Methods:

For each of the two spectral images DC and Polymer, 8 clustering methods (kmeans, mean shift, ward, agglomerative clustering, dbscan, hdbscan, birch, and gaussian mixture) from the sklearn package were trained and evaluated compared to the ground truth.

Results:

Table and Figures shown at end.

For the figures, the left side shows the scatter plots of the pixels from the first and second bands. The reflectance of these bands is noted in the x and y labels and the colors correspond to the clusters predicted. The right side shows the first 100 rows of the image colored by the predicted clusters. (Figures were cut in half and shown side by side to provide better readability)

Overall the longest running models were ward, agglomerative clustering, and hdbscan. The fastest were nkmeans and dbscan. The best performing methods were different on each image. Given the small sample space of the polymer image which unlike the DC image, did not contain most of (or any of) the ground truth classes in that sample, the evaluation metrics are not very useful. Because of this I will focus on the DC image for my result comments. Mean shift and dbscan had essentially no similarities to the ground truth as shown by their AMIS and ARS scores as well as the plotted image. The agglomerative clustering performed the next worse in evaluation scores and we can see in the image it produces that the clusters are less refined and broader than the other images. The rest of the clustering methods performed relatively the same. **So given this, I would chose nKmeans as the best model given it produced the best results at the fastest time.**

Limitations:

- Many of the clustering methods caused the kernel to die during run likely due to memory constraints (I.e. dbscan, birch, etc.) when training on a very small sample but predicting on the entire image. Because of this, I limited the training and prediction to a sample of the data (all rows and bands but

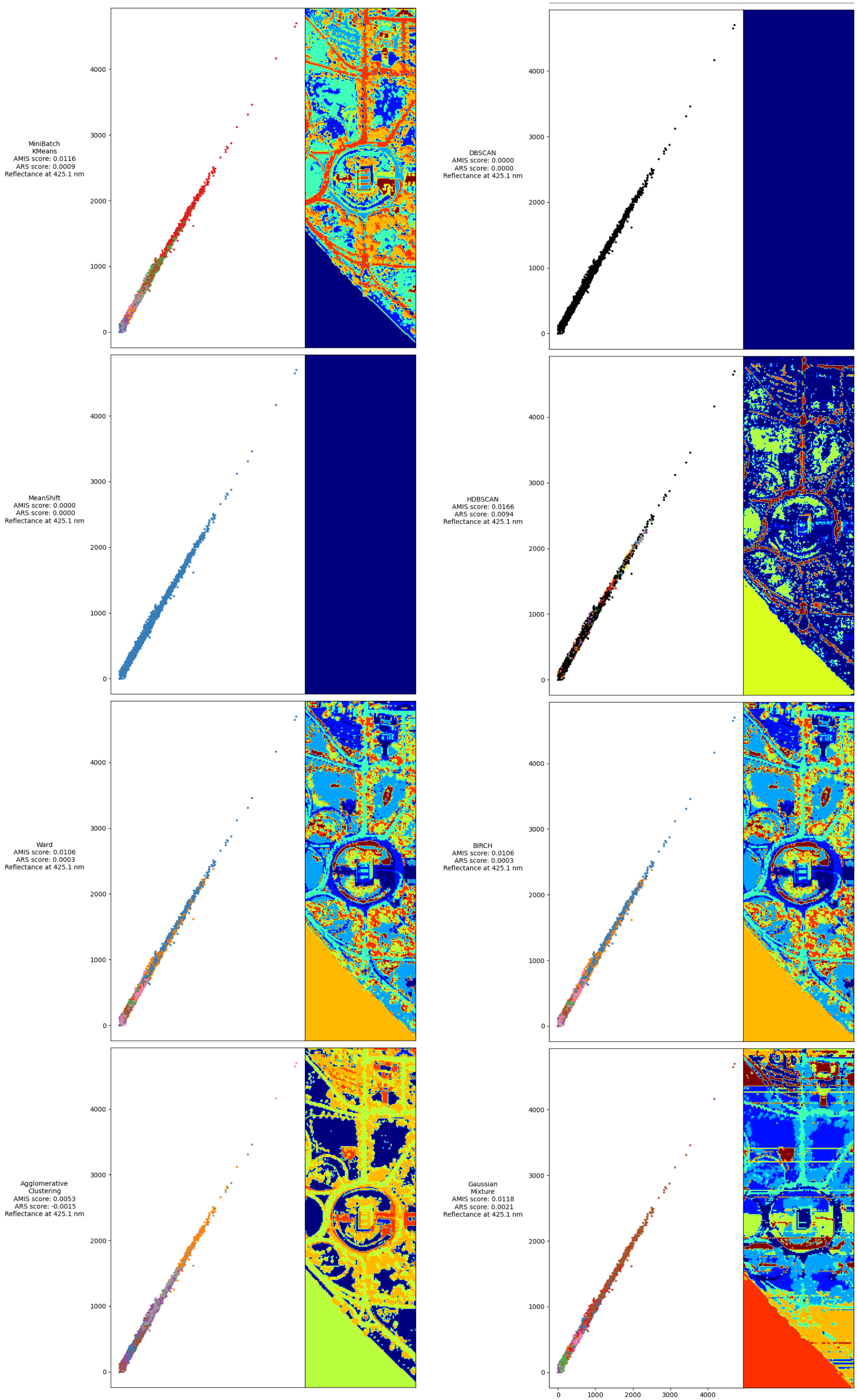
only the first 100 columns). This allowed for enough portion of the image to compare between clusters and a reasonable run-time. However, even with this small sample, three of the clustering methods (affinity propagation, spectral clustering, and optics) took too long (affinity propagation took over 459 minutes and still did not finish!). Because of this, I chose to remove these clustering methods from the final result in order to compare the rest of the methods on a sample of a useful size.
- The evaluation methods used may have bias towards the clustering methods with the same amount of clusters as the ground truth.

Improvements:

- I chose to use all bands as to not loose information, however, using PCA on the bands would have led to less memory and compute need while also keeping most of that information for training.
- I could have tuned the parameters on the individual clustering models more as for this exercise I chose to use the default parameters aside from selecting the number of clusters where possible.
- Testing alternative evaluation metrics to compare the cluster predictions to ground truth as not every clustering method (i.e. dbscan) predicts a given number of clusters which may cause bias against the models with a different numbers of classes from the ground truth.
- A cloud-based machine with more compute and memory than my personal computer may have allowed for the more memory intensive clustering methods to run on the sample.

Remote Ser

Figure 1.
DC
Image



Remote Sensi

Figure 2. Polymer Image

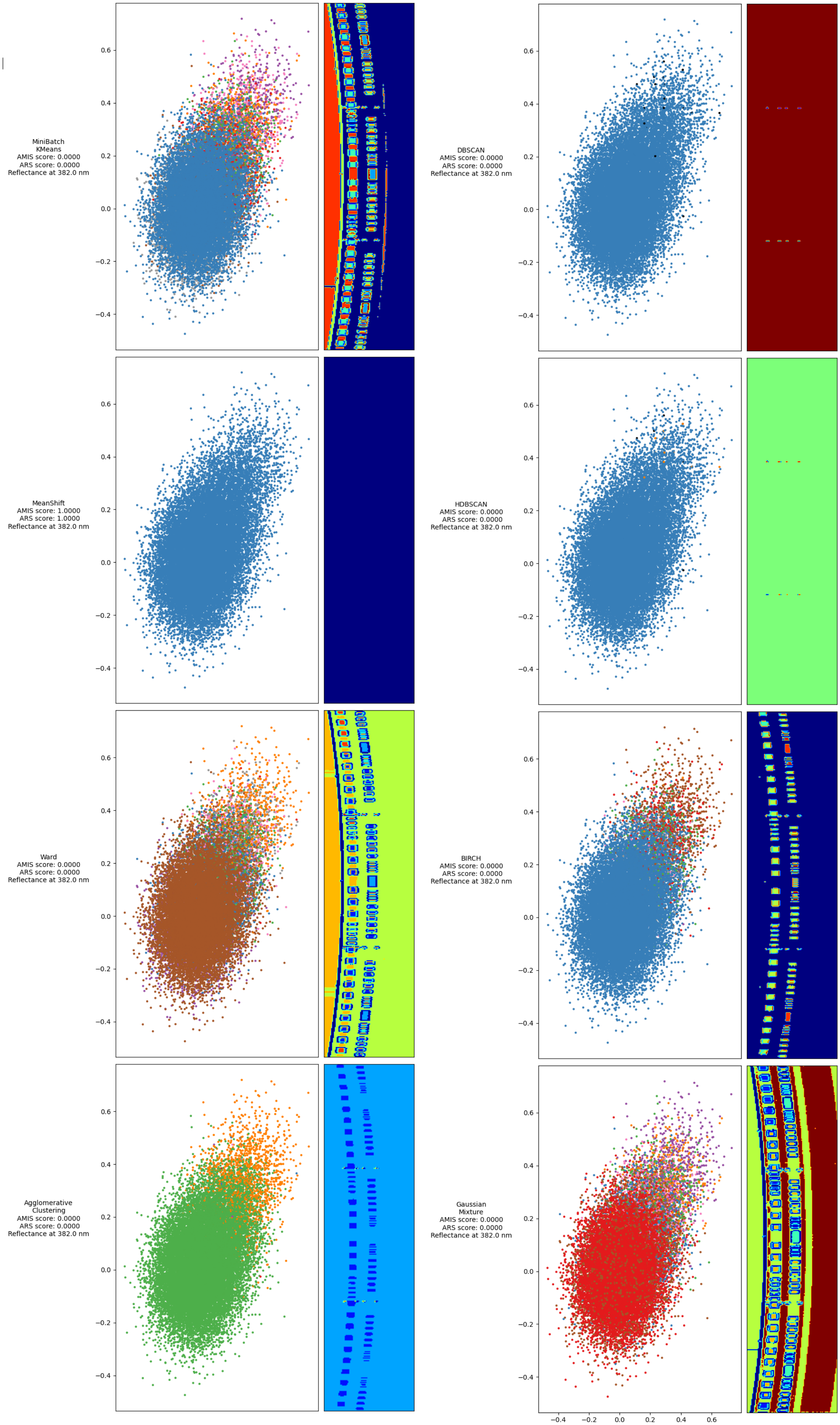


Table 1. Two evaluation metrics (Adjusted Mutual Info Score (AMIS) and Adjusted Rand Score (ARS)) and the combined run time for the model training predictions for each clustering method and image.

image	algorithm	amis	ars	time (s)
DC	MiniBatch\nKMeans	0.011582	0.000937	0.034682
DC	MeanShift	0.000000	0.000000	16.964519
DC	Ward	0.010622	0.000273	28.504380
DC	Agglomerative\nClustering	0.005287	-0.001499	26.892263
DC	DBSCAN	0.000000	0.000000	0.793465
DC	HDBSCAN	0.016557	0.009373	40.379908
DC	BIRCH	0.010622	0.000273	26.628665
DC	Gaussian\nMixture	0.011828	0.002131	21.474930
Polymer	MiniBatch\nKMeans	0.000000	0.000000	0.086364
Polymer	MeanShift	1.000000	1.000000	41.972381
Polymer	Ward	0.000000	0.000000	130.175778
Polymer	Agglomerative\nClustering	0.000000	0.000000	129.594754
Polymer	DBSCAN	0.000000	0.000000	21.988005
Polymer	HDBSCAN	0.000000	0.000000	383.369022
Polymer	BIRCH	0.000000	0.000000	0.489146
Polymer	Gaussian\nMixture	0.000000	0.000000	72.143228