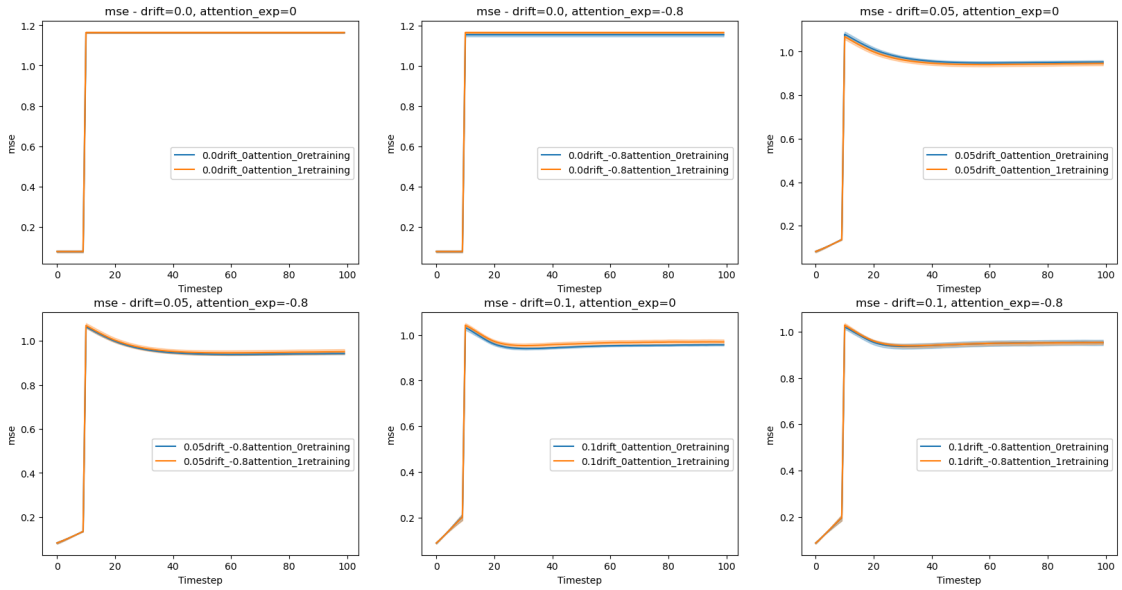
[ ]:
```
metric_key = "mse"

fig, axs = plt.subplots(1, 2, figsize=(15, 5))

graph_metrics_by_axis(axs[0], results, metric_key, single_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[0].set_ylabel(y_labels[metric_key])
axs[0].set_xlabel("Timestep")
axs[0].set_title(f"{metric_key} - Single Training")
axs[0].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)

graph_metrics_by_axis(axs[1], results, metric_key, repeated_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[1].set_ylabel(y_labels[metric_key])
axs[1].set_xlabel("Timestep")
axs[1].set_title(f"{metric_key} - Repeated Training")
axs[1].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)
```

[ ]:
```
<matplotlib.legend.Legend at 0x16a09abb0>
```



[ ]:
```
metric_key = "mse"

fig, axs = plt.subplots(2, 3, figsize=(20, 10))

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_key, model_key_pairs[i],␣
 ↪id_to_readable, mult_sd=0.5)
```

```
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"{metric_key} -␣
↪drift={models[model_key_pairs[i][0]][0]},␣
↪attention_exp={models[model_key_pairs[i][0]][1]}")
```



[ ]:
```
# single_training_keys = ['0.0drift_0attention_0retraining', '0.0drift_-0.
↪8attention_0retraining', '0.05drift_0attention_0retraining', '0.05drift_-0.
↪8attention_0retraining', '0.1drift_0attention_0retraining', '0.1drift_-0.
↪8attention_0retraining']
# print(single_training_keys)

# metric_key = "mse"
# # graph_relative_to_ideal(results, "mse", single_training_keys,␣
↪id_to_readable, absolute_measure=False, mult_sd=1.0)
# graph_metrics(results, metric_key, single_training_keys, id_to_readable,␣
↪mult_sd=0.5)
# plt.ylabel(y_labels[metric_key])
# plt.xlabel("Timestep")
# plt.legend(facecolor='white', framealpha=1, loc='upper right',␣
↪bbox_to_anchor=(1.7, 1.0))
# plt.ylim(-0.1, 1.75)
# plt.xlim(0, 60)
# plt.title(f"Single Training - {metric_key}")
# if not DEBUG:
#     plt.savefig("figures/repeated_training_sim_pair.pdf", bbox_inches =␣
↪"tight")
```

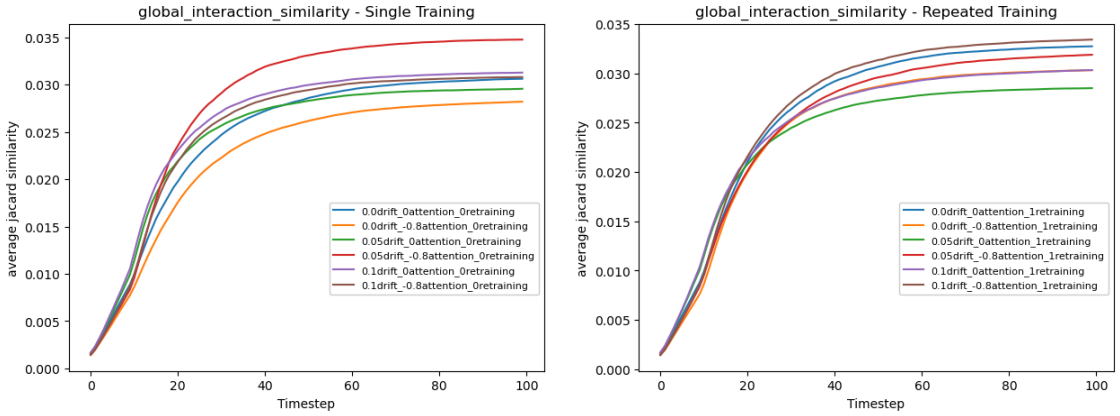
```
# repeated_training_keys = list(set(model_keys) - set(single_training_keys))
# print(repeated_training_keys)

# metric_key = "mse"
# # graph_relative_to_ideal(results, "mse", single_training_keys,␣
 ↪id_to_readable, absolute_measure=False, mult_sd=1.0)
# graph_metrics(results, metric_key, repeated_training_keys, id_to_readable,␣
 ↪mult_sd=0.5)
# plt.ylabel("Change in Jaccard Index")
# plt.xlabel("Timestep")
# plt.legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1.7, 1.0))
# plt.ylim(-0.1, 1.75)
# plt.xlim(0, 60)
# plt.title(f"Repeated Training - {metric_key}")
# if not DEBUG:
#     plt.savefig("figures/repeated_training_sim_pair.pdf", bbox_inches =␣
 ↪"tight")
```

2 Graphing Interaction Similarity

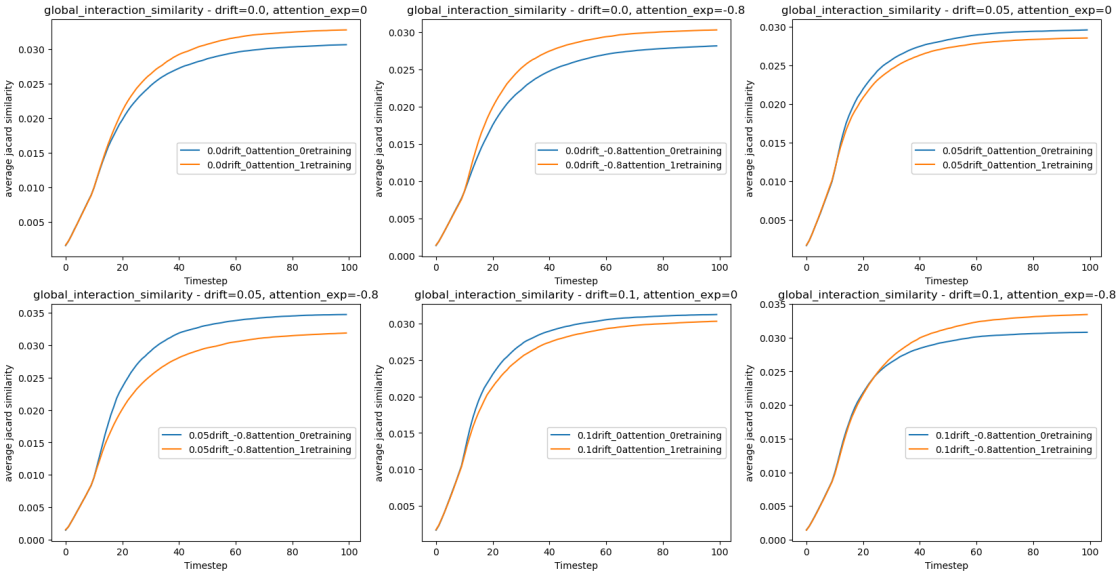
2.0.1 Graphing Global Interaction Similarity

[ ]:
```
metric_key = "global_interaction_similarity"

fig, axs = plt.subplots(1, 2, figsize=(15, 5))

graph_metrics_by_axis(axs[0], results, metric_key, single_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[0].set_ylabel(y_labels[metric_key])
axs[0].set_xlabel("Timestep")
axs[0].set_title(f"{metric_key} - Single Training")
axs[0].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)

graph_metrics_by_axis(axs[1], results, metric_key, repeated_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[1].set_ylabel(y_labels[metric_key])
axs[1].set_xlabel("Timestep")
axs[1].set_title(f"{metric_key} - Repeated Training")
axs[1].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)
```

[ ]:
```
<matplotlib.legend.Legend at 0x16b281d00>
```



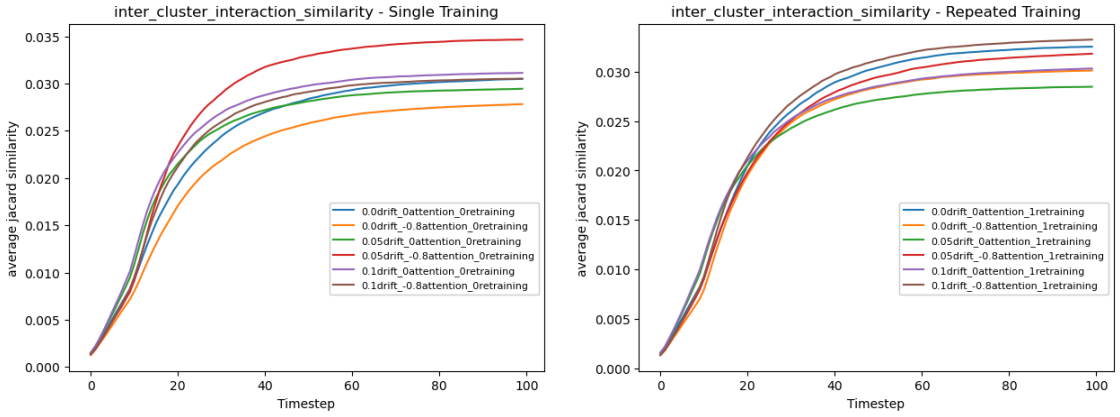
[ ]:
```
metric_key = "global_interaction_similarity"

fig, axs = plt.subplots(2, 3, figsize=(20, 10))

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_key, model_key_pairs[i],␣
 ↪id_to_readable, mult_sd=0)
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"{metric_key} -␣
 ↪drift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```



2.0.2 Graphing Inter -Cluster Interaction Similarity

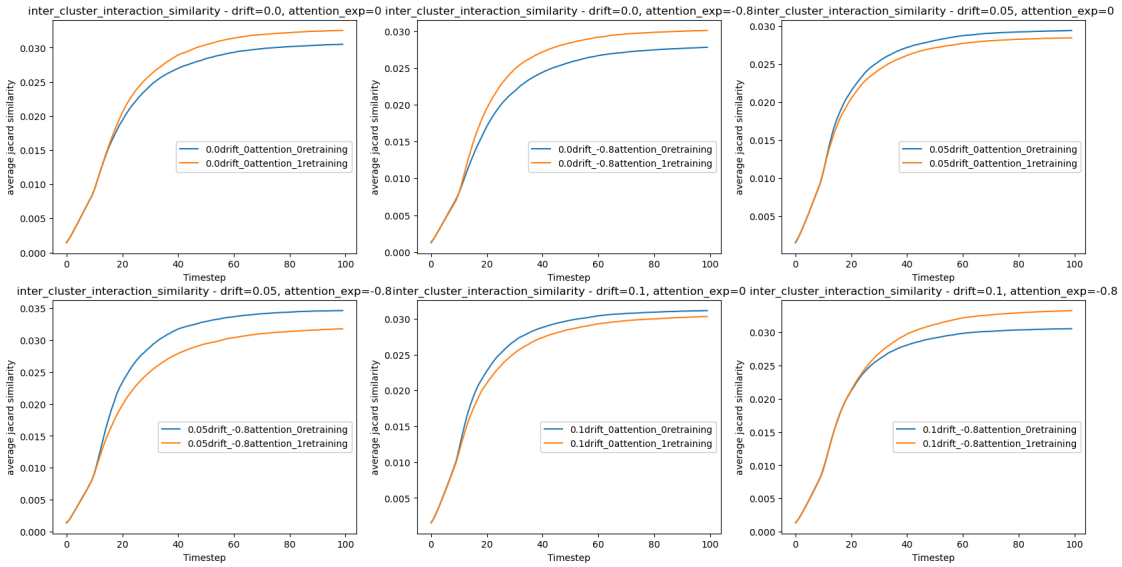
[ ]:
```
metric_key = "inter_cluster_interaction_similarity"

fig, axs = plt.subplots(1, 2, figsize=(15, 5))

graph_metrics_by_axis(axs[0], results, metric_key, single_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[0].set_ylabel(y_labels[metric_key])
axs[0].set_xlabel("Timestep")
axs[0].set_title(f"{metric_key} - Single Training")
axs[0].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)

graph_metrics_by_axis(axs[1], results, metric_key, repeated_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[1].set_ylabel(y_labels[metric_key])
axs[1].set_xlabel("Timestep")
axs[1].set_title(f"{metric_key} - Repeated Training")
axs[1].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)
```

[ ]:
```
<matplotlib.legend.Legend at 0x16a862a90>
```



[ ]:
```
metric_key = "inter_cluster_interaction_similarity"

fig, axs = plt.subplots(2, 3, figsize=(20, 10))

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_key, model_key_pairs[i],␣
 ↪id_to_readable, mult_sd=0)
```

```
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"{metric_key} -␣
 ↪drift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

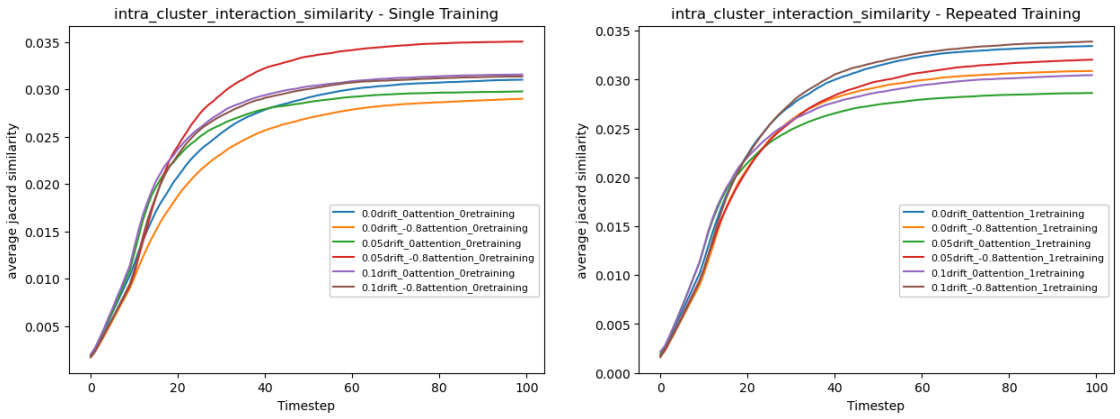


2.0.3 Graphing Intra -Cluster Interaction Similarity

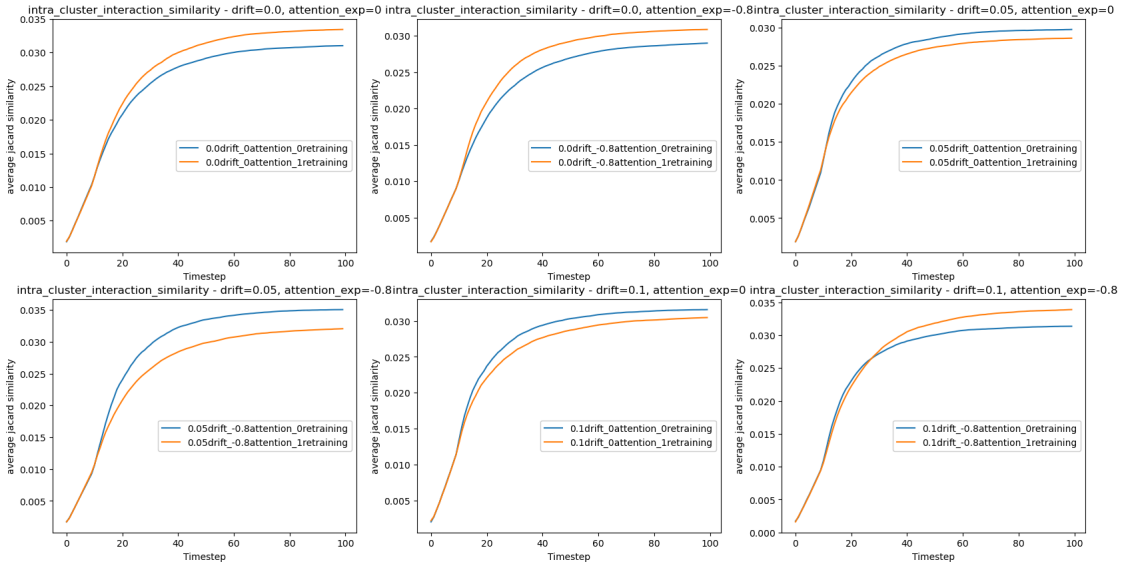
[ ]:
```
metric_key = "intra_cluster_interaction_similarity"

single_training_keys = [key[0] for key in model_key_pairs]
repeated_training_keys = [key[1] for key in model_key_pairs]

fig, axs = plt.subplots(1, 2, figsize=(15, 5))

graph_metrics_by_axis(axs[0], results, metric_key, single_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[0].set_ylabel(y_labels[metric_key])
axs[0].set_xlabel("Timestep")
axs[0].set_title(f"{metric_key} - Single Training")
axs[0].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)

graph_metrics_by_axis(axs[1], results, metric_key, repeated_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[1].set_ylabel(y_labels[metric_key])
axs[1].set_xlabel("Timestep")
axs[1].set_title(f"{metric_key} - Repeated Training")
```

```
axs[1].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)
```

[ ]: <matplotlib.legend.Legend at 0x16b6effd0>



[ ]:
```
metric_key = "intra_cluster_interaction_similarity"

fig, axs = plt.subplots(2, 3, figsize=(20, 10))

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_key, model_key_pairs[i],␣
 ↪id_to_readable, mult_sd=0)
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"{metric_key} -␣
 ↪drift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

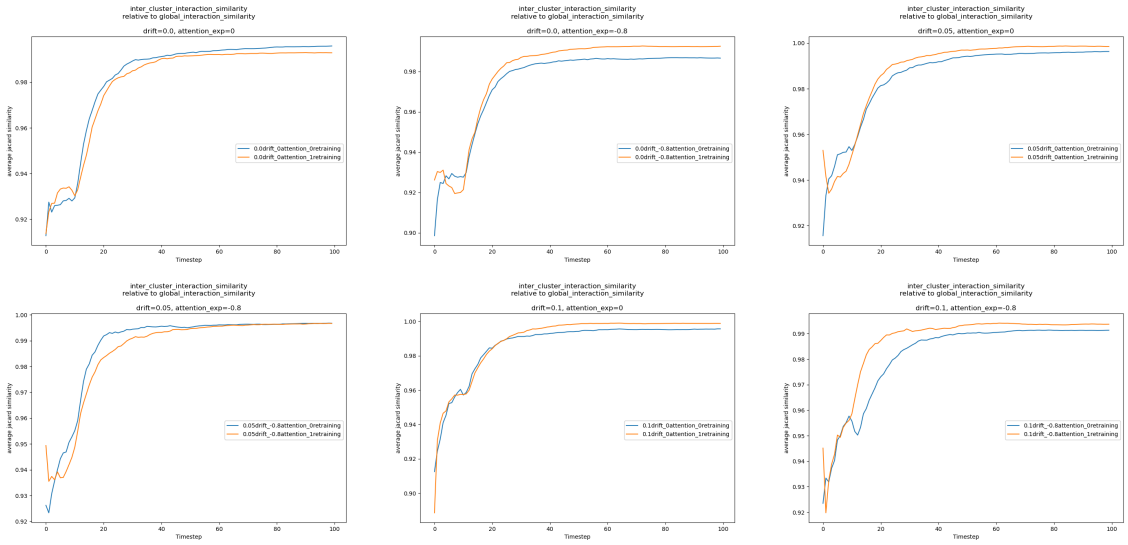


2.0.4 Graphing Intra -Cluster Interaction Similarity and Inter -Cluster Interaction Similarity relative to Global Interaction Simarilty

[ ]:
```
global_metric_key = "global_interaction_similarity"
```

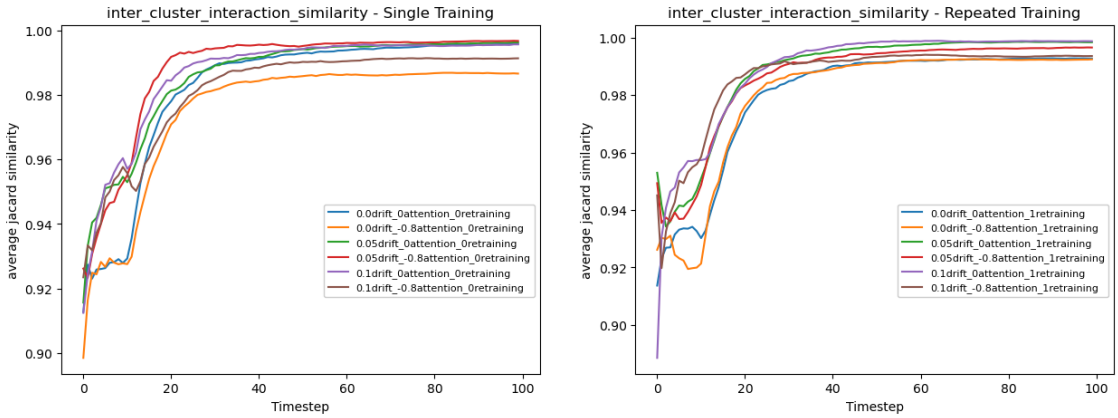
Graphing Inter -Cluster Interaction Similarity relative to Global Interaction Simarilty Single training vs. Retraining

[ ]:
```
metric_key = "inter_cluster_interaction_similarity"

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_relative_to_global_by_axis(curr_ax, results, global_metric_key,␣
 ↪metric_key, model_key_pairs[i], id_to_readable, mult_sd=0)
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"{metric_key}\nrelative to␣
 ↪{global_metric_key}\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```



[ ]:
```
metric_key = "inter_cluster_interaction_similarity"

single_training_keys = [key[0] for key in model_key_pairs]
repeated_training_keys = [key[1] for key in model_key_pairs]

fig, axs = plt.subplots(1, 2, figsize=(15, 5))

# graph_relative_to_global_by_axis(curr_ax, results, global_metric_key,␣
 ↪metric_key, model_key_pairs[i], id_to_readable, mult_sd=0)
graph_relative_to_global_by_axis(axs[0], results, global_metric_key,␣
 ↪metric_key, single_training_keys, id_to_readable, mult_sd=0)
axs[0].set_ylabel(y_labels[metric_key])
axs[0].set_xlabel("Timestep")
axs[0].set_title(f"{metric_key} - Single Training")
axs[0].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)

graph_relative_to_global_by_axis(axs[1], results, global_metric_key,␣
 ↪metric_key, repeated_training_keys, id_to_readable, mult_sd=0)
axs[1].set_ylabel(y_labels[metric_key])
axs[1].set_xlabel("Timestep")
axs[1].set_title(f"{metric_key} - Repeated Training")
axs[1].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)
```

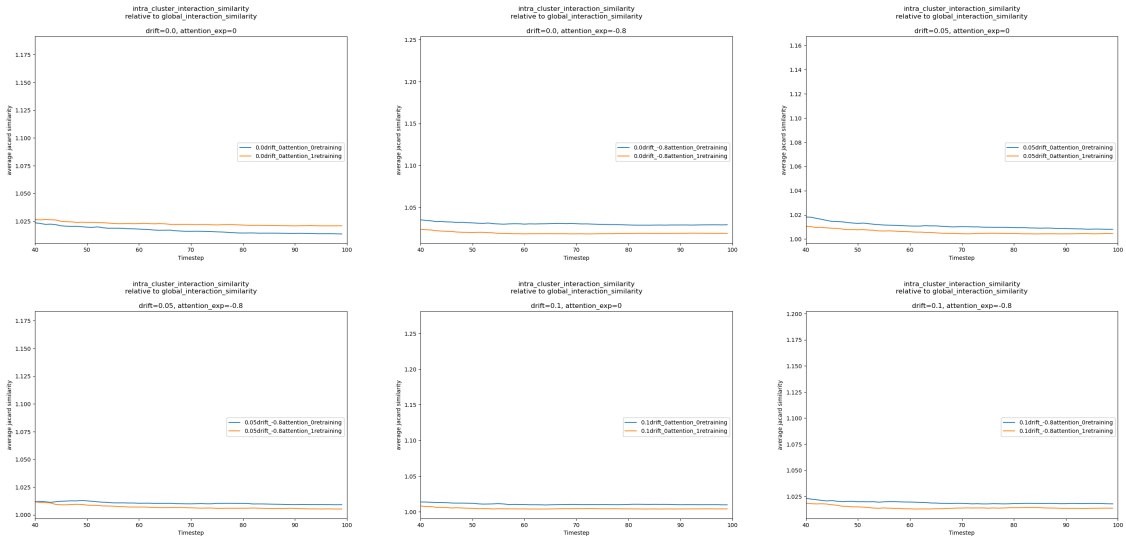
[ ]:
```
<matplotlib.legend.Legend at 0x16b153dc0>
```



[ ]:
```
# inter_cluster_interaction_similarity =␣
 ↪results['inter_cluster_interaction_similarity']['0.1drift_-0.
 ↪8attention_1retraining'].mean(axis=0)
# global_interaction_similarity = results['global_interaction_similarity']['0.
 ↪1drift_-0.8attention_1retraining'].mean(axis=0)

# diff = np.divide(inter_cluster_interaction_similarity,␣
 ↪global_interaction_similarity)

# plt.plot(diff)
```

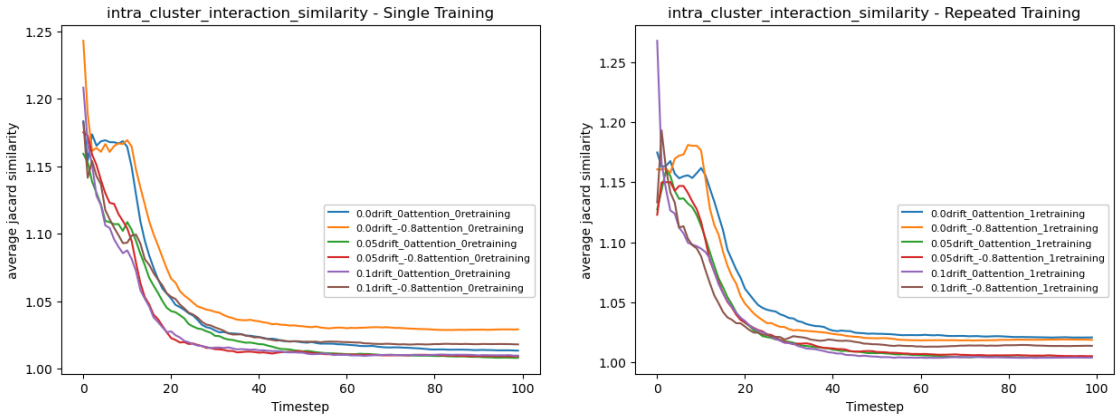
Graphing Intra -Cluster Interaction Similarity relative to Global Interaction Simarilty Single training vs. Retraining

[ ]:
```
metric_key = "intra_cluster_interaction_similarity"

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_relative_to_global_by_axis(curr_ax, results, global_metric_key,␣
 ↪metric_key, model_key_pairs[i], id_to_readable, mult_sd=0)
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"{metric_key}\nrelative to␣
 ↪{global_metric_key}\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
    # curr_ax.set_ylim(-0.1, 1.75)
    curr_ax.set_xlim(40, 100)
```



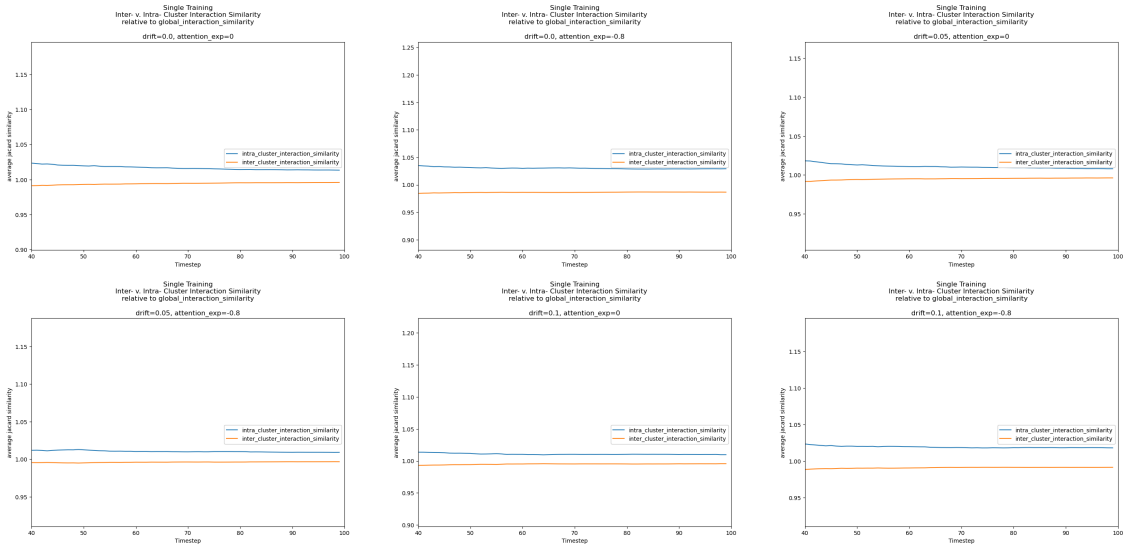
[ ]:
```
metric_key = "intra_cluster_interaction_similarity"

single_training_keys = [key[0] for key in model_key_pairs]
repeated_training_keys = [key[1] for key in model_key_pairs]

fig, axs = plt.subplots(1, 2, figsize=(15, 5))

# graph_relative_to_global_by_axis(curr_ax, results, global_metric_key,␣
 ↪metric_key, model_key_pairs[i], id_to_readable, mult_sd=0)
graph_relative_to_global_by_axis(axs[0], results, global_metric_key,␣
 ↪metric_key, single_training_keys, id_to_readable, mult_sd=0)
axs[0].set_ylabel(y_labels[metric_key])
axs[0].set_xlabel("Timestep")
axs[0].set_title(f"{metric_key} - Single Training")
axs[0].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)

graph_relative_to_global_by_axis(axs[1], results, global_metric_key,␣
 ↪metric_key, repeated_training_keys, id_to_readable, mult_sd=0)
axs[1].set_ylabel(y_labels[metric_key])
axs[1].set_xlabel("Timestep")
axs[1].set_title(f"{metric_key} - Repeated Training")
axs[1].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)
```

[ ]:
```
<matplotlib.legend.Legend at 0x16b37eee0>
```

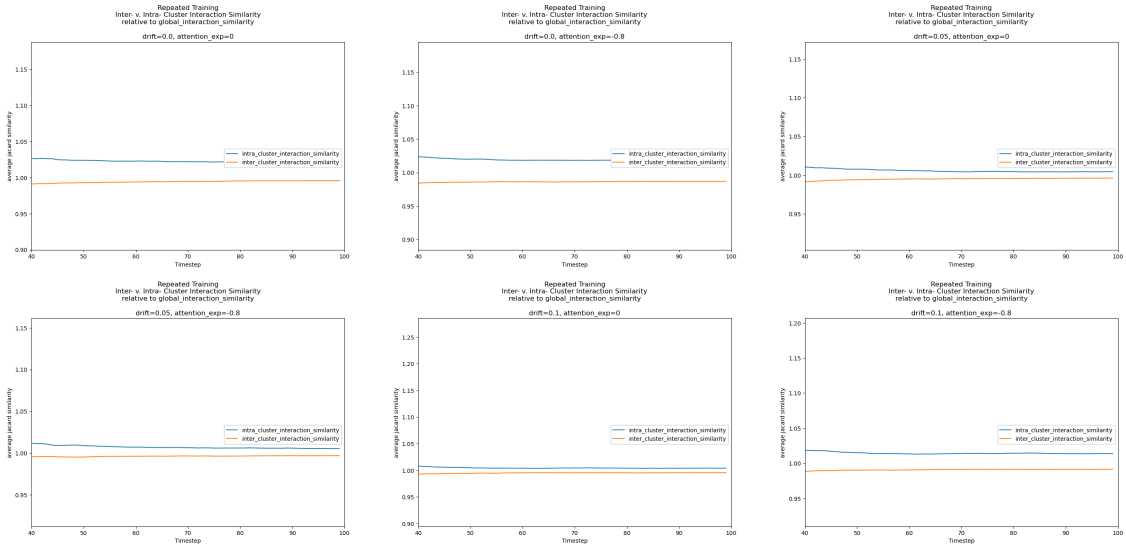


Graphing Cluster Interaction Similarity relative to Global Interaction Simarilty Inter- vs. Intra-

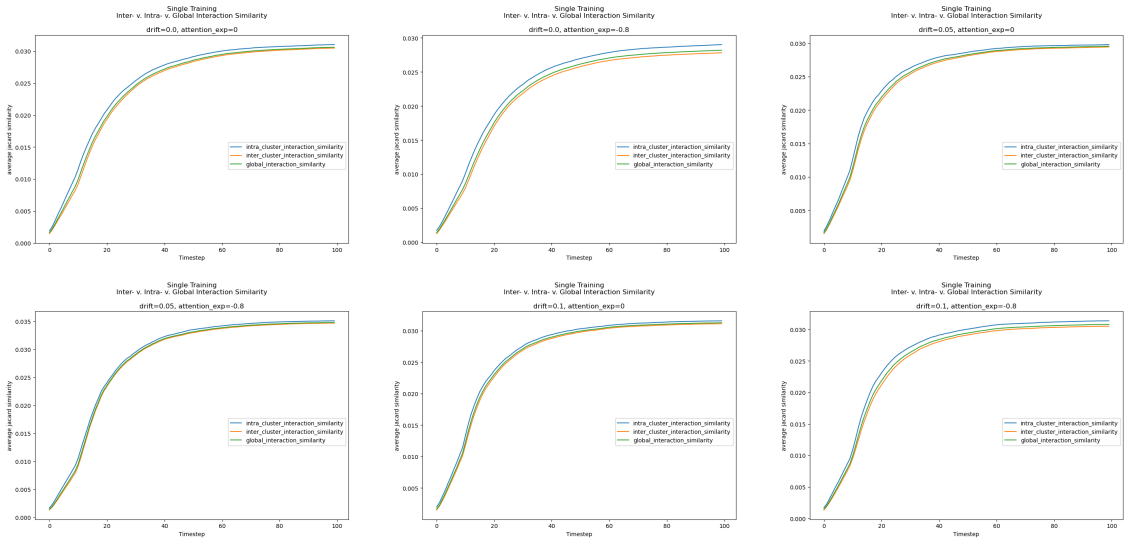
[ ]:
```
"""
Single training
"""
metric_keys = ["intra_cluster_interaction_similarity",␣
 ↪"inter_cluster_interaction_similarity"]

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_relative_to_global_by_axis(curr_ax, results, global_metric_key,␣
 ↪metric_keys[0], (model_key_pairs[i][0],), id_to_readable, mult_sd=0,␣
 ↪graph_by="metric")
    graph_relative_to_global_by_axis(curr_ax, results, global_metric_key,␣
 ↪metric_keys[1], (model_key_pairs[i][0],), id_to_readable, mult_sd=0,␣
 ↪graph_by="metric")
    curr_ax.set_ylabel(y_labels[metric_keys[0]])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Single Training\nInter- v. Intra- Cluster Interaction␣
 ↪Similarity\nrelative to␣
 ↪{global_metric_key}\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
    # curr_ax.set_ylim(-0.1, 1.75)
    curr_ax.set_xlim(40, 100)
```



[ ]:
```
"""
Repeated training
"""
metric_keys = ["intra_cluster_interaction_similarity",␣
 ↪"inter_cluster_interaction_similarity"]

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_relative_to_global_by_axis(curr_ax, results, global_metric_key,␣
 ↪metric_keys[0], (model_key_pairs[i][1],), id_to_readable, mult_sd=0,␣
 ↪graph_by="metric")
    graph_relative_to_global_by_axis(curr_ax, results, global_metric_key,␣
 ↪metric_keys[1], (model_key_pairs[i][0],), id_to_readable, mult_sd=0,␣
 ↪graph_by="metric")
    curr_ax.set_ylabel(y_labels[metric_keys[0]])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Repeated Training\nInter- v. Intra- Cluster Interaction␣
 ↪Similarity\nrelative to␣
 ↪{global_metric_key}\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
    # curr_ax.set_ylim(-0.1, 1.75)
    curr_ax.set_xlim(40, 100)
```



[ ]:
```
"""
Single training
"""
metric_keys = ["intra_cluster_interaction_similarity",␣
 ↪"inter_cluster_interaction_similarity", "global_interaction_similarity"]

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_keys[0],␣
 ↪(model_key_pairs[i][0],), id_to_readable, mult_sd=0, graph_by="metric")
    graph_metrics_by_axis(curr_ax, results, metric_keys[1],␣
 ↪(model_key_pairs[i][0],), id_to_readable, mult_sd=0, graph_by="metric")
    graph_metrics_by_axis(curr_ax, results, metric_keys[2],␣
 ↪(model_key_pairs[i][0],), id_to_readable, mult_sd=0, graph_by="metric")
    curr_ax.set_ylabel(y_labels[metric_keys[0]])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Single Training\nInter- v. Intra- v. Global Interaction␣
 ↪Similarity\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

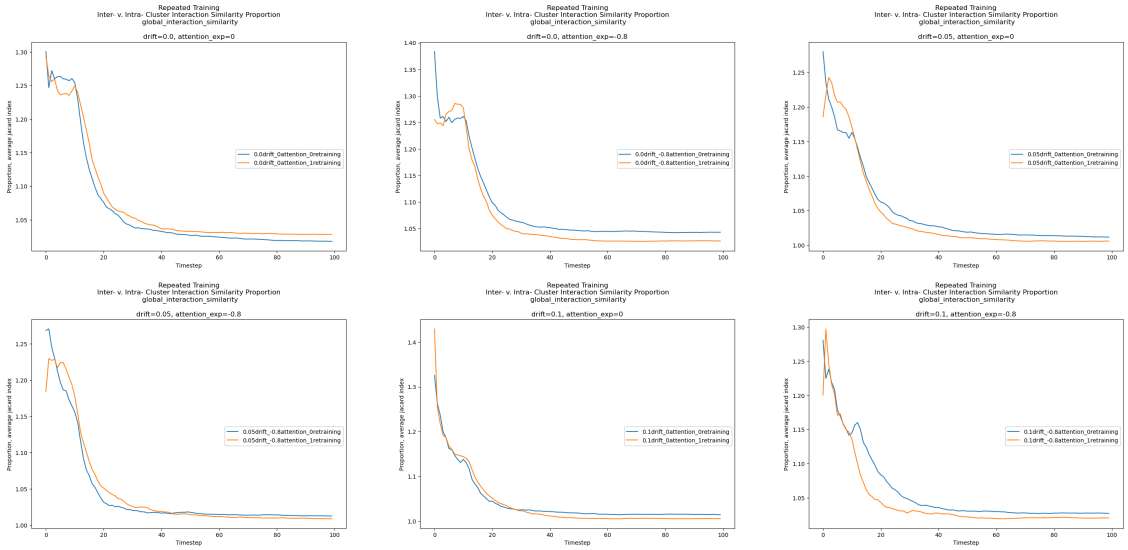


[ ]:
```
"""
Repeated training
"""
metric_keys = ["intra_cluster_interaction_similarity",␣
 ↪"inter_cluster_interaction_similarity", "global_interaction_similarity"]

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    print(model_key_pairs[i][1])
    graph_metrics_by_axis(curr_ax, results, metric_keys[0],␣
 ↪(model_key_pairs[i][1],), id_to_readable, mult_sd=0, graph_by="metric")
    graph_metrics_by_axis(curr_ax, results, metric_keys[1],␣
 ↪(model_key_pairs[i][1],), id_to_readable, mult_sd=0, graph_by="metric")
    graph_metrics_by_axis(curr_ax, results, metric_keys[2],␣
 ↪(model_key_pairs[i][1],), id_to_readable, mult_sd=0, graph_by="metric")
    curr_ax.set_ylabel(y_labels[metric_keys[0]])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Repeated Training\nInter- v. Intra- v. Global␣
 ↪Interaction Similarity\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

Intra-Cluster:Inter-Cluster Interaction Similarity, Single training v. Repeated training

[ ]:
```
numerator = "intra_cluster_interaction_similarity"
denominator = "inter_cluster_interaction_similarity"
```

```
fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_relative_to_global_by_axis(curr_ax, results, denominator, numerator,␣
 ↪model_key_pairs[i], id_to_readable, mult_sd=0)
    curr_ax.set_ylabel("Proportion, average jacard index")
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Repeated Training\nInter- v. Intra- Cluster Interaction␣
 ↪Similarity Proportion\n␣
 ↪{global_metric_key}\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
    # curr_ax.set_ylim(-0.1, 1.75)
    # curr_ax.set_xlim(40, 100)
```



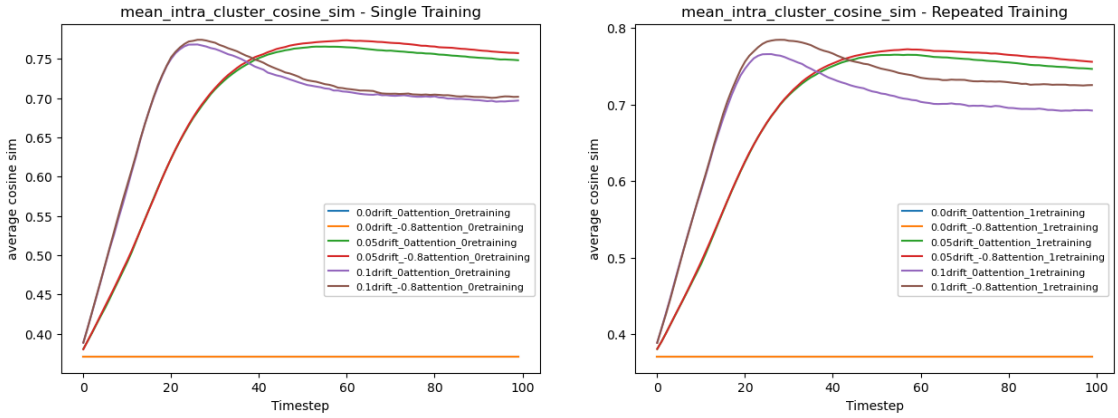
3 Graphing Cosine Similarity

3.0.1 Graphing Intra -Cluster Cosine Similarity

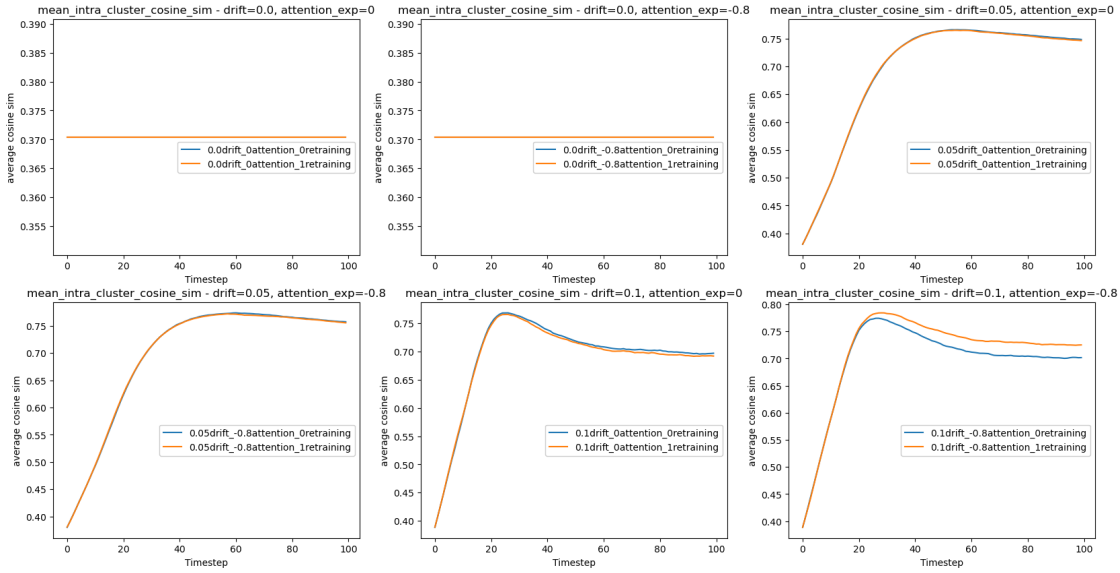
[ ]:
```
metric_key = "mean_intra_cluster_cosine_sim"

single_training_keys = [key[0] for key in model_key_pairs]
repeated_training_keys = [key[1] for key in model_key_pairs]

fig, axs = plt.subplots(1, 2, figsize=(15, 5))
```

```
graph_metrics_by_axis(axs[0], results, metric_key, single_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[0].set_ylabel(y_labels[metric_key])
axs[0].set_xlabel("Timestep")
axs[0].set_title(f"{metric_key} - Single Training")
axs[0].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)

graph_metrics_by_axis(axs[1], results, metric_key, repeated_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[1].set_ylabel(y_labels[metric_key])
axs[1].set_xlabel("Timestep")
axs[1].set_title(f"{metric_key} - Repeated Training")
axs[1].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)
```

[ ]: <matplotlib.legend.Legend at 0x16b55bb80>



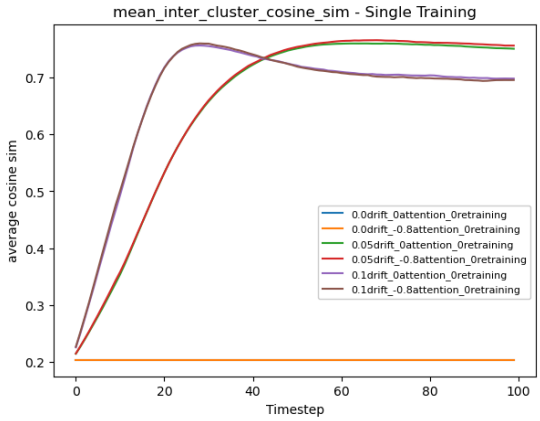
[ ]:
```
metric_key = "mean_intra_cluster_cosine_sim"

fig, axs = plt.subplots(2, 3, figsize=(20, 10))

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_key, model_key_pairs[i],␣
 ↪id_to_readable, mult_sd=0)
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"{metric_key} -␣
 ↪drift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

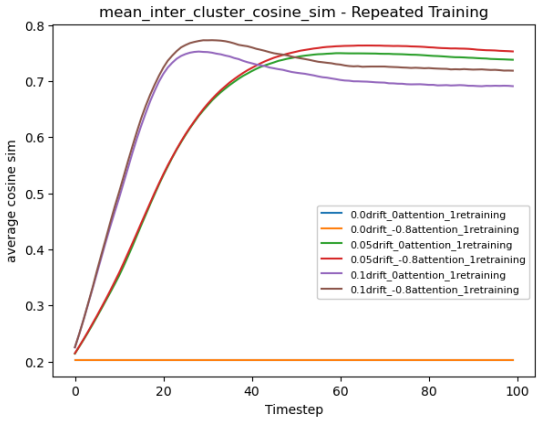


3.0.2 Graphing Inter -Cluster Cosine Similarity

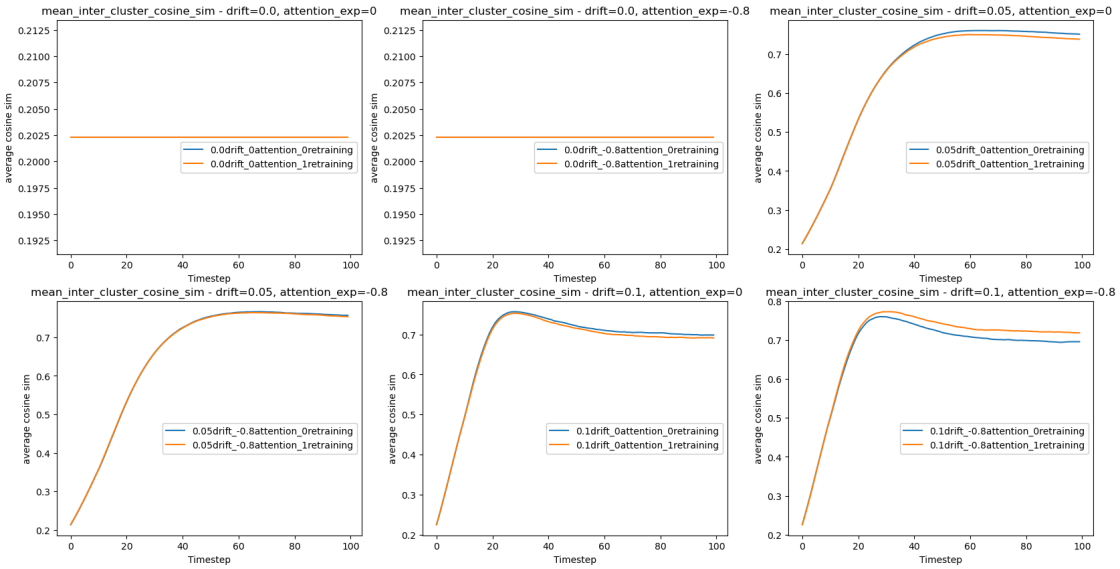
[ ]:
```
metric_key = "mean_inter_cluster_cosine_sim"

fig, axs = plt.subplots(1, 2, figsize=(15, 5))

graph_metrics_by_axis(axs[0], results, metric_key, single_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[0].set_ylabel(y_labels[metric_key])
axs[0].set_xlabel("Timestep")
axs[0].set_title(f"{metric_key} - Single Training")
axs[0].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)

graph_metrics_by_axis(axs[1], results, metric_key, repeated_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[1].set_ylabel(y_labels[metric_key])
axs[1].set_xlabel("Timestep")
axs[1].set_title(f"{metric_key} - Repeated Training")
axs[1].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)
```

[ ]:
```
<matplotlib.legend.Legend at 0x16c36d460>
```





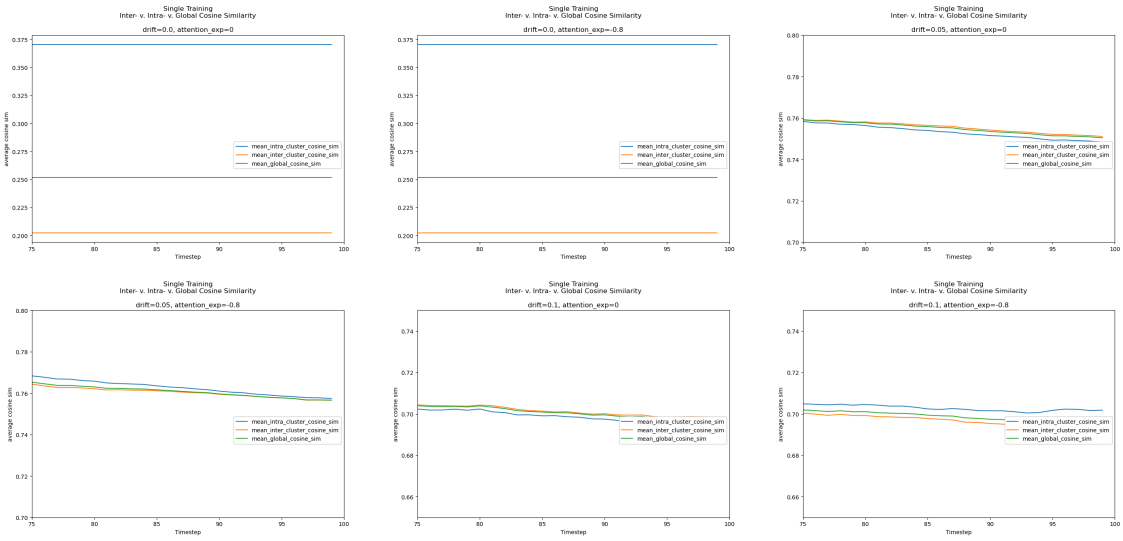
[ ]:
```
metric_key = "mean_inter_cluster_cosine_sim"

fig, axs = plt.subplots(2, 3, figsize=(20, 10))

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_key, model_key_pairs[i],␣
↪id_to_readable, mult_sd=0)
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"{metric_key} -␣
↪drift={models[model_key_pairs[i][0]][0]},␣
↪attention_exp={models[model_key_pairs[i][0]][1]}")
```



Graphing Cluster Cosine Similarity relative to Global Cosine Simarilty

[ ]:
```
"""
Single training
"""
metric_keys = ["mean_intra_cluster_cosine_sim",␣
 ↪"mean_inter_cluster_cosine_sim", "mean_global_cosine_sim"]

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    print((model_key_pairs[i][0],))
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_keys[0],␣
 ↪(model_key_pairs[i][0],), id_to_readable, mult_sd=0, graph_by="metric")
    graph_metrics_by_axis(curr_ax, results, metric_keys[1],␣
 ↪(model_key_pairs[i][0],), id_to_readable, mult_sd=0, graph_by="metric")
    graph_metrics_by_axis(curr_ax, results, metric_keys[2],␣
 ↪(model_key_pairs[i][0],), id_to_readable, mult_sd=0, graph_by="metric")

    curr_ax.set_xlim(75, 100)
    if i > 3:
        curr_ax.set_ylim(0.65, 0.75)
    elif i > 1:
        curr_ax.set_ylim(0.7, 0.8)
    # if i > 1:
    #     curr_ax.set_ylim(0.6, 0.8)
    curr_ax.set_ylabel(y_labels[metric_keys[0]])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Single Training\nInter- v. Intra- v. Global Cosine␣
 ↪Similarity\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")

# fig.tight_layout()
```

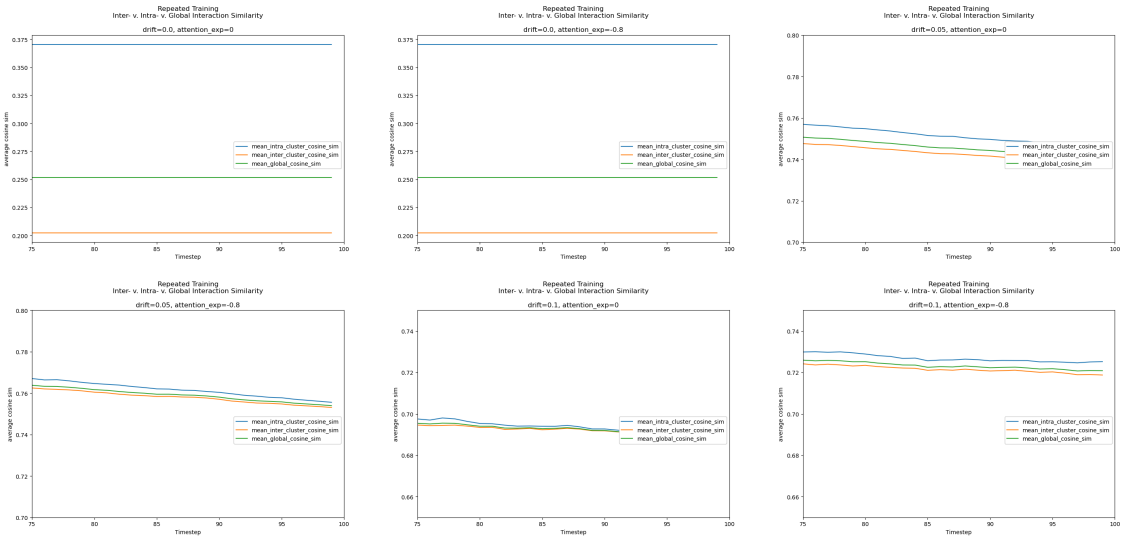
```
('0.0drift_0attention_0retraining',)
('0.0drift_-0.8attention_0retraining',)
('0.05drift_0attention_0retraining',)
('0.05drift_-0.8attention_0retraining',)
('0.1drift_0attention_0retraining',)
('0.1drift_-0.8attention_0retraining',)
```



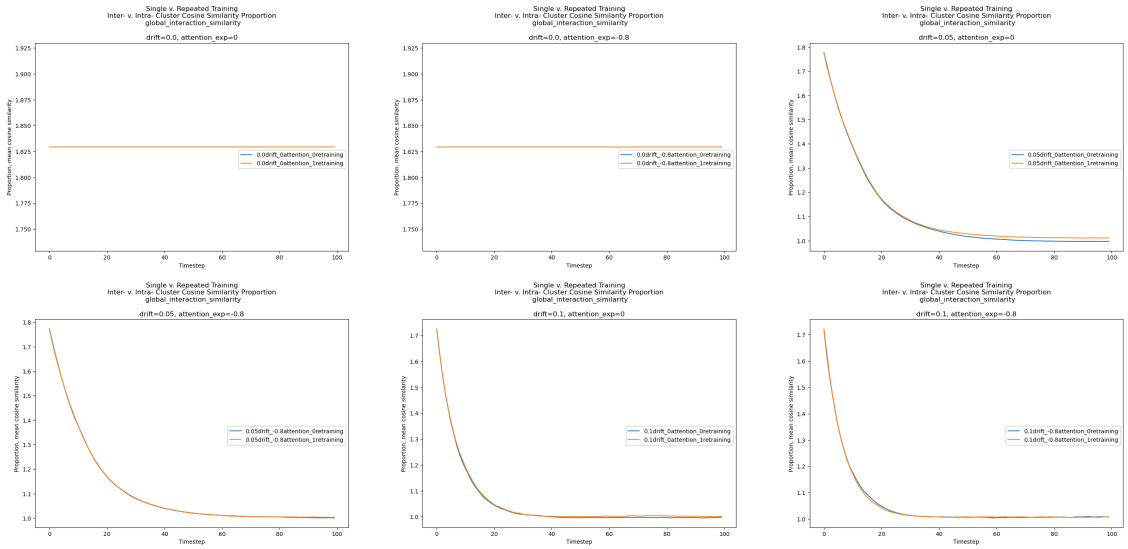
[ ]:
```
"""
Repeated training
"""
metric_keys = ["mean_intra_cluster_cosine_sim",␣
 ↪"mean_inter_cluster_cosine_sim", "mean_global_cosine_sim"]

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    print(model_key_pairs[i][1])
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_keys[0],␣
 ↪(model_key_pairs[i][1],), id_to_readable, mult_sd=0, graph_by="metric")
    graph_metrics_by_axis(curr_ax, results, metric_keys[1],␣
 ↪(model_key_pairs[i][1],), id_to_readable, mult_sd=0, graph_by="metric")
    graph_metrics_by_axis(curr_ax, results, metric_keys[2],␣
 ↪(model_key_pairs[i][1],), id_to_readable, mult_sd=0, graph_by="metric")

    curr_ax.set_xlim(75, 100)
    if i > 3:
        curr_ax.set_ylim(0.65, 0.75)
    elif i > 1:
        curr_ax.set_ylim(0.7, 0.8)
    # if i > 1:
    #     curr_ax.set_ylim(0.6, 0.8)
    curr_ax.set_ylabel(y_labels[metric_keys[0]])
    curr_ax.set_xlabel("Timestep")
```

```
    curr_ax.set_title(f"Repeated Training\nInter- v. Intra- v. Global␣
 ↪Interaction Similarity\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

```
0.0drift_0attention_1retraining
0.0drift_-0.8attention_1retraining
0.05drift_0attention_1retraining
0.05drift_-0.8attention_1retraining
0.1drift_0attention_1retraining
0.1drift_-0.8attention_1retraining
```



[ ]:
```
numerator = "mean_intra_cluster_cosine_sim"
denominator = "mean_inter_cluster_cosine_sim"

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_relative_to_global_by_axis(curr_ax, results, denominator, numerator,␣
 ↪model_key_pairs[i], id_to_readable, mult_sd=0)
    curr_ax.set_ylabel("Proportion, mean cosine similarity")
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Single v. Repeated Training\nInter- v. Intra- Cluster␣
 ↪Cosine Similarity Proportion\n␣
 ↪{global_metric_key}\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
    # curr_ax.set_ylim(-0.1, 1.75)
    # curr_ax.set_xlim(40, 100)
```

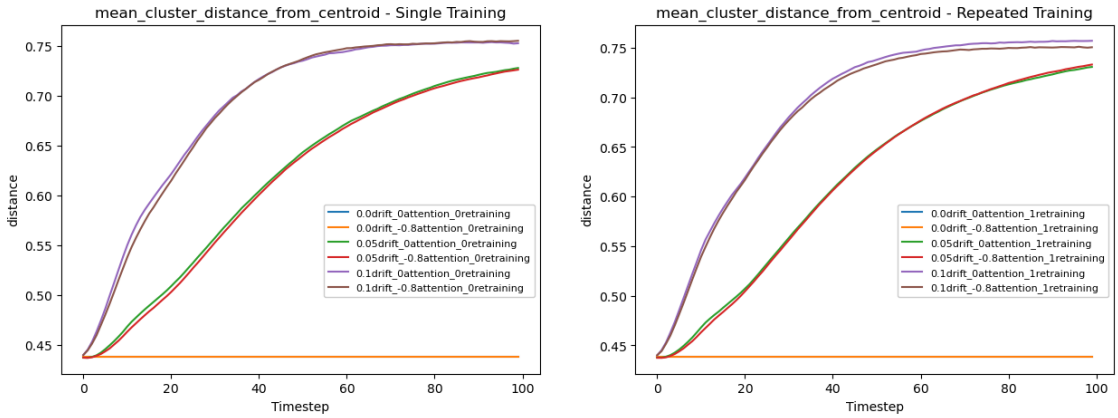


4 Graphing Distance from centroid

4.0.1 Graphing mean_cluster_distance_from_centroid

[ ]:
```
metric_key = "mean_cluster_distance_from_centroid"

single_training_keys = [key[0] for key in model_key_pairs]
repeated_training_keys = [key[1] for key in model_key_pairs]

fig, axs = plt.subplots(1, 2, figsize=(15, 5))

graph_metrics_by_axis(axs[0], results, metric_key, single_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[0].set_ylabel(y_labels[metric_key])
axs[0].set_xlabel("Timestep")
axs[0].set_title(f"{metric_key} - Single Training")
axs[0].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)

graph_metrics_by_axis(axs[1], results, metric_key, repeated_training_keys,␣
 ↪id_to_readable, mult_sd=0)
axs[1].set_ylabel(y_labels[metric_key])
axs[1].set_xlabel("Timestep")
axs[1].set_title(f"{metric_key} - Repeated Training")
axs[1].legend(facecolor='white', framealpha=1, loc='upper right',␣
 ↪bbox_to_anchor=(1, 0.5), fontsize="8",)
```

[ ]:
```
<matplotlib.legend.Legend at 0x177f22e50>
```

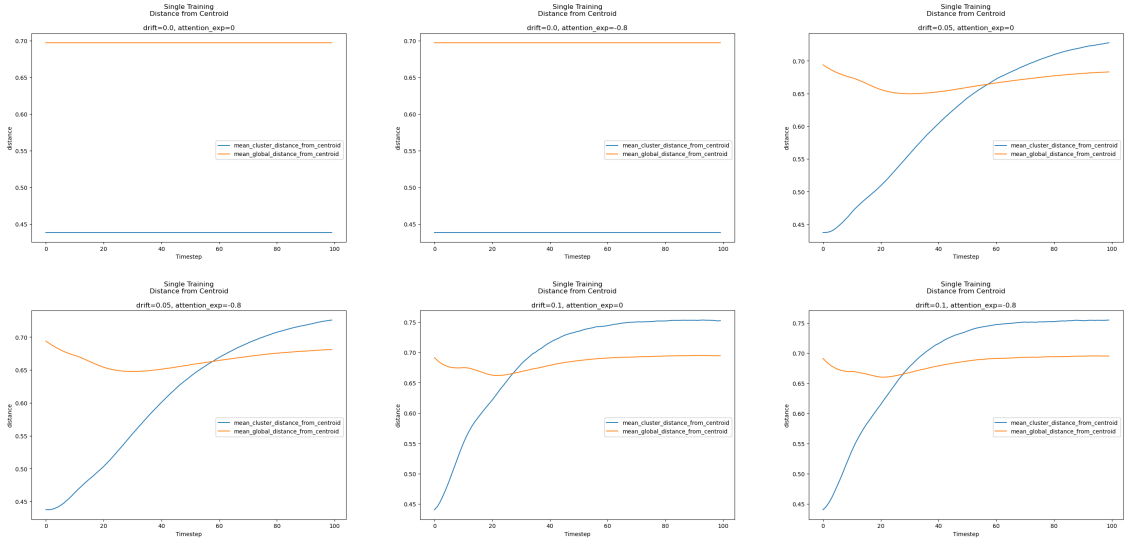


Graphing Intra-Cluster Distance relative to Global Distance

[ ]:
```
"""
Single training
"""
metric_keys = ['mean_cluster_distance_from_centroid',␣
 ↪'mean_global_distance_from_centroid']

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    print(model_key_pairs[i][0])
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_keys[0],␣
 ↪(model_key_pairs[i][0],), id_to_readable, mult_sd=0, graph_by="metric")
    graph_metrics_by_axis(curr_ax, results, metric_keys[1],␣
 ↪(model_key_pairs[i][0],), id_to_readable, mult_sd=0, graph_by="metric")

    # curr_ax.set_xlim(75, 100)
    # if i > 3:
    #     curr_ax.set_ylim(0.3, 0.6)
    # elif i > 1:
    #     curr_ax.set_ylim(0.4, 0.8)
    # if i > 1:
    #     curr_ax.set_ylim(0.3, 0.8)
    curr_ax.set_ylabel(y_labels[metric_keys[0]])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Single Training\nDistance from␣
 ↪Centroid\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

```
0.0drift_0attention_0retraining
```

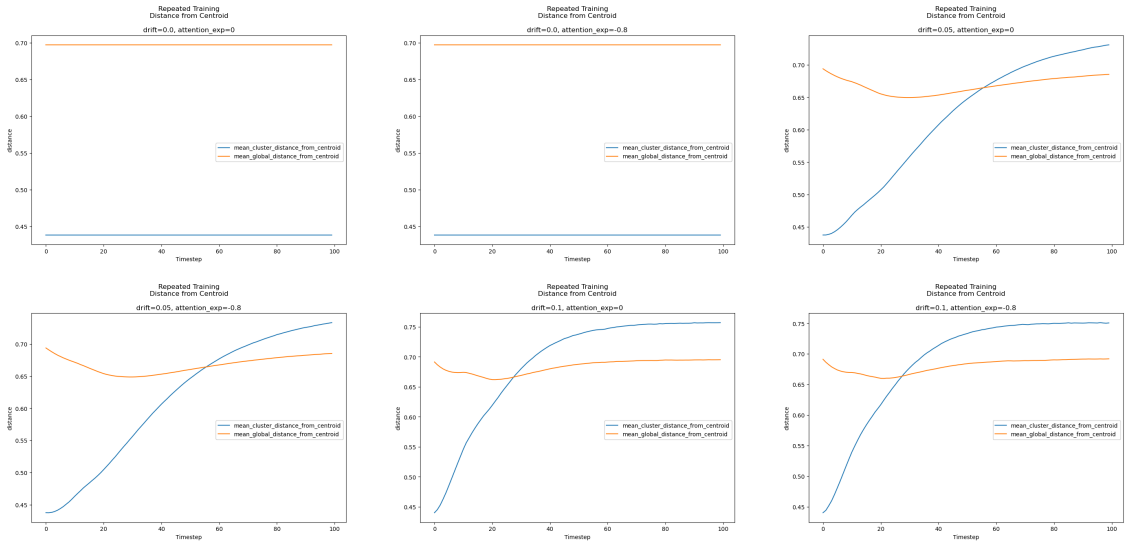
```
0.0drift_-0.8attention_0retraining
0.05drift_0attention_0retraining
0.05drift_-0.8attention_0retraining
0.1drift_0attention_0retraining
0.1drift_-0.8attention_0retraining
```



[ ]:
```
"""
Repeated training
"""
metric_keys = ['mean_cluster_distance_from_centroid',␣
 ↪'mean_global_distance_from_centroid']

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    print(model_key_pairs[i][1])
    curr_ax = axs[int(i >= 3), i%3]
    graph_metrics_by_axis(curr_ax, results, metric_keys[0],␣
 ↪(model_key_pairs[i][1],), id_to_readable, mult_sd=0, graph_by="metric")
    graph_metrics_by_axis(curr_ax, results, metric_keys[1],␣
 ↪(model_key_pairs[i][1],), id_to_readable, mult_sd=0, graph_by="metric")

    # curr_ax.set_xlim(75, 100)
    # if i > 3:
    #     curr_ax.set_ylim(0.3, 0.6)
    # elif i > 1:
    #     curr_ax.set_ylim(0.4, 0.8)
    # if i > 1:
```

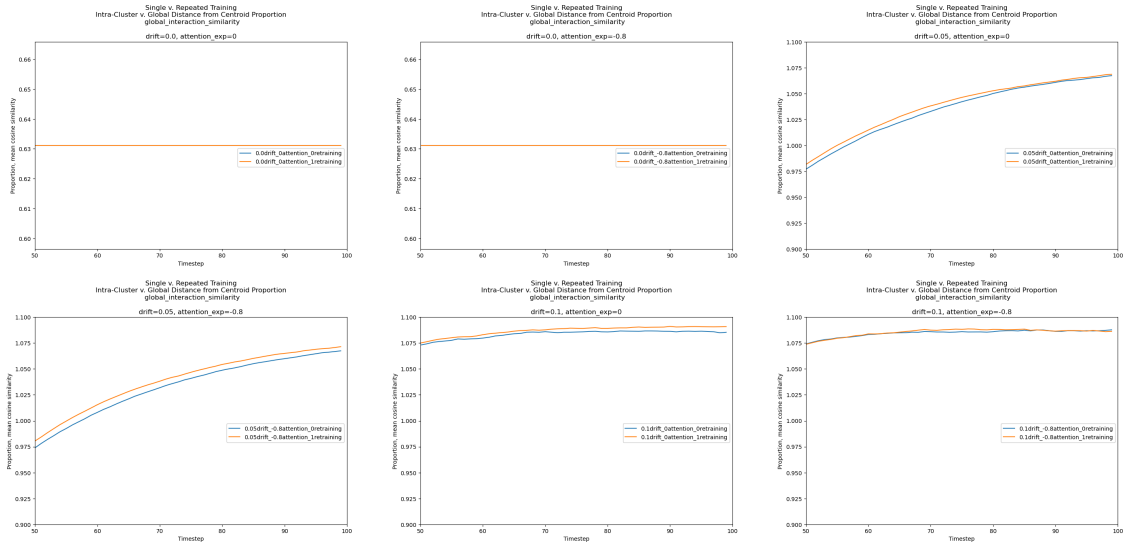
```
    #     curr_ax.set_ylim(0.3, 0.8)
    curr_ax.set_ylabel(y_labels[metric_keys[0]])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Repeated Training\nDistance from␣
 ↪Centroid\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

```
0.0drift_0attention_1retraining
0.0drift_-0.8attention_1retraining
0.05drift_0attention_1retraining
0.05drift_-0.8attention_1retraining
0.1drift_0attention_1retraining
0.1drift_-0.8attention_1retraining
```



[ ]:
```
numerator = "mean_cluster_distance_from_centroid"
denominator = "mean_global_distance_from_centroid"

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    curr_ax = axs[int(i >= 3), i%3]
    graph_relative_to_global_by_axis(curr_ax, results, denominator, numerator,␣
 ↪model_key_pairs[i], id_to_readable, mult_sd=0)
    curr_ax.set_xlim(50, 100)
    if i > 1:
        curr_ax.set_ylim(0.9, 1.1)
    curr_ax.set_ylabel("Proportion, mean cosine similarity")
    curr_ax.set_xlabel("Timestep")
```

```
    curr_ax.set_title(f"Single v. Repeated Training\nIntra-Cluster v. Global␣
↪Distance from Centroid Proportion\n␣
↪{global_metric_key}\n\ndrift={models[model_key_pairs[i][0]][0]},␣
↪attention_exp={models[model_key_pairs[i][0]][1]}")
    # curr_ax.set_ylim(-0.1, 1.75)
    # curr_ax.set_xlim(40, 100)
```



[ ]:

5 Analyzing histogram data

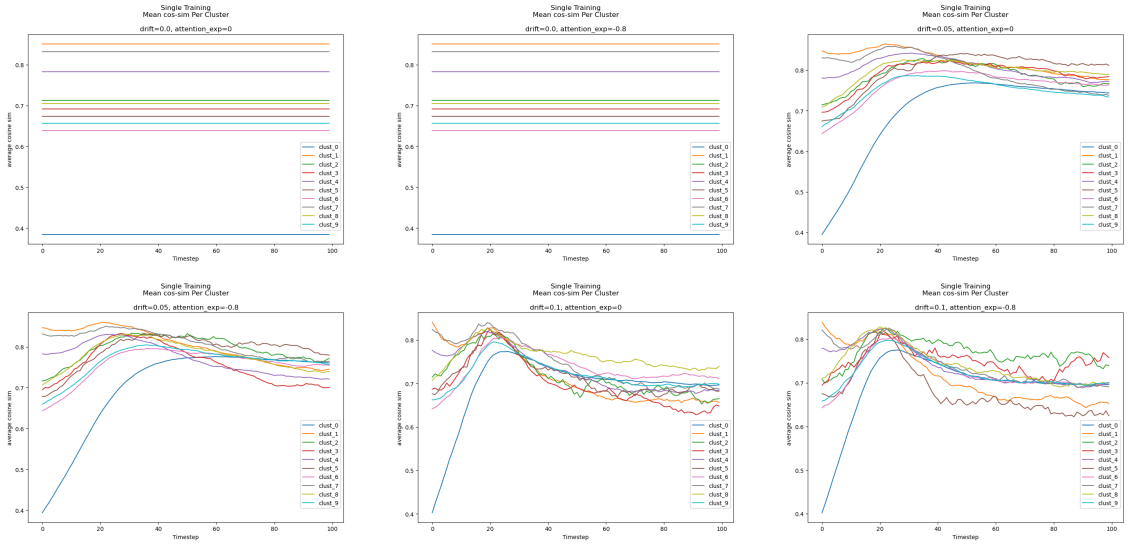
Graphing mean_cosine_sim_per_cluster

[ ]:
```
n_clusters = 10
label_map = dict()
for i in range(n_clusters):
    label_map[i] = f"clust_{i}"
```

[ ]:
```
"""
Single training
"""
metric_key = 'mean_cosine_sim_per_cluster'

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    print(model_key_pairs[i][0])
    curr_ax = axs[int(i >= 3), i%3]
```

```
    graph_histogram_metric_by_axis(curr_ax, results, metric_key,␣
 ↪(model_key_pairs[i][0],), label_map, mean_sigma=0, mult_sd=0, conf_sigma=0)
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Single Training\nMean cos-sim Per␣
 ↪Cluster\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

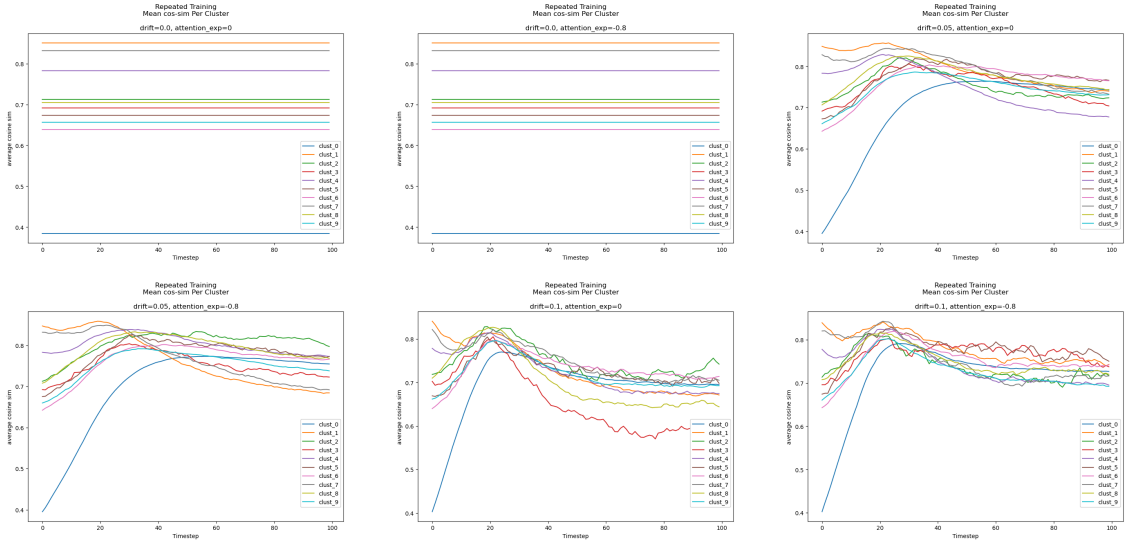
```
0.0drift_0attention_0retraining
0.0drift_-0.8attention_0retraining
0.05drift_0attention_0retraining
0.05drift_-0.8attention_0retraining
0.1drift_0attention_0retraining
0.1drift_-0.8attention_0retraining
```



[ ]:
```
"""
Repeated training
"""
metric_key = 'mean_cosine_sim_per_cluster'

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    print(model_key_pairs[i][1])
    curr_ax = axs[int(i >= 3), i%3]
    graph_histogram_metric_by_axis(curr_ax, results, metric_key,␣
 ↪(model_key_pairs[i][1],), label_map, mean_sigma=0, mult_sd=0, conf_sigma=0)
    curr_ax.set_ylabel(y_labels[metric_key])
```

```
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Repeated Training\nMean cos-sim Per␣
 ↪Cluster\n\ndrift={models[model_key_pairs[i][0]][0]},␣
 ↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

```
0.0drift_0attention_1retraining
0.0drift_-0.8attention_1retraining
0.05drift_0attention_1retraining
0.05drift_-0.8attention_1retraining
0.1drift_0attention_1retraining
0.1drift_-0.8attention_1retraining
```

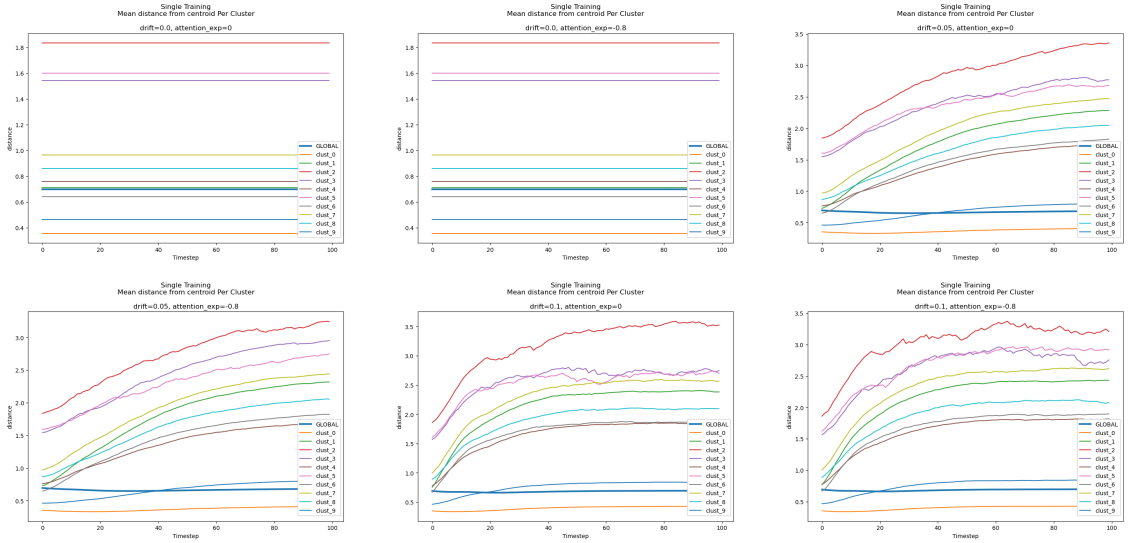


Graphing `mean_distance_from_centroid_per_cluster`

[ ]:
```
"""
Single training
"""
metric_key = 'mean_distance_from_centroid_per_cluster'

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
    #␣
 ↪print(results['mean_global_distance_from_centroid'][model_key_pairs[i][0]].
 ↪mean(axis=0).shape)
    # break
    print(model_key_pairs[i][0])
    curr_ax = axs[int(i >= 3), i%3]
```

```
    curr_ax.
↪plot(results['mean_global_distance_from_centroid'][model_key_pairs[i][0]].
↪mean(axis=0), label="GLOBAL", linewidth=3)
    graph_histogram_metric_by_axis(curr_ax, results, metric_key,␣
↪(model_key_pairs[i][0],), label_map, mean_sigma=0, mult_sd=0, conf_sigma=0)
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Single Training\nMean distance from centroid Per␣
↪Cluster\n\ndrift={models[model_key_pairs[i][0]][0]},␣
↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

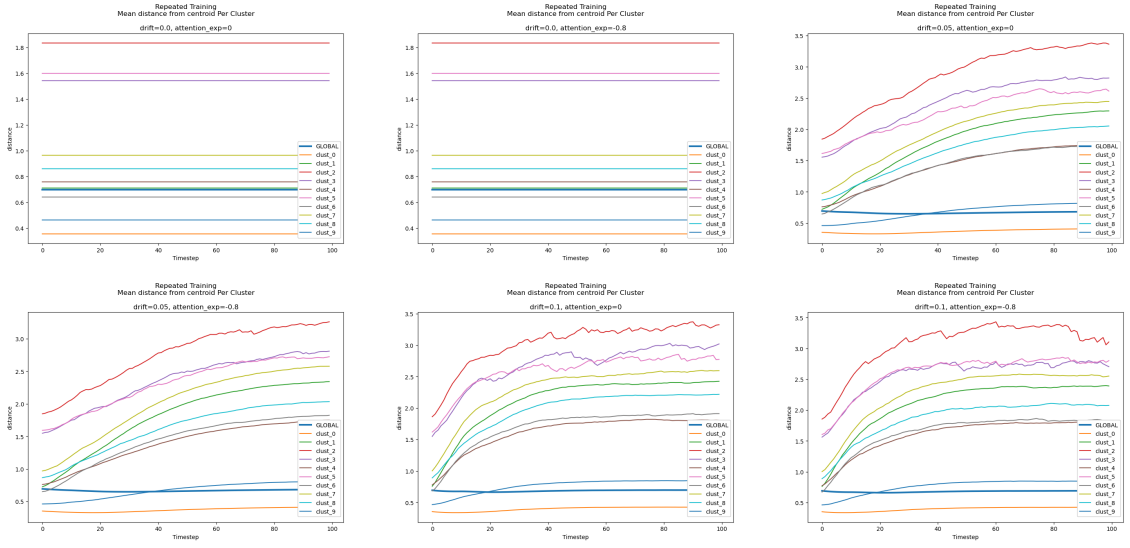
```
0.0drift_0attention_0retraining
0.0drift_-0.8attention_0retraining
0.05drift_0attention_0retraining
0.05drift_-0.8attention_0retraining
0.1drift_0attention_0retraining
0.1drift_-0.8attention_0retraining
```



[ ]:
```
"""
Repeated training
"""
metric_key = 'mean_distance_from_centroid_per_cluster'

fig, axs = plt.subplots(2, 3, figsize=(30, 15))
fig.tight_layout(pad=10.0)

for i in range(len(model_key_pairs)):
```

```
    #␣
↪print(results['mean_global_distance_from_centroid'][model_key_pairs[i][0]].
↪mean(axis=0).shape)
    # break
    print(model_key_pairs[i][1])
    curr_ax = axs[int(i >= 3), i%3]
    curr_ax.
↪plot(results['mean_global_distance_from_centroid'][model_key_pairs[i][1]].
↪mean(axis=0), label="GLOBAL", linewidth=3)
    graph_histogram_metric_by_axis(curr_ax, results, metric_key,␣
↪(model_key_pairs[i][1],), label_map, mean_sigma=0, mult_sd=0, conf_sigma=0)
    curr_ax.set_ylabel(y_labels[metric_key])
    curr_ax.set_xlabel("Timestep")
    curr_ax.set_title(f"Repeated Training\nMean distance from centroid Per␣
↪Cluster\n\ndrift={models[model_key_pairs[i][0]][0]},␣
↪attention_exp={models[model_key_pairs[i][0]][1]}")
```

```
0.0drift_0attention_1retraining
0.0drift_-0.8attention_1retraining
0.05drift_0attention_1retraining
0.05drift_-0.8attention_1retraining
0.1drift_0attention_1retraining
0.1drift_-0.8attention_1retraining
```



[ ]:
```
results.keys()
```

[ ]:
```
dict_keys(['mse', 'interaction_spread', 'global_interaction_similarity',
'inter_cluster_interaction_similarity', 'intra_cluster_interaction_similarity',
'mean_global_cosine_sim', 'mean_intra_cluster_cosine_sim',
```

```
'mean_inter_cluster_cosine_sim', 'mean_cosine_sim_per_cluster',
'mean_cluster_distance_from_centroid', 'mean_global_distance_from_centroid',
'mean_distance_from_centroid_per_cluster'])
```

[ ]: