Supplemental Information

Accurate Characterization of Mixed Plastic Waste using Machine Learning and Fast Infrared Spectroscopy

Stas Zinchik,[1] Shengli Jiang,[2] Søren Friis,[3] Fei Long,[1] Lasse Høgstedt,[3] Victor M.Zavala,[2] Ezra Bar-Ziv[1*]

[1] Department of Mechanical Engineering, Michigan Technological University, Houghton, MI, USA, 49931

[2] Department of Chemical and Biological Engineering, University of Wisconsin-Madison, Madison, WI, USA, 53706

[3] NLIR ApS, Hirsemarken 1, Farum, Denmark, 3520

Number of pages: 14
Number of tables: 1
Number of figures: 19

Dataset

Raw Spectra Data

We have 5,000 spectra for each plastic type. We have 14 plastic types including pure ABS, HIPS, PA12, PC, PE, PET, PLA, PMMA, POM, PP, PS, PVC, and blends of ABS/PC and PP/PE/PS. We have a total of 700,000 spectra. The ratio of training:validation:testing is 0.56:0.24:0.2, which means we have 392,000 training spectra, 168,000 validation spectra and 140,000 testing spectra. The normalized spectra are shown in Figure S1.

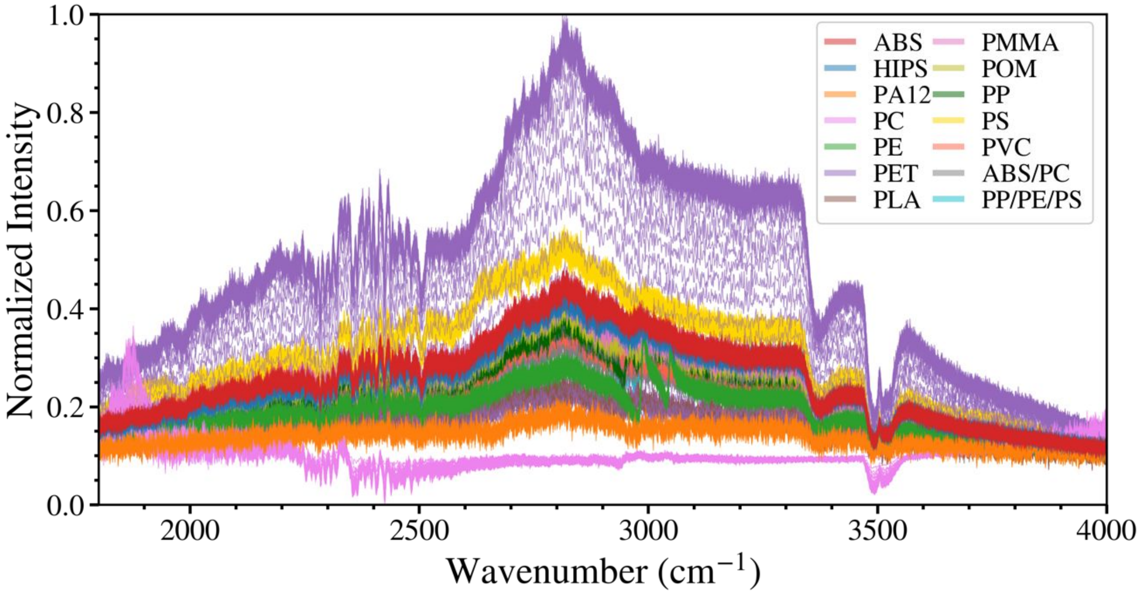

Figure S1. Normalized infrared spectral intensities of various plastic materials. Each spectrum is a vector of length 1600. The resulting spectra contain a large amount of noise.

Conversion to GAF Matrices

The conversion of spectra to GASF and GADF matrices is illustrated in Figure S2.

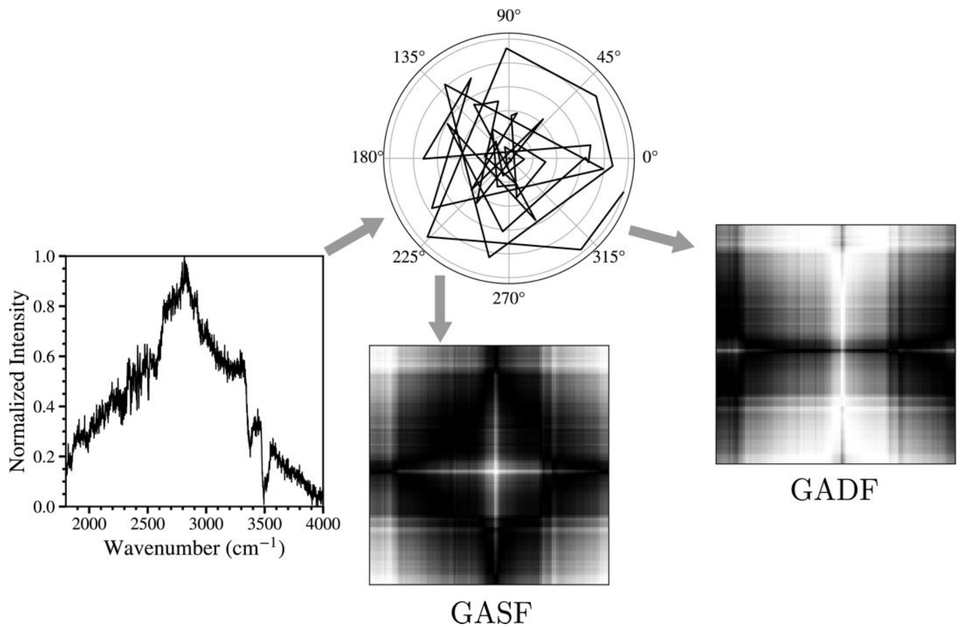

Figure S2. Transformation from 1D signal to GASF and GADF matrices. The 1D signal is first mapped to a polar coordinate system and then converted to GASF and GADF matrices. The encoding of one-dimensional signals into GAF matrices captures the relationship between signal intensities at different wavenumbers.

Five-fold Cross-validation

The original data set is randomly divided into five subsets of equal size. In these five subsets, one subset is kept as the test set and the remaining four subsets are used as training data. The cross-validation process is repeated five times, and each of the five subsets is used exactly once as the test data.

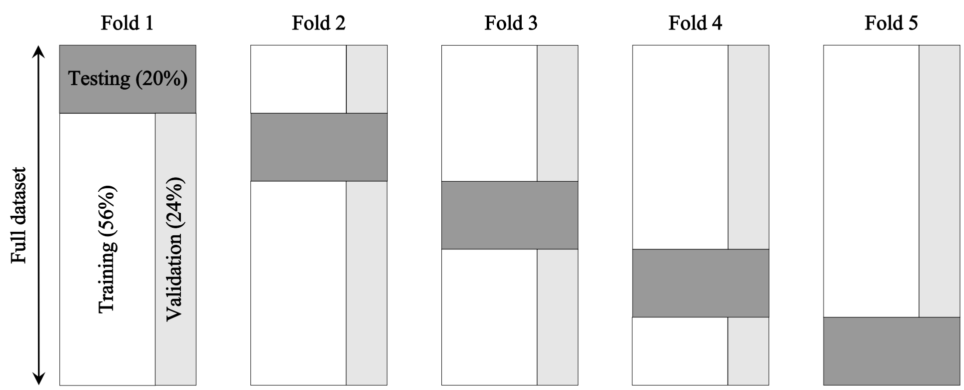

Figure S3. Schematic diagram of the 5-fold cross-validation procedure used to train and test the model. The ratio of training to test is 4:1. Within the training set, we randomly select 30% of the data as the validation set to tune the parameters of the model.

Computation Platform

All CNN training was performed on the University of Wisconsin-Madison's High Throughput Computing Center (CHTC) servers. The GPU was an NVIDIA GeForce RTX 2080 Ti. The detailed information about the server can be found on https://chtc.cs.wisc.edu/gpu-jobs. The tests were performed on a desktop with AMD Ryzen 7 3800X, NVIDIA GeForce RTX 2080 Super 8GB and 16GB of DDR4 memory.

Hyperparameter Optimization

We used grid search to optimize the hyperparameters for CNN training. The hyperparameters we optimized included learning rate, batch size, monitor function of the Keras model checkpoint and number of filters in each convolution layer.

Learning rate determines the step size of each iteration while moving towards the minimum value of the loss function. It was chosen from {0.0005, 0.001, 0.005}. Batch size is the number of training examples utilized in one iteration. It was chosen from {64, 128, 256}. The monitoring function records a metric during training and saves the weights of the model when the metric is optimal. It was chosen from {‘val_loss’, ‘val_acc’}. Here the loss was categorical cross-entropy, and we wanted to minimize the validation set loss or maximize validation set accuracy. The number of filters in each convolutional layer was chosen from {16, 32, 64}. In total, we have 54 different combinations of hyperparameters. The best hyperparameter combinations are shown in Table S1.

For support vector machine, the regularization parameter is chosen from {1, 2, …, 10} and the optimal value is 5. For random forest, the number of trees is chosen from {10, 20, …, 100} and the optimal value is 100. For KNN, the number of neighbors is chosen from {1, 2, …, 19} and the optimal value is 19.

Table S1. The best hyperparameters for each CNN model.

Model	Input size	Learning rate	Batch size	Model checkpoint	Number of filters	Accuracy
1D CNN	(1600, 1)	0.0005	256	val_loss	16	0.9997 ± 0.0001
2D CNN	(25, 25, 2)	0.001	64	val_loss	32	0.9923 ± 0.0002
2D CNN	(50, 50, 2)	0.0005	256	val_acc	32	0.9987 ± 0.0003
2D CNN	(75, 75, 2)	0.0005	128	val_loss	16	0.9992 ± 0.0001
2D CNN	(100, 100, 2)	0.0005	256	val_loss	32	0.9995 ± 0.0001

Convolutional Neural Network

Model Architecture

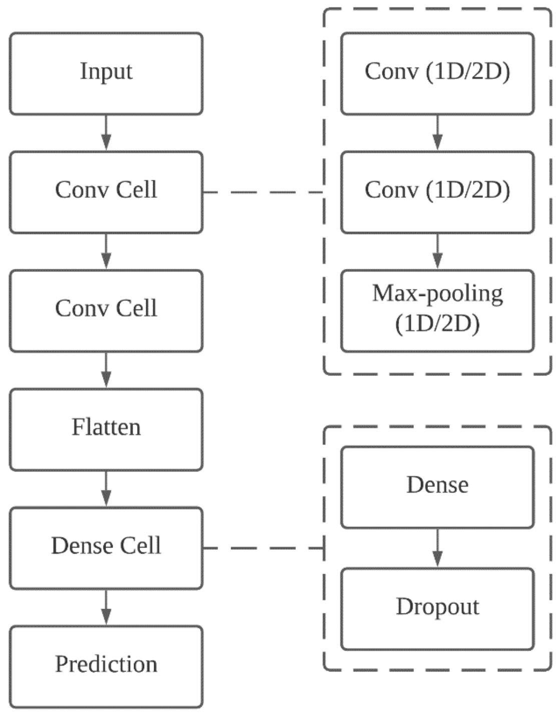

Figure S4. The model architecture of the 1D CNN and 2D CNN. The model has an input for 1D spectra or 2D GAF matrices and an output for the plastic type prediction. For 1D CNN, the input has a size of 1600. For 2D CNN, the input has a size of $25 \times 25 \times 2$, $50 \times 50 \times 2$, $75 \times 75 \times 2$ or $100 \times 100 \times 2$, where 2 represents the channels of GASF and GADF. The output for classification is the probability of an input in a plastic class. The Conv Cell contains two 1D or 2D convolutional layers and a 1D or 2D max-pooling layer. The Dense Cell includes a dense layer and a dropout layer. All Conv Cells and Dense Cells are identical. The detailed hyperparameters are listed in Table S1.

Learning Curves

The learning curves were analyzed to ensure convergence of training and validation. Figure SX shows the training and validation losses for the five models used for classification, and the categorical cross-entropy is the loss function under 5-fold cross-validation. The detailed hyperparameters for these five models are shown in Table SX. The training loss (solid line) for all CNNs reaches zero by 50 iterations, indicating that the models are fully trained. Some lines were cut off before 50 iterations because we used an early stop with 20 iterations of patience. Specifically, we considered the model overfitted and stopped training when the validation accuracy metric did not improve after 20 iterations. The validation loss (dashed line) also converged before 50 iterations, indicating that the model was not overfitted.

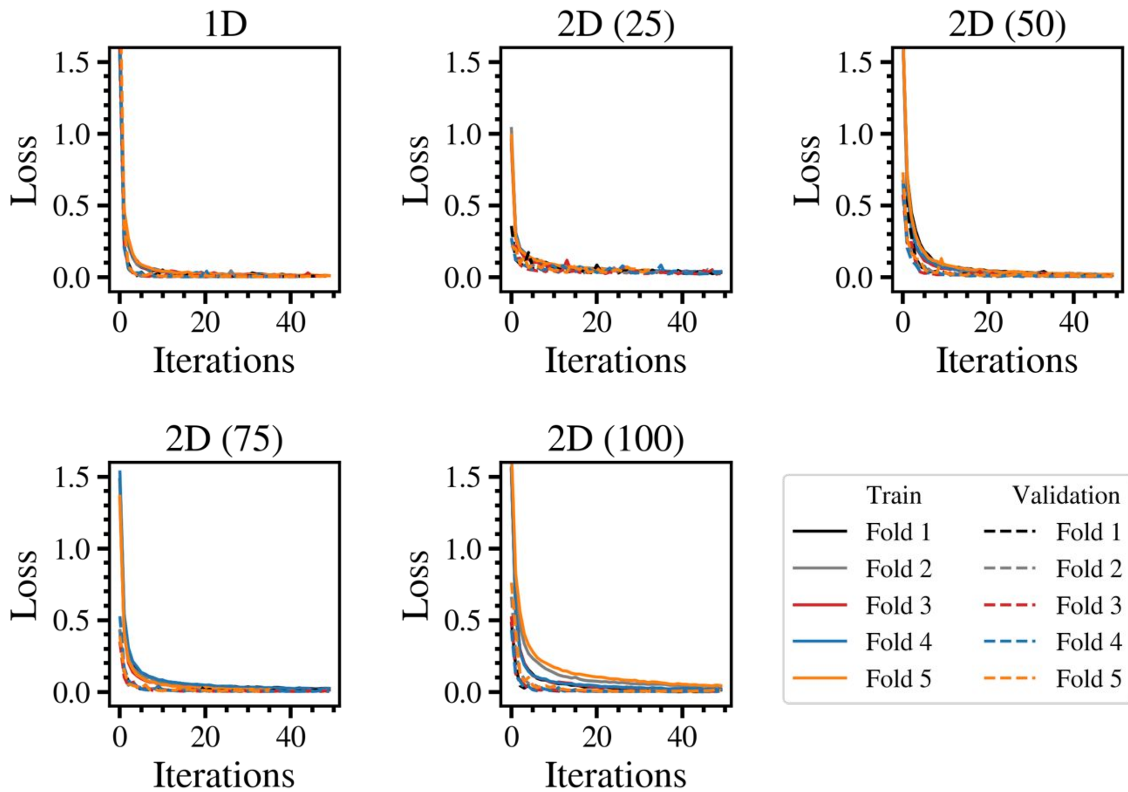

Figure S5. Example learning curves for CNN models. Training (solid lines) and validation (dotted lines) are shown across 50 iterations to predict the plastic types. Five models with different hyperparameters (detailed in Table S1) are used for comparison.

Model Complexity

CNNs are prone to overfitting because they contain a large number of parameters. In this study, the ratio of parameters to the number of training samples ranged from 10.67 to 42.67 for 1D CNNs and from 0.66 to 53.67 for 2D CNNs, depending on the specific model. Specifically, the number of parameters ranged from 41,8242 to 1,672,722 for 1D CNNs and from 25,774 to 2,103,758 for 2D CNNs. The number of training samples was 39,200. To prevent overfitting, we used regularization methods such as dropout and early stopping.

Confusion Matrix

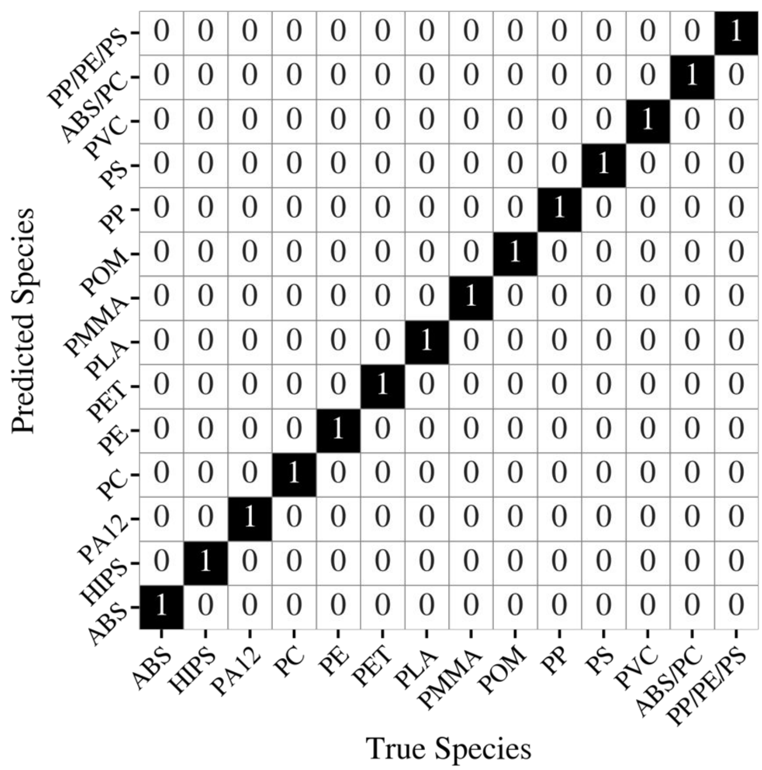

Figure S6. Confusion matrix of PlasticNet (2D) with an input size of 100 × 100 × 2. The overall accuracy is 100%. Each column represents a true plastic type, and each row represents a model-predicted plastic type. The entries along the diagonal lines are where the plastic types are correctly classified. The model predicts not only pure plastics, but also plastics containing blue pigments (PS), binary blends (ABS/PC) and ternary blends (PP/PE/PS).

Multi-label Training

Among the 14 plastic types, we have 12 pure plastic types that are ABS, HIPS, PA12, PC, PE, PET, PLA, PMMA, POM, PP, PS, PVC, and two mixtures of ABS/PC and PP/PE/PS. In multi-label training, we consider the labels as a binary vector in $\mathbb{Z}^n$, where n is the number of pure plastic types. Each entry of this vector represents the presence (1) and absence (0) of a plastic type. For example, the mixture of PP/PE/PS has three entries equal to 1 and other entries equal to 0, indicating the presence of PP, PE and PS only. The activation function of the prediction layer is sigmoid $\sigma(x) = \frac{1}{1+e^{-x}}$. The loss function is binary cross-entropy, which is as follows

$$\mathcal{L}(y, \hat{y}) = -\frac{1}{n}\sum_{i=1}^{n} y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i) \qquad (1)$$

where $y \in \mathbb{Z}^n$ is the true label and $\hat{y} \in \mathbb{R}^n$ is the predicted label probability. Each entry in $\hat{y}$ is activated by the sigmoid function.

Revalidation

We changed the optical system to examine the robustness of the algorithm. The test setup is shown in Figure S7. The revalidation model is the same as the one in Table S1 with an input size of $100 \times 100 \times 2$.

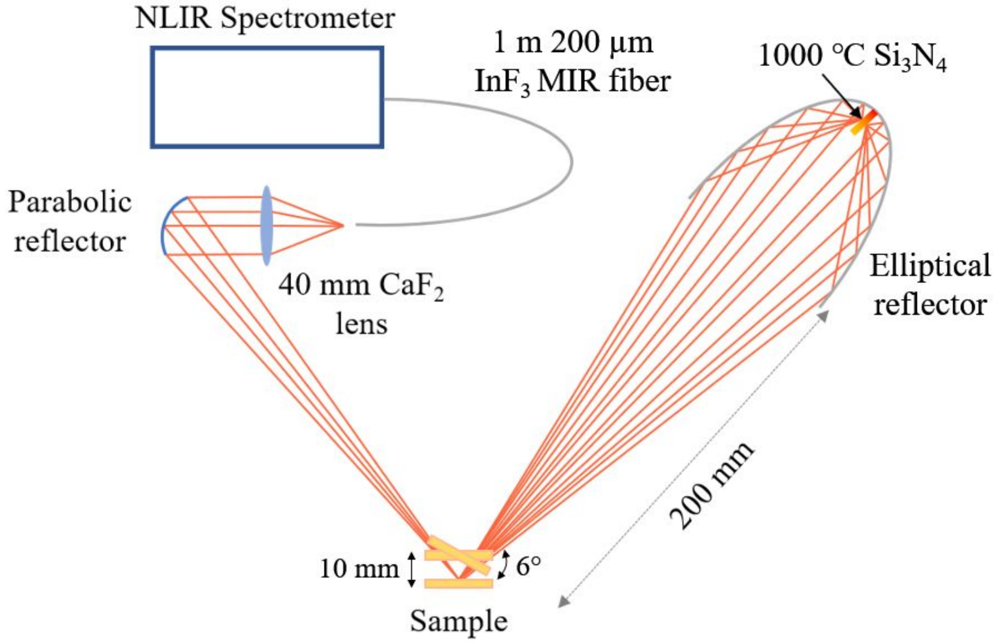

Figure S7. Robustness test with a 200 μm core indium fluoride (InF_3) fiber. The sample position is adjusted out-of-focus for 10 mm, and the sample orientation is adjusted for 6°.

Saliency Analysis

Here, we show saliency maps for all other plastic types.

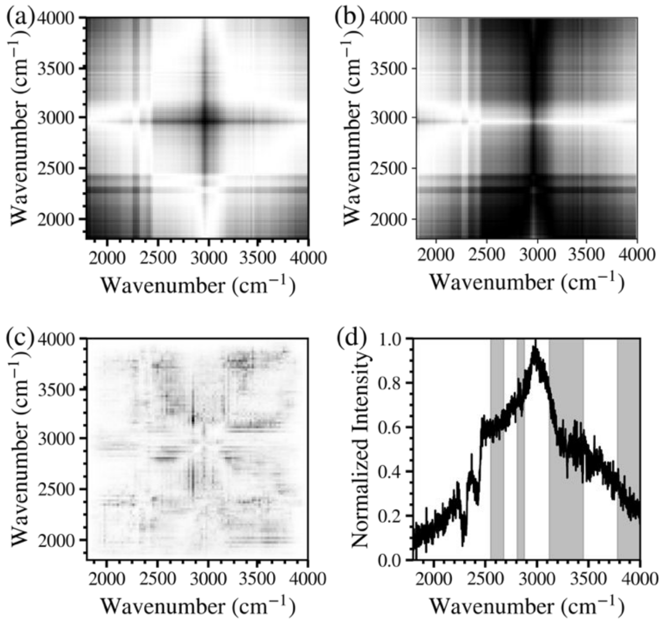

Figure S8. Saliency analysis for ABS.

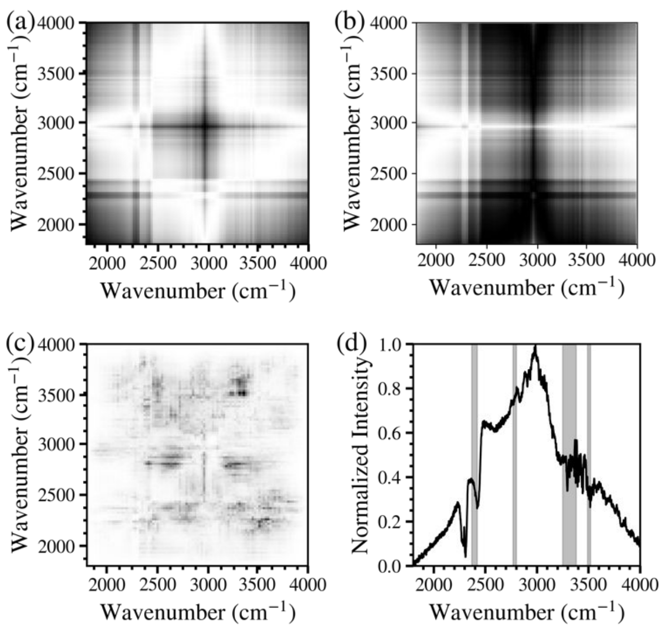

Figure S9. Saliency analysis for HIPS.

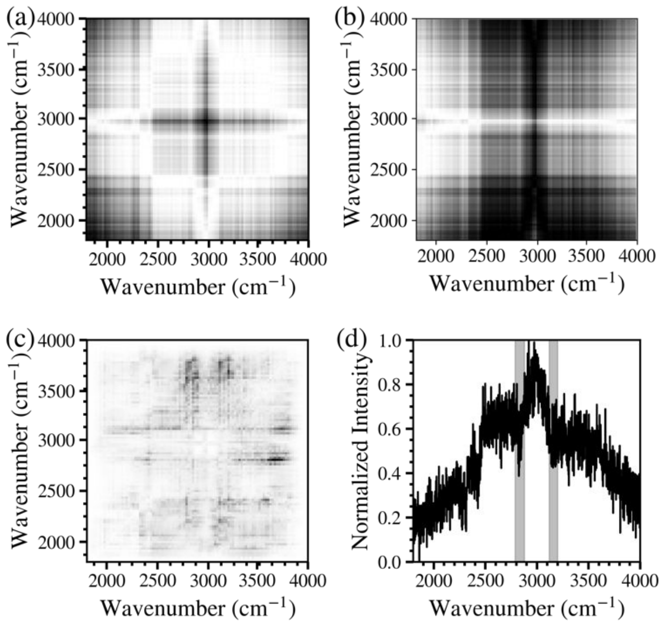

Figure S10. Saliency analysis for PA12.

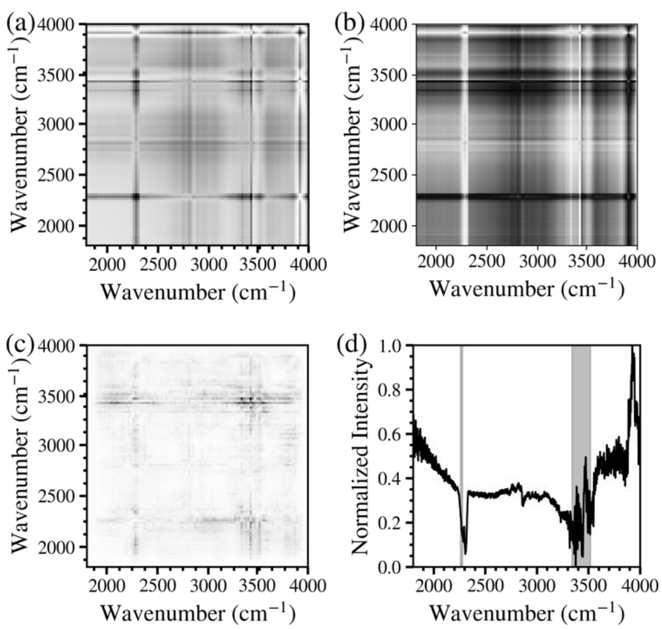

Figure S11. Saliency analysis for PC.

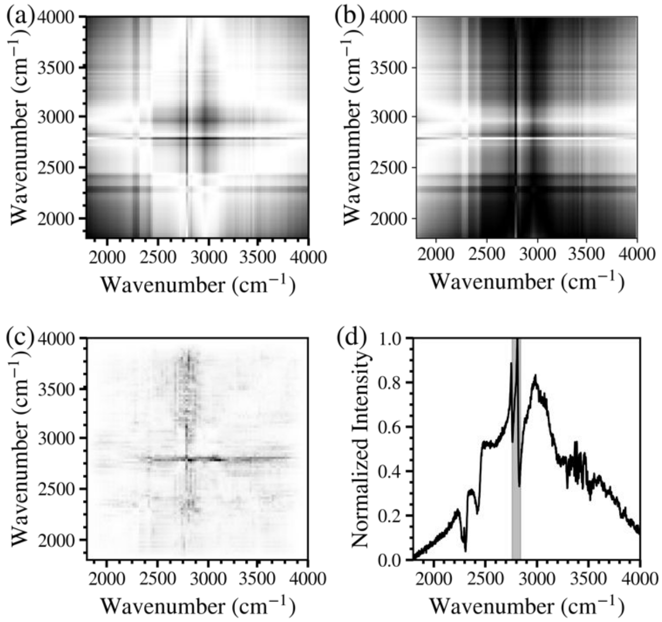

Figure S12. Saliency analysis for PE.

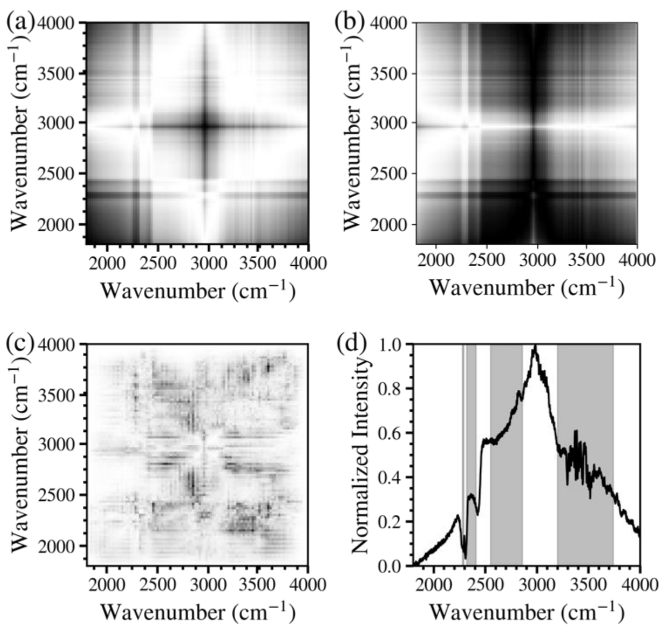

Figure S13. Saliency analysis for PET.

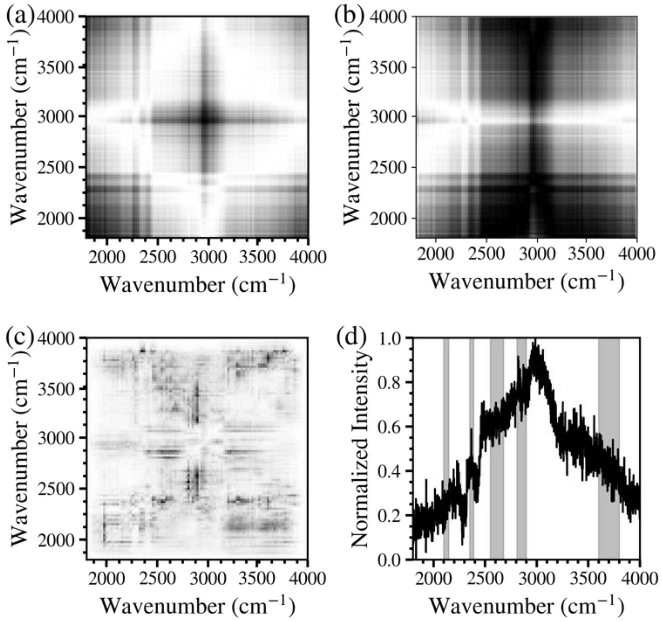

Figure S14. Saliency analysis for PLA.

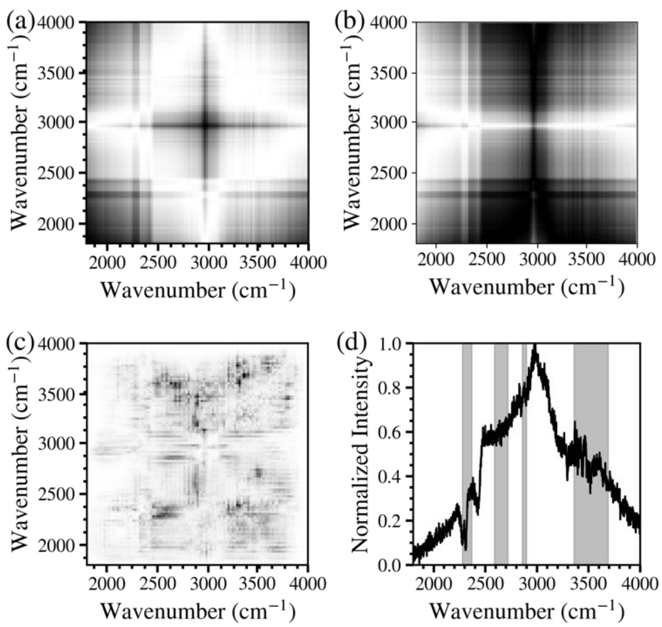

Figure S15. Saliency analysis for PMMA.

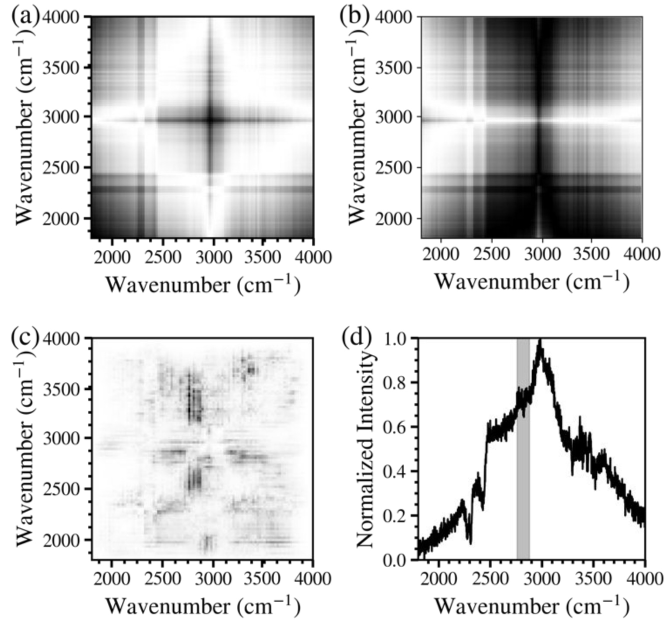

Figure S16. Saliency analysis for POM.

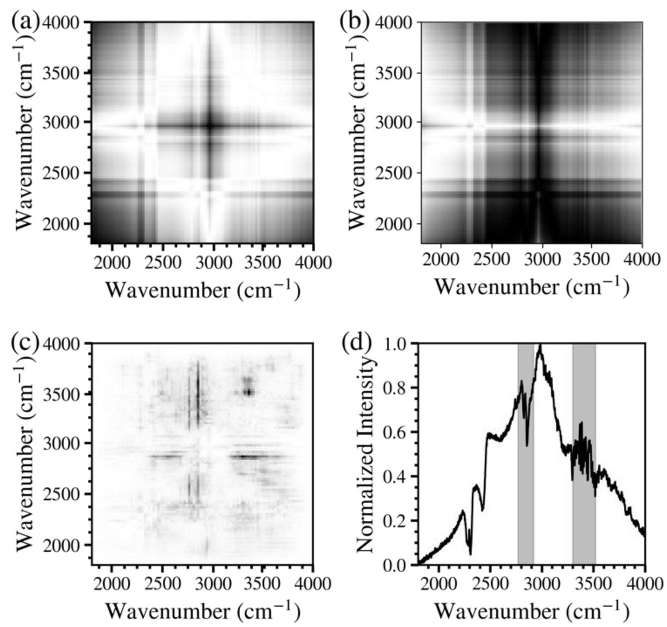

Figure S17. Saliency analysis for PP.

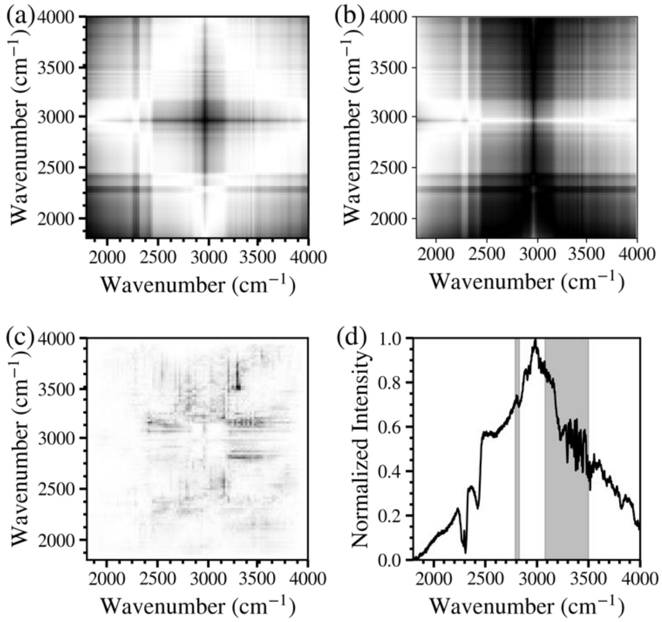

Figure S18. Saliency analysis for PS.

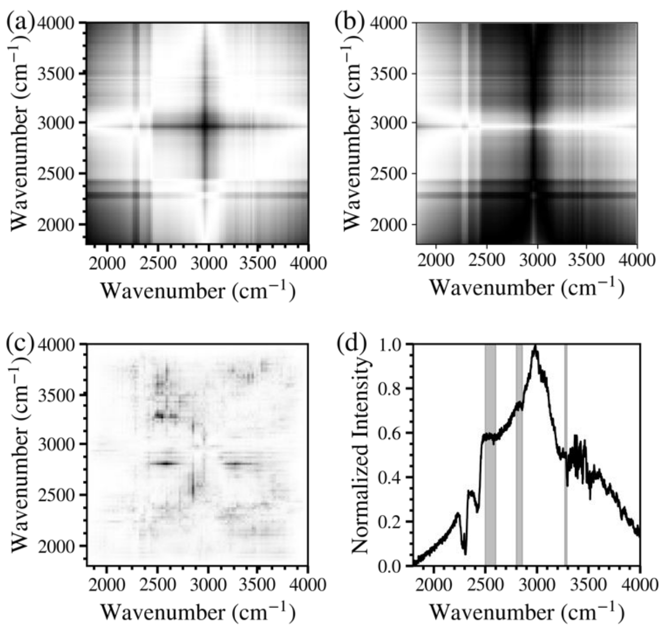

Figure S19. Saliency analysis for PVC.