UNIVERSITY of WASHINGTON

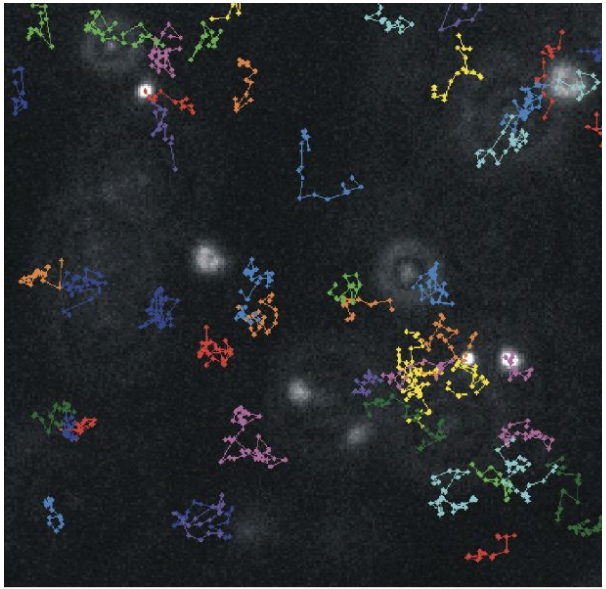
TrackONauts:
Multiple Particle Tracking Data Quality Toolkit

Meng Qin, Aileen Sun, Garrett McPheron, Khanh Ha

Introduction

Motivation: The Nance lab needs to train machine learning models using MPT data outputs to predict biological parameters.

Aim: Use computational methods to assess the quality of MPT data used in machine learning models



Data Separation

1. data_separation.read_feature(feature_data_path, feature_files)
2. data_separation.filter_feature(feature_list, feature_data_path, feature_files)
3. data_separation.remove_nans_feature(feature_list, feature_data_path, feature_files)
4. data_separation.read_json(json_data, feature_list, feature_data_path, feature_files)
5. data_separation.put_together(json_data, feature_list, feature_data_path, feature_files)

{'features_P70_40nm_s3_v3':

	alpha	D_fit	kurtosis	asymmetry1	asymmetry2	asymmetry3	\
0	3.000219e-01	1.466714	1.827982	0.831748	0.214532	0.234618	
1	3.201792e-01	34.234700	1.986628	0.871043	0.185748	0.268897	
2	1.724717e+00	0.283085	3.071816	0.997034	0.027251	0.594843	
3	4.628234e+00	0.000021	1.718922	0.786380	0.244962	0.203225	
4	1.447265e+00	0.054312	2.213652	0.999969	0.002783	0.682137	
...	...	...	...	...	...	...	
2193	1.897588e+00	0.103256	2.241314	0.654671	0.324825	0.139105	
2194	3.233752e-01	12.654308	1.724502	0.926729	0.137917	0.338241	
2195	3.860281e+00	0.000017	10.665880	0.998613	0.018629	0.623794	
2196	4.896067e-01	2.383485	3.919235	0.802734	0.234260	0.213751	
2197	6.769934e-08	14.525734	2.208495	0.992910	0.042176	0.548766	

	AR	elongation	boundedness	fractal_dim	...	Mean straightness	\
0	1.377828	0.274220	0.208504	2.180911	...	0.146740	
1	1.844000	0.457701	0.169997	2.007813	...	0.126079	
2	5.022024	0.800877	0.032339	1.367141	...	0.223336	
3	2.448351	0.591562	0.063546	1.763388	...	0.223336	
4	15.441644	0.935240	0.015247	1.908143	...	0.090795	
...	...	...	...	...	...	...	
2193	2.358700	0.576038	0.157919	1.918276	...	0.137948	
2194	2.538380	0.606048	0.048917	1.430580	...	0.175666	
2195	8.642961	0.884299	0.053421	1.486450	...	0.084041	
2196	1.957279	0.489087	0.121116	2.006209	...	0.137948	
2197	3.804139	0.737128	0.094830	1.693099	...	0.084041	

	Mean MSD_ratio	Mean Deff1	Mean Deff2	Track_ID	X	\
0	0.478852	1.910137	1.054939	0.0	456.755044	
1	1.023370	4.419614	0.879117	1.0	307.691753	
2	0.624219	5.257431	NaN	2.0	240.963795	
3	0.624219	5.257431	NaN	3.0	209.178269	
4	0.330886	1.253643	1.082479	4.0	213.264787	
...	...	...	...	...	...	
2193	0.230909	2.306063	0.446560	2317.0	2011.101311	
2194	0.090573	2.583338	NaN	2318.0	1711.064665	
2195	1.728507	0.991768	0.011171	2319.0	1976.195912	
2196	0.230909	2.306063	0.446560	2320.0	1961.472493	
2197	1.728507	0.991768	0.011171	2321.0	2000.926364	

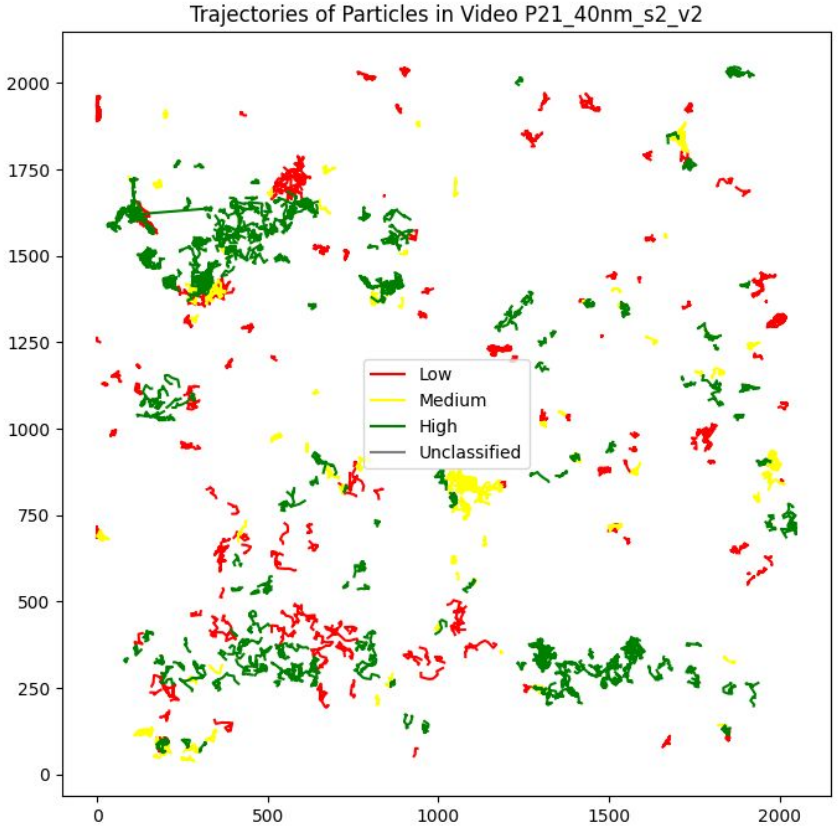
	Y	frames	Quality	Category
0	1975.544731	20.0	1.000000	high
1	1754.359393	16.0	1.000000	high
2	1691.939292	24.0	1.000000	high
3	1700.414792	30.0	1.000000	high
4	1586.784347	207.0	1.000000	high
...	...	...	...	...
2193	103.153933	19.0	1.000000	high
2194	399.234135	27.0	1.000000	high
2195	171.886109	28.0	0.933333	high
2196	106.022479	23.0	1.000000	high
2197	144.330820	23.0	0.800000	high

Visualization

1)
video_quality_map.
merge_data
(feature_path, msd_path, json_path)

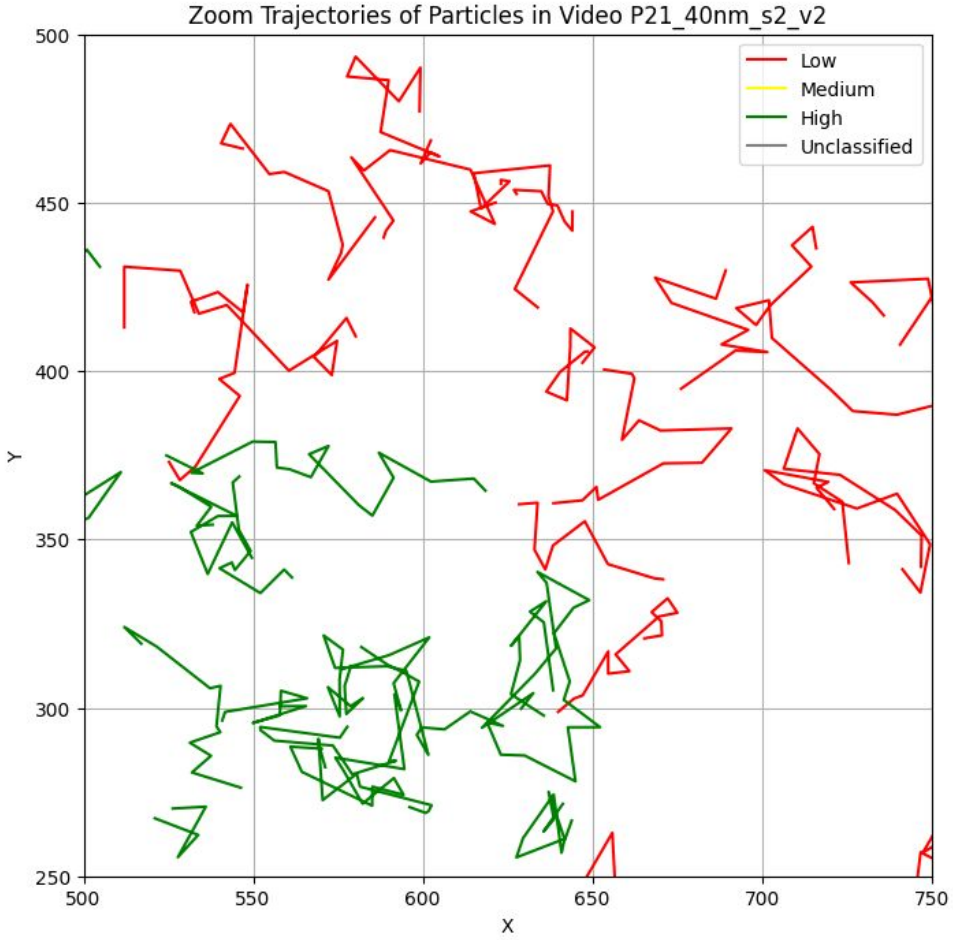
→ [**merge_df**, msd_data, **quality_data**]

2)
video_quality_map.
trajectory_plot
(**merge_df**, vid_code, save = None)



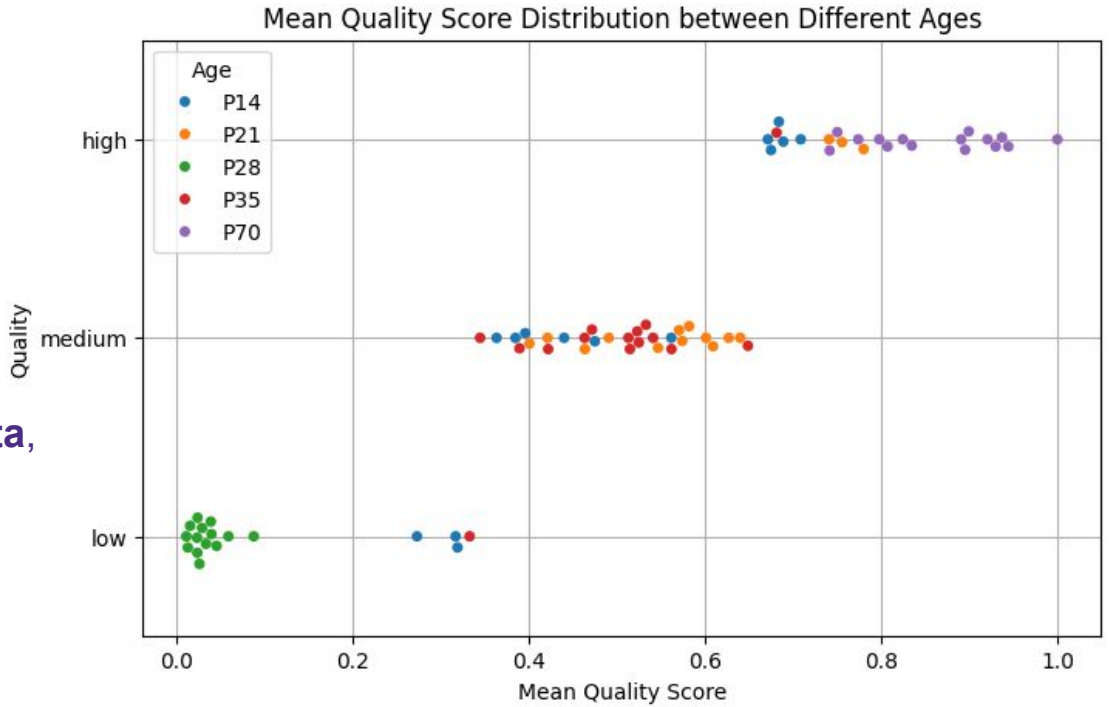
Visualization

3)
video_quality_map.
zoom_trajectory_plot
(**merge_df**, vide_code,
x1, x2, y1, y2, save = None)



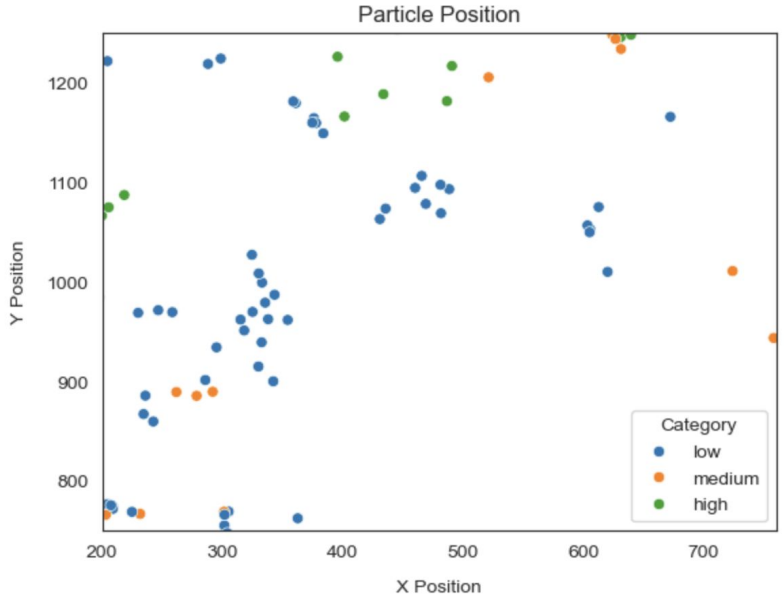
Visualization

4)
video_quality_map.
distribution_by_age
(feature_path, msd_path, **quality_data**,
save = None)



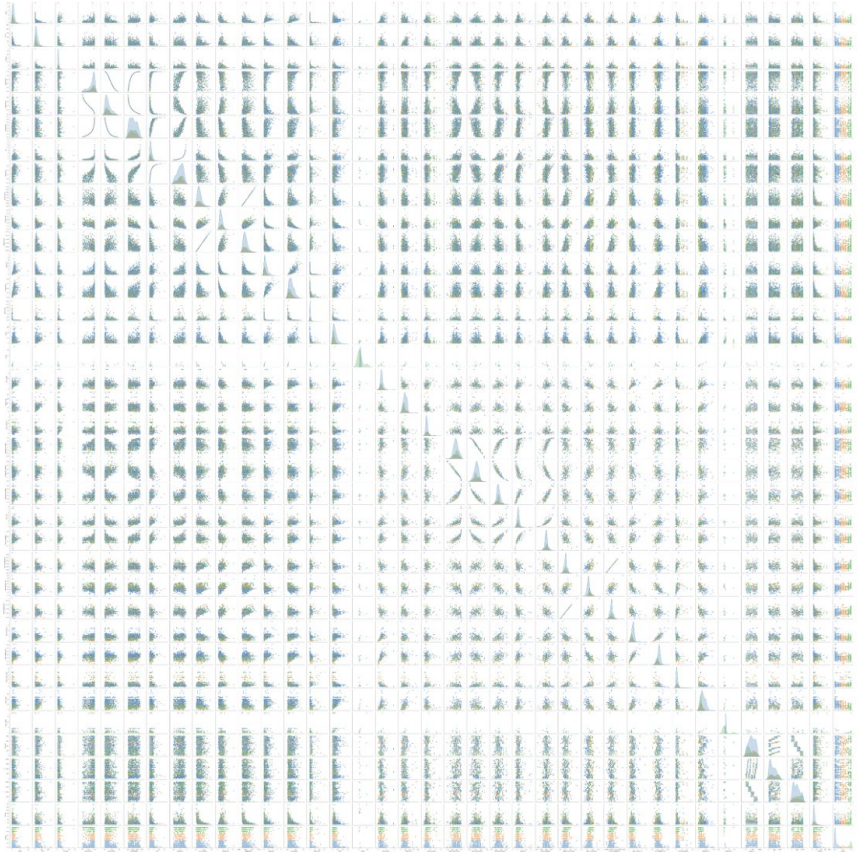
Visualization

TrackONautsVis.position_plot(dataframe, x="X", y="Y", title="Particle Position", x_bounds="", y_bounds="")



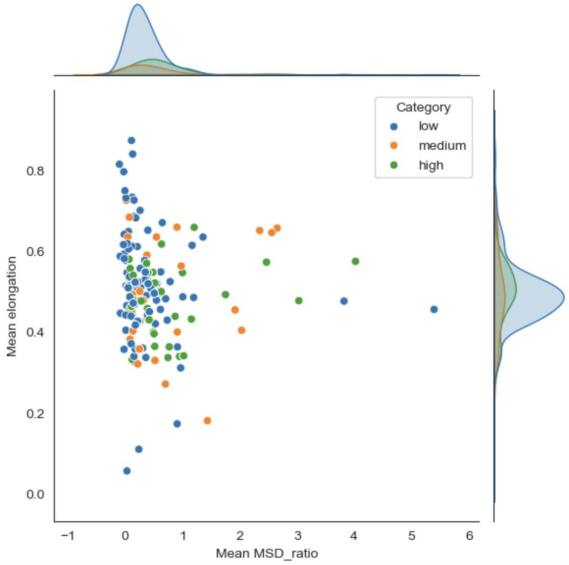
Visualization

TrackONautsVis.pairwise_plot (dataframe)

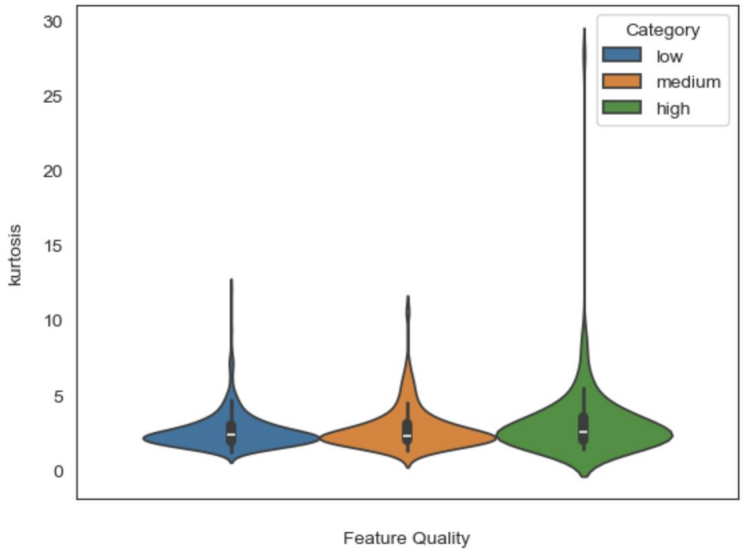


Visualization

TrackONauts.pair_plot(dataframe, feature1="", feature2 ="")



TrackONauts.violin_plot(dataframe, feature="")



Statistical Processing

Pearson pairwise correlation

TrackONautsStats.pairwise_correlation(dataframe)

⇧ **TrackONautsStats.corr_rowi_vs_all(row_i, dataframe)**

⇧ **TrackONautsStats.corr_rowi_rowj(row_i,row_j)**

Statistical Processing

Descriptive Statistics

TrackONautsStats.multi_df_feat_descriptive_statistics(dataframes, features)

⇧ **TrackONautsStats.feature_descriptive_statistics(dataframe, features)**

features: “all_features” or list of features

Mean, median, maximum, minimum, standard deviation, and variance

Can remove comment hash in code to specify quantile

Statistical Processing

Outliers

TrackONautsStats.feature_outliers(dataframe, features, outlier_method)

outlier_method: “STD multiplier” or “IQR”

Statistical Processing

TrackONauts.py example Jupyter Notebook

https://github.com/MPT-project-W24/track_o_nauts/blob/main/examples_TrackONautsStats.ipynb

Future Work

TrackONautsStats.py

- Currently does not process with data quality column
- Inferential statistics tool
- Add function to handle clustering methods

Code-free interface

Flagging low quality data using machine learning

Questions?