EXPLOITING SPARSITY AND DICTIONARY LEARNING TO EFFICIENTLY CLASSIFY MATERIALS IN HYPERSPECTRAL IMAGERY

by

Andrew E. Pound

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Electrical Engineering

Approved:

Dr. Jacob Gunther Major Professor	Dr. Todd Moon Committee Member
Dr. Adele Cutler Committee Member	Dr. Mark R. McLellan Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2014

Abstract

Exploiting Sparsity and Dictionary Learning to Efficiently Classify Materials in Hyperspectral Imagery

by

Andrew E. Pound, Master of Science

Utah State University, 2014

Major Professor: Dr. Jacob Gunther
Department: Electrical and Computer Engineering

Hyperspectral imaging (HSI) produces spatial images with pixels that, instead of consisting of three colors, consist of hundreds of spectral measurements. The dimensionality of the data collected is extremely high, thus making analysis difficult. Frequently, dimension reduction techniques are incorporated in the HSI signal processing chain as a preprocessing step in order to reduce the dimensionality of the data. This reduction and change of basis can occlude the physics of the system.

This research explores the utility of representing the high-dimensional HSI data in a learned dictionary basis for the express purpose of material identification and classification. Multiclass classification is performed on the transformed data using the RandomForests algorithm. Performance results are reported.

In addition to classification, single material detection is considered also. Commonly used detection algorithm performance is demonstrated on both raw radiance pixels and HSI represented in dictionary-learned bases. Comparison results are shown which indicate that detection on dictionary-learned sparse representations perform as well as detection on radiance. In addition, a different method of performing detection, capitalizing on dictionary learning is established and performance comparisons are reported, showing gains over traditional detection methods.

(121 pages)

Public Abstract

Exploiting Sparsity and Dictionary Learning to Efficiently Classify Materials in Hyperspectral Imagery

by

Andrew E. Pound, Master of Science

Utah State University, 2014

Major Professor: Dr. Jacob Gunther
Department: Electrical and Computer Engineering

Hyperspectral imaging (HSI) produces spatial images with pixels that, instead of consisting of three colors, consist of hundreds of spectral measurements. Because there are so many measurements for each pixel, analysis of HSI is difficult. Frequently, standard techniques are used to help make analysis more tractable by representing the HSI data in a different manner.

This research explores the utility of representing the HSI data in a learned dictionary basis for the express purpose of material identification and classification. Multiclass classification is performed on the transformed data using the RandomForests algorithm. Performance results are reported.

In addition to classification, single material detection is considered also. Commonly used detection algorithm performance is demonstrated on both raw radiance pixels and HSI represented in dictionary-learned bases. Comparison results are shown which indicate that detection on dictionary-learned sparse representations perform as well as detection on radiance. In addition, a different method of performing detection, capitalizing on dictionary learning is established and performance comparisons are reported, showing gains over traditional detection methods.

Acknowledgments

I would like to express my gratitude to Dr. Gunther and Dr. Moon, who have both helped me in innumerable ways. I am especially grateful for their patience and encouragement.

I am also very thankful to my parents, Wes and Brenda Pound, for teaching me to always try my hardest and to reach for what I want most.

I also want to say thanks to my family. I'm grateful for three wonderful boys that I love to spend time with. Austin, Gavin, and Jordan, I love you all. And I would be remiss to not express my love and gratitude to my wonderful wife for her patience, love, and support that she has provided while I have been doing this research and working on this thesis. Debra, thank you. I love you with all my heart.

For Vectron!

Andrew E. Pound

Contents

List of Tables

List of Figures

Chapter 1

Hyperspectral Imaging

1.1 Introduction

A standard color image captured by a common camera is a spatially oriented image recording light using three bands, one each centered in the colors, red, green, and blue. Multispectral imaging extends this approach, capturing images using more bands and incorporating more information that can be used for interesting applications, such as historical document restoration. Hyperspectral imaging (HSI) further extends this to the capture of spatial images using hundreds of spectral bands. Often the bands are narrow and close enough together to warrant calling the samples in the wavelength dimension a spectrum. The data collected from these images is in the form of a cube, with the first two dimensions representing the spatial dimensions captured, and the third dimension comprising of the wavelength information. An example of a datacube, as they are called, can be seen in fig. 1.1. The objective of using HSI is to take advantage of the high spectral resolution to

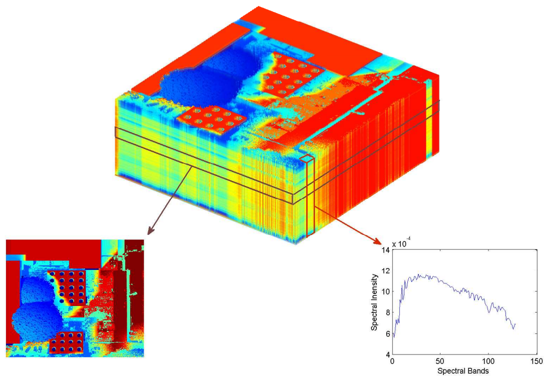


Fig. 1.1: An example of a datacube. The first two dimensions represent the spatial dimensions and the third provides the wavelength, or spectral, content.

aid in the identification and differentiation of materials in a scene of interest.

HSI is used in many different fields, including geological surveys, medicine, forestry [2–4], and defense efforts [5–7]. Each of these fields utilizes the spectrum obtained to aid in image segmentation and analysis, providing more accurate and reliable information than that which can be gained from image processing on standard monochrome or tri-color images.

Much of the early research in HSI was based in geological remote sensing. It is frequently used for detecting mines and geological exploration [8, 9]. Of particular interest to the geological researchers is the identification and mapping of minerals [10, 11]. A number of papers have been published that provide a good overview of these subjects [12, 13].

There has been significant interest in utilizing HSI in a wide variety of applications in the medical field. Researchers have applied HSI to help in diagnoses [14], in the characterization of kidney stones [15] and tumors [16, 17], and even in the study of cancer [18, 19]. Many different areas of the medical field are finding use in spectral characterization of tissues and samples. More references and results are reported in the review papers [20–22].

1.2 Physics of HSI

Hyperspectral Imaging is a passive remote sensing technique. That is, a hyperspectral system does not illuminate the scene in order to take a measurement, as do active sensing systems (RADAR, SEM, STM, etc.). A hyperspectral system measures the amount of radiation emitted or reflected by the objects in the scene.

The measurement that the sensor actually receives is a measurement of radiance for each spectral wavelength bin. This is a conglomeration of many different terms and influences. The radiance incident on the sensor comes from many different sources. Solar radiance is scattered off of the atmosphere, the targets in the scene, and also of of nearby objects outside of the scene, before it is collected by the camera. At times, radiance can be reflected by multiple surfaces before being scattered into the sensor field of view. Each of these reflections is called a bounce. The more bounces experienced, the more atmosphere the radiance had traveled through, attenuating the radiance more than radiance that is directly scattered into the sensor. Neglecting multi-bounce terms of more than one bounce gives a radiance that depends on eight different terms that are then cascaded with the sensor effects. This equation is developed and derived in detail by Schott [1]. The governing equation, what Schott refers to as the "*big equation*" is

$$\begin{aligned} L &= L_A + L_D + L_B + L_E + L_G + L_H + L_C + L_F \\ &= \left[\overbrace{E_{s\lambda} cos(\sigma) \tau_1(\lambda) \frac{r(\lambda)}{\pi}}^{L_A} + \overbrace{\epsilon(\lambda) L_{T\lambda}}^{L_D} + \overbrace{F\,(E_{ds\lambda}}^{L_B} + \overbrace{E_{d\epsilon\lambda})\,\frac{r_d(\lambda)}{\pi}}^{L_E} \right. \\ &\quad \left. + \underbrace{(1-F)\,(L_{bs\lambda}}_{L_G} + \underbrace{L_{b\epsilon\lambda})\,r_d(\lambda)}_{L_H} \right] \tau_2(\lambda) + \underbrace{L_{us\lambda}}_{L_C} + \underbrace{L_{u\epsilon\lambda}}_{L_F}. \end{aligned} \tag{1.1}$$

The terms are identified with labels which correspond to the paths illustrated in fig. 1.2, demonstrating how it reaches the sensor. A deeper understanding can be obtained by reading Schott's excellent book [1], particularly focusing on Chapters 3 and 4.

The data used in this study use long-wave infrared (LWIR) measurements. Because self emission dominates in the LWIR region, many of the terms of the "*big equation*" drop out or are negligible. Thus, a LWIR pixel measurement at wavelength λ comprised of a single material is generally described by

$$L(\lambda) = [\epsilon L_e(T) + (1-\epsilon)\,(F L_d + (1-F)\,L_b)]\,\tau + L_u. \tag{1.2}$$

Each term is dependent on λ, but this is dropped for easier notation. Also note, subscripts are different, allowing for easier identification as detailed hereafter. The material emissivity is denoted ϵ, and the reflectance is represented by $(1-\epsilon)$. The radiance emitted from the material is incorporated in the ϵL_e term and is generally a blackbody curve modulated by the emissivity. The strong dependence on temperature is indicated. τ is the atmospheric transmissivity from the target to the sensor. The terms L_d and L_u refer to the downwelling and upwelling radiances, respectively. Downwelling is the effect of radiation originating

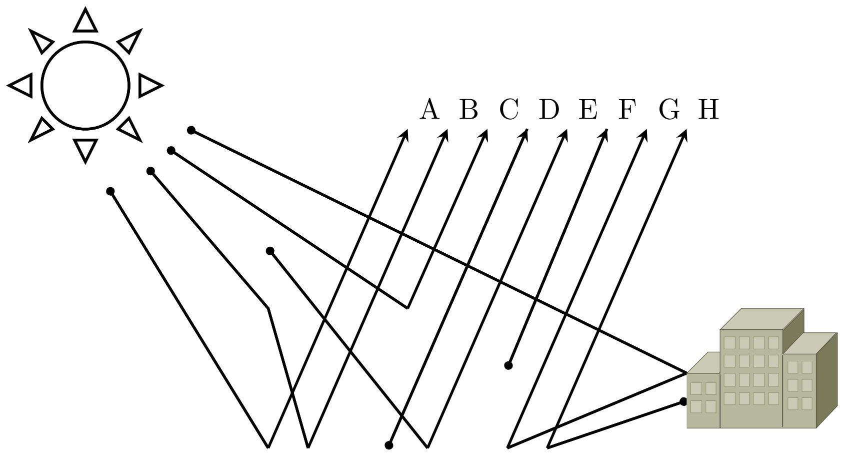


Fig. 1.2: Radiance paths reaching a sensor. Each path corresponds to a term in eq. (1.1). (This figure is based on one in Schott's book [1].)

from (or reflected off of) the atmosphere and then subsequently reflected off the target in the scene. Upwelling is a similar effect that accounts for radiation that is emitted or scattered off the atmosphere between the target and the sensor. These down- and upwelling radiance terms encompass the energy originating from the atmosphere before and after being reflected off the target. L_b is the background radiance, collecting any nearby radiating source that reflects off the target and is then scattered into the sensor. Each of these radiances are possibly dependent on many factors including, but not limited to, temperature and atmospheric conditions. If there is little radiance incident on the target from any strongly emitting nearby sources, or the target is unobstructed and not reflecting such radiance, the L_b background term can, and often is, dropped, leaving the simpler equation

$$L(\lambda) = \left[\epsilon L_e(T) + (1-\epsilon)L_d\right]\tau + L_u. \tag{1.3}$$

Stacking these up gives us the radiance vector of a single HSI pixel $\mathbf{l}$, where each element corresponds to a single wavelength λ. This is shown with the explicit dependence on wavelength in the equation

$$\mathbf{l} = \begin{bmatrix} l_{\lambda_1} & l_{\lambda_2} & \dots & l_{\lambda_N} \end{bmatrix}^T, \quad \text{where} \quad l_\lambda = \left[\epsilon_\lambda L_{e\lambda}(T) + (1-\epsilon_\lambda)L_{d\lambda}\right]\tau_\lambda + L_{u\lambda}. \tag{1.4}$$

In the above equations, the material specific parameter is the emissivity. The emissivity is an intrinsic property of each material and provides the information to determine the type of material that the pixel is a measurement of. The process of extracting the emissivities from the radiances involves separating the atmosphere and temperature dependencies from the emissivities in some way. This is a difficult problem and is an active area of research. Many of the currently used methods perform significant and costly pre-processing to mitigate atmospheric, downwelling, and upwelling effects.

One of the goals for this research is explore the possibility of extracting identifying information from the radiances, instead of requiring emissivities in order to provide reasonable results.

1.2.1 Brightness-Temperature

Industry and national lab experiments are often performed on emissivities, because emissivity is an intrinsic characteristic of each material. Our data is in radiance units (microflicks), so it is a difficult problem to separate the temperature, emissivity, and atmospheric components even in the simplified regime of this experiment.

In order to use something similar to emissivities, we have done a portion of this work on brightness temperature profiles produced from the radiance pixels.

The method to calculate the BT of a radiance pixel is by inverting Planck's law

$$E_\lambda = \frac{2hc^2}{\lambda^5} \frac{1}{e^{\frac{hc}{\lambda k_B T}} - 1}, \tag{1.5}$$

to get

$$T = \frac{ch}{k_B \lambda} \frac{1}{\ln\left(\frac{c^2 h}{\lambda^5 E_\lambda} + 1\right)}. \tag{1.6}$$

This does have some adverse effects on the data. Because we have not compensated for any atmospheric effects, they are still seen in the data. These can mask the true emissivities, although the data will "look" more like emissivities.

The measurements are now considered a temperature in Kelvin (K), so we often remove the mean, to put each measurement around the same temperature. This is analogous to removing the blackbody from the radiance data. Figure 1.3 shows a comparison between radiance pixels and BT pixels. A histogram of an entire cube is shown in fig. 1.4, demonstrating that many of the pixels in a cube lie on the blackbody curve at a specific temperature. The transformation to brightness temperature effectively removes the curve from the pixels.

1.2.2 Multiple Materials

Often HSI pixels cannot be assumed to be pure. That is, the target resolution is not sufficient to assume that a pixel is comprised of a singular material, but is instead a mixture of materials. For example, imaging a patch of grass would collect the spectral content of the grass, but some portion of the underlying dirt or rock would also likely be captured. This mixing of materials in a single pixel is often modeled in a simple linear mixing model. Given the radiance measures $\mathbf{l}_i$ of each the k separate materials, the pixel is often given as

$$\mathbf{r} = L\boldsymbol{\alpha}, \quad L = \begin{bmatrix} \mathbf{l}_1, \mathbf{l}_2 \ldots \mathbf{l}_k \end{bmatrix}, \tag{1.7}$$

where $\boldsymbol{\alpha}$ is a vector of abundance values indicating the proportion of the pixel that each material occupies or comprises. The $\mathbf{l}_i$ vectors represent spectrally pure or representative class spectra. Often, constraints are placed on the elements of $\boldsymbol{\alpha}$ such as $\sum_i \alpha_i = 1$ and $\alpha_i \geq 0,\ \forall i$. This mixing model is decent if the materials mixed in a single pixel do not have too much interaction, that is, the spectral features are not dependent upon each other or reflected off of each other.

1.3 Current Methods for Hyperspectral Image Unmixing

Significant research has been done on methods to be able to unmix hyperspectral pixels. Many of these popular techniques require or assume that pixels exist in the scene that are purely one material or class, and that every material in the scene is represented in this way.

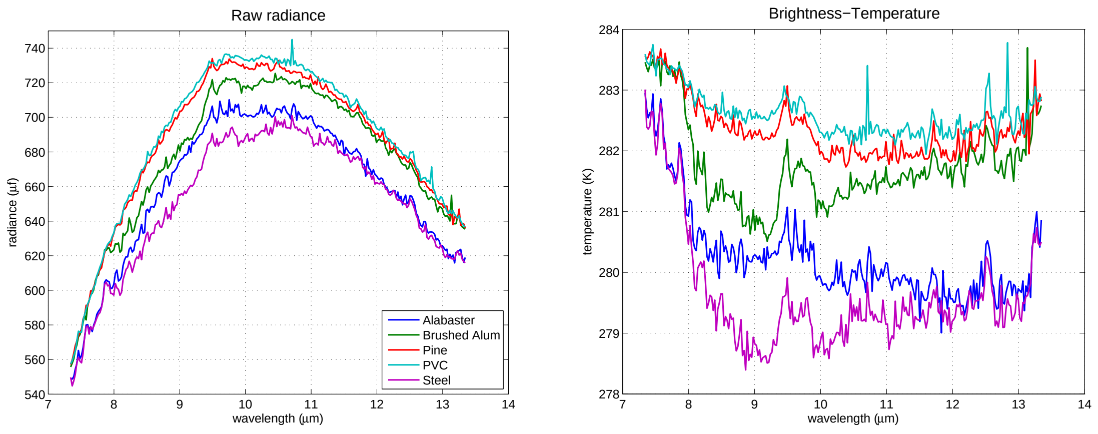


Fig. 1.3: A comparison between radiance pixels (left) and the same pixels converted to brightness-temperature (right).

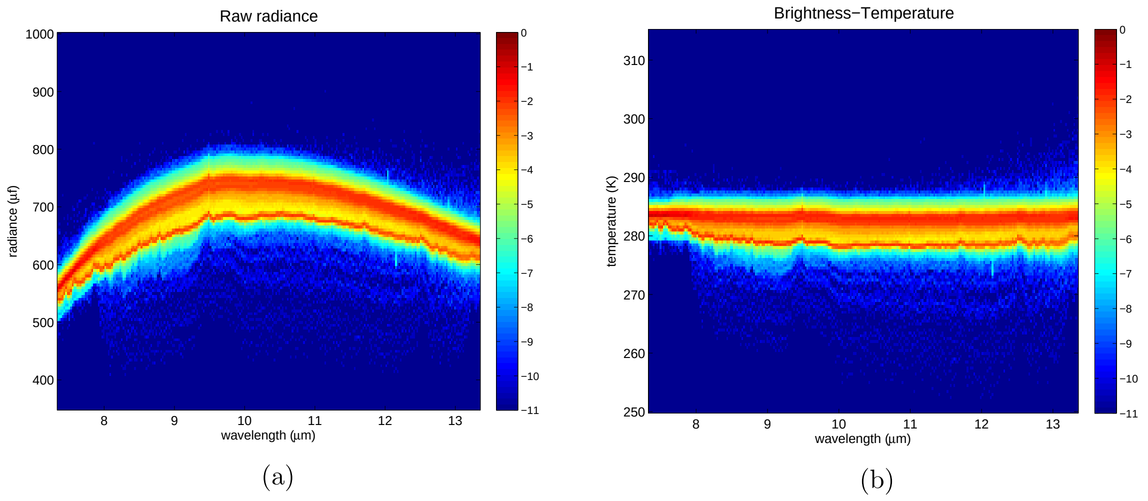


Fig. 1.4: A heatmap showing the distribution of the (a) radiance and (b) brightness-temperature (BT) $\log_{10}$ histograms of all pixels in an HSI cube. It can be seen that a large amount of the pixels fall on a blackbody curve, and that the BT removes it.

Many algorithms exist which focus on the segmentation of the image into constituent materials. Two main goals divide the algorithms into two classes: those which look to identify anomalies/targets, and those which work to assign to each pixel a class, identifying it as a certain material (or a group of classes, identifying it as those materials). Each class of algorithms works to achieve its objective with minimal misclassification error. Many of the algorithms are based on the idea that the data lie on a low-dimensional manifold that is embedded within the higher-dimensional dataspace. For this reason, there has been a large amount of research in the field of dimension reduction and manifold embedding/unfolding as applied to HSI data. Some of these algorithms that look for a lower dimensional representation are NFINDR [23], ISOMAP [24], and Locally Linear Embedding [25].

Many of the methods developed to aid in hyperspectral unmixing or segmentation are focused on what are known as endmembers. Endmembers are pixels from the data which form a linear basis for which the other pixels in the scene can be represented. Often it is assumed that there exists at least one pure pixel of each material present in a given scene, and that these are identified as endmembers.

N-FINDR

In 1999, Winter proposed identifying the "pure" elements of the scene, referring to these as endmembers, and representing the rest as combinations of these [23]. His algorithm is a main stay in the HSI community and is called *N-FINDR*. The manner in which he proposed to find these endmembers was to randomly select a data vector to be a part of the endmember set $\mathbf{E}$, and compute the volume of the simplex formed using the candidate endmembers as vertices using

$$V(\mathbf{E}) = \frac{1}{(M-1)!}\text{abs}\left(|\mathbf{E}|\right), \tag{1.8}$$

and

$$\mathbf{E} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ \mathbf{e}_1 & \mathbf{e}_2 & \dots & \mathbf{e}_N \end{bmatrix}. \tag{1.9}$$

$V(\mathbf{E})$ is the calculated volume encompassed by the N candidate endmembers $\mathbf{e}_i$. Then each candidate endmember is replaced with every other pixel available in the dataset. If the volume increases, then it stays in the candidate set. Each pixel is substituted into each endmember position to find the largest volume possible.

When the largest volume simplex is found, then other pixels are represented as linear combinations of the identified endmembers. The coefficients found in the unmixing are the abundance of each of the endmembers. Winter suggests utilizing either a least squares or a non-negatively constrained least squares method to perform the unmixing [23].

This algorithm requires a foreknowledge of the number of expected endmembers N, and can only return that number of them. It also requires preprocessing in order to reduce the dimension of the hyperspectral pixels down to $N-1$. This is needed in order to calculate the determinant of $\mathbf{E}$, which must be a square matrix. In addition to these limitations it returns pixels from the scene as the endmembers, explicitly assuming that there exist pure pixels of each material or endmember in the scene and that the other pixels are mixtures of the extracted endmembers.

ISOMAP

Isometric Feature Mapping (ISOMAP) is actually not specifically designed for hyperspectral information but for any very high dimension data that lies upon a low-dimension manifold embedded in a much higher dimensional space [24]. The method is similar to Multi-Dimensional Scaling (MDS) in that it calculates a distance between each point and calculates an embedding that preserves as much as possible these inter-point distances. But where ISOMAP differs is in the calculation of the inter-point distances. It uses a pseudo-geodesic distance defined by nearest neighbors on the graph which represents the embedded manifold.

One criticism of this method for dimensionality reduction is that measurement noise may push points sufficiently far from the embedded surface manifold, that the algorithm will not return meaningful results.

Independent and Principal Component Analysis

When no information is given as to the materials present in the pixels, then some form of blind unmixing is needed. Independent Component Analysis (ICA) is a blind unmixing algorithm that works to decompose the given data into statistically independent sources. Usually this separation into statistically int dependent sources is done by maximizing some measure of non-gaussianity or a minimization of mutual information. It has been investigated as a means of unmixing HSI data, see for example [26]. Stites [27] offers a good review of a number of papers that strive to perform spectral unmixing with ICA. Stites develops two algorithms to deal with the scale ambiguity that is inherently a part of ICA. He then demonstrates the suitability of using ICA as an unmixing algorithm for hyperspectral data.

Principal Components Analysis (PCA) is a dimension reduction method that strives to represent a data set along dimensions which capture the most variance [28]. Often PCA is used as a method to remove noise from measurements, by projecting the data onto the space spanned by the first k principal components. This implicitly assumes that the other dimensions are nuisance dimension, or noise in the data. A danger in doing this is that it may remove more than just noise. PCA has been utilized to improve HSI classification. See for example Radarmel and Shan's paper [29] and the cited works therein.

NMF

Non-negative matrix factorization (NMF) has been used to identify endmembers or materials and to segment an image into its constituents. NMF is the factorization of the data matrix into two matrices which are defined to be nonnegative, that is have all components or elements that are nonnegative.

Esser *et al.* use Non-negative Matrix Factorization to formulate the endmember type problem into a tractable algorithm. They use Alternating Direction Method of Multipliers for the numerical optimization (ADMM) [30]. Li *et al.* utilize a kernel NMF to unmix HSI data under an assumption of a nonlinear mixing model [21]. They also demonstrate the ability of two other constraints to improve performance. They test their algorithm on both synthetic and real hyperspectral datasets. Wang *et al.* [31] propose a new constraint on the

NMF problem that measure the dissimilarity between endmember spectra by calculating the difference of the gradients between the candidate endmembers. They provide experiments presenting the superiority of this compared to other NMF-based unmixing methods.

1.4 Goals of This Work

The main goal of this research is to investigate the ability of sparse representations to parallel the physics of hyperspectral data and provide new information to classification and detection algorithms. Specifically, this thesis will evaluate the ability of the Random-Forests classifier to distinguish between classes of materials based on their sparse signatures represented by dictionary learning.

1.5 Motivations for This Work

As has been discussed, many techniques exist that project HSI data into another basis or form to be able to more tractably process. Whether that is by dimension reduction using PCA or manifold unfolding utilizing ISOMAP, these techniques do not mirror, and sometimes occlude the physics of the initial system. It is hoped that the bases learned from dictionary learning algorithms and the sparse representations obtained will mimic the physical properties of the imaging system, providing improved insight and information.

1.6 Contributions of the Thesis

The main contributions of this research are first, the charaterization of the performance of RandomForests on abundance vectors for hyperspectral classification. Second, various matched filters were evaluated for their performance in material detection utilizing sparse representations based on dictionaries learned on hyperspectral pixels. Also, a novel sparsity based detection algorithm is described and evaluated with respect others surveyed. These characterizations demonstrate the efficacy in performing dictionary learning on hyperspectral information to form a sparse representation.

The remainder of the thesis is organized in the following manner. The next chapter details the Denali Material Survey dataset, the particular hyperspectral dataset that this thesis uses for its testing. Chapter 3 describes classification and describes the RandomForests classification algorithm, along with reporting results of the RandomForests algorithm on the Denali hyperspectral data. Chapter 4 introduces dictionary learning and describes some common learning algorithms. This is followed by Chapter 5, where we investigate the idea of pairing the sparse representations and dictionary learning algorithms of the previous chapter with the RandomForests classifier. Results are presented which describe the performance of classifying on the sparse representations. Next, in Chapter 6, the methodology of detection as a method for identification of a material is explained and a novel approach to detection is presented that utilizes dictionary learning and sparse representations. Results are presented demonstrating the performance of the detector along with common detectors applied to hyperspectral data. Finally, Chapter 7 formalizes our conclusions and lays out future directions of research.

Chapter 2
Denali Dataset

2.1 Hyperspectral Dataset

The research that we have done is based on hyperspectral data. Much of the tests and trials that have been performed are in relation to an HSI dataset that was provided by Lawrence Livermore National Laboratory (LLNL). The dataset was collected by a Long-wave Infrared (LWIR) Hyperspectral imaging device called Denali. The collection of cubes generated by the device was a long-term experiment executed for the purpose of examining the spectral characteristics of various materials over time. In this chapter, the Denali instrument will be described, the dataset will be explained, and some difficulties of the dataset will be demonstrated along with any techniques used to mitigate effects of these problems that were manifested in the data.

2.2 The Denali Instrument

The Denali instrument collects data in the LWIR region of the electromagnetic (EM) spectrum, ranging from $7.34\mu m$ to $13.34\mu m$. It is a dispersive hyperspectral instrument, which means that it collects its measurements utilizing some sort of a diffraction lens or grating. It measures 258 bands, producing band centers that have roughly $23nm$ of center separation [32].

2.3 Material Survey Experiment

The dataset that we are working with was provided to us by Lawrence Livermore National Laboratory. The experiment from which we have data (often called the Material Survey) was investigating changes in spectra of materials over time. The instrument was positioned in a mechanical room on the sixth floor of a building, and collected a hyperspectral image of a board with various materials on it located approximately 270 meters away. It had an approximate field of view (FOV) of 1.5 degrees in the horizontal direction. Denali was set up to take an image approximately every 10-11 minutes. The images were collected over a period of about one and a half months with a few more days a few months later [32].

2.3.1 Distribution of the Cubes

The data cubes were taken over 53 days from December 13, 2007 to March 20, 2008 in two runs. The first run spanned the time from December 13, 2007 through February 2, 2008. The second, shorter run took place from March 19, 2008 to March 20, 2008.There are 5743 data cubes total. The cubes are not quite all uniformly distributed across the days of the project. Figure 2.1 shows this distribution over the period of the experiment. In fig. 2.2, the distribution of the cubes over the hours of the day are given.

2.3.2 Hyperspectral Model Setup for Denali Material Survey

From eq. (1.4), the model can be simplified somewhat because of the particulars of our dataset. First, the upwelling term may be neglected ($L_u \approx 0$, because the sensor was positioned in such close proximity to the target, that there is assumed to be negligible contributions to the radiance. Likewise, the transmissivity of the that same atmosphere is assumed to be unity for the same reasons ($\tau \approx 1$). Making these assumptions allows us to use the radiance model given by equation

$$\mathbf{l} = \begin{bmatrix} l_{\lambda_1} \\ l_{\lambda_2} \\ \vdots \\ l_{\lambda_N} \end{bmatrix}, \quad \text{where} \quad l_\lambda = \epsilon_\lambda L_{e\lambda}(T) + (1 - \epsilon_\lambda) L_{d\lambda}. \tag{2.1}$$

This expression only accounts for pure pixel images. In order to use this for mixed pixel applications, we need to employ the Linear Mixing Model (LMM). In the LMM, we

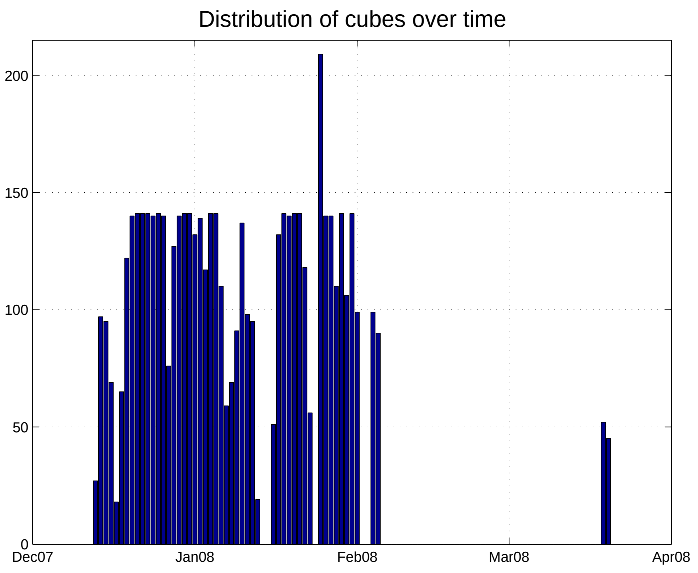


Fig. 2.1: The distribution of the datacubes collected across time. The two separate runs can be seen. On the days for which there were measurements, they are not *always* uniform; some days have a much larger number of measurements than others.

adjust the radiance from each material with the proportion with which it covers the pixel. As from eq. (1.7), we can model a full pixel as a linear combination of its constituent parts. That is

$$\mathbf{r} = L\boldsymbol{\alpha}, \qquad L = \begin{bmatrix} \mathbf{l}_1 & \mathbf{l}_2 & \dots & \mathbf{l}_k \end{bmatrix}, \qquad \boldsymbol{\alpha} \succcurlyeq 0, \qquad \sum_i \alpha_i = 1. \tag{2.2}$$

2.3.3 Brightness-Temperature as an Analogue to Emissivity

Much of the research surrounding HSI uses the emissivity of the material as the true material-specific signature. In the material survey validation paper, Lawson *et al.* perform their analysis on what they consider a psuedo-emissivity. They performed their usual calibrations to the raw data, and converted the radiance values to brightness-temperature measurements. Then a vegetated, green region of the scene was chosen to use as a flat field measurement over the central portion of the spectrum measured. This produced an emissivity of sorts: an emissivity with respect to the gray body of the vegetation, instead of with respect to a true blackbody. They demonstrated that many of the materials exhibited similar features to emissivity spectra measured in a laboratory setting [32].

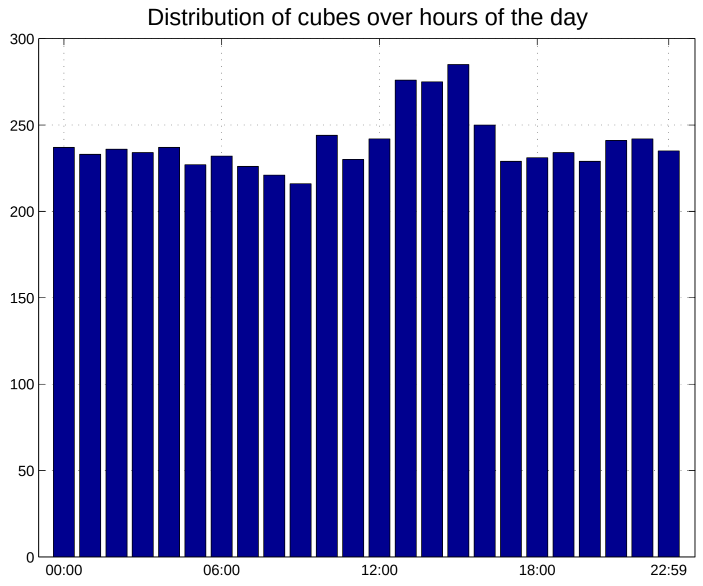


Fig. 2.2: The distribution of the datacubes as a function of time of day. They are approximately evenly distributed.

2.3.4 The Materials

The subject of the study was a board upon which were affixed a number of different samples of various materials. There were 27 materials laid out, and a couple of other "targets of opportunity" were also present. The materials were affixed to a piece of plywood/particle board that had been painted white. The materials were pieces 12 inches square.

The materials fell into roughly three main categories: natural minerals, metals, and man-made materials. The man-made materials included plastics, fabrics, and concrete/construction materials. A listing of the materials mounted on the board is given in Table 2.1.

Any numbering of the materials that appears in our research can be identified by numbering in Table 2.2 and in fig. 2.3. A color image is shown next to the diagram to provide a reference for the materials.

2.3.5 Spectra of the Materials

The measurements of the materials that we are working with is all in the LWIR region. Because of this, our measurements are dominated by emission as opposed to reflectance. This leads to most of the data having a very definite blackbody appearance. This is demon-

Table 2.1: Classification of the materials into the general categories represented. (*) indicates a material not numbered in fig. 2.3.

Rocks/Minerals	Metals	Plastics	Wood	Concrete/Asphalt/Clay	Fabrics
Slate	Painted Steel	PVC	Pine	Concrete	Camo Net
Marble	Bare Steel	HDPE	Painted Pine*	Cinderblock	Carbon Fabric
Granite	Brushed Aluminum	ABS		Asphalt Shingle	
Travertine	Polished Aluminum	Tyvex		Terra Cotta	
Alabaster		Blue Tarp		Aggregrate	
Aggregate					
Calcite					
Limestone					
Quarzite					
Soapstone					
Sand					

Table 2.2: Enumeration of the materials on the board.

First Row		Second Row		Third Row		Fourth Row	
Painted Steel	1	Cinderblock	2	Marble	3	Granite	4
Steel	5	Concrete	6	Trap	7	Travertine	8
Asphalt	9	Slate	10	Terra Cotta	11	Alabaster	12
PVC	13	HPDE	14	ABS	15	Aggregate	16
Tyvek	17	Pine	18	Aluminum	19	Calcite	20
Camo Net	21	Carbon	22	Polished Aluminum	23	Limestone	24
				Soapstone	25	Quartzite	26
						Sand	27

strated when looking at a histogram of the spectra of an entire cube together such as in fig. 2.4.

Each material also has spectral features that distinguish it. Examples of some materials are presented in fig. 2.5.

2.4 Difficulties of the Dataset

There are issues with registration from cube to cube. Each cube is completely independent and even within the dataset, there are issues that make registration difficult. At some points in the dataset, the camera is zoomed in or out so that the panel with the materials occupies more or less of the entire image. This means that each material is comprised of a different number of pixels from cube to cube. It also means that a straight index-based registration is not sufficient. A morphing or warping would be necessary to co-register the

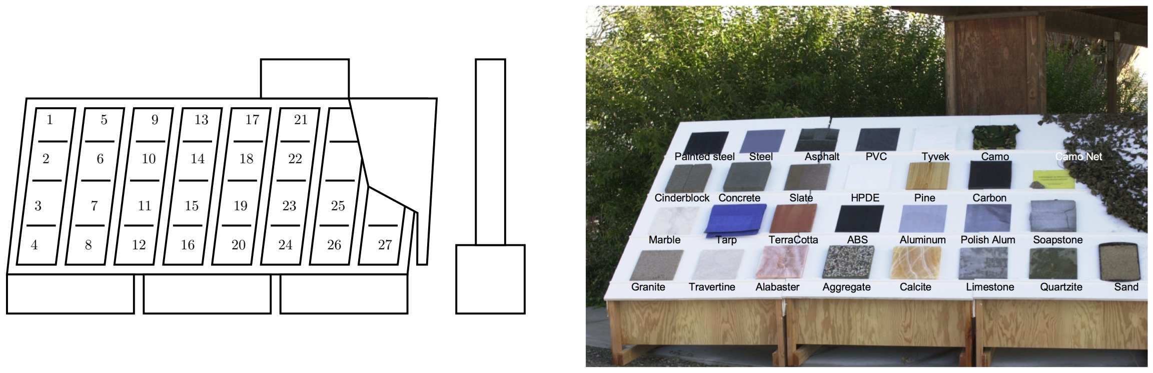


Fig. 2.3: The panel and the materials measured in the Material Survey Experiment. The left is a diagram specifying the numbering used throughout this research. The right is an optical photograph of the board.

entire dataset.

2.4.1 Translation

There are a number of times during the course of the study in which the Denali instrument was moved and even zoomed in or out. Consider the images in fig. 2.6. This inconsistency in the dataset makes registration across the entire dataset more difficult. This type of registration could be compensated for using state of the art registration algorithms, but they were beyond the scope of this study.

In order to be able to use multiple cubes, a time period was found consisting of slightly more than a week in which the Field of View (FOV) did not change significantly, thus enabling a coarse registration from cube to cube.

2.4.2 Warping

Another issue that complicated the co-registration of the Denali Material Survey cubes was an occasional warping of the panel materials' size. In fig. 2.7, the non-uniformity of the size of the material pieces on the panel (especially on the left side) can clearly be seen. The cubes shown are in consecutive order.

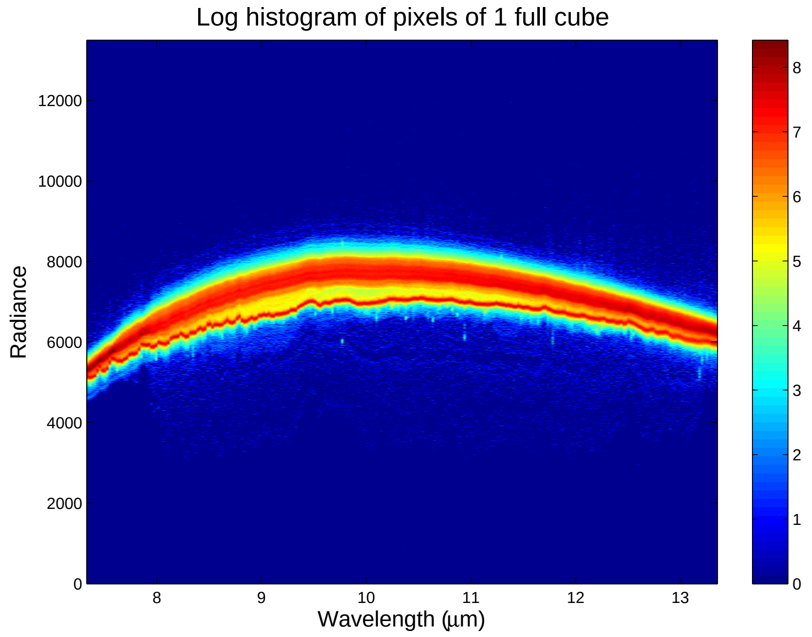


Fig. 2.4: Histogram of the spectral content of an entire hyperspectral cube. The underlying blackbody curve is very evident.

2.4.3 Striping

Hyperspectral data often have artifacts that manifest as stripes and bad pixels. Many different methods are proposed to correct or minimize the effects of these bad pixels and other artifacts (see for example, Leathers and Downes [33], Manolakis *et al.* [34], Fischer *et al.* [35], and Kieffer [36]). In the Denali dataset, often a whole row can be seen that is significantly different than those around it. See, for example, fig. 2.8. In order to mitigate some of the effects of these stripes, when choosing the data from a material, a median filter was used to be able to avoid being influenced too much by erroneous and spurious data. Figure 2.9 shows an example of utilizing the median filter to correct for a “striped” row.

2.4.4 Edge Feathering

The Material Survey dataset that was provided had a interesting phenomena in that the columns of the image did not always line up together correctly. This produced an irregular boundary for each of the materials in the scene and distorted horizontal lines in the images. An example of this “edge feathering” can be seen in fig. 2.10.

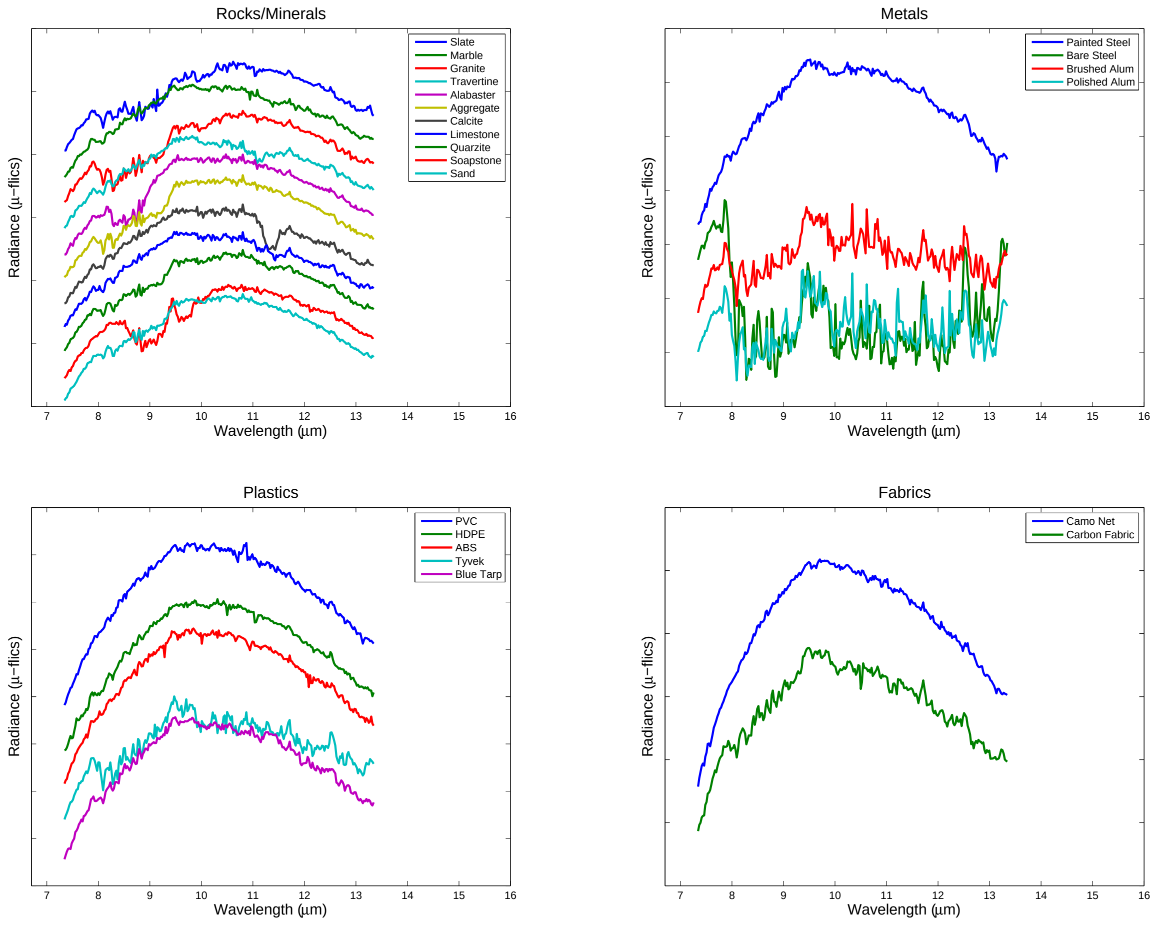


Fig. 2.5: A sampling of the spectrum of various materials from the Denali Material Survey dataset. The y-axis is radiance with an offset to allow for viewing of the spectral features of the materials. These spectra were all generated from the same cube and were median filtered.

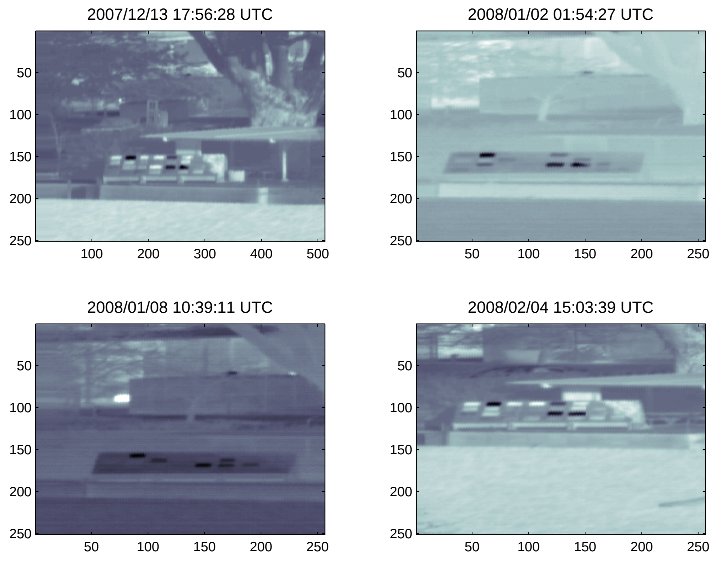


Fig. 2.6: Example of translation and zooming variations across the dataset.

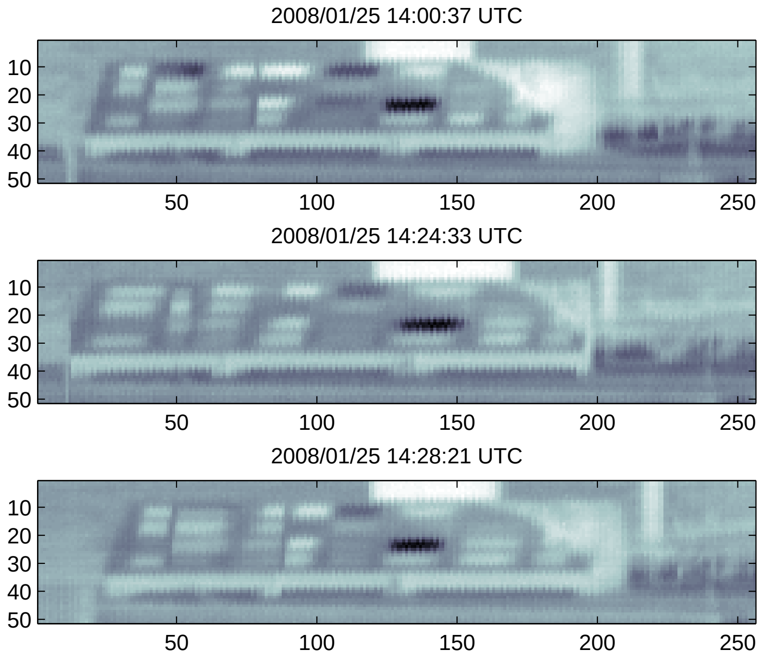


Fig. 2.7: Example of the amount of warping that is present in various portions of the Material Survey dataset.

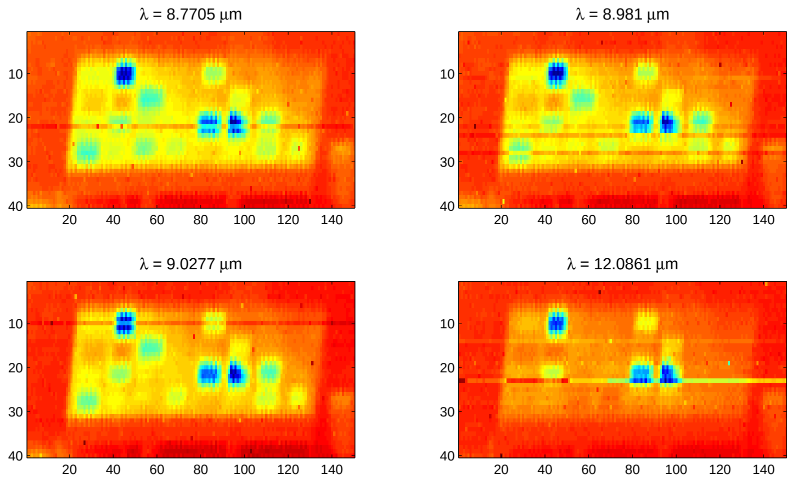


Fig. 2.8: Some examples of striping found within a single cube at various wavelengths.

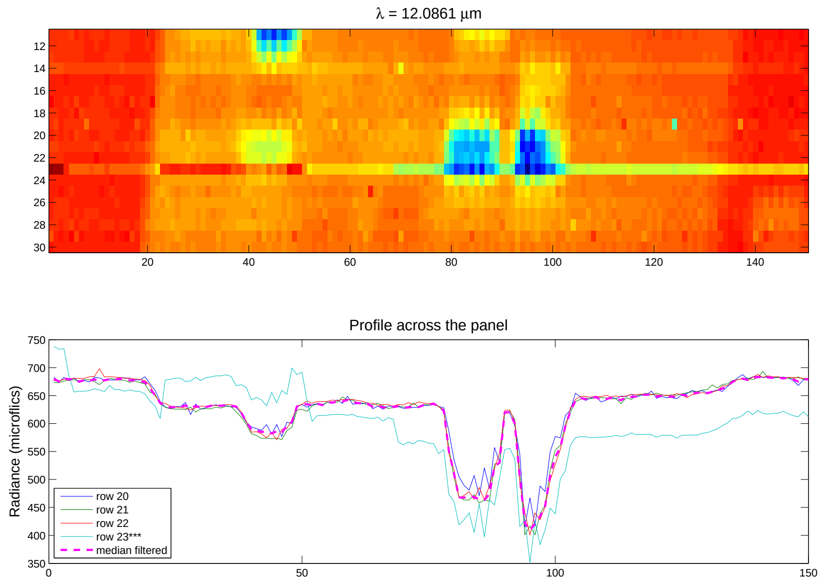


Fig. 2.9: Utilizing a median filter to beat down a noisy row in the the materials of interest.

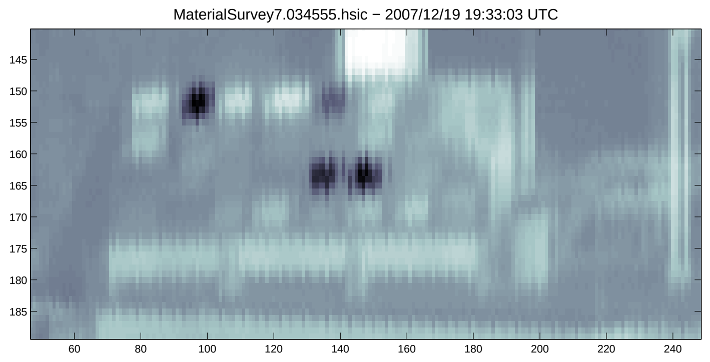


Fig. 2.10: In this pseudo-broadband image of the panel, the edge feather errors can be seen especially well in the darker materials.

Some work was done to correct this with a few different possible methods. Figure 2.11 shows some results. The algorithms proved to not be quite reliable enough to apply across the dataset as a whole, so the solution to this problem that was incorporated into most of the experiments was to sample from the middle, inner portion of the materials, i.e. staying away from the edges.

Lawrence Livermore have informed us that this is no longer an issue with new data obtained from this sensor.

2.5 The 9-Days Dataset

In order to address some of the issues of registration, a subset of the data was identified in which the zoom was approximately the same. This comprised approximately 9 days from December 18, 2007 to December 26, 2007.

The panels were extracted from the full cubes by doing a simple cross correlation registration. These panels were then saved as MATLAB .mat files.

Extra Registration

When extracting the materials from the panels, an extra registration was employed to ensure that the material of interest was as pure as possible. An area was defined around the predetermined most likely center for each material. This area was then shifted around with a distance penalty to find a local registration that maximizes the purity of the patch.

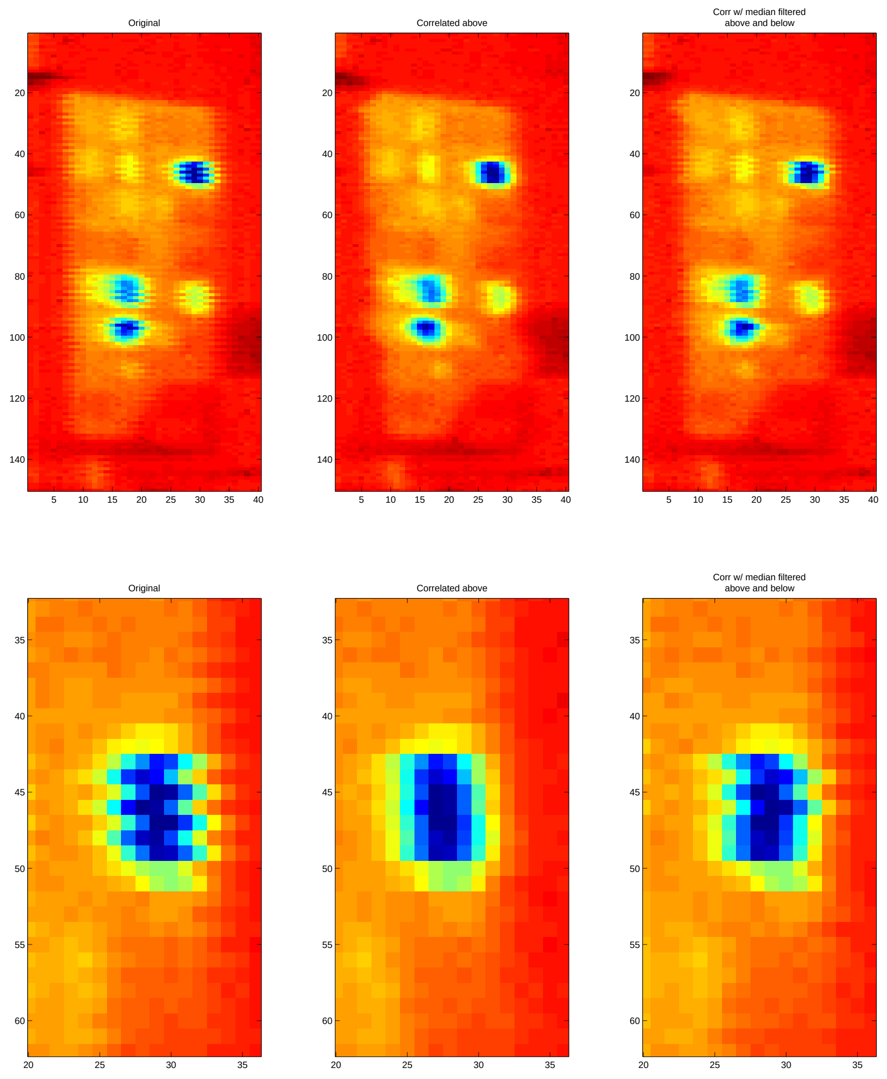


Fig. 2.11: Examples of methods to mitigate the edge feathering errors. The top left shows the original image and the lower left a zoomed in area. The middle shows the results of a correlation alignment calculated line by line with the line above it. The right does a correlation above and below line by line and also incorporates a median filtering process.

Even Smaller Dataset

There appeared to be too much variation, when running tests on this smaller subset of the data, and thus a smaller hand-picked set of material spectra were chosen to provide a more cohesive and consistent dataset on which to do the tests. The manner in which this set was built was that the material spectra were extracted from all of the 9-days dataset and then plotted on top of each other in histogram fashion, quantizing, and binning in the radiance direction. Then a representative spectrum was chosen and the N closest spectra to this example were ranked accordingly. This was done independently for each material, and thus the spectra were chosen from many different cubes, but there was guaranteed some self consistency in each material. The cubes with the highest ranking were chosen as the new dataset.

Artificial Datasets

Sometimes a ground truth set is extremely helpful. For some experiments, a dataset was created which is built upon the Linear Mixing Model (LMM). Given a pool of data as a library, a pixel can be produced as a linear combination of pixels from the library. In our tests, we used the median filtered material spectra from the 9 days and used them to create a dataset. This was used to test for a material of interest. It allowed for specification of abundance and scene coverage along with a few other parameters. This artificial data was used in testing the matched filter detections derived.

The median filtered data from the small, 9 day dataset was used as a dictionary to build hyperspectral pixels from. The data were labeled by material. Each dataset built in this fashion features a particular material of interest. This allows for testing for detection of this material in varying abundance percentages and in varying scene coverage amounts.

Chapter 3
Classification

3.1 Classifiers

Classification is the “systematic arrangement in groups or categories according to established criteria” [37]. In the manner referred to in this study, it encompasses both the act of determining the criteria by which the classification should happen and also the act of assigning data into classes utilizing the criteria found. The set of criteria which divide the data in the different classes, along with any mechanism needed to do the classifying, is termed the classifier. In the statistical machine learning literature the type of classification that is treated in this thesis is commonly referred to as supervised learning, because it incorporates a “training” set with which to develop the classifier. Commonly a “test” set is then used to evaluate the effectiveness of the classifier on new data. Both the training and the test sets usually have known labels/classes associated with them, in order to facilitate the determination of the error of the classifier.

3.2 Types of Classifiers

3.2.1 Binary and Multiclass Classifiers

A classical binary classification problem is a two-class problem. There are only two outcomes possible. They may be labeled as 0 or 1, or 1 and -1, or 1 and 2, and sometimes the label has to do with the classification algorithm. The binary problem has been studied extensively and is well understood. Multiclass classifiers are classifiers which distinguish between more than two different classes. Some multiclass classifiers are generalizations of binary classifiers (*i.e.* support vector machines [38]). Some methods have also been proposed to utilize binary classifiers in a coded fashion to create a multiclass classifier [39,40].

3.2.2 Linear Classifiers

Linear classifiers try to determine some linear boundary that divides the data into the separate classes with some minimum measure of error. These methods include Logistic Regression, Linear Discriminant Analysis (LDA), and perceptrons. Support Vector Machines are frequently listed as linear classifiers also, although they can be made to learn a nonlinear decision boundary using various different kernels [28].

Quadratic Discriminant Analysis (QDA) isn't technically a linear classifier, but is a direct extension of the ideas of LDA.

3.2.3 Probabilistic Models

There exist many different types of probabilistic classifiers that have different approaches to modeling the data and finding a classification. The methods range from nearest neighbor methods and naive Bayes models to maximum likelihood and modern Bayesian methods. Probabilistic models can be fit to the data, possibly incorporating any prior information to assist in the classification. Although these methods have found great success in the machine learning community, we focus on tree-based methods in this research because this does not make assumptions about an underlying model.

3.2.4 Tree-Based Methods

Tree-based classification methods are essentially a subspace partitioning scheme. A tree will define a split at each branching and assign a direction to each datum arriving at that split. At the leaves or ends of the branches, some prediction is formed based on the data that arrived at the terminal node. Candidates for the prediction could be a majority voting scheme or a histogram approximation of a probability function.

3.3 RandomForests

RandomForests is an ensemble machine learning algorithm first proposed by Leo Breiman in 1996 [41]. It is a decision tree-based algorithm which builds a collection, or forest, of trees and then aggregates the results of the individual trees into a full prediction. It natively and seamlessly can accommodate multiclass classification problems. It builds its classifying space from the subspace partitioning of the underlying classifiers (the decision trees that form the forest).

3.3.1 Tree Concepts

Background

A classification tree is an acyclic graph that is organized in a hierarchical manner. As such it is composed of nodes and edges. Edges are the connections between the nodes in the graph. Most common classification trees use a binary tree structure, that is, each node in the tree can only have 0 or 2 children. A node that has no children is called a "leaf" node. Any other node in a decision or classifying tree is considered a "splitting" node, because a decision or split is made at these points.

In decision trees there are two main modes of operation, a "training" phase and a "testing" phase. During the training phase, the tree structure is built to perform the classification according to the configuration setup of the forest and the training set given. During the testing phase, new data are entered into the tree and the projected response or classification is given as an output from the tree [42].

Building a Decision Tree

The framework for building a tree is that all of the training data are passed down the tree and split apart at each node, with portions of the the data going down each child branch to be split at the next nodes. The splitting is done according to some set criteria. Examples of these criteria could be minimization of a classification or regression error, maximization of a purity measure, or maximization of entropy. As the data filters down the tree, the error continues to become better, until some stopping criterion is met, such as a class purity threshold or a set error threshold, or a depth restriction.

It has been shown that tree-based models can overfit on the training data, losing the ability to predict the outcome or response from new test data. Many methods have been proposed to reduce overfitting including pruning and refactoring [28]. Another method to reduce or eliminate overfitting is to use an ensemble approach.

Ensemble Methods

An ensemble is a group of classifiers which are aggregated together to produce a result that is often more accurate than any of the single classifiers by themselves. Frequently a simple, fast classifier is used as the basis for the ensemble. Examples are logistic regression models, or classification/decision trees. Sometimes different and varied classifiers are put together, but it is more common to only include a single type of classifier with some variation. These variations can be obtained using techniques such as bagging, which introduces variation in the datasets or randomization inside the classifiers themselves.

3.3.2 The RF Algorithm

Breiman proposed RandomForests as an ensemble tree-based method that was able to be efficiently grown, yet also outperforms single decision tree-based classifiers. The algorithm is presented below.

Making RF Random

In order to produce a number of trees that are different, random sampling is incorporated in two important places in the building of the trees. The first place is bootstrap aggregating, commonly called bagging. For each tree that needs to be built, a uniform random sampling with replacement of the training set is generated and this becomes the dataset with which the tree is built. Breiman states that on average about one third of the samples are left out of each tree [41].

Another way in which RandomForests randomness is in a random selection of the candidate split variables. RF searches over a randomly chosen subset of the variables to find the best split. This subset is chosen independently at each split node. The number of variables to randomly choose is the parameter *mtry*.

Final Prediction

RandomForests produces its final prediction by aggregating the results from each individual tree and doing a majority vote. The class with the largest group of trees that predicted it, is the prediction that is given from the forest as a whole. This is not the only method of forming a prediction from a forest, but this is the default behavior and the method used in this paper.

3.4 Classification of HSI Pixels

3.4.1 Previous Work

Applying classifiers to hyperspectral data pixels and cubes is a broad and open area of research. A number of papers evaluated and published their findings of applying numerous signal processing techniques and various classifiers to hyperspectral pixels.

Many varied and different techniques have been applied to hyperspectral pixel classification. Bidhendi *et al.* uses fuzzy clustering to classify pixels [43], and Kuo *et al.* evaluate the usefulness of both a K-nearest neighbors and a Gaussian classifier approach [44]. Another interesting approach is the application of Kalman filtering on a single HSI pixel to classify it. Wang *et al.* develop a couple different Kalman Filter-based methods and compare their performance with other Kalman filtering based techniques [45].

Du and Chakrarvarty [46] introduces an unsupervised classifier based on the idea of a blind source separation problem. They used a mutual information and orthogonality encouraging cost function and incorporated a Neyman-Pearson step to estimate the number of classes, thereby reducing the complexity of the algorithm. They showed preliminary results images demonstrating the feasibility of the method.

Support Vector Machines

A number of people have applied Support Vector Machines (SVMs) to the hyperspectral classification problem. SVMs are essentially a margin maximization classifier; they try to maximize a measure of the margin between two classes. Through use of the "kernel trick" it can be used to find nonlinear boundaries between the classes.

Melgani and Bruzzone [47] compare using an SVM for classification of hyperspectral pixels against both a K-nearest neighbors and a radial basis function neural network classifier. The SVM performs favorably in classifying the AVIRIS data. They also report multiclass SVM performance in a 1 versus 1 and a 1 versus all framework along with hierarchical frameworks.

Lennon *et al.* uses an SVM to do classification and compares its performance to a spectral angle mapping classifier and a Gaussian maximum likelihood classifier [48]. It is shown that a prefiltering of the data with a Maximum Noise Fraction transform and an anisotropic filter dramatically improves results in homogeneous areas.

Tarabalka *et al.* [49] use a 1-vs-1 SVM strategy, along with a Markov Random Field (MRF) for incorporating spatial information, to classify hyperspectral pixels. They showed favorable results in the few variations that they tested on AVIRIS and ROSIS data. They attribute their gains to the use of neighboring pixels' class information. Fauvel *et al.* [50] also incorporates spatial information using a self-complementary area filter and a derived two-kernel SVM strategy. They demonstrated gains above that of using the SVM on spectral information only.

Demir and Ertürk propose using Relevance Vector Machines (RVMs) in the place of SVMs for classification of hyperspectral pixels [51]. They demonstrate that RVMs offer approximately the same classifying power as the SVMs with a significant drop in the number of significant vectors (relevance and support vectors). This allows for testing times to be dramatically faster.

RandomForests

Both Ham *et al.* [52] and Joelsson *et al.* [53] use RF to perform classification on hyperspectral pixels. Ham *et al.* performs his experiments using data collected from the AVIRIS visual and near infrared sensor, whereas Joelsson *et al.* uses data from the ROSIS instrument, a mid-wavelength infrared hyperspectral camera. They both present a comparison between a couple of different RF implementations. They evaluate the general CART framework, which is close to what we use, and also a RF that uses Binary Hierarchical Classifier (BHC) as the underlying tree structure. Joelsson *et al.* discusses the classification error based on the ROSIS data, with both ensembles demonstrating classification accuracies generally within the 90% range. They do not show any comparison with any other classifiers or data. Ham *et al.* does the same comparison and in addition compares against a best-basis BHC and a random subspace BHC. In both cases, RF CART performs favorably, although the BHC RF performs even better in certain circumstances. Joelsson *et al.* concludes that the RF BHC implementation is a good classifier and performs very well on certain problems, where as the CART implementation is a good classifier for any general problem and shines when the problem does not lend itself to using the BHC implementation.

3.4.2 Classifying on Denali Data

In order to have a base line to which we can compare, RF was run on the original pixels. The results are shown in Table 3.1.

Method of Choosing Data

The Denali dataset covers a significant time period and numerous variations of environmental factors. We use a method to select some cubes from the full set that have the most consistent representation across all materials and with respect to the average of the dataset.

In order to identify the most consistent cubes, a histogram of all the cubes was constructed for each material, using all nine spatial samples available. Then the maximum radiance bins were identified and the distance from this “best” sample was taken for each cube. The cubes with the smallest distance were used as the dataset for this experiment. Half were set aside as test cubes, and half as the training set.

RF Test

From each selected cube, the materials were loaded and selected from the set center of the material and the eight adjoining pixels. This gave nine sample spectra for each material from each of the cubes for 90 samples of each of the 27 materials overall. The total size of the training set was 2430. The test set was the same size as training set.

RandomForests was run on both the radiance and brightness-temperature data. Also taken into consideration is the scaling of the data. In order for the SPAMS libraries to take advantage of the data, scaling is often required. To accurately portray the effect of this on the RF output, RF output error from a few variants of this scaling are included.

Radiance Data Variations

For the radiance data, RandomForests was run on the radiance data itself, with no pre-scaling. Then a few different methods of scaling the data were considered. The first variation considered was the pixel-wise mean-removed radiance. Each pixel mean was calculated and this mean was removed from the pixel, producing a zero mean pixel. The second is where the training set and the test set are normalized independently by the Frobenius norm of each dataset divided by the size of that set. This gives an approximate unit norm for each pixel with respect to the dataset to which it belongs. The third is where the test set is scaled by the normalizing constant of the training set, where the normalizing constant is calculated as in the second method.

Brightness-Temperature Variations

In addition to testing with radiance, we also tried testing with brightness temperature converted pixels. We tested with the full brightness temperature, and also with mean removed brightness temperature pixels. In addition to these we also tried the normalizations as done with the radiance data, in that we tested on BT pixels with each set (training and testing) being normalized to average unit norm. And finally, we also tested with both sets being normalized by the training set normalization factor.

Error Results

The results were averaged over twenty runs for each variation of the data described above. RandomForests on the raw radiance data produced an average error of 4.4% for just about every method except for the two which had a zero mean. For the two tests which were zero mean, the average error was between 3% and 4%. Interestingly, the out-of-bag estimates of the error for each of these were significantly less than the other test sets. The results can be seen in the fig. 3.1 and tabulated in Table 3.1.

From these results we see that scaling the datasets independently produces datasets that are significantly different. We can also see that even when using the best scaling, the results of using the scaling does not significantly improve the error over the raw radiance vectors.

3.4.3 Confusion Matrices

The confusion matrix for the RandomForests algorithm on the original raw hyperspectral data is shown in fig. 3.2. The confusion matrices are a method to determine the performance of a classifier. The columns correspond to each true class. They are stochastic, that is, they sum to unity. The rows correspond to the classification assigned to each data vector by the RandomForests classifier. Thus, cell $c_{i,j}$ of the confustion matrix specifies the percentage of class j that was classified as class i.

The diagonal down the middle of the confusion matrix are the correct classifications. The total error can be found by counting the number of test vecotrs that are off the diagonal.

Table 3.1: The tabulated results of running RandomForests on a training set of 2430 samples and a test set of 2430 samples. See text for descriptions of each of the datasets.

	OOB			Test		
Data	Min Error	Mean Error	Max Error	Min Error	Mean Error	Max Error
Raw	3.0453	3.3642	3.7037	3.9918	4.4033	4.7325
Raw samp-wise 0-mean	1.0700	1.2613	1.4403	2.7984	3.1379	3.4568
Independent Norm	3.0453	3.3868	3.6626	4.2387	4.4753	4.9383
Scaled "Norm"	3.0041	3.3395	3.6214	4.0741	4.4362	4.8148
B-T of raw	3.1276	3.3868	3.7037	4.1564	4.4486	4.6914
B-T samp-wise 0-mean	1.5226	1.8004	2.0576	3.6214	3.9794	4.3210
B-T less Train Mean	3.0041	3.3601	3.6626	3.9506	4.3601	4.6091
B-T less Test Mean	3.0041	3.3601	3.6626	3.9506	4.3909	4.7325

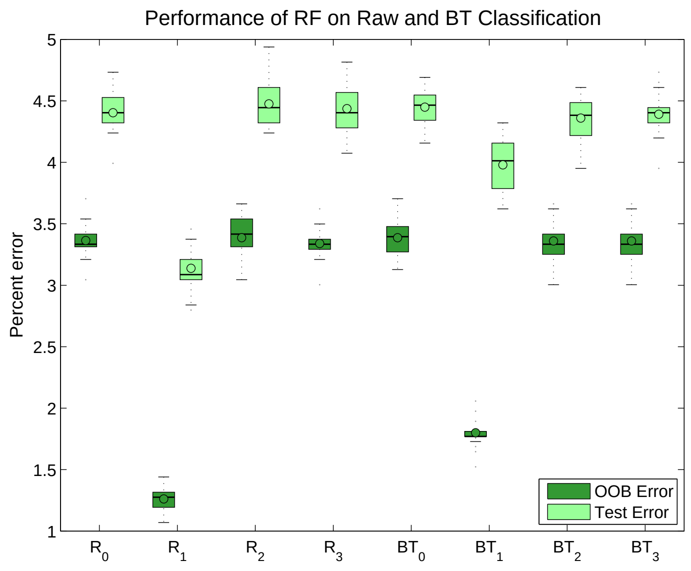
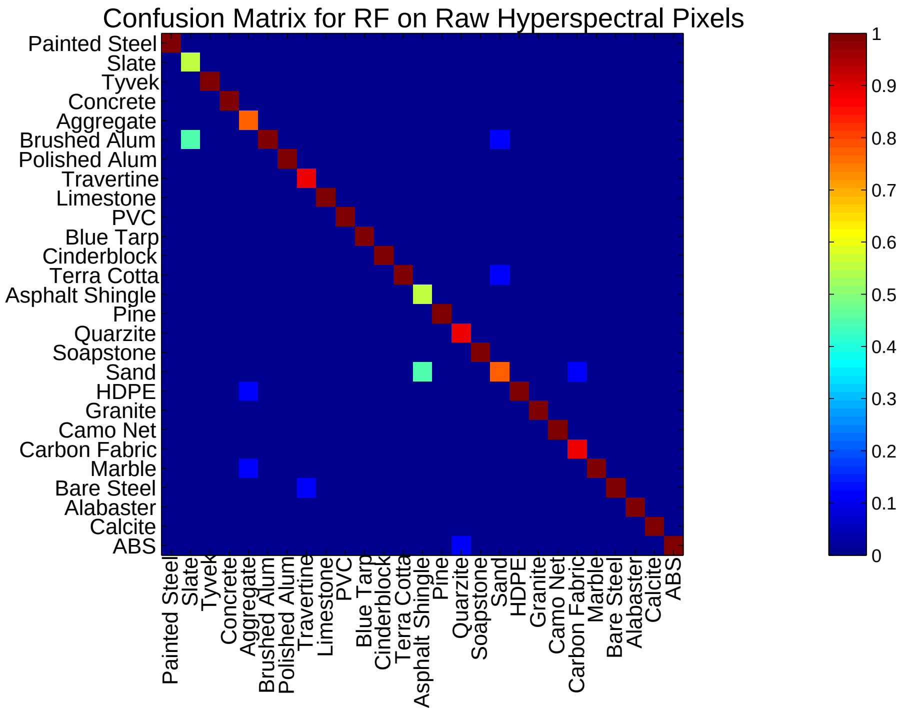


Fig. 3.1: This shows the results of running RF twenty times on each of the following data: R_0) the Raw HSI pixels, R_1) Raw sample-wise 0-mean, R_2) Independent Norm, R_3) Scaled "norm", BT_0) BT of the raw, BT_1) BT with a sample-wise 0-mean, BT_2) BT with the Training set mean removed, BT_3) BT with the Test set Mean removed.



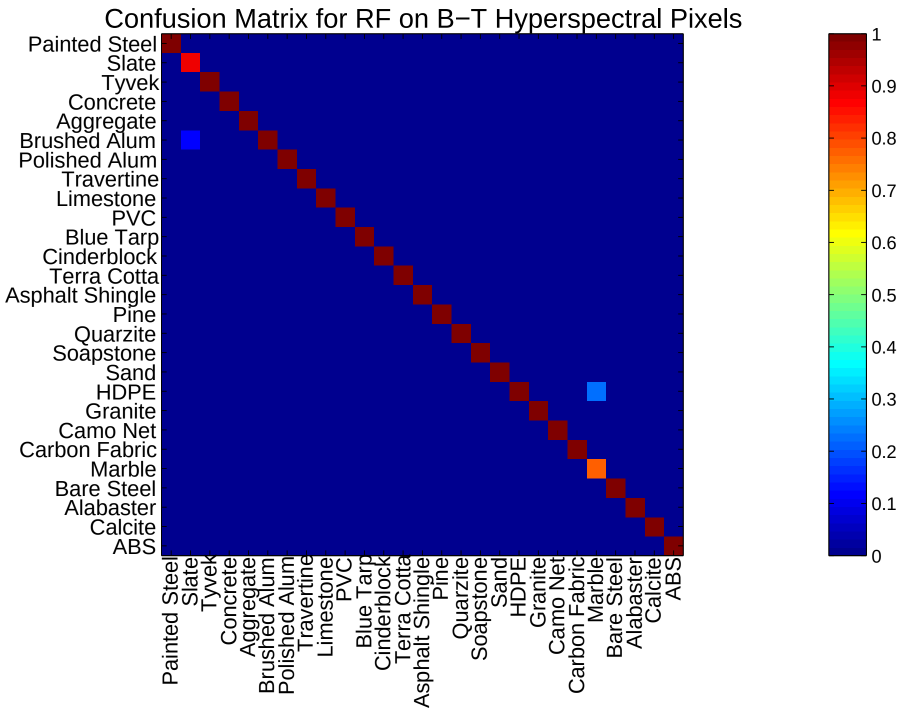


Fig. 3.2: Confusion matrices of a few of the testing and training sets described in the text.

Chapter 4
Dictionary Learning

Many signal processing algorithms focus on representing data in a different basis. That is, projecting a data vector into another dataspace that may be more informative. Examples of this are Fourier representations, wavelet transformations, Graham-Schmidt orthoganalization, and principle components analysis. Each of these processes project signals onto (or creates) a basis which represents the data in a different fashion. These alternate bases can provide a valuable way to look at the data and new insights can be made. Some data can be represented as a mixture of a small set of causes. Of numerous different possibilities, each datum is only a combination of very few sources. This situation is what sparked interest in Dictionary Learning. Dictionary Learning (DL) strives to represent data in a different basis that admits a small number of nonzero coefficients. That is, the coefficient vectors are sparse, which follows the model better that other common methods of decomposition that use all or most of the available sources or basis vectors.

4.1 Origins

Dictionary learning is the learning of a basis for which the given data can be represented with low error and that produces sparse coefficient vectors. The sparsity constraint enforces the idea that only a few basis elements contribute in representing any one datum.

This technique is often posed in the form of an optimization problem. Frequently, it is presented as

$$\left[\hat{D},\ \{\boldsymbol{\alpha}_i\}\right] = \arg \min_{D,\,\{\boldsymbol{\alpha}_i\}} \sum_i \|\boldsymbol{\alpha}_i\|_0 \quad \text{subject to} \quad \|\mathbf{x}_i - D\boldsymbol{\alpha}_i\|_2^2 < \lambda \ \forall \ i, \tag{4.1}$$

or some variation of this. Essentially, this finds the basis or "dictionary" $\hat{D}$ that minimizes the number of components in each coefficient vector $\boldsymbol{\alpha}_i$, subject to a bound on the representation error. The use of the 0-pseudo-norm in the problem makes this NP-hard [54]. To attempt to solve this, numerous methods have been proposed that will find an approximation to the solution. Greedy methods, relaxations, matrix factorizations, and gradient descent methods are just a few of the types of algorithms that have been developed by the community. Most of these break the problem into a 2-step iterative process: fixing the dictionary and solving for the abundance coefficients, and then fixing the coefficients and solving for the dictionary by an update that converges to a viable solution.

The general DL problem is often relaxed to use a 1-norm instead of the 0-norm, because the 1-norm is the convex hull of the 0-norm [54]. From this, an alternating algorithm can be used by fixing each of the dictionary and the abundances in turn while optimizing the other.

4.2 Methods of Doing the Learning

In this section a few different algorithms for performing dictionary learning are introduced. Each performs the dictionary learning differently than the others and seeks to improve on the others in distinctive ways.

4.2.1 Olshausen and Field

Olshausen was one of the first to really explore this field. His method was drawn from a Bayesian framework, in that he proposed the problem in such a way that it became an issue of drawing from a difficult distribution [55].

In this framework, it was shown that by maximizing the average log-likelihood of a draw from this distribution, using a sparse and statistically independent prior, a set of basis functions is found that can be used to sparsely reconstruct the given images. They worked to solve

$$\phi^* = \arg\min_{\phi} \left\langle \min_{a} E\left(I, a|\phi\right) \right\rangle, \tag{4.2}$$

where $E(I,a|\phi)$, is the log-likelihood re-cast into an energy framework. The definition of E is given as

$$\begin{aligned} E\left(I,a|\phi\right) &= -\log P\left(I|a,\phi\right)P\left(a\right) \\ &= \sum_{\mathbf{x}}\left[I\left(\mathbf{x}\right)-\sum_i a_i\phi_i\left(\mathbf{x}\right)\right]^2+\lambda\sum_i S\left(a_i\right). \end{aligned} \tag{4.3}$$

Olshausen and Field then describe the solving of this by first holding the basis functions ϕ_i fixed and solving for the a_i's by finding the equilibrium solution to the differential equation

$$\dot{a}=\sum_{\mathbf{x}}\phi_i\left(\mathbf{x}\right)r\left(\mathbf{x}\right)-\lambda S'\left(a_i\right), \tag{4.4}$$

where $r\left(\mathbf{x}\right)$ denotes the residual image. Then the basis is solved for while holding the abundance coefficients fixed. Olshausen and Field implement this algorithm in a neural network, using Hebbian updates of the ϕ_i, and using the network structure to calculate the a_i.

4.2.2 K-SVD by Aharon *et al.*

K-SVD, a popular DL algorithm, was proposed by Aharon *et al.* [56]. It is a generalization of the K-means algorithm. It works by updating the dictionary by dropping out the atom being adjusted and using the rest of the dictionary to calculate an update for the current atom.

The K-SVD is a generalization of the K-means algorithm. Where the K-means algorithm limited the columns of the abundance matrix to being from the trivial basis (*i.e.*, $\mathbf{a}_i=\mathbf{e}_k$), the K-SVD algorithm allows for more than one element to be non-zero. Essentially looking for the solution to the optimization problem

$$\min_{D,X}\left\{\|Y-DX\|_F^2\right\} \qquad \text{subject to} \qquad \forall i, \|\mathbf{x}_i\|_0\le T_0, \tag{4.5}$$

where T_0 is some limit imposed on the number of non-zero elements allowed in each abundance vector $\mathbf{x}_i$. The K-SVD algorithm solves the optimization problem by first assuming the dictionary D is set and performing a sparse coding step to update the abundances X. Specifically, any pursuit algorithm can be chosen to solve for this portion, such as matching pursuit (MP), orthogonal matching pursuit (OMP), basis pursuit (BP), or the focal underdetermined system solver (FOCUSS). Then for a fixed abundance matrix, the dictionary is updated by decomposing the product into a summation of rank-one matrices,

$$\|Y - DX\|_F^2 = \|Y - \sum_{j=1}^{K} \mathbf{d}_j \mathbf{x}_T^j\|_F^2. \tag{4.6}$$

$\mathbf{x}_T^j$ denotes the kth *row* of X, that is the kth column of D is multiplied by the kth row of X, forming a rank-1 matrix. This decomposition allows for defining an error matrix E that is a summation of all but one of the columns of D,

$$E_k = Y - \sum_{j \neq k} \mathbf{d}_j \mathbf{x}_T^j, \tag{4.7}$$

leaving us to minimize the problem of

$$\|E_k - \mathbf{d}_k \mathbf{x}_T^k\|. \tag{4.8}$$

Now by using the singular value decomposition (SVD) the closest rank-1 matrix can be computed, but some careful selection of the portions of E_k and $\mathbf{x}_T^k$ are necessary to enforce the sparsity constraint over the support of $\mathbf{x}_T^k$. This careful handling allows for updating of both the dictionary atom (column) and the abundance vector simultaneously. This update is performed for each dictionary atom and then the process is repeated, starting with another sparse coding step. Even though the K-SVD updates both the dictionary and the abundances, the sparse coding step is still necessary to allow for the support to change after the dictionary is updated.

4.2.3 Online DL by Mairal *et al.*

Mairal *et al.* propose an online dictionary learning algorithm that can efficiently work with very large (possibly infinite) datasets, capitalizing on a treatment of a training set as stochastic draws [57]. They also use an alternating algorithm to solve the dictionary learning problem. The dictionary update portion ends up solving the following problem,

$$\begin{aligned} D_t &= \arg\min_{\mathbf{D}\in C} \frac{1}{t}\sum_{i=1}^{t}\left(\frac{1}{2}\|\mathbf{x}_i - \mathbf{D}\boldsymbol{\alpha}_i\|_2^2 + \lambda\|\boldsymbol{\alpha}_i\|_1\right) \\ &= \arg\min_{\mathbf{D}\in C} \frac{1}{t}\left(\frac{1}{2}\operatorname{Tr}\left(\mathbf{D}^T\mathbf{D}\mathbf{A}_t\right) - \operatorname{Tr}\left(\mathbf{D}^T\mathbf{B}_t\right)\right), \end{aligned} \tag{4.9}$$

where

$$\begin{aligned} \mathbf{A}_t &= \mathbf{A}_{t-1} + \boldsymbol{\alpha}_t\boldsymbol{\alpha}_t^T \\ \mathbf{B}_t &= \mathbf{B}_{t-1} + \mathbf{x}_t\boldsymbol{\alpha}_t^T. \end{aligned} \tag{4.10}$$

This algorithm specifically works to update the dictionary in an "online" manner. That is, it works to update the dictionary sequentially on new training data at each iteration, treating each new datum as a draw from a probability distribution. Mairal *et al.* demonstrate that this online method is very effective with very large training sets, as in image and video processing applications. They also suggest a mini-batch improvement that increases convergence speed.

The algorithm of Mairal *et al.* is the basis for the libraries that we used in the majority of our experiments. More of this particular set of libraries is described in section 4.4.

4.3 DL on HSI

A number of researchers have applied sparsity inducing decompositions to HSI data and have had varied success. One of the powerful ideas of applying these techniques to HSI is that it follows the actual physical situation of the imaging system while also providing a dimension reduction of sorts. The following are highlights of various papers that applied dictionary learning techniques to HSI and a brief synopsis of their results.

Charles *et al.* [58] look at a gradient descent algorithm and focus on how well a sparse dictionary does at representing hyperspectral content. In a follow-on journal paper [59], they also consider the possibility of reconstructing HSI from multispectral measurements. They show that dictionaries are well suited for representing HSI data.

Gillis and Plemmons add in a sparsity-inducing term into the nonnegative matrix underapproximations minimization problem [60]. They show that the use of this term contributes to increased performance with respect to identifying and separating endmembers.

Möller *et al.* first present the problem as $X \in \mathcal{R}^{m,d}$, d is the number of pixels, m is the number of spectral bands, and look for a solution that is of the form

$$X \approx AS,$$

where A is the endmember dictionary, and S captures the abundance information [61]. In order to make the results more *physically meaningful*, the endmember dictionary is chosen out of the data itself, similar to the endmember idea used in the hyperspectral community. They use a combination of clustering and nonnegative matrix factorization techniques to solve

$$\min_{\substack{A \geq 0, S \geq 0 \\ \left\|A_j - \tilde{A}_j\right\|_2 < r_g}} \frac{1}{2} \left\|AS - X\right\|_F^2 + < R_w \sigma, S >,$$

and column normalization of A after each iteration. r_g is the radius (or diameter) around the j^{th} cluster around $\tilde{A}_j$, and the last term is used to induce sparsity into the S matrix.

Iordache *et al.* illustrate the usefulness of using a sparse representation and a dictionary to represent hyperspectral data [62]. They use a known spectral library from the U.S. Geological Survey (USGS) as the dictionary and perform sparse coding to identify which materials are selected from the library. They specifically call attention to some shortcomings, such as the (likely) possibility that the lab-measured library and the field-measured data could be very different. They explored using these techniques on both simulated and real data. The real data that they used was previously atmospherically corrected, but still needed some calibration on their part in order to use in their experiments. They report encouraging results, but are cautious to express that many aspects are still to be explored.

Xing *et al.* use dictionary learning for the purpose of hyperspectral cube reconstruction [63]. This is essentially the compressive sensing problem. They show very good results for reconstruction and demonstrate the efficacy of using dictionary learning for this purpose. They are not concerned with the discrimination of the pixels or with extracting endmembers from the hyperspectral data.

4.4 SPAMS (SPArse Modeling Software)

The code that we use to perform the dictionary learning is the Sparse Modeling Software (SPAMS). SPAMS is an open source set of C libraries that implement the online learning techniques outlined in Mairal's paper on online dictionary learning [64]. The library is interfaced to MATLAB through *mex*, allowing for easy access and analysis of the data returned. We chose this library because it incorporated both the dictionary learning, and also has an efficient implementation of the Least Absolute Shrinkage and Selection Operator (LASSO), to use for the sparse coding step. In addition to these steps, the open nature of the license allows us to modify the code as needed.

SPAMS has several different modes of operation for the dictionary learning task. After trying a few of the available modes, we ended up using a mode that finds the dictionary by solving

$$\min_{D \in \mathcal{C}} \frac{1}{n} \sum_{i=1}^{n} ||\boldsymbol{\alpha}_i||_1 \quad s.t. \quad ||\mathbf{x}_i - D\boldsymbol{\alpha}_i||_2^2 \leq \lambda. \tag{4.11}$$

The set $\mathcal{C}$ is the convex set of matrices that conform to the requirements of being a dictionary. We obtained the best results by preprocessing the cubes to make each pixel in the cube have unit energy on average. That is,

$$\mathbf{x}_i = \frac{\mathbf{p}_i}{||P||_F/n} = \frac{\mathbf{p}_i}{P_F}, \tag{4.12}$$

where $\mathbf{p}_i$ is a pixel of the cube, $P \in R^{m \times n}$ is a matrix formed of all the pixels stacked column-wise, and $P_F = ||P||_F/n$. Additional constraints enforced on the dictionary were that of less than or equal to unit energy dictionary atoms, and positivity of the components of the $\boldsymbol{\alpha}_i$'s.

Other modes investigated were minimizing the 2-norm error while bounding the 1-norm of the the abundances, and these same methods with the mean removed and the positivity constraints relaxed. These other modes were experimented with, but did not yield such easy and intuitive explanations as to the interpretability of the parameters.

4.4.1 Solving Over the Support Returned by *mexLasso*

The Lasso results from the function *mexLasso* are the results of a greedy algorithm. One way of interpreting the result from Lasso is that it returns the support over which the "true" solution can be found. When there exist constraints such as nonnegativity, the least squares problem solved over the lasso support must also be constrained.

In general, we found that the results obtained from this extra least-squares step were more accurate and fit the data better than the raw lasso results. Figure 4.1 shows this in detail. The optimization problem that is solved in the *mexLasso* is the sparse coding problem

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \|\boldsymbol{\alpha}\|_1 \qquad \text{subject to} \qquad \|\mathbf{x} - D\alpha\|_2 \leq \lambda, \tag{4.13}$$

but it only pushes the solution just inside the constraint of λ. Figure 4.1 shows a histogram of a full cube that has been reconstructed and the residual error calculated. That is,

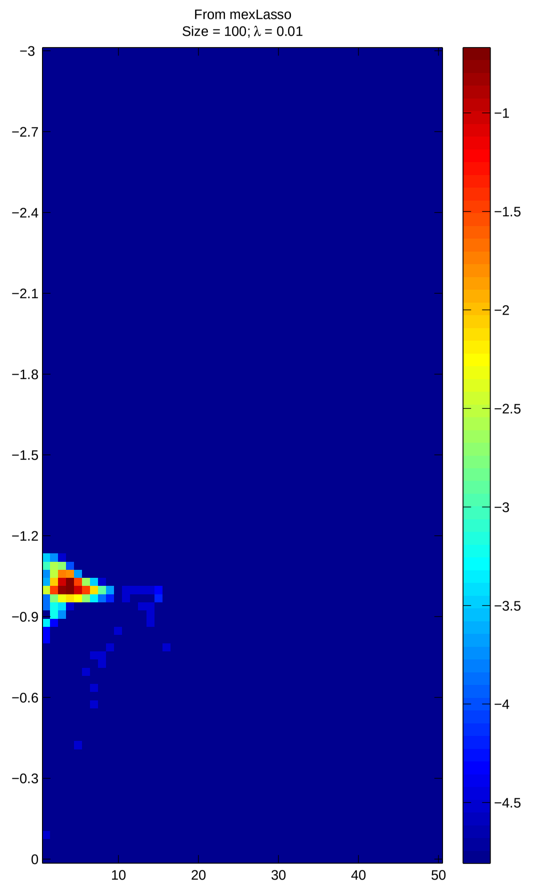
$$\mathbf{r}_i = \mathbf{x}_i - D\hat{\boldsymbol{\alpha}}_i, \tag{4.14}$$

is the residual error vector of the representation by the dictionary D and the abundance vector $\boldsymbol{\alpha}_i$. The histogram shows the log base 10 of the 2-norm error ($\log_{10}[\|\mathbf{r}_i\|_2]$) of each pixel reconstructed. The x-axis bins the pixels according to the sparsity of the abundance vector $\boldsymbol{\alpha}_i$. The bin color demonstrated the number of pixels that fall within that error-sparsity bin as a fraction of the total cube. The colorbar is expressed in percentage (*i.e.*, $.1 = 10\%$).

In the diagram to the left in fig. 4.1, the tight clustering of the vectors at the algorithm bound of the 2-norm representation error of $\lambda = 0.01$ can be seen. When least squares is applied over the support of the abundance vectors (the nonzero components), the error can be reduced —sometimes even by an order of magnitude. This is shown in the right portion of fig. 4.1, where the tight clustering has been smeared out down into the lower error region beyond the λ bound. Because of this better representation we always perform this additional step when reconstructing the data from the sparse representation.

4.4.2 Results of Dictionary Learning

The dictionary learning procedures are doing what we expect, in that they are able to represent the data in a sparse manner with low error. The constraints that we employ give us a natural interpretation of the error as a bound on the two-norm difference from the original data vector. Solving for the support first, and then resolving over the support produces even better error results. In our tests the sparsity was very low (*i.e.*, on the order of ten or fewer atoms active out of more than one hundred atoms). In the next chapter, we discuss the results of representing the HSI data in the dictionary learned basis specifically for classifying by material.



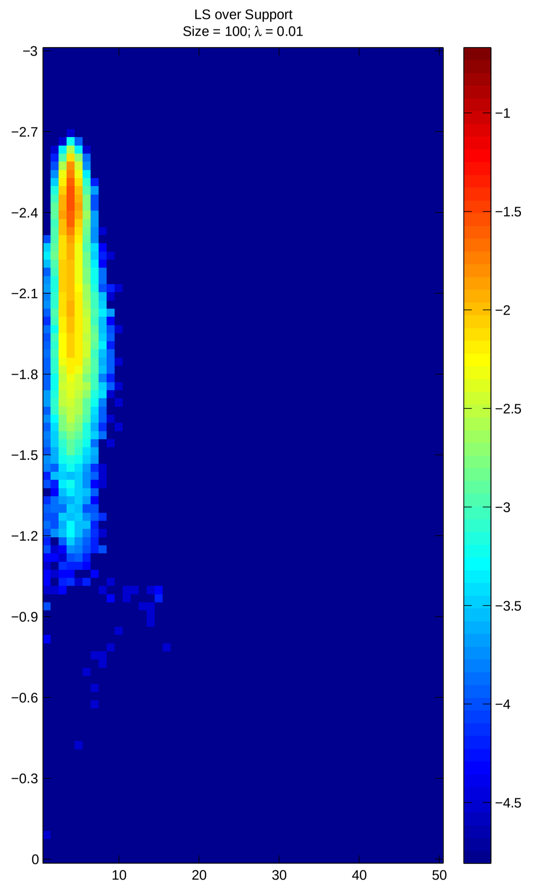


Fig. 4.1: The histogram of the number of reconstructed pixels of a cube that have a given error with a given number of active dictionary atoms. The x-axis specifies the number of non-zeros in the abundance vector, and the y-axis shows the $\log_{10}(\|\hat{\mathbf{x}}\|)$ of the reconstructed pixel, $\hat{\mathbf{x}}$. The constraint $\lambda = 0.01$, specifies the bound on the 2-norm error in the DL algorithm.

Chapter 5

DL+Classification

Dictionary Learning initially was devised for representational purposes. The sparser an abundance or feature vector could be, the greater the compression rate that was achieved in the new basis. DL holds importance in signal compression, storage, and transmission as long as the dictionary is able to still represent the signal with the same fidelity as before. As the technology matured, the focus shifted from finding efficient learning algorithms to being able to take advantage of the sparsity for tasks such as classification.

The main point of this thesis is the marriage of DL with a classifier to be able to detect and classify materials in a hyperspectral scene. The idea behind this is that DL imposes constraints on the data that reflect the physics of the problem, and can also affect some noise suppression, and ultimately enhancing the performance of the classifier.

5.1 Previous Work

In the DL literature there have been some attempts to classify on abundances and also to incorporate the classification into the DL process to provide a more discriminative dictionary.

5.1.1 Classifying on Sparse Representations

Wright *et al.* [65] utilize the idea that a certain number of classes and corresponding training samples are given. Thus, what we would call a dictionary can be formed from the samples for each of the classes,

$$A_i = \begin{bmatrix} \mathbf{v}_{i,1}, & \mathbf{v}_{i,1}, & \cdots, & \mathbf{v}_{i,1} \end{bmatrix} \in \mathbb{R}^{m \times n_i}. \tag{5.1}$$

and the full dictionary is comprised of each individual dictionary

$$A = \begin{bmatrix} A_1, & A_2, & \dots & A_k \end{bmatrix}. \tag{5.2}$$

Their algorithm then consists of utilizing sparse coding to solve the classical sparse coding representation problem

$$\mathbf{y} = A\mathbf{x}. \tag{5.3}$$

Ideally, they hoped that $\mathbf{x}$ would be of the form

$$\mathbf{x} = \begin{bmatrix} 0, & \dots, & 0, & \alpha_{i,1}, & \alpha_{i,2}, & \dots & \alpha_{i,n_i}, & 0, & \dots, & 0 \end{bmatrix}^T, \tag{5.4}$$

where all the entries are zero except the $\alpha_{i,j}$ that pertain to the ith class's portion of A.

They lay out the interpretation and consider small dense noise in the data, and provide an algorithm which they call Sparse Representation Classification (SRC) which selects the class for which the residual 2-norm error of the sparse representation is smallest.

The application of this classifier was tested on face recognition on a few different databases. They compare their results with a variety of other classifiers, including Nearest Neighbors (NN) and Nearest Subspace (NS) paired with different data preprocessing/dimension reduction routines (for example, Principle Components Analysis (PCA) and Independent Component Analysis (ICA)). They report marked gains over the other techniques.

In an extension of this main work they investigate their method with regard to occlusion and corruption in the face recognition. They exhibit a very strong robustness to these types of data issues, with the SRC being able to successfully classify even with very large portions of the data removed or corrupted.

They also define a measure they call the sparsity concentration index (SCI). This gives a measure of how well the test abundance vector is represented by any one class dictionary. If the vector is not able to be successfully represented in a single one of the subspaces of the class dictionaries, it is likely that the test vector is of a class not included in those in the dictionaries and is thus rejected as an outlier.

The dictionaries used in this effort were not actually learned dictionaries, but sample vectors of the different classes. It is assumed that the samples in each dictionary A_i span (or at least get close) the space that that particular class occupies. Although Wright *et al.* [65] do not explicitly do any dictionary learning in their algorithm, it can be directly extended to include dictionary learning.

5.1.2 Incorporation of Class Information into DL Algorithms

Supervised and Task-Driven Dictionary Learning

Mairal *et al.* [64, 66] incorporate classification into the DL framework by solving the following problem

$$\min_{\mathbf{D}\in D, \mathbf{W}\in W} E_{\mathbf{y},\mathbf{x}} \left\{ l_s\left(\mathbf{y}, \mathbf{W}, \boldsymbol{\alpha}^\star\left(\mathbf{x}, \mathbf{D}\right)\right)\right\}, \tag{5.5}$$

where $E_{\mathbf{y},\mathbf{x}}$ denotes expectation over the joint distribution of $\mathbf{y}$ and $\mathbf{x}$, and $\boldsymbol{\alpha}^\star\left(\mathbf{x}, \mathbf{D}\right)$ is the solution of the elastic-net problem

$$\boldsymbol{\alpha}^\star\left(\mathbf{x}, \mathbf{D}\right) \overset{\triangle}{=} \arg\min_{\boldsymbol{\alpha}} \frac{1}{2}\|\mathbf{x} - \mathbf{D}\boldsymbol{\alpha}\|_2^2 + \lambda_1\|\boldsymbol{\alpha}\| + \frac{\lambda_2}{2}\|\boldsymbol{\alpha}\|_2^2, \quad \text{with} \quad \boldsymbol{\alpha} \in \mathbb{R}^p. \tag{5.6}$$

The formulation allows for an analysis of the differentiability of the loss function, even though it depends on $\alpha^\star$ which is not differentiable, because it is uniformly Lipshitz continuous and differentiable almost everywhere. The only exception is where the support of $\boldsymbol{\alpha}^\star$ changes. The author refers the reader to the proofs in the paper appendices for a better treatment and increased understanding. Because of the assumptions and differentiability propositions, gradient update formulas can be derived and used to update both the dictionary $\mathbf{D}$ and the classification parameters $\mathbf{W}$, allowing for an efficient algorithm to build both. They show that these better tools perform favorably compared to their heuristics approach in their previous paper [67].

Discriminative K-SVD

Zhang and Li incorporate the discrimination and classification information into the dataspace, effectively expanding the dimensionality of the data [68]. The usual K-SVD algorithm [56] solves

$$\{D, \alpha\} = \arg\min_{D,\alpha} \|Y - D\alpha\|_2 \quad \text{subject to} \quad \|\alpha\|_0 \leq T \tag{5.7}$$

in a greedy fashion. Y is the data that is to be represented using the dictionary D and the sparse abundance vector α. Zhang and Li add to each data vector y_i a label vector $h_i = [0, 0, \cdots, 1, \cdots, 0]^T$, such that the 1 indicates the class or label of the corresponding data vector. They consider a linear classifier such that

$$H = W\alpha, \tag{5.8}$$

where W are the classifier parameters. Putting this together with the dictionary learning, they propose the following joint dictionary learning and classifier learning optimization problem

$$\{D, W, \alpha\} = \arg\min_{D,W,\alpha} \left\| \begin{pmatrix} Y \\ \sqrt{\gamma}\, H \end{pmatrix} - \begin{pmatrix} D \\ \sqrt{\gamma}\, W \end{pmatrix} \alpha \right\|_2 \quad \text{subject to} \quad \|\alpha\|_0 \leq T. \tag{5.9}$$

In this algorithm, γ provides a weighting factor to tune or trade-off the importance between the DL and classification training. This classification training plus dictionary learning problem still fits within the K-SVD algorithm framework and utilizes the K-SVD to learn the stacked matrix of D and $\sqrt{\gamma}\, W$.

The classification is performed in a post processing step. It requires a renormalization of the D and W, such that

$$D' = \left[\frac{d_1}{\|d_1\|_2}, \quad \frac{d_2}{\|d_2\|_2}, \quad \cdots, \quad \frac{d_k}{\|d_k\|_2}, \right] \qquad \text{and} \qquad W' = \left[\frac{w_1}{\|d_1\|_2}, \quad \frac{w_2}{\|d_2\|_2}, \quad \cdots, \quad \frac{w_k}{\|d_k\|_2}, \right]. \tag{5.10}$$

Then a sparse coding step to solve for the representative abundance is employed to give

$$\{\alpha'\} = \arg\min_{\alpha} \|y - D'\alpha\|_2 + \sigma\|\alpha'\|_0. \tag{5.11}$$

This sparse approximation can be solved in any number of ways such as those discussed previously in the Dictionary Learning chapter. When the sparse abundance α' vector is known, then the final classification is made by forming the label vector

$$l = W'\alpha'. \tag{5.12}$$

The largest of all the elements of this label vector indicates the classification assigned by the linear classifier.

Zhang and Li test this discriminative learning on face recognition. They report consistent improvement over classification applied after KSVD dictionary learning and over the sparse representation classifier of Wright *et al.* [65].

5.1.3 Applications to HSI

Nonnegative Matrix Underapproximation Incorporating Sparsity

Gillis and Plemmons [60] improve upon the nonnegative matrix factorization (NMF) technique used to factor HSI data into two nonnegative matrices recursively. The general NMF problem is

$$\min_{U\geq 0, V\geq 0} \|X - UV^T\|_F^2. \tag{5.13}$$

The underapproximation algorithm seeks to solve the above using a recursive method, extracting a single column of U and a single column of V at a time, for example, solving

$$\min_{u\geq 0, v\geq 0} \|X - uv^T\|_F^2 \quad \text{subject to} \quad uv^T \leq M. \tag{5.14}$$

Their contribution involves incorporating a sparsity inducing term in the optimization problem. By forcing the u vector to be sparse, then Sparse Nonnegative Underapproximations (sNMU) could overcome difficulties in separating endmembers when blurring, limited resolution or other data artifacts obscure or corrupt the HSI pixels. They report increased performance as compared to NMF and regular NMU in identifying the four main endmembers in a HYDICE hyperspectral image.

Variation on Euclidean Distance Classifier

Castrodad *et al.* [69] use Dictionary Learning as a variation on an Euclidean Distance Classifier by building a dictionary of each class and classifying a pixel as the material that corresponds to the dictionary that represents the pixel with the least amount of error.

They incorporate subpixel classification by performing an extra sparse coding step over all of the material dictionaries. One of the main focuses of this paper is the accurate reconstruction of a cube based upon a small number of the pixels (*i.e.*, the compressed sensing problem). They show that using extreme sub-sampling, and reconstruction based on the dictionary learning, classification is still possible. They show that the mixed pixel variation of their classifier produces clearer and more regular boundaries.

5.2 Novel Approach

To pair a classifier and DL, it requires first, a learned dictionary, then representing the pixels in this dictionary to produce abundance vectors. Lastly, the classifier is run on these abundance vectors to produce a prediction of the materials' class.

5.2.1 Classification Using RandomForests on Abundance Vectors

Selection of Datasets

The dataset that we used for this experiment is obtained from the 9 days of stable FOV, allowing for much easier extraction of the data to train and test on. The materials of each cube in the 9-days set (approximately 1100) were extracted and were plotted in a histogram to determine which cubes had the most correlation between materials. The true candidate spectra for each material was identified as the maximum ridge across the histogram space. Then each material was scored on how well its own materials matched with the candidate spectrum. The cubes with the highest score were the ones chosen for the test. Ten cubes were selected for training and ten were selected for testing.

Training the Dictionary

The panel for each of the training cubes were concatenated together to provide a training set of 60,000 examples of HSI pixels to learn a dictionary of 140 elements. Of those 140 elements, approximately 20-30 provide information on the materials on the panel. Figure 5.1 shows the covered of the 20 to 30 atoms that contribute to the materials on the board. Figures 5.2 to 5.8 show the abundance maps of each dictionary element of an 140 element dictionary as reconstructed from a separate panel image. Numerous of the atoms target one or more material, and it is hoped can be used to discriminate it from the rest.

The materials on the board preferentially utilize different dictionary elements. The amount of usage of the dictionary elements can be seen in fig. 5.9.

Running RF on the Data

The testing consisted of forming the dataset as detailed, using raw radiance data. A

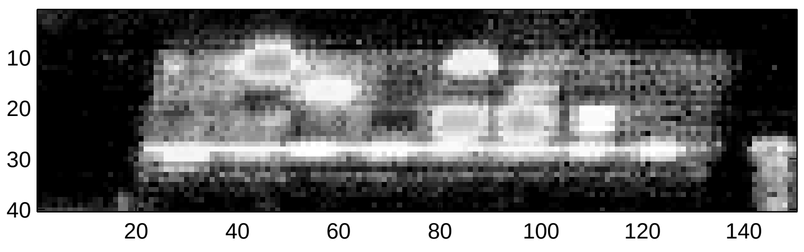


Fig. 5.1: An agglomeration of the abundance maps of approximately 30 atoms of from a learned dictionary of 140 atoms. These atoms are showing most of the materials on the board.

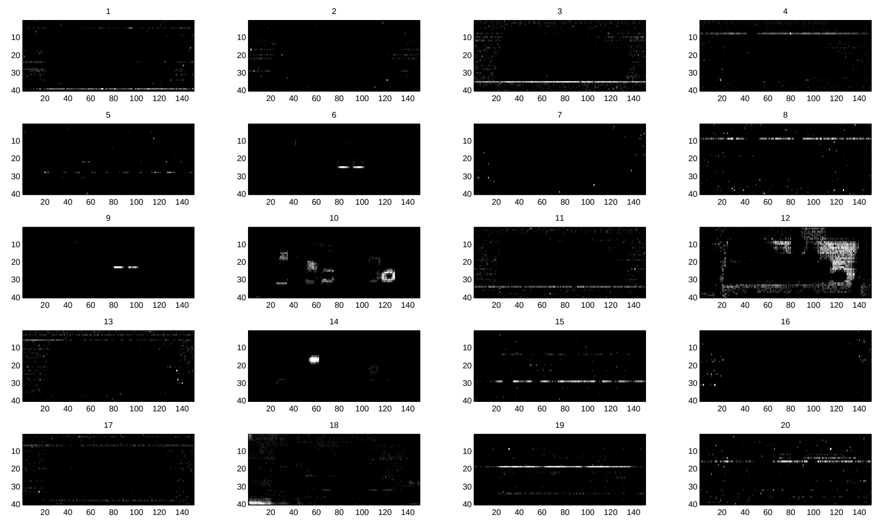


Fig. 5.2: The abundance maps pertaining to atoms 1-20 of a trained dictionary of 140 atoms.

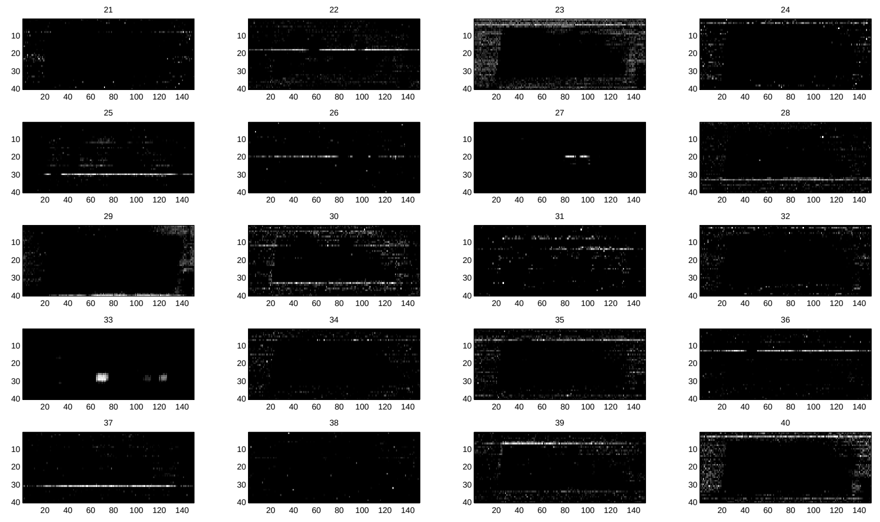


Fig. 5.3: The abundance maps pertaining to atoms 21-40 of a trained dictionary of 140 atoms.

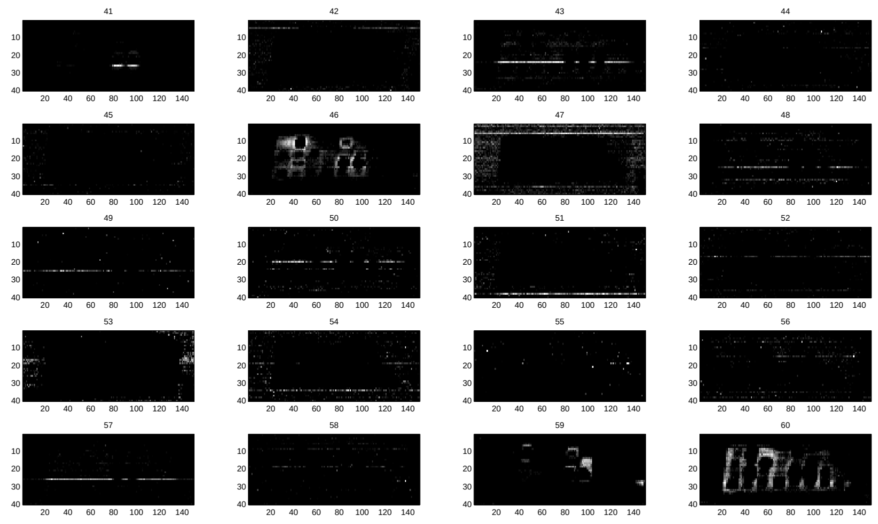


Fig. 5.4: The abundance maps pertaining to atoms 41-60 of a trained dictionary of 140 atoms.

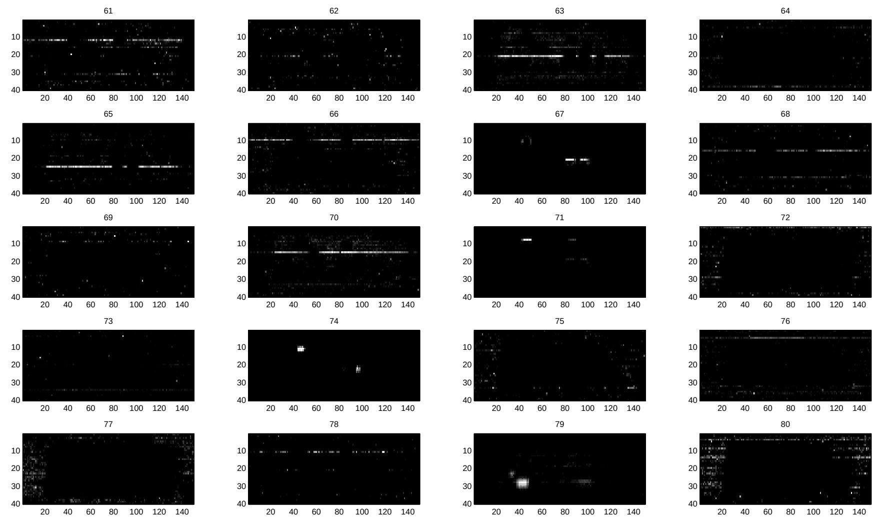


Fig. 5.5: The abundance maps pertaining to atoms 61-80 of a trained dictionary of 140 atoms.

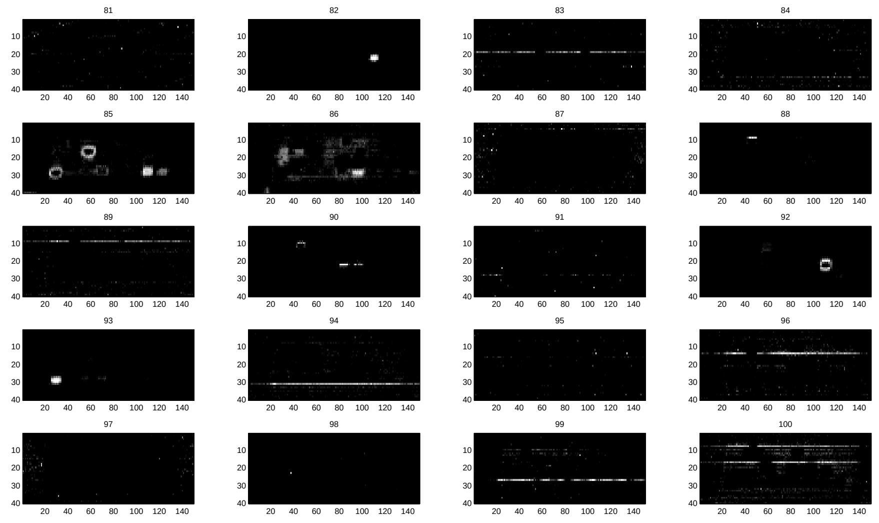


Fig. 5.6: The abundance maps pertaining to atoms 81-100 of a trained dictionary of 140 atoms.

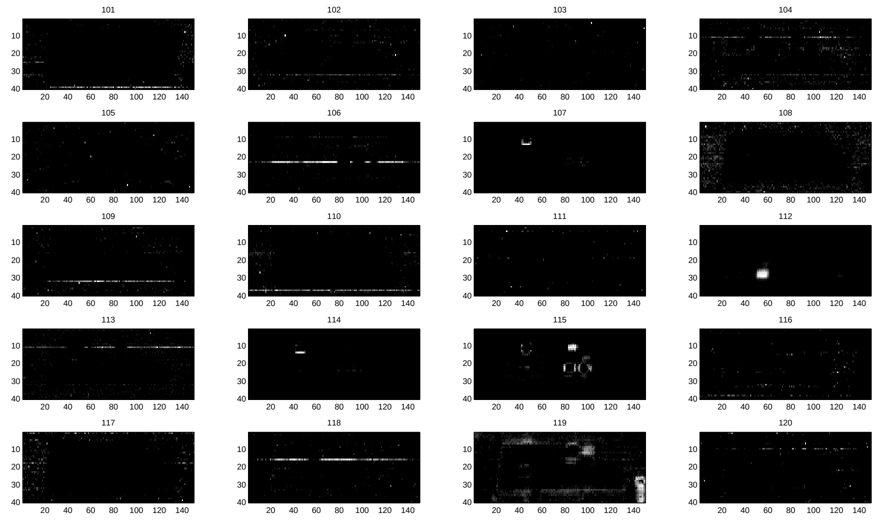


Fig. 5.7: The abundance maps pertaining to atoms 101-120 of a trained dictionary of 140 atoms.

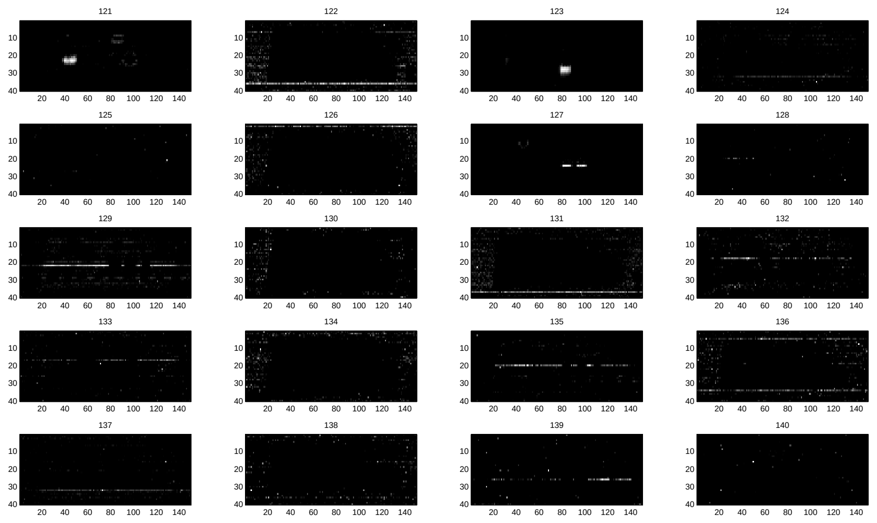


Fig. 5.8: The abundance maps pertaining to atoms 121-140 of a trained dictionary of 140 atoms.

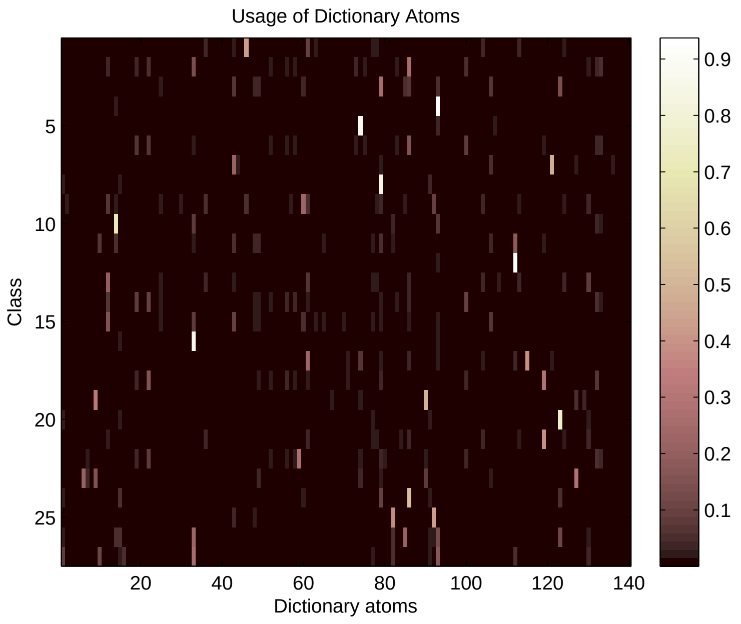


Fig. 5.9: This shows the average usage of each vector in each class. The y-axis is a listing of each of the 27 materials and the x-axis is the dictionary atoms. It can be seen that different materials preferentially utilize different atoms of the dictionary.

zero-mean version is used also, in order to compare with the results in the previous chapter of using RF on just the data. Then a dictionary is learned for each type of data that will be tested. The data is represented in terms of those dictionaries to produce abundance vectors. Subsequently, forests were trained using both the raw data and the abundances from the training portions of the datasets. Then the test data was sent down the forest and a prediction was given. The results are detailed in the next section.

5.2.2 Results

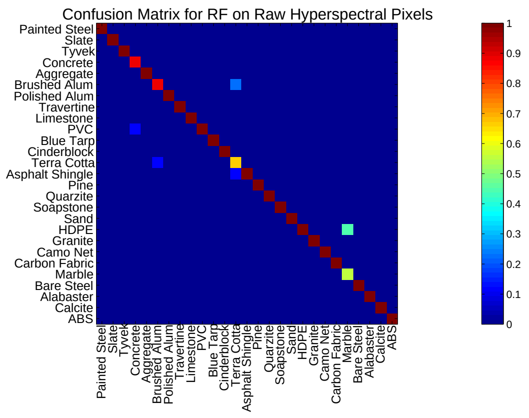
When classifying on the abundance vectors, the accuracy hoped for was not attained. The RF classifier had decent performance by itself on the raw radiance data. As can be seen in Table 5.1, RandomForests on the raw hyperspectral pixels had an error of approximately 4%, whereas when classifying on the abundance vectors obtained from sparse coding with the learned dictionary only achieved about 14% error. As a check, the original pixels that were tested were reconstructed from the dictionary using

$$\hat{\mathbf{x}}_i = D\boldsymbol{\alpha}_i \qquad \forall \quad i, \tag{5.15}$$

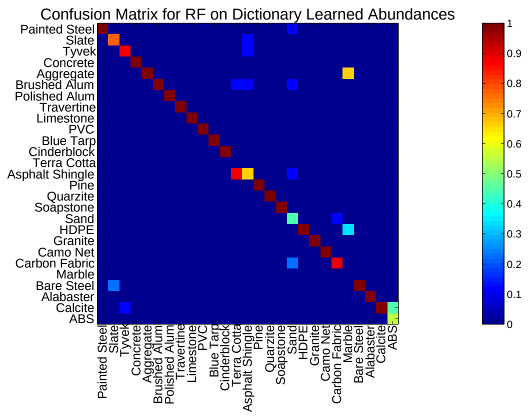
for the dictionary D and abundance vectors $\boldsymbol{\alpha}_i$. Because of the DL algorithm, the reconstructed 2-norm error is less than approximately 1% per pixel on average. This was verified. Then these reconstructed pixels were put through the same process as the raw data, building a classifier and testing. The results at approximately 8% error were better than classifying on just the abundance vectors, but still did not improve over classifying on the raw hyperspectral pixels. The confusion matrices of the tests preformed are shown in fig. 5.10. Confusion matrices were explained in detail in section 3.4.3. We hoped to see

Table 5.1: Error results for RF with DL.

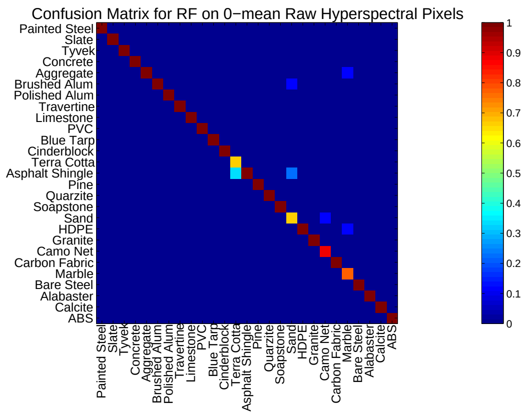
	Raw	DL Abundances	Reconstructed
Radiance	0.0370	0.1399	0.0823
0-mean Radiance	0.0370	0.107	0.0740



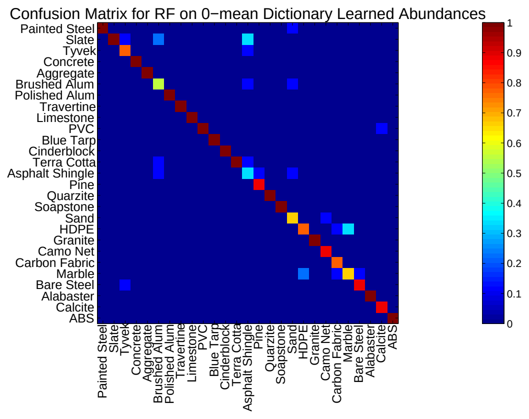
(a) Raw radiance RF



(b) Raw DL RF



(c) 0-mean Raw radiance RF



(d) 0-mean DL RF

Fig. 5.10: Confusion matrices for each of the tests performed.

confusion between similar types of materials, but that does not seem to be the case in any of the tests performed. The closest that we see is the confusion between Sand and Asphalt Shingle in the 0-mean raw radiance test.

Thus, DL is not producing abundances that are useful for RF classifying. In order to test this a bit further, the residual of the reconstruction are then used to classify. The residuals are calculated using the $\hat{\mathbf{x}}$ from equation eq. (5.15) as

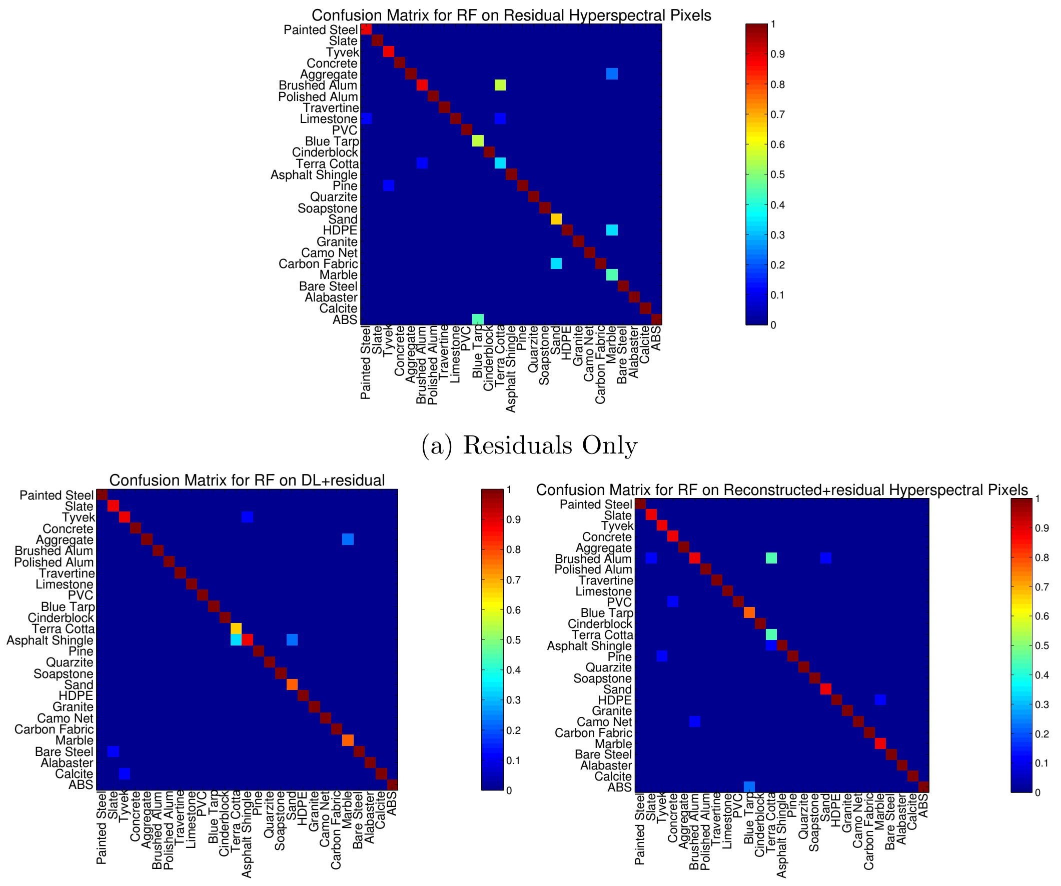
$$\mathbf{x}_r = \mathbf{x} - \hat{\mathbf{x}}. \tag{5.16}$$

Then we follow the same procedure as the other tests. The error obtained from RandomForests with regards to residuals is shown in Table 5.2. The residuals had an error rate around 8.6%, which definitely is better than the RandomForests on the abundances by themselves (although not as good as the raw HSI pixels). So from this we can conclude that some of the information RandomForests is using for the classification is lost in that 1% reconstruction error from the Sparse coding using the learned dictionary.

Additional tests were performed making use of the residual vectors. We concatenated the abundances with the residuals to see whether RandomForests is able to regain any of the information lost. We also concatenated the reconstructed vectors with the residuals to test whether that would also provide any gains. The results of each of these test are shown in Table 5.2, also. Each of them obtained better error rates than DL abundances by themselves and the reconstructed pixels but neither regained enough to overtake using RandomForests on the raw HSI pixels. The abundance vectors with the residuals had an error rate of approximately 4.1% and the reconstructed pixels with the residuals obtained 5.4%. Figure 5.11 shows the confusion matrices for the tests involving the residual vector.

Table 5.2: Error results obtained from utilizing the residual vectors $\mathbf{x}_r$.

Test data type	Error rate
Residual only	0.086
Abundances and Residuals	0.041
Reconstructed and Residuals	0.054



(a) Residuals Only

(b) DL Abundances and Residuals

(c) Reconstructed Spectra and Residuals

Fig. 5.11: The confusion matrices for the test involving the residuals.

It was considered whether these results were because of the particular dictionary learning algorithm that was chosen. In order to test this, the same test was performed using the K-SVD package by Rubenstein *et al.* [70]. The results were consistent with those reported using the SPAMS package.

5.3 Conclusion

The RandomForests classifier did not perform as well as expected in distinguishing between materials when utilizing the sparse representations. The RandomForests classifier on the raw data outperformed the classifier on the sparse data significantly. The cause was found to be that the residual information that was lost in the sparse representation contained a portion of that information that was needed by the RandomForests algorithm to realize the matrial classification. This loss of valuable information shows that the pure reconstructive dictionary learning algorithms, such as those tested, lacked the possibility to identify which portions of the hyperspectral information was necessary for the classification. This maybe should not be much of a surprise, because these algorithms do not include any explicit classifying information into their optimization cost functions. It is possible that some of the proposed algorithms that incorporate the classifying information into their cost functions may provide the type of sparse representations that RandomForests would then be able to use to improve the classification error.

Chapter 6
Detection

6.1 Detection of Materials of Interest

Detection is a binary classification problem. Instead of considering all of the materials in a particular scene, the objective is to determine where, if at all, a certain material is located. This is different from the other classification tests that we have studied, because the focus is on one material at a time, neglecting the others, or aggregating them into the background.

In our studies, we look at detection of each of the 27 materials of the Denali dataset using similar detection algorithms to those used commonly on hyperspectral images.

6.1.1 General Target Detection

Target detection, referred as just detection from henceforth, is generally posed in the framework of hypothesis testing. Neyman-Pearson detection is performed using a likelihood ratio test and utilizes two hypotheses. The first hypothesis, often called the null hypothesis, is denoted H_0, and generally indicates that the signal of interest or the target is not present. The alternative hypothesis, H_1, is where the target is modeled as being present. Often these are specified as

$$\begin{aligned} H_0 &: \text{target/material not present,} \\ H_1 &: \text{target/material present.} \end{aligned} \tag{6.1}$$

Frequently, some distributions are assumed with some (possibly unknown) parameters in order to facilitate prediction. Let $l_0(\mathbf{x}|\boldsymbol{\theta}_0)$ represent the likelihood of the distribution for the null hypothesis and $l_1(\mathbf{x}|\boldsymbol{\theta}_1)$ be the likelihood for the distribution for when the target is present, then we can form the ratio

$$D(\mathbf{x}) = \frac{l_1(\mathbf{x}|\boldsymbol{\theta}_1)}{l_1(\mathbf{x}|\boldsymbol{\theta}_0)}. \tag{6.2}$$

This is called the Likelihood Ratio Test (LRT). This ratio of likelihoods then becomes our detector. If this is low, then it is much more likely that the null hypothesis is correct and vice versus for a high value of the GLRT. This detection criteria and framework is often called the Neyman-Pearson criteria. Then for a given detector D, the test becomes

$$D(\mathbf{x}) \underset{H_0}{\overset{H_1}{\gtrless}} \eta, \tag{6.3}$$

where η is some significance threshold which governs the trade-off between the probability of detection P_D and the probability of false alarm P_{FA}.

In our experiments we compare the performance of two well known, well used matched filters: the adaptive matched filter and the adaptive coherence estimator. Both are used frequently as detection algorithms, and both have broad applications across many different signal processing fields, and both are used in hyperspectral detection [71, 72]. Both of these algorithms are described along with various others by Theiler and Foy [73].

6.1.2 Adaptive Matched Filter

The adaptive matched filter can be developed using the hypothesis that the mean of the distribution that is being looked at has changed when the target is present. That is, we work under the hypotheses

$$\begin{aligned} H_0 &: \mathbf{x} = \boldsymbol{\mu}_b + \mathbf{n} && \sim N\left(\boldsymbol{\mu}_b, \Sigma_b\right) \\ H_1 &: \mathbf{x} = a\boldsymbol{\mu}_t + \mathbf{n} && \sim N\left(a\boldsymbol{\mu}_t, \Sigma_b\right). \end{aligned} \tag{6.4}$$

By forming the likelihood ratio between these hypothesis, we obtain the detector

$$D_{AMF}(\mathbf{x}) = \frac{\left(\boldsymbol{\mu}_t - \boldsymbol{\mu}_b\right)^T \Sigma_b^{-1} \left(\mathbf{x} - \boldsymbol{\mu}_b\right)}{\sqrt{\left(\boldsymbol{\mu}_t - \boldsymbol{\mu}_b\right)^T \Sigma_b^{-1} \left(\boldsymbol{\mu}_t - \boldsymbol{\mu}_b\right)}} \underset{H_0}{\overset{H_1}{\gtrless}} \eta_{AMF}. \tag{6.5}$$

6.1.3 Adaptive Coherence Estimator

The adaptive coherence estimator (ACE) deals with separating the subspaces of the background and the target. This allows a richer model for the target, and does not restrict it to a single mean vector, but allows it to vary over a subspace denoted by $\mathbf{S}$. The hypotheses assumed for this detector are

$$\begin{aligned} H_0 &: \mathbf{x} = \boldsymbol{\mu}_b + \mathbf{n} && \sim N\left(\boldsymbol{\mu}_b, \Sigma_b\right) \\ H_1 &: \mathbf{x} = \mathbf{S}\mathbf{a} + \beta\mathbf{n} && \sim N\left(\mathbf{S}\mathbf{a}, \beta^2\Sigma_b\right). \end{aligned} \tag{6.6}$$

This gives rise to the detector

$$D_{ACE}(\mathbf{x}) = \frac{\left(\mathbf{x}-\boldsymbol{\mu}_b\right)^T \Sigma_b^{-1}\mathbf{S}\left(\mathbf{S}^T\Sigma_b^{-1}\mathbf{S}\right)^{-1}\mathbf{S}^T\Sigma_b^{-1}\left(\mathbf{x}-\boldsymbol{\mu}_b\right)}{\left(\mathbf{x}-\boldsymbol{\mu}_b\right)^T \Sigma_b^{-1}\left(\mathbf{x}-\boldsymbol{\mu}_b\right)} \underset{H_0}{\overset{H_1}{\gtrless}} \eta_{ACE}. \tag{6.7}$$

If the subspace $\mathbf{S}$ is one-dimensional, that is, $\mathbf{S} = \mathbf{s}$ a vector, then the detector can be simplified to

$$D_{ACE-1}(\mathbf{x}) = \frac{\left(\mathbf{s}^T\Sigma_b^{-1}(\mathbf{x}-\boldsymbol{\mu}_b)\right)^2}{\left(\mathbf{s}^T\Sigma_b^{-1}\mathbf{s}\right)\left((\mathbf{x}-\boldsymbol{\mu}_b)^T\Sigma_b^{-1}(\mathbf{x}-\boldsymbol{\mu}_b)\right)^2} \underset{H_0}{\overset{H_1}{\gtrless}} \eta_{ACE}. \tag{6.8}$$

6.1.4 Receiver Operator Characteristic Curves

Each detection algorithm has a performance for a given set of parameters and a threshold. The threshold can be swept over and a performance curve can be made, comparing the probability of detection P_d with the probability of false alarm P_{fa}. These performance curves are called receiver operator characteristic (ROC) curves. As the detection rate goes up, the false alarm rate increases also. Often, then a determination of the allowable amount of false alarms are set, which determines the level of detection possible. The ROC curves allow for a detection scheme-agnostic statistic that enables comparison between any two detection algorithms. The algorithm which has higher P_d at the set or allowable P_{fa} is the better algorithm to use.

Another method to judge the appropriateness of an algorithm is by using the area under the ROC curve (AUC) as a measure of effectiveness. A test which performs extremely well, will be in the upper left corner of the ROC curve "box." The upper limit for the AUC is one, for a perfect detector at a zero false alarm rate. Everything else will have an AUC value less than one. The ROC curve itself provides more information, because there exists an ambiguity in the AUC, because there may be different ROC curves that produce the same AUC, but as a whole, the AUC provides a convenient quick look at detector performance.

6.1.5 Sparse Representations = Fast Filtering

An advantage of representing the data in a sparse representation is that the filtering can be implemented and optimized to be very efficient. Most of the calculations of the the filter output can avoided by only calculating the nonzero products in the filtering step. Depending on the the sparsity of the representation, this could produce dramatic speed up over filtering the data in the raw, dense representation.

6.2 Detection in HSI

Manolakis *et al.* [72] investigate numerous different proposed detection algorithms for HSI. They first consider various models of hyperspectral data, including probabilistic and physics-based models. Of the probabilistic models, they consider a few different distributions, namely the multivariate normal, the multivariate t-distribution and the elliptically contoured family of distributions.

They then give an overview of detection basics and introduces the concepts involving generalized likelihood ratio tests. Expanding on this, they introduce a few notable algorithms including the quadratic detector, the target covariance shrinkage detector, the matched and adaptive matched filter detectors, and the anomaly detector, all based on and assuming Gaussian data. Manolakis *et al.* then give an introduction to other types of detectors that utilize additive signal models, replacement signal models, additive subspace methods, and other non-Gaussian distributions.

Manolakis *et al.* conclude stating that it is difficult to compare all of the detection algorithms to each other, and postulate that (in the spirit of Occam's Razor) the well understood algorithms with good theoretical properties provide good performance. The small performance gains from more sophisticated methods are often offset by difficulties in assessing the limitations and uncertainties of how the algorithm performs in a particular setting.

Nasrabadi takes a similar route in his survey paper [71]. He first explores the common anomaly detectors, such as the RX anomaly detector, and also variants of it. Then he summarizes a kernel RX detector, and expounds upon the merits of the support vector data description detector. He gives results showing the performance of each of these on some HYDICE hyperspectral data.

Next, Nasrabadi considers subspace and matched filtering detection strategies. He also details some of the challenges that these target detectors face, describing the difficulties in detecting by comparison, if the target signature is significantly different in the field than that in the spectral library.

Lastly, Nasrabadi describes a method of applying the sparse representation classifier (SRC) of Wright *et al.* [65] to hyperspectral data and adapted for detection. The detection algorithm is based on the residual from the reconstruction of the hyperspectral data from the dictionary and the abundance vectors. Nasrabadi then ends by outlining several directions of future research, including the development of a dictionary that has the desired qualities.

Chen *et al.* use sparse coding on a composite 2-class dictionary to perform detection [74]. They form the dictionary from

$$D = \begin{bmatrix} A_b & A_t \end{bmatrix}, \tag{6.9}$$

where A_b is a "dictionary" formed of background pixels, and A_t is a subspace formed from pixels designated as targets. They use a double windowing method to form the adaptive localized background dictionary. This captures and allows for local variations in the background subspace.

They use the large dictionary and calculate a sparse abundance vector γ by

$$\hat{\gamma} = \arg\min \|\gamma\|_0 \quad \text{subject to} \quad A\gamma = x. \tag{6.10}$$

This abundance vector is partitioned according to the two class dictionaries

$$\gamma = \begin{bmatrix} \hat{\alpha} \\ \hat{\beta} \end{bmatrix}. \tag{6.11}$$

A residual measure is then formed from these sparse abundance vectors and their respective dictionaries

$$r_b(x) = \|x - A_b\hat{\alpha}\|_2 \qquad \text{ans} \qquad r_t(x) = \|x - A_t\hat{\beta}\|_2. \tag{6.12}$$

Their detection or classification method is based on a ratio of the size of the residuals, yielding

$$D(x) = \frac{r_b(x)}{r_t(x)} \gtrless \delta. \tag{6.13}$$

A large value of $D(x)$ indicates a target pixel. They report detection performance that exceeds that of both spectral matched filters and matched subspace detectors in both qualitative and quantitative measures.

6.3 Testing Procedures

For the tests performed, a synthetic dataset was manufactured in the following manner. A material to be tested is selected from the 27 available materials. A certain amount of the dataset is set aside to include the material of interest. This amount is the set coverage amount and can range anywhere from 0-100%. The pixels set aside to contain the material of interest are then filled with an abundance fraction of a sample drawn uniformly from the material of interest median filtered data. In the tests abundances from 15% up to 35% were used. Then the rest of the pixel was filled with pixel spectra drawn randomly from the other material types. The rest of the dataset not containing the material of interest was uniformly drawn and mixed from the remaining materials.

6.4 A Sparse Representation-Based Detector

In addition to evaluating the most commonly used matched filters specified above, this paper details another method that is loosely based on the concepts of self-taught learning. Self-taught learning was introduced by Raina *et al.* [75] in 2007. Self-taught learning strives to use a large amount of unlabeled data to find a useful description of the data as a whole. Then the (possibly a small limited number of) labeled data are then represented in this data descriptive format. Then use these new representations to perform classification, or in our case, detection.

This approach allows for use of a large amount of data to find a suitable decomposition, for example a learned dictionary, and then still uses really only a small number of labeled components, as is common in a hyperspectral library.

6.4.1 Learning the General Dictionary

In order to build a dictionary that incorporated a good amount of hyperspectral content, a dictionary was built from a large number of hyperspectral pixels, some of which may be the material for which you have designated as the material of interest. In our experiments, we constructed it with smaller dictionaries built around different classes that we were interested. With 27 different material on the Denali panel, we used all the materials on the panel over a large time interval for building the dictionaries.

6.4.2 Representing the Labeled Data

Then the reference library spectra are represented using sparse coding and the dictionary built from the unlabeled data. In this particular method, each library spectra $\mathbf{l}_i$ that is available is represented in the dictionary representation, yielding a "new" library $L \in \mathbb{R}^{N_s \times N_L}$, where N_s is the dimension of the hyperspectral data and N_L is the number of spectra in the library. This library is now in the basis of the dictionary that was learned based on the large number of unlabeled data.

The data to be tested is also represented in the dictionary basis, in addition to the library. That is, sparse coding is perform on the data to produce a dataset of abundance vectors

$$A = \arg\min_A \|X - DA\|, \quad \text{where} \quad A = \begin{bmatrix} \boldsymbol{\alpha}_1, & \boldsymbol{\alpha}_2, & \ldots, & \boldsymbol{\alpha}_M \end{bmatrix}, \tag{6.14}$$

and each $\boldsymbol{\alpha}_i$ is an abundance vector for the ith data vector $\mathbf{x}_i$. Thus A is a sparse matrix made up of abundances.

6.4.3 Class-Wise Concentration-Based Detection

The method developed here utilized the same ideas as the matched filter, but instead of only filtering on the spectra in the library of the material of interest, the data to be tested A is filtered with the *entire* library. Thus we receive back a vector for each data-vector filtered

$$\mathbf{r}_i = MF(\boldsymbol{\alpha}_i, L) = \begin{bmatrix} MF(\boldsymbol{\alpha}_i, L_1) \\ MF(\boldsymbol{\alpha}_i, L_2) \\ \vdots \\ MF(\boldsymbol{\alpha}_i, L_{N_L}) \end{bmatrix}. \tag{6.15}$$

Instead of the usual thresholding or finding the element that produces the maximum in each $\mathbf{r}_i$, more post processing is performed to determine whether the $\mathbf{x}_i$ is of the material of interest. The detection algorithm detailed is based on the idea that if a material is "close" to a certain class in the library, then it will score higher in the matched filters of that class than the others.

Given a vector $\mathbf{r}_i$, produce the normalized vector $\hat{\mathbf{r}}_i = \mathbf{r}_i/\|\mathbf{r}_i\|$. Then partition the vector into the portions which correspond to the library spectra of the material of interest. That is, let the vector $\hat{\mathbf{r}}_i$ be decomposed as (without the i subscript for easier notation)

$$\hat{\mathbf{r}} = \hat{\mathbf{r}}_t + \hat{\mathbf{r}}_{\tilde{t}}, \tag{6.16}$$

where

$$\begin{aligned}\hat{\mathbf{r}}_t &= \Big[[0, \quad \ldots, \quad 0, \quad \hat{r}_k, \quad \ldots, \quad \hat{r}_{k+j}, \quad 0, \quad \ldots, \quad 0\Big]^T \\ \hat{\mathbf{r}}_{\tilde{t}} &= \Big[[\hat{r}_1, \quad \ldots, \quad \hat{r}_{k-1}, \quad 0, \quad \ldots \quad 0, \quad \hat{r}_{k+j+1}, \quad \ldots, \quad \hat{r}_{N_L}\Big]^T .\end{aligned} \tag{6.17}$$

The j terms begining at the kth term correspond to the material of interest from the library. Thus, the vector $\hat{\mathbf{r}}_t$ has the elements that provide the matched filter outputs from the material of interest and the vector $\hat{\mathbf{r}}_{\tilde{t}}$ has the other materials' matched filter output.

Then a detection is performed by considering the ratio test

$$D_E(\hat{\mathbf{r}}) = \frac{\|\hat{\mathbf{r}}_t\|_2}{\|\hat{\mathbf{r}}_{\tilde{t}}\|_2} \underset{H_0}{\overset{H_1}{\gtrless}} \eta_E. \tag{6.18}$$

This compares the concentration of the MF outputs in the material of interests' portion of the library versus the rest of the library. If a test vector is higher in the material of interest's portion and lower in the rest of the library's MF outputs, then the vector is considered as the material of interest. Because of this utilization of the class-wise concentration, we refer to this algorithm as Class-wise Concentration Match Filter (CCMF).

The matched filter outputs can be either of the described filters or even other scoring algorithms. The tests performed for this research utilized both the AMF and the ACE filters.

6.5 Comparison of MFs

6.5.1 Metals

I will discuss and explain the results shown in fig. 6.1. Results are fairly typical for all the materials presented. In all of these diagrams, a curve closer to the upper left corner indicates higher performance. The upper left figure shows the results of using the Class-wise Concentration Match Filter (CCMF) decribed previously. As can be seen, the DL CCMF provides some performance gain over using the CCMF on the raw data as evidenced by the curves being "above" the others. For most materials, this is the case, although nor all. The upper right ROC curve demonstrates the Adaptive Matched Filter (AMF) on both raw radiance and DL abundance inputs. It can be seen that the dictionary-learned AMF performance is a bit better than that of the AMF on the raw data. The lower ROC curves show the performance of the Adaptive Cosine Estimator (ACE) filter when used on both raw radiances and DL abundances as inputs. The lower left is the usual ACE filter utilizing the true covariance of the data, whereas the right neglects any covariance measurements, substituting the identity matrix in for the covariance in the ACE filtering calculation. Both demonstrate that dictionary learning can have a positive effect on the detection algorthms, because the algorithms performed on the dictionary learned sparse datahave better performance (*i.e.*, are above the other curves). Overall, the CCMF on DL abundances seems to give the best performance over the AMF and ACE filters.

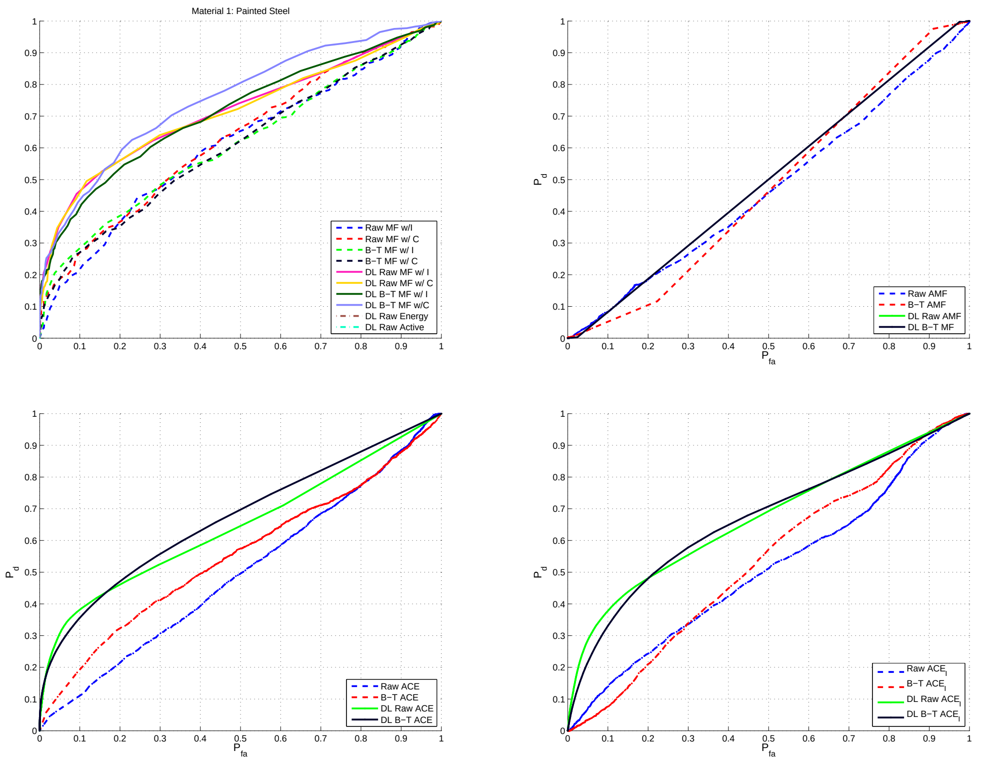


Fig. 6.1: Material 1: Painted Steel.

The other metals are shown in figs. 6.2 to 6.4. They detail the performance on bare steel (fig. 6.2), brushed aluminum (fig. 6.3), and polished aluminum (fig. 6.4). The interpretation is similar to that of painted steel above.

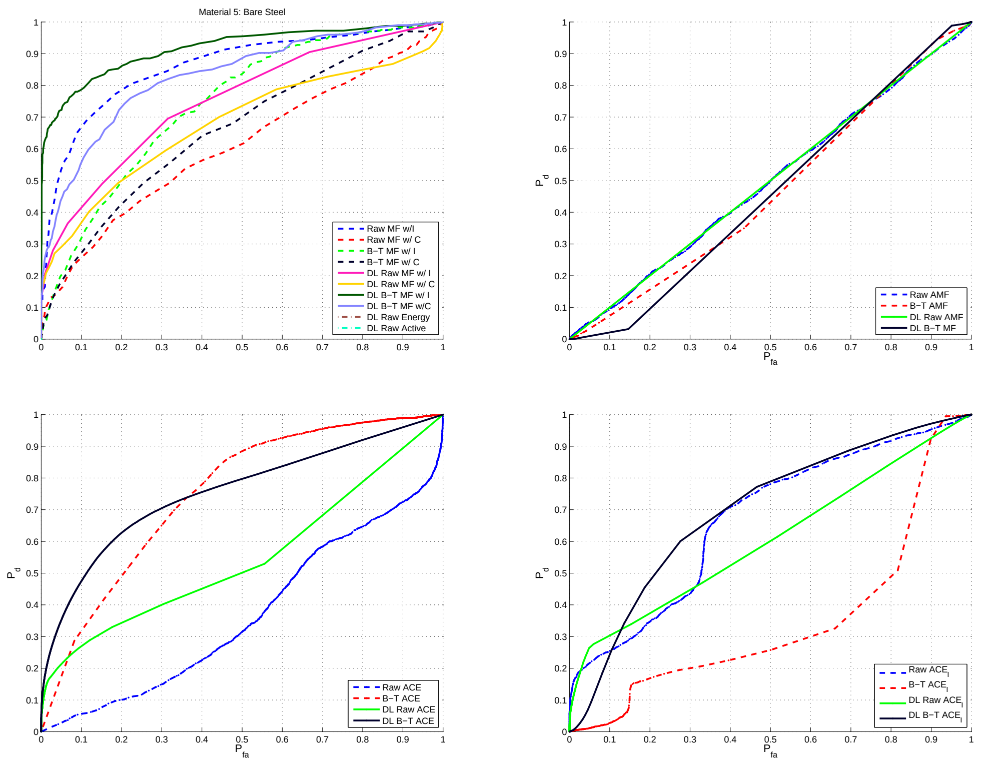


Fig. 6.2: Material 5: Bare Steel.

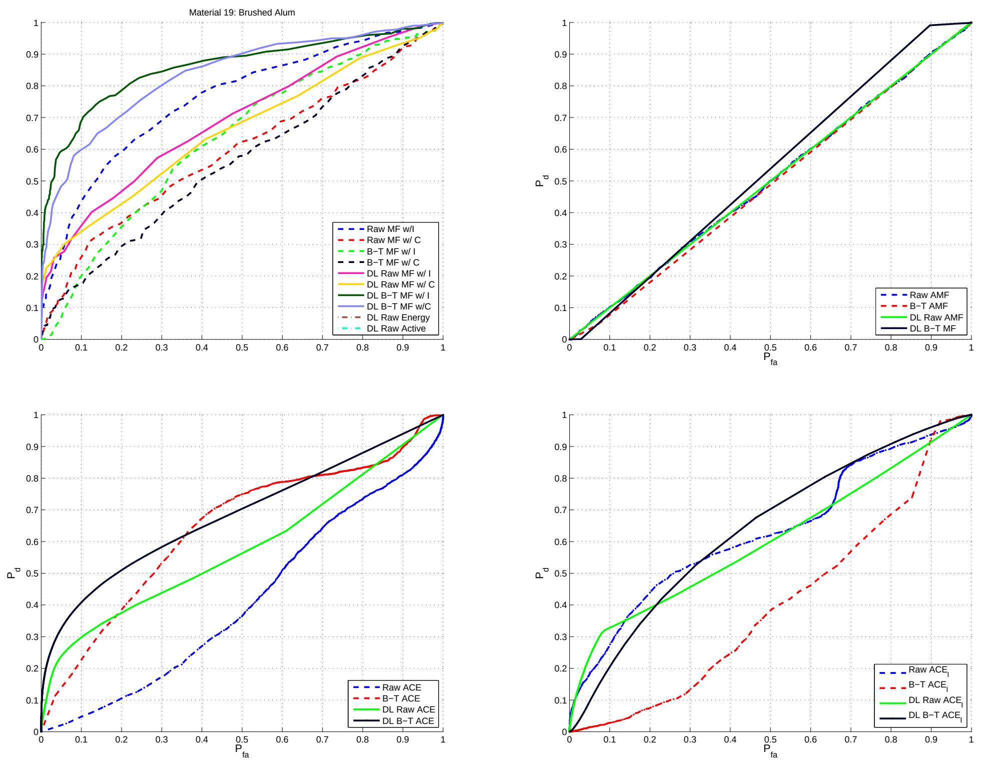


Fig. 6.3: Material 19: Brushed Aluminum.

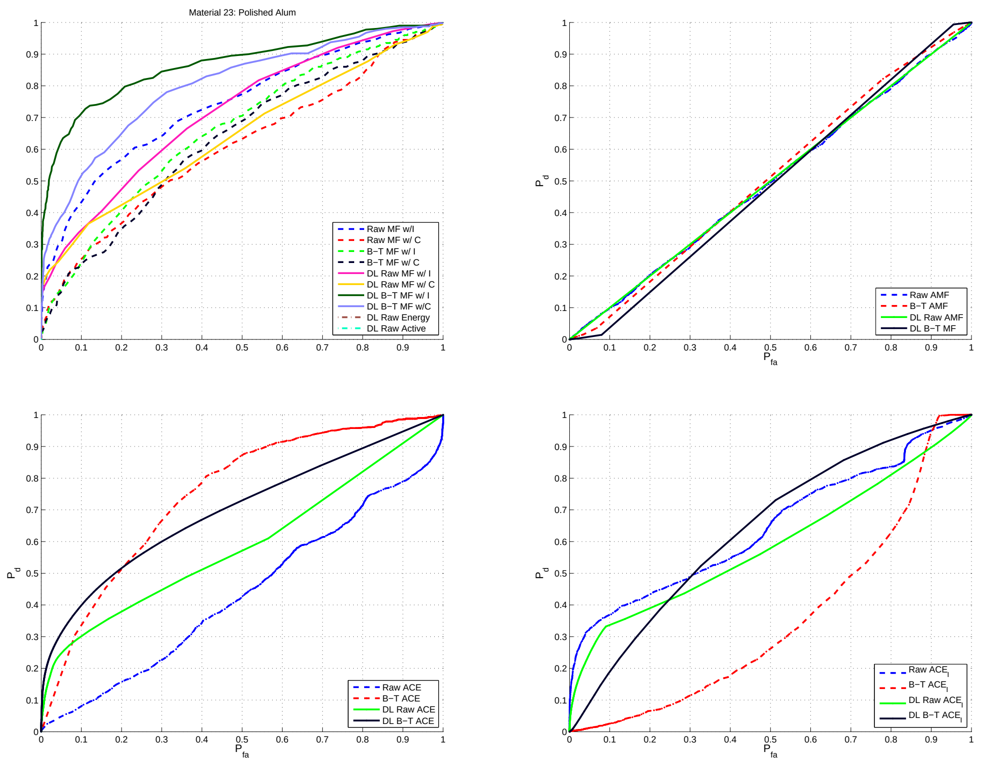


Fig. 6.4: Material 23: Polished Aluminum.

6.5.2 Rocks/Minerals

The natural rocks and minerals are shown in figs. 6.5 to 6.14. They detail the performance on marble (fig. 6.5), granite (fig. 6.6), travertine (fig. 6.7), slate (fig. 6.8), alabaster (fig. 6.9), calcite (fig. 6.10), limestone (fig. 6.11), soapstone (fig. 6.12), quartzite (fig. 6.13), sand (fig. 6.14). The interpretation is similar to that of painted steel above.

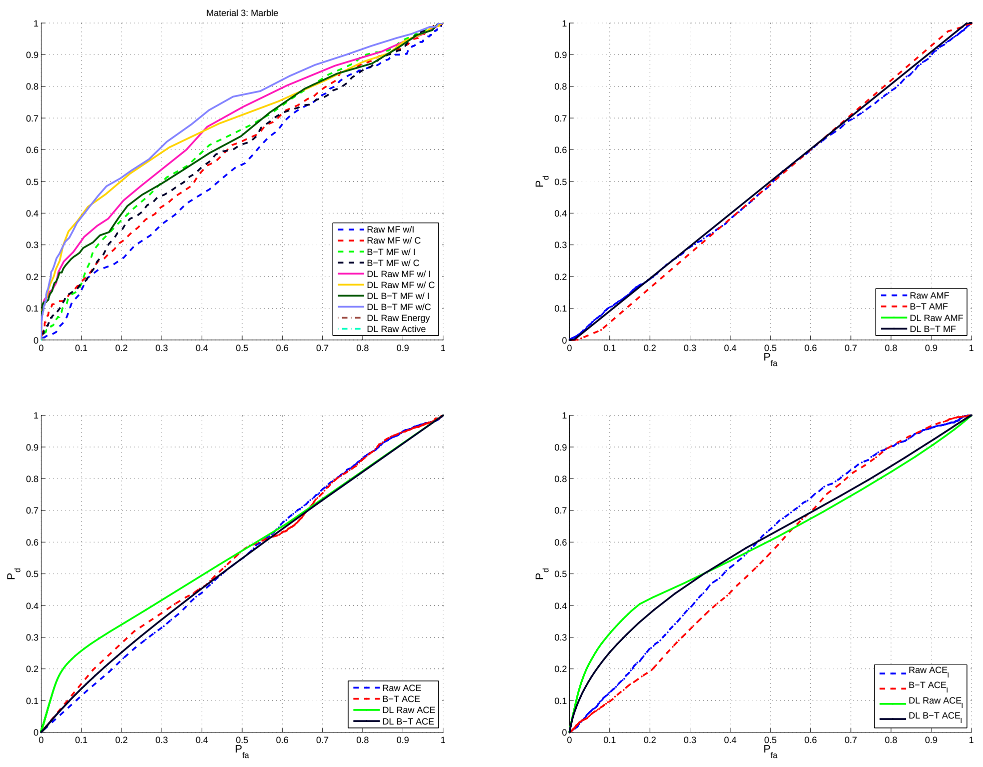


Fig. 6.5: Material 3: Marble.

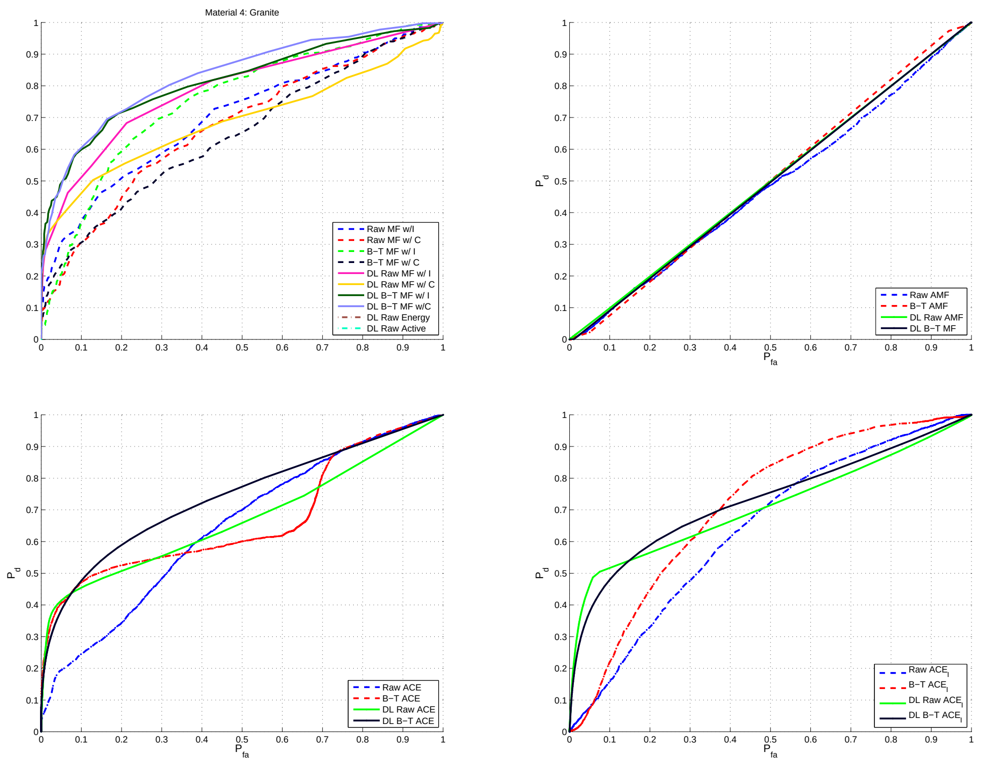


Fig. 6.6: Material 4: Granite.

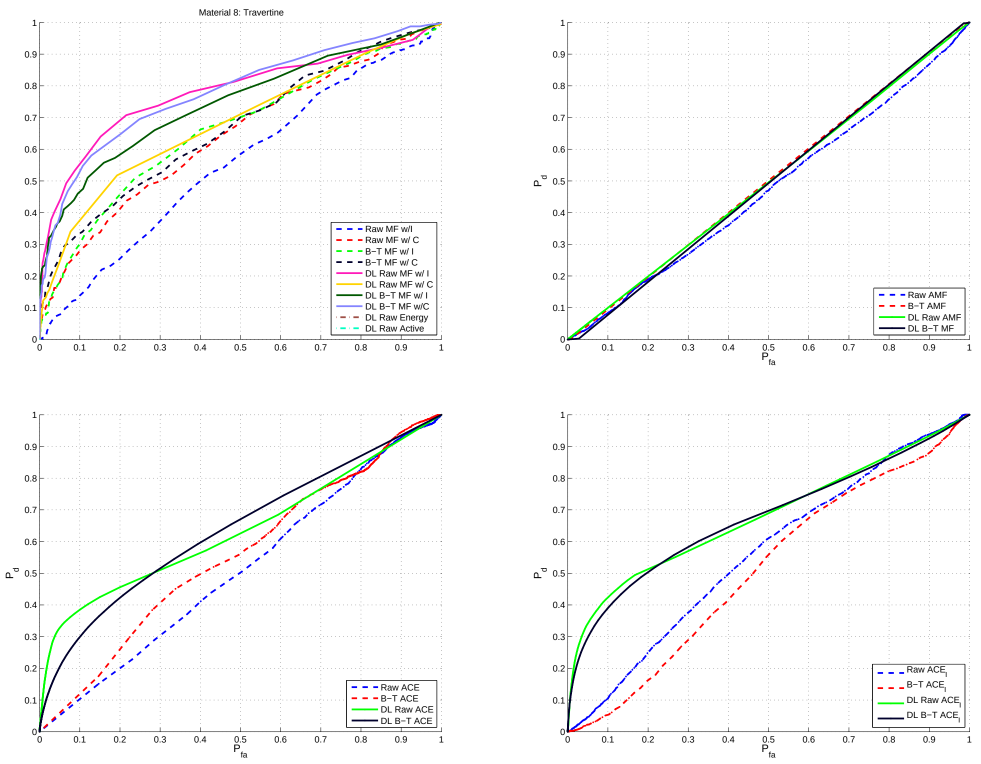


Fig. 6.7: Material 8: Travertine.

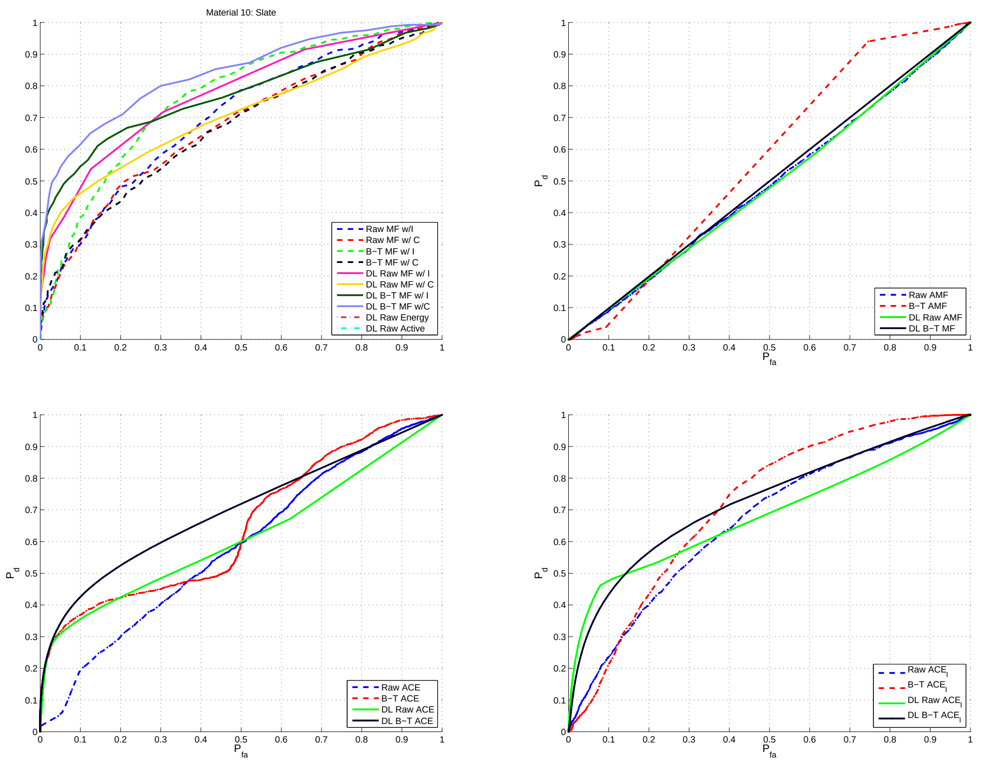


Fig. 6.8: Material 10: Slate.

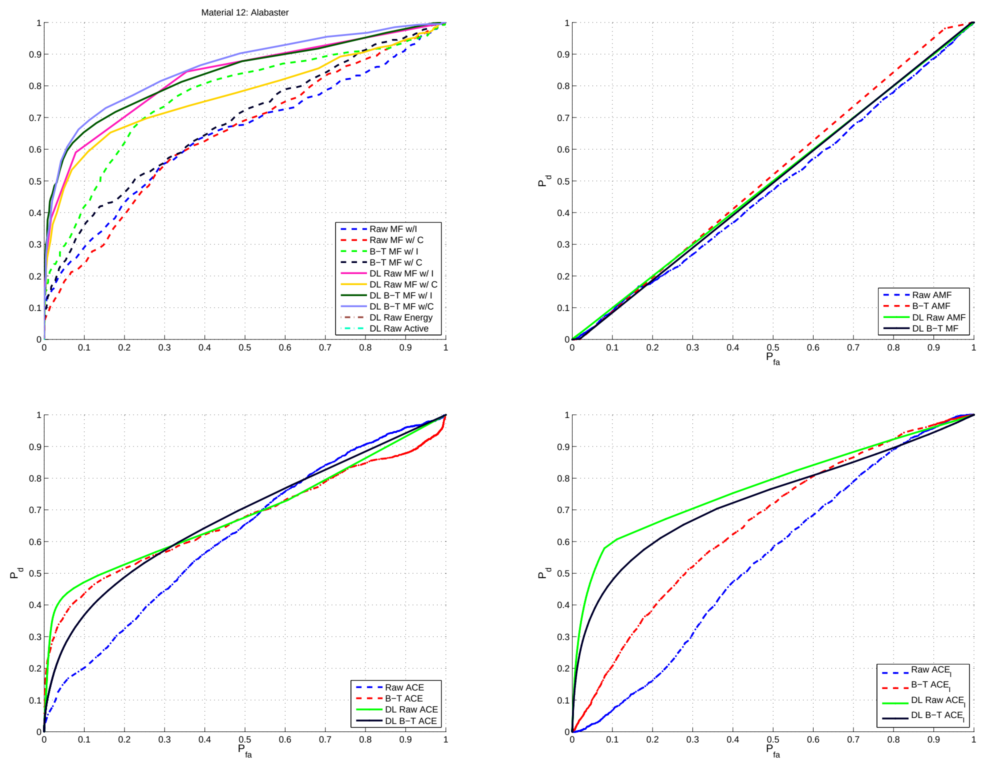


Fig. 6.9: Material 12: Alabaster.

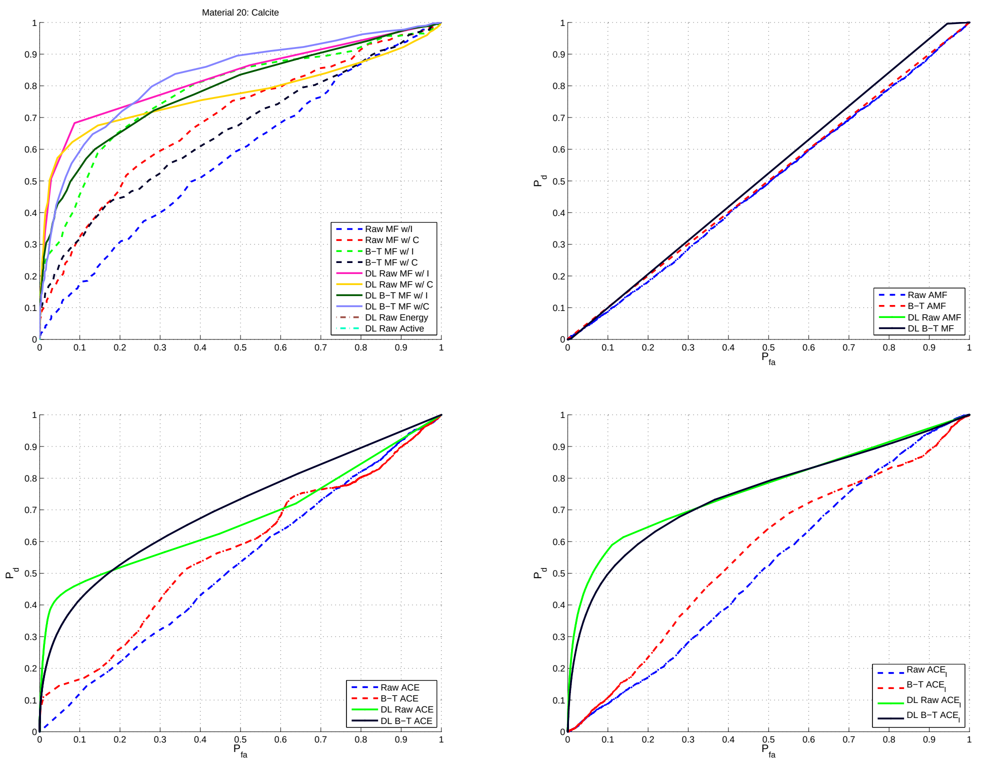


Fig. 6.10: Material 20: Calcite.

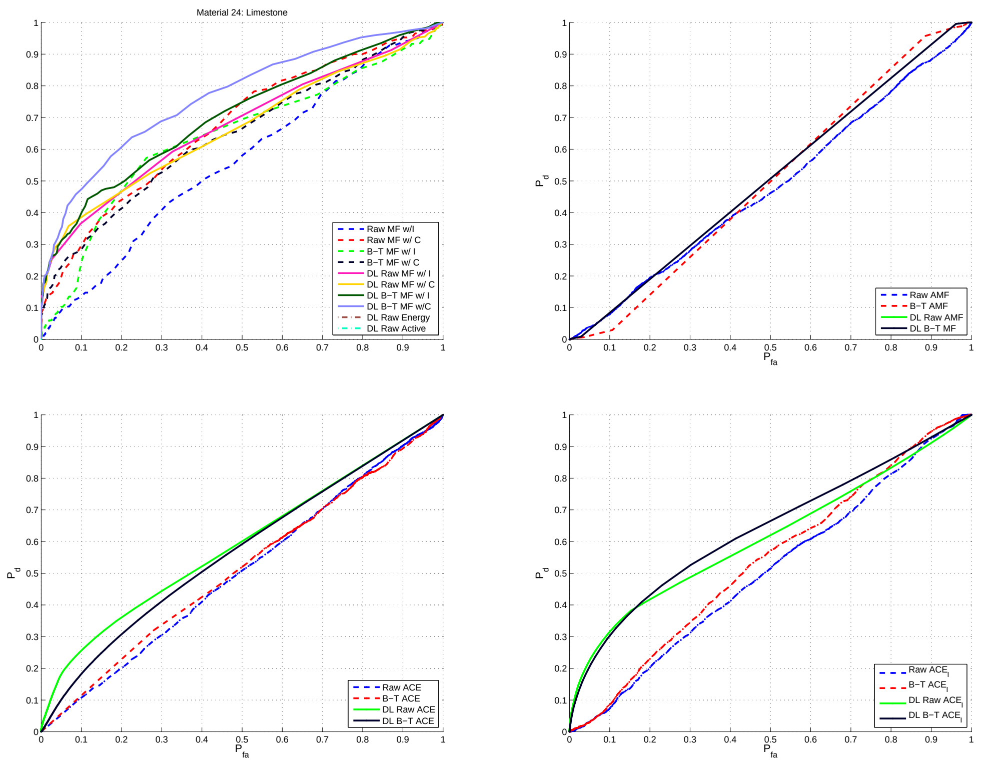


Fig. 6.11: Material 24: Limestone.

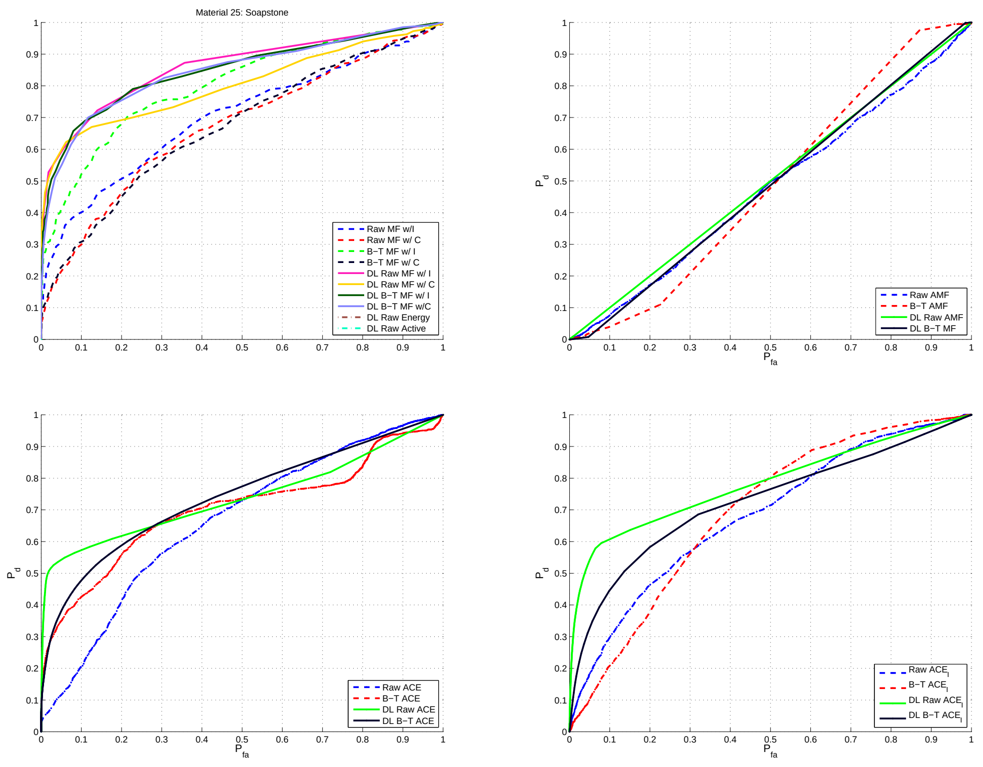


Fig. 6.12: Material 25: Soapstone.

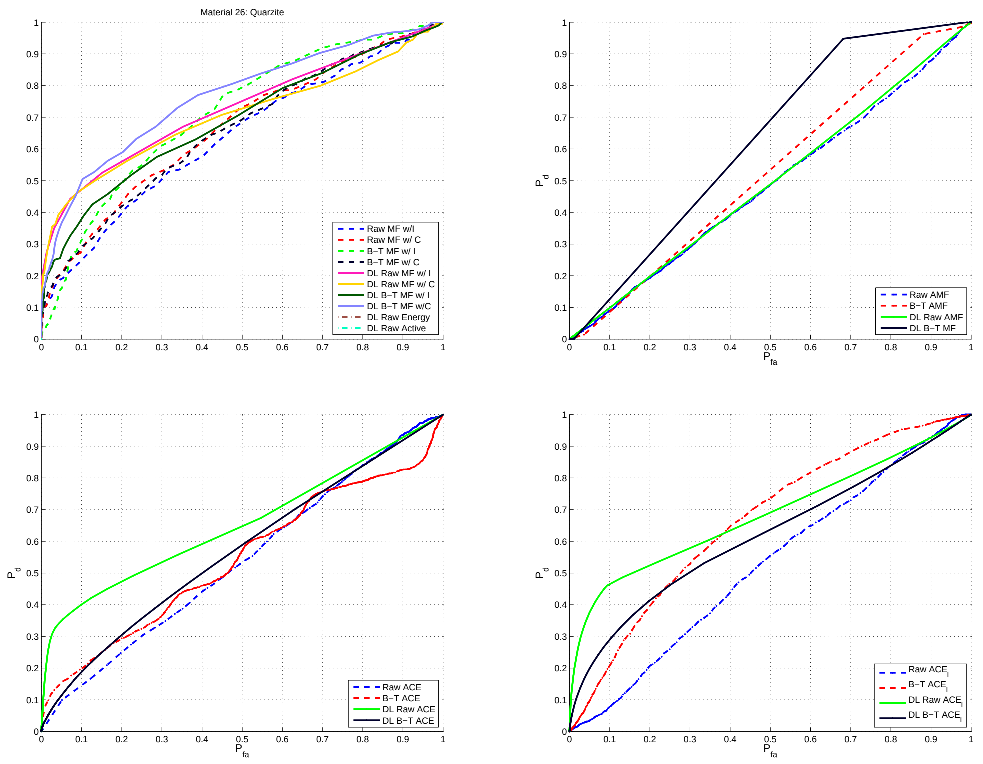


Fig. 6.13: Material 26: Quartzite.

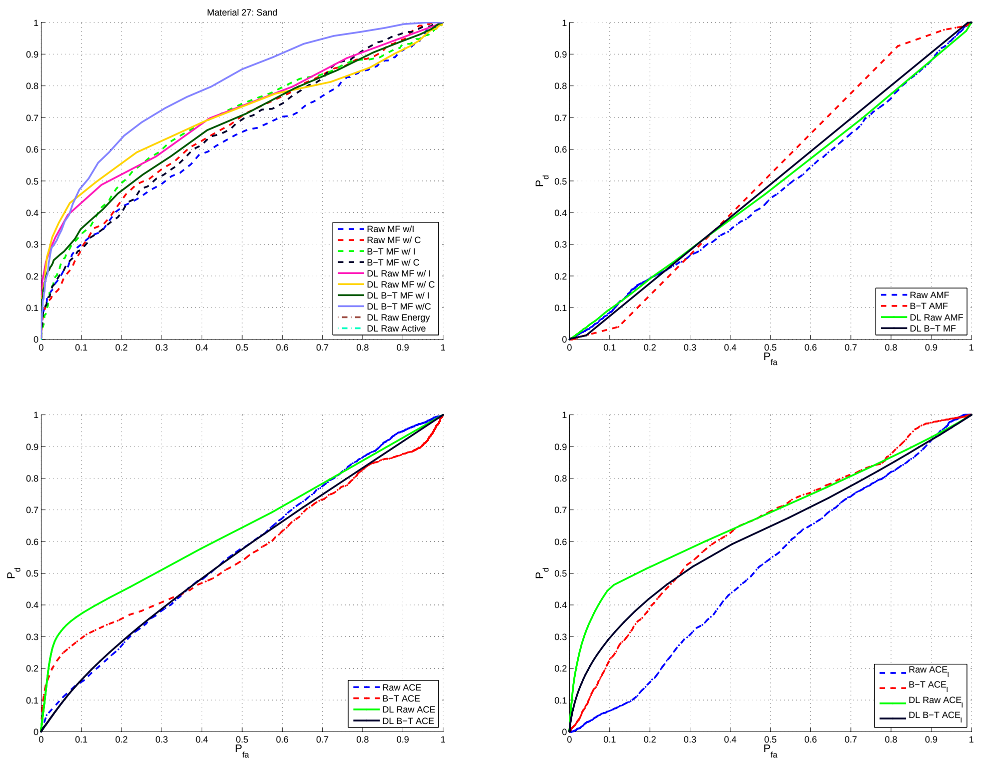


Fig. 6.14: Material 27: Sand.

6.5.3 Concrete/Asphalt/Clay

The man-made construction material types are shown in figs. 6.15 to 6.19. They detail the performance on cinder block (fig. 6.15), concrete (fig. 6.16), asphalt shingle (fig. 6.17), terra cotta (fig. 6.18), and aggregate (fig. 6.19). The interpretation is similar to that of painted steel above.

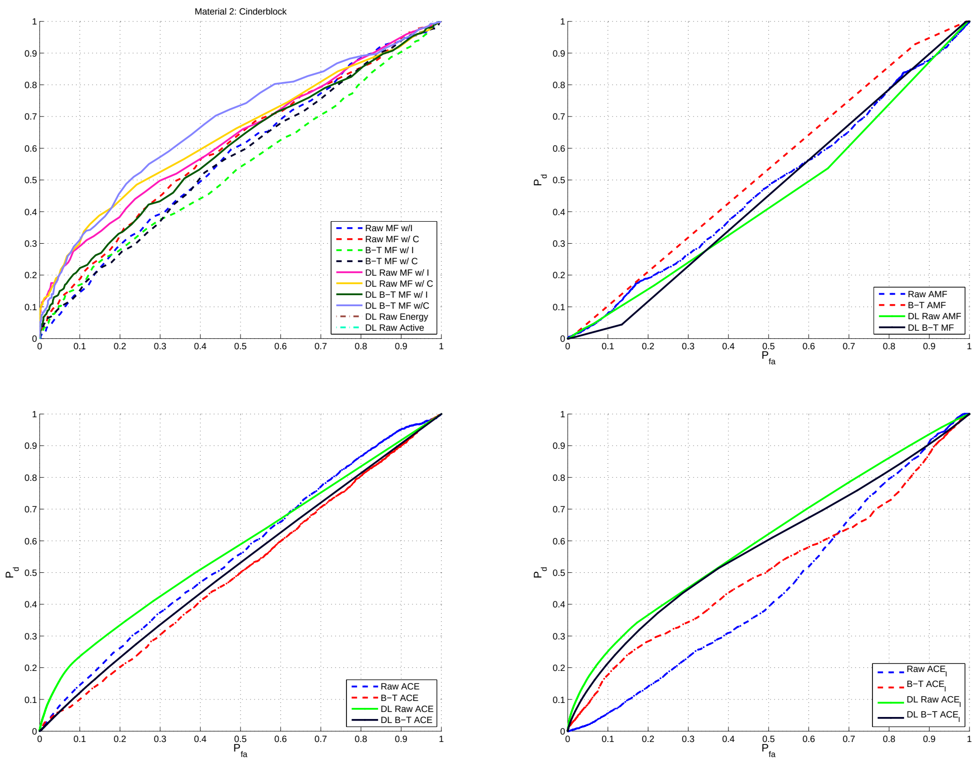


Fig. 6.15: Material 2: Cinderblock.

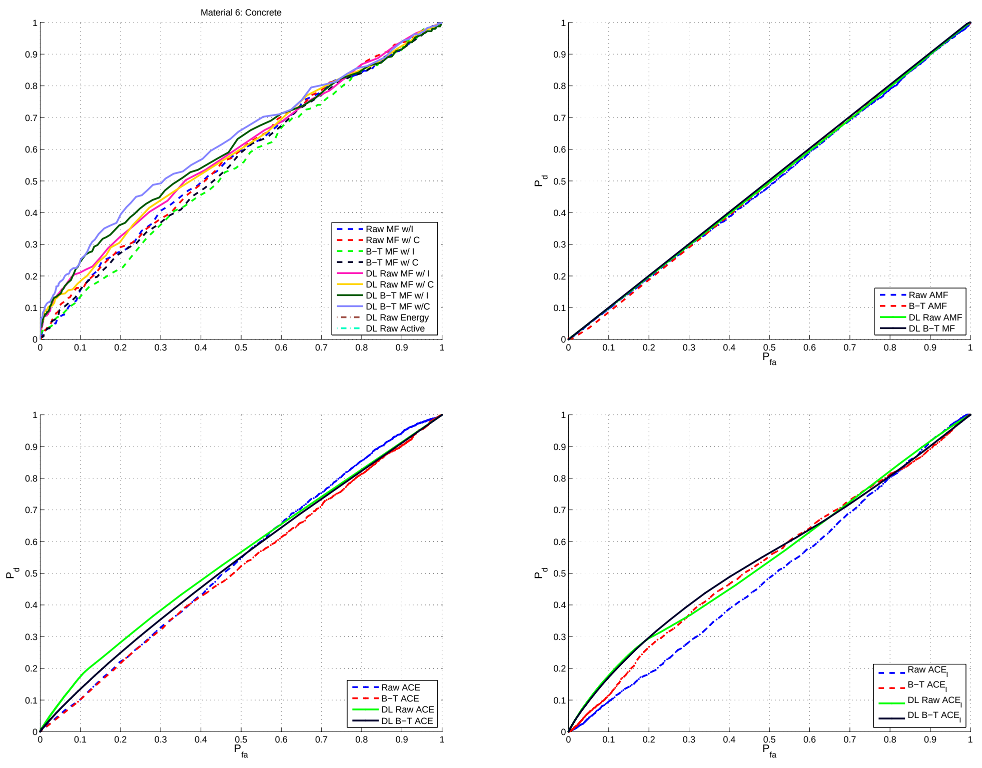


Fig. 6.16: Material 6: Concrete.

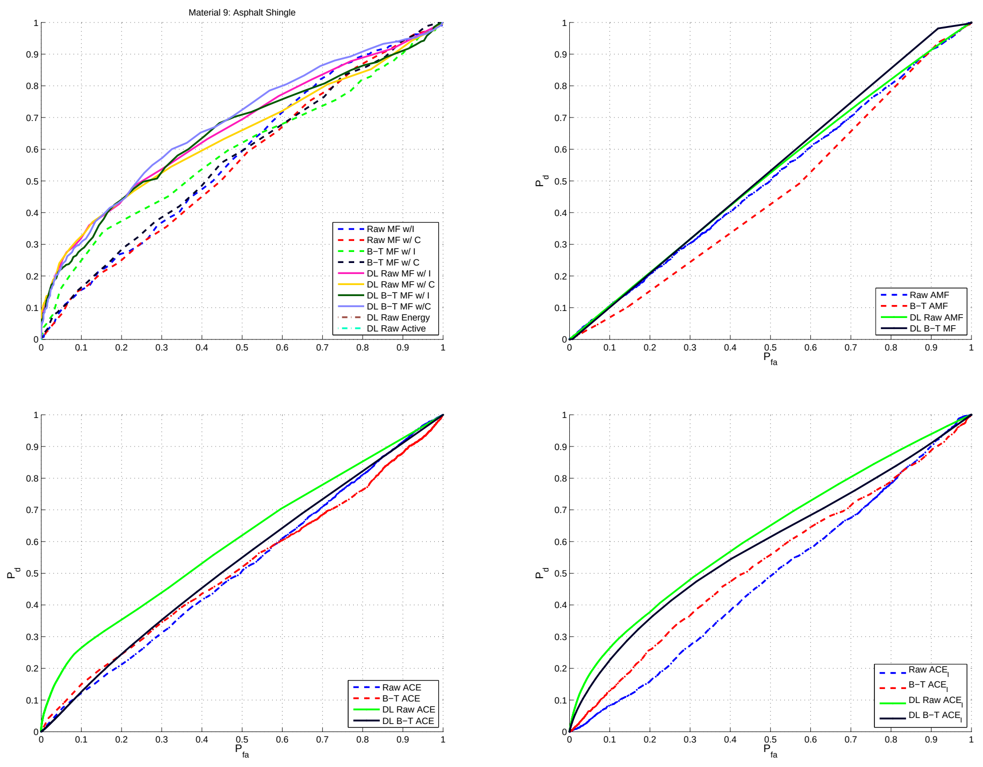


Fig. 6.17: Material 9: Asphalt Shingle.

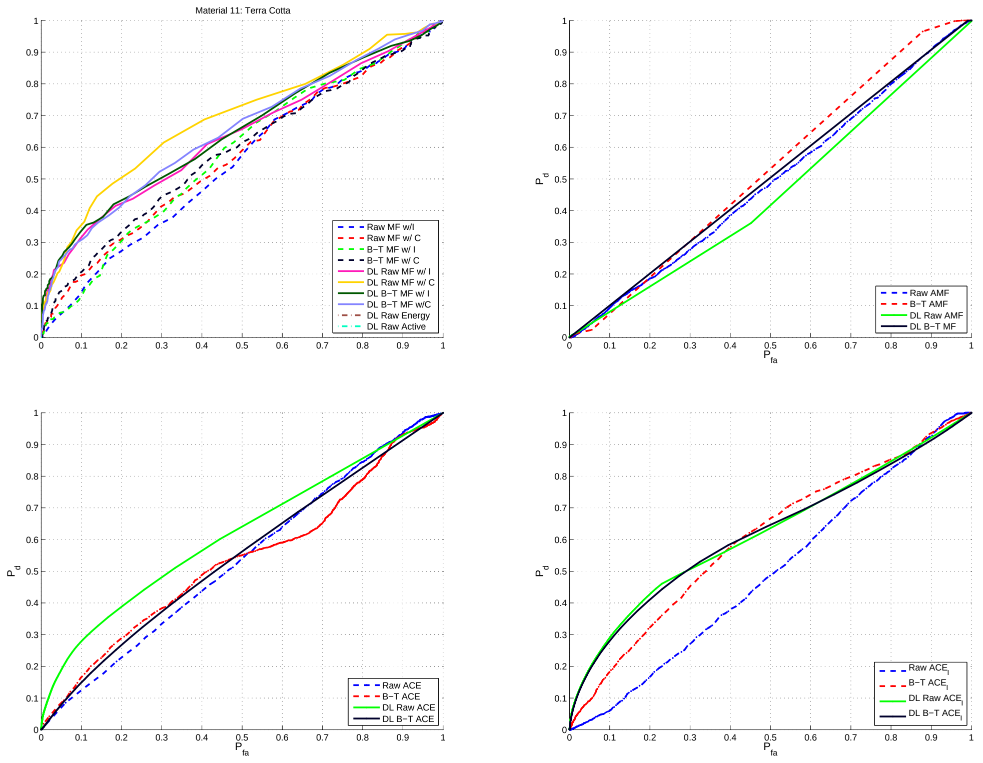


Fig. 6.18: Material 11: Terra Cotta.

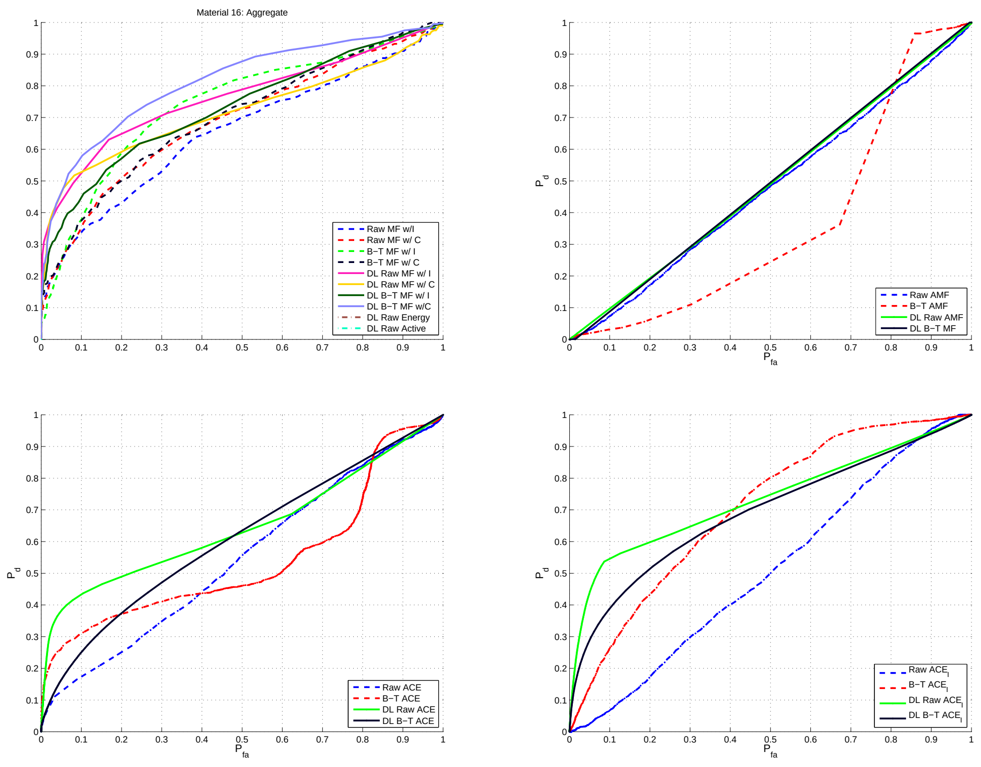


Fig. 6.19: Material 16: Aggregate.

6.5.4 Plastics

The man-made plastics material performance are shown in figs. 6.20 to 6.24. They detail the performance of PVC (fig. 6.20), HDPE (fig. 6.21), ABS (fig. 6.22), Tyvek (fig. 6.23), and blue tarp (fig. 6.24). The interpretation is similar to that of painted steel above.

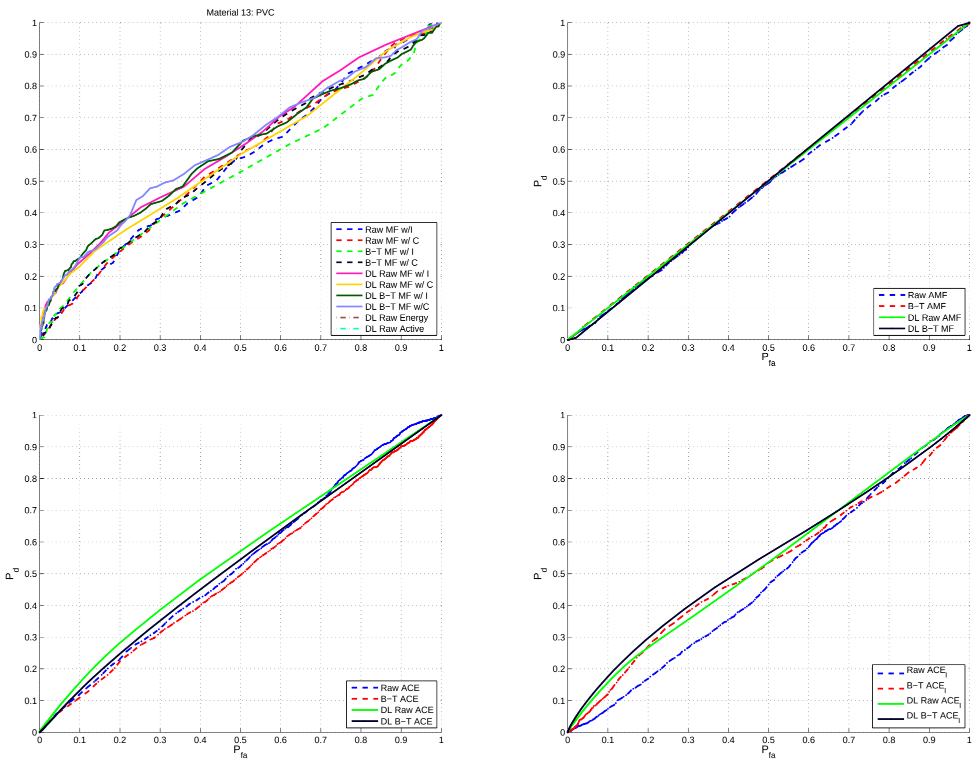


Fig. 6.20: Material 13: PVC.

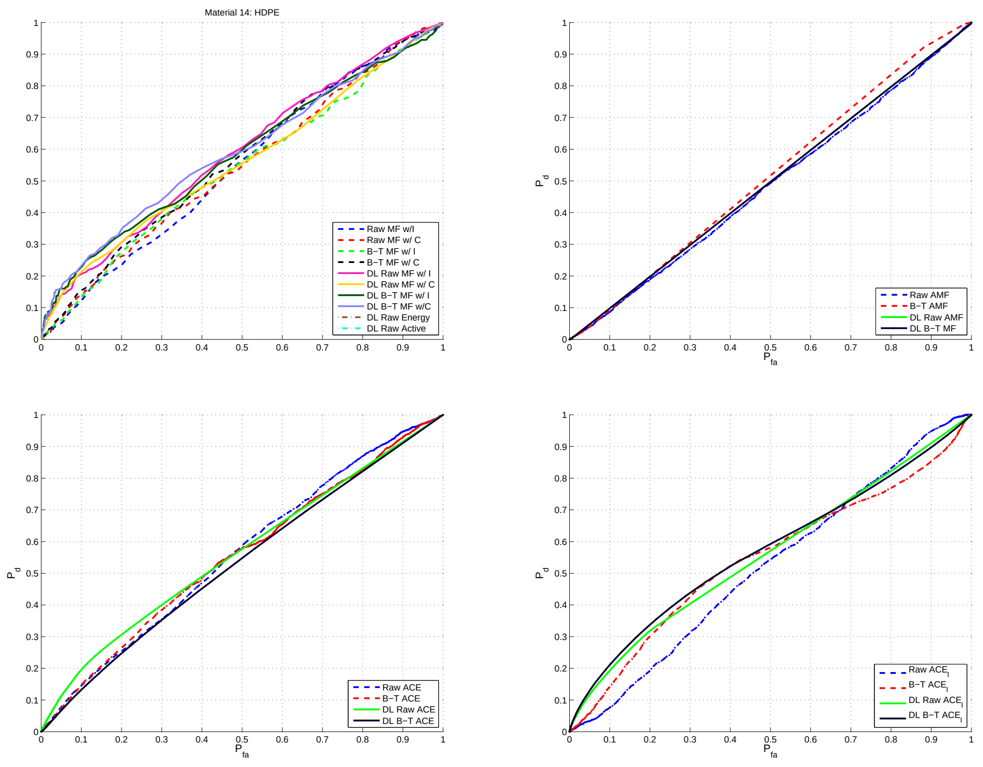


Fig. 6.21: Material 14: HDPE.

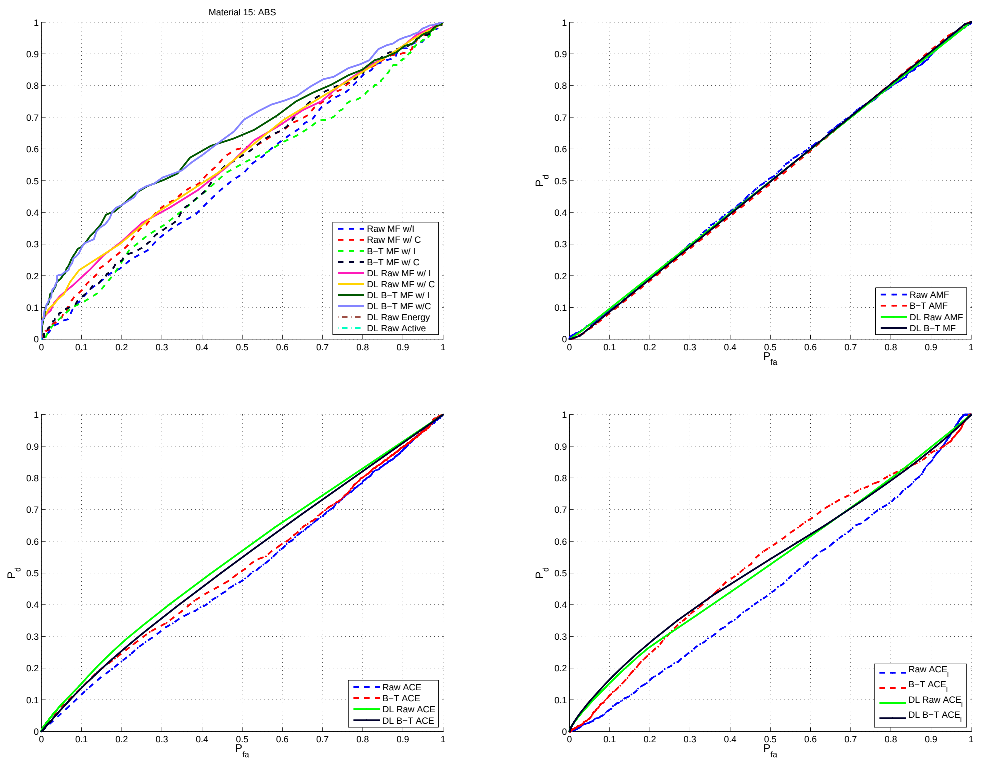


Fig. 6.22: Material 15: ABS.

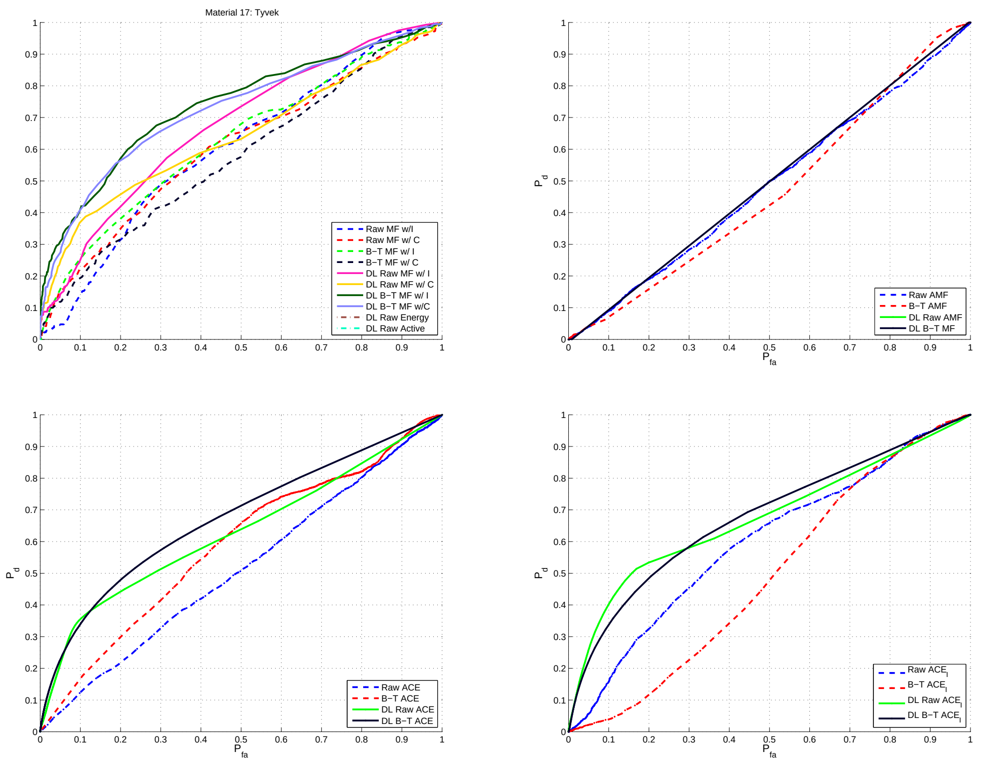


Fig. 6.23: Material 17: Tyvek.

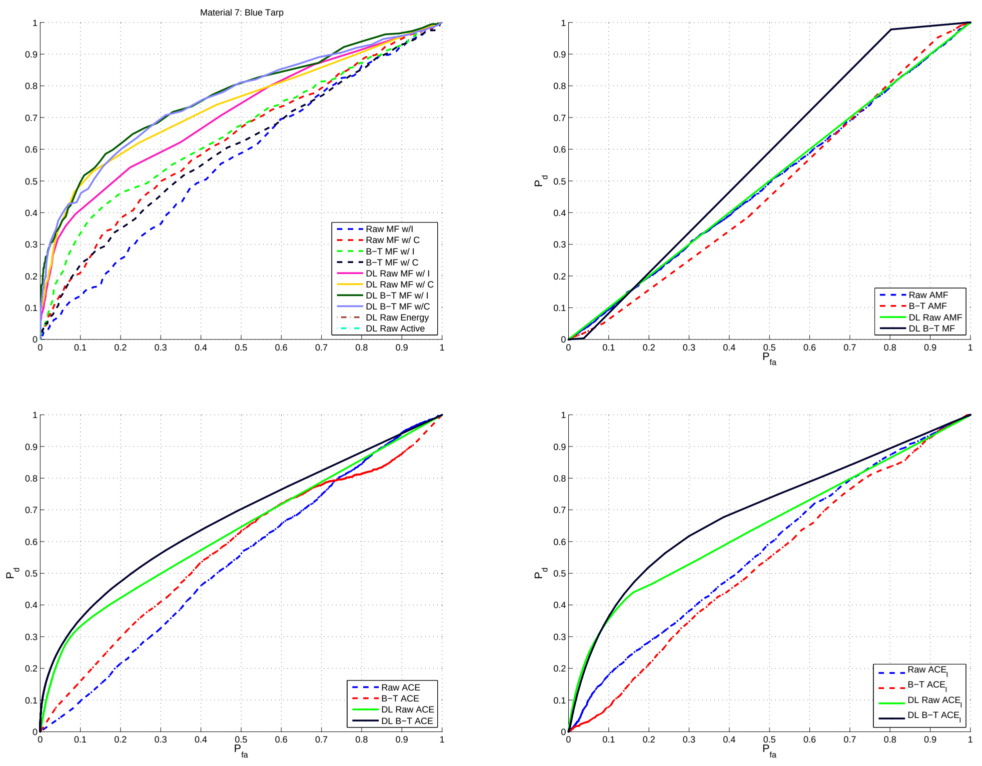


Fig. 6.24: Material 7: Blue Tarp.

6.5.5 Wood

The natural pine performance is shown in fig. 6.25. The interpretation is similar to that of painted steel above.

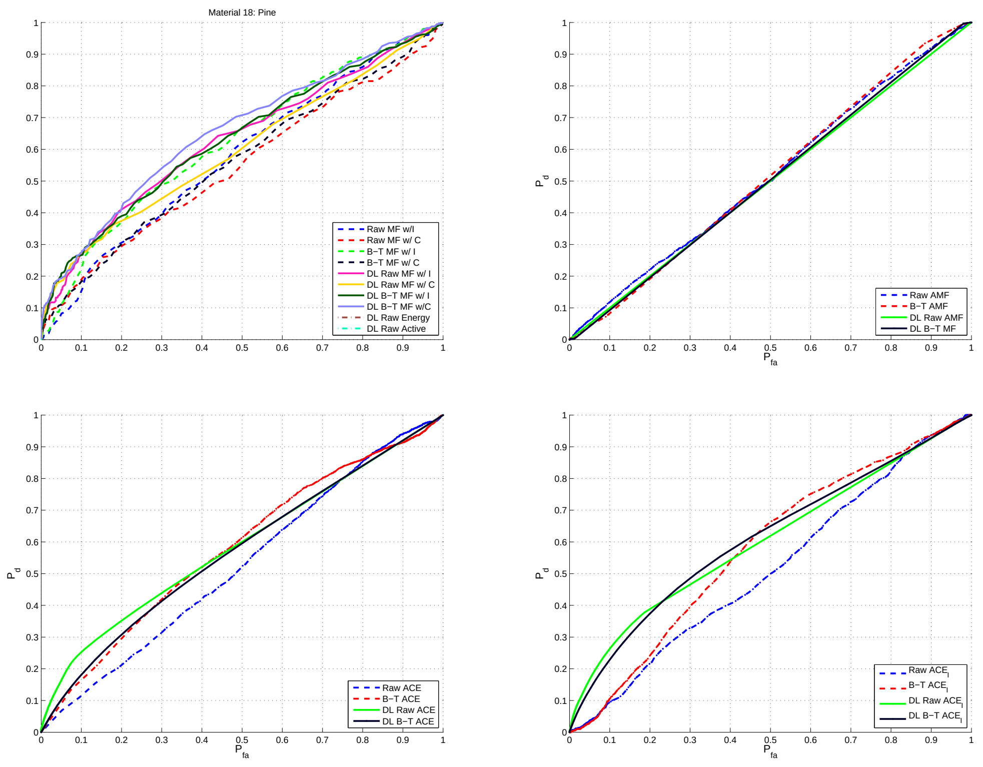


Fig. 6.25: Material 18: Pine.

6.5.6 Fabrics

The performance of the man-made fabrics is shown in figs. 6.26 to 6.27. Figure 6.26 shows the performance with camo net, whereas fig. 6.27 details performance of carbon fabric. The interpretation is similar to that of painted steel above.

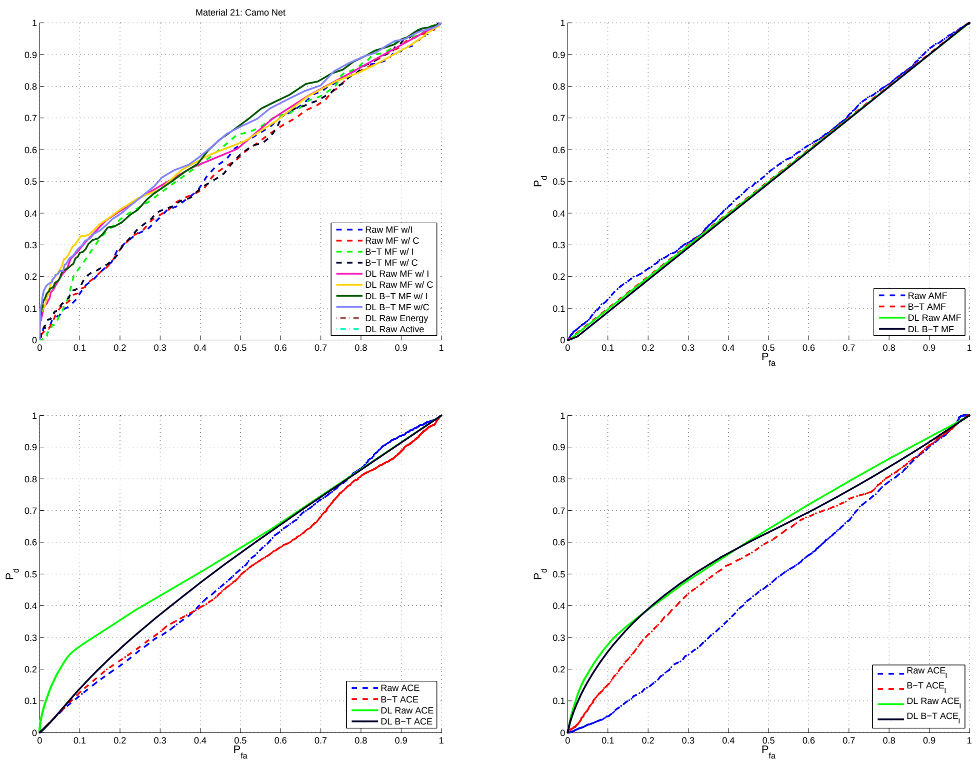


Fig. 6.26: Material 21: Camo Net.

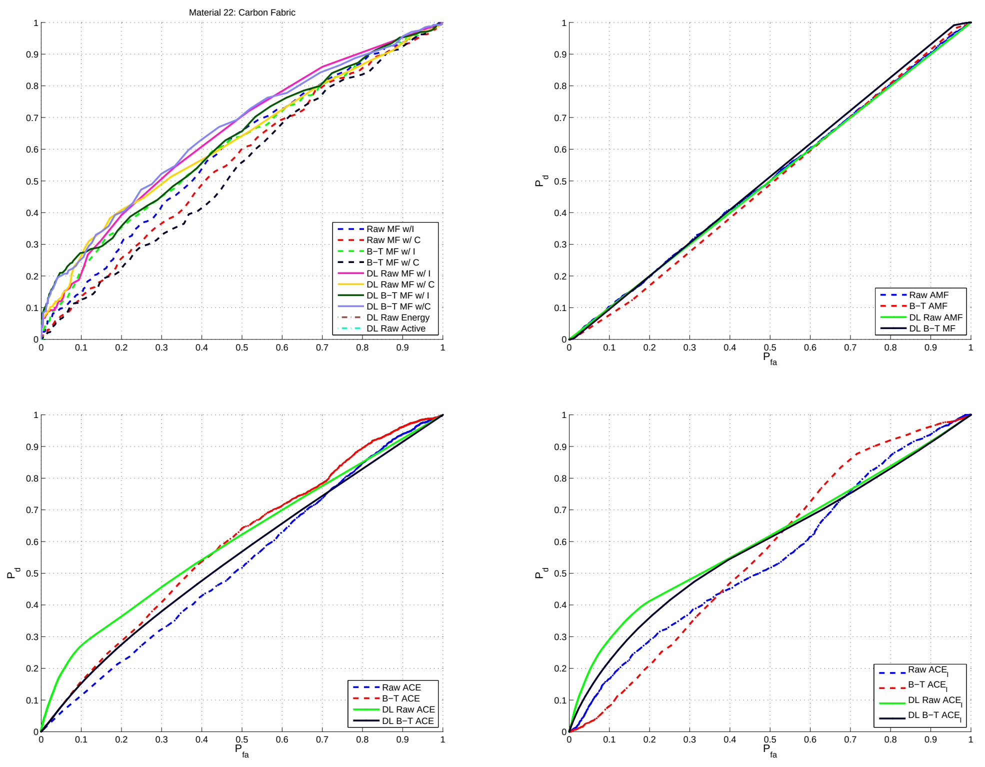


Fig. 6.27: Material 22: Carbon Fabric.

Chapter 7
Conclusions and Future Work

7.1 Discussion of Results

7.1.1 DL+Classification Results

The inability of the dictionary learning algorithm to provide RandomForests with the information necessary for the classification was an unforeseen result. The expectation that the dictionary learning could be useful because it closely paralleled the underlying physics of the hyperspectral model was definitely proved optimistic in the cases tried. Many possible things may be the cause of this. When analyzing these results the dictionaries and abundance maps show that the dictionaries tend to represent noise in the majority of the atoms. The atoms also tended to model well some of the imperfections in the data, such as the streaks and spikes that indicate bad pixels. It remains to be seen if these results would be the same if working on "cleaner" data with these imperfections.

7.1.2 Sparse Detection Results

While the classification results were disheartening, the sparse detection results were encouraging. The ability of the dictionary learning and self-taught learning framework to provide a basis for detection in hyperspectral data is very much a success. In addition, it fits into the realities of hyperspectral sensing, in that there exists a lot of unlabeled data, and few labeled examples of each class. Yet, by taking advantage of sparse representations and matched filtering, good performance gains were realized.

7.2 Future Directions

With the dictionary learning not adding any classification power to RandomForests, it raises the question: "Why not?" And a portion of that is definitely captured in the learning algorithms' rejection of important information, *i.e.*, leaving classifying information in the residuals, but there may be other problems that impair RandomForests from being able to utilize the abundances provided. One possibility is that the RandomForests implementation used is the stock Fortran code available from Breiman [76] and his colleague Adele Cutler. This code does not take into account that the data is very sparse, and because of this, it is very likely splitting nodes on variables that are all 0s. This may or may not affect the trees or forests, but maybe a sparsity aware version of RandomForests may be able to avoid this issue altogether.

In addition, the dictionaries could include class information in a number of different manners. As referred to previously, there exist dictionary learning algorithms that also incorporate a discrimination in the optimization problem. This seems to be the direction in which future research may be warranted. It may be that the newer, discriminating dictionary learning algorithms provide much better abundances that improve upon using just the raw radiances.

As for detection, different methods of building the dictionary can still be explored. A deeper investigation into how the properties (*i.e.*, the number of atoms, the learning algorithm, non-negativity, etc.) of the dictionary affect the performance of the detection algorithm is still a future topic of research. Also, a comparison could be made between this algorithm and that proposed by Nasrabadi [71] and the algorithm by Chen *et al.* [74].

7.3 Final Conclusion

Dictionary learning and sparse representations have many uses and have proven to be an effective tool for representing hyperspectral data. We have seen, though, that it is not always useful in all data processing chains. In particular, it is important to investigate whether the dictionary is representing HSI data in a manner that is useful for post-processing steps, especially where classification is the end product. Dictionary learning can provide a very useful representation for detection, when detection is the desired deliverable. This representation can lead to improved performance using classical matched filtering techniques, but really performs well when used in a self-taught learning framework.

References

[1] J. R. Schott, *Remote Sensing : The Image Chain Approach: The Image Chain Approach*, 2nd ed. New York: Oxford University Press, 2007.

[2] M. L. Clark, D. A. Roberts, and D. B. Clark, "Hyperspectral discrimination of tropical rain forest tree species at leaf to crown scales," *Remote Sensing of Environment*, vol. 96, no. 3, pp. 375–398, 2005.

[3] M. Govender, K. Chetty, and H. Bulcock, "A review of hyperspectral remote sensing and its application in vegetation and water resource studies," *Water Sa*, vol. 33, no. 2, 2007.

[4] A. Ghiyamat and H. Z. Shafri, "A review on hyperspectral remote sensing for homogeneous and heterogeneous forest biodiversity assessment," *International Journal of Remote Sensing*, vol. 31, no. 7, pp. 1837–1856, 2010.

[5] R. Anderson, W. Malila, R. Maxwell, and L. Reed, "Military utility of multispectral and hyperspectral sensors," Defense Technical Information Center Document, Tech. Rep., 1994.

[6] J.-P. Ardouin, J. Lévesque, and T. A. Rea, "A demonstration of hyperspectral image exploitation for military applications," in *2007 10th International Conference on Information Fusion*. IEEE, 2007.

[7] H. Xu and X.-j. Wang, "Applications of multispectral/hyperspectral imaging technologies in military," *Infrared and Laser Engineering*, vol. 36, no. 1, p. 13, 2007.

[8] E. M. Winter, M. J. Schlangen, A. P. Bowman, M. R. Carter, C. L. Bennett, D. J. Fields, W. D. Aimonetti, P. G. Lucey, J. Johnson, K. A. Horton, T. J. Williams, A. D. Stocker, A. Oshagan, A. T. DePersia, and C. J. Sayre, "Experiments to support the development of techniques for hyperspectral mine detection," in *Aerospace/Defense Sensing and Controls*. International Society for Optics and Photonics, 1996.

[9] F. A. Kruse, "Advances in hyperspectral remote sensing for geologic mapping and exploration," in *Proceedings 9th Australasian Remote Sensing Conference, Sydney, Australia*, 1998.

[10] R. G. Resmini, M. E. Kappus, W. S. Aldrich, J. C. Harsanyi, and M. Anderson, "Mineral mapping with HYperspectral Digital Imagery Collection Experiment (HYDICE) sensor data at Cuprite, Nevada, U.S.A." *International Journal of Remote Sensing*, vol. 18, no. 7, pp. 1553–1570, 1997.

[11] F. F. Sabins, "Remote sensing for mineral exploration," *Ore Geology Reviews*, vol. 14, no. 3, pp. 157–183, 1999.

[12] F. D. van der Meer, H. M. van der Werff, F. J. van Ruitenbeek, C. A. Hecker, W. H. Bakker, M. F. Noomen, M. van der Meijde, E. J. M. Carranza, J. B. de Smeth, and T. Woldai, "Multi- and hyperspectral geologic remote sensing: a review," *International Journal of Applied Earth Observation and Geoinformation*, vol. 14, no. 1, pp. 112–128, 2012.

[13] F.-p. GAN and R.-s. WANG, "The application of the hyperspectral imaging technique to geological investigation," *Remote Sensing for Land & Resources*, vol. 4, p. 014, 2007.

[14] M. E. Martin, M. B. Wabuyele, M. Panjehpour, M. N. Phan, B. F. Overholt, and T. Vo-Dinh, "Hyperspectral fluorescence imaging system for biomedical diagnostics," in *Biomedical Optics 2006*. International Society for Optics and Photonics, 2006.

[15] F. Blanco, M. Lpez-Mesas, S. Serranti, G. Bonifazi, J. Havel, and M. Valiente, "Hyperspectral imaging based method for fast characterization of kidney stone types," *Journal of Biomedical Optics*, vol. 17, no. 7, 2012.

[16] J. W. Uhr, M. L. Huebschman, E. P. Frenkel, N. L. Lane, R. Ashfaq, H. Liu, D. R. Rana, L. Cheng, A. T. Lin, G. A. Hughes, X. J. Zhang, and H. R. Garner, "Molecular profiling of individual tumor cells by hyperspectral microscopic imaging," *Translational Research*, vol. 159, no. 5, pp. 366–375, 2012.

[17] B. S. Sorg, B. J. Moeller, O. Donovan, Y. Cao, and M. W. Dewhirst, "Hyperspectral imaging of hemoglobin saturation in tumor microvasculature and tumor hypoxia development," *Journal of Biomedical Optics*, vol. 10, no. 4, 2005.

[18] S. V. Panasyuk, S. Yang, D. V. Faller, D. Ngo, R. A. Lew, J. E. Freeman, and A. E. Rogers, "Medical hyperspectral imaging to facilitate residual tumor identification during surgery," *Cancer Biology & Therapy*, vol. 6, no. 3, pp. 439–446, 2007.

[19] H. Akbari, L. V. Halig, H. Zhang, D. Wang, Z. G. Chen, and B. Fei, "Detection of cancer metastasis using a novel macroscopic hyperspectral method," in *SPIE Medical Imaging*. International Society for Optics and Photonics, 2012.

[20] J. Freeman, F. Downs, L. Marcucci, E. Lewis, B. Blume, and J. Rish, "Multispectral and hyperspectral imaging: applications for medical and surgical diagnostics," in *Engineering in Medicine and Biology Society, 1997. Proceedings of the 19th Annual International Conference of the IEEE*, vol. 2, 1997.

[21] Q. Li, X. He, Y. Wang, H. Liu, D. Xu, and F. Guo, "Review of spectral imaging technology in biomedical engineering: achievements and challenges," *Journal of biomedical optics*, vol. 18, no. 10, p. 100901, 2013.

[22] G. Lu and B. Fei, "Medical hyperspectral imaging: a review," *Journal of Biomedical Optics*, vol. 19, no. 1, p. 010901, 2014.

[23] M. E. Winter, "N-FINDR: An algorithm for fast autonomous spectral end-member determination in hyperspectral data," in *Proceedings of the SPIE*, M. R. Descour and S. S. Shen, Eds., vol. 3753, no. 1. SPIE, 1999.

[24] J. B. Tenenbaum, V. d. Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[25] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[26] J. D. Bayliss, J. A. Gualtieri, and R. F. Cromp, “Analyzing hyperspectral data with independent component analysis,” *Proceedinds of the SPIE*, vol. 3240, pp. 133–143, 1998.

[27] M. R. Stites, “Utilization of a Probabilistic Model for Improved Hyperspectral Unmixing,” Ph.D. dissertation, Utah State University, Logan, UT, 2011.

[28] T. Hastie, R. Tibshirani, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, 2nd ed. New York: Springer-Verlag, 2008.

[29] C. Rodarmel and J. Shan, “Principal component analysis for hyperspectral image classification,” *Surveying and Land Information Science*, vol. 62, no. 2, pp. 115–122, 2002.

[30] E. Esser, M. Möller, S. Osher, G. Sapiro, and J. Xin, “A convex model for non-negative matrix factorization and dimensionality reduction on physical space,” *ArXiv e-prints*, vol. Statistics:Machine Learning, p. 14, Feb. 2011.

[31] N. Wang, B. Du, and L. Zhang, “An endmember dissimilarity constrained non-negative matrix factorization method for hyperspectral unmixing,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 6, no. 2, pp. 554–569, 2013.

[32] J. Lawson, J. Conger, and L. Balick, “Investigation of temporal variation of lwir solid signatures using denali,” Lawrence Livermore National Laboratory, Los Alamos National Laboratory, Tech. Rep. LLNL-TR-403504, May 2008.

[33] R. Leathers and T. Downes, “Scene-based nonuniformity correction and bad-pixel identification for hyperspectral vnir/swir sensors,” in *IEEE International Conference on Geoscience and Remote Sensing Symposium*, Jul. 2006.

[34] D. G. Manolakis, D. Marden, J. P. Kerekes, and G. A. Shaw, “Statistics of hyperspectral imaging data,” *Proceedings of the SPIE*, vol. 4381, pp. 308–316, 2001.

[35] A. D. Fischer, T. V. Downes, and R. Leathers, “Median spectral-spatial bad pixel identification and replacement for hyperspectral swir sensors,” *Proceedings of the SPIE*, vol. 6565, 2007.

[36] H. H. Kieffer, “Detection and correction of bad pixels in hyperspectral sensors,” *Proceedings of the SPIE*, vol. 2821, pp. 93–108, 1996.

[37] Merriam-Webster, *Merriam-Webster’s Collegiate Dictionary*, 10th ed. Merriam-Webster, 2000.

[38] K. Crammer and Y. Singer, “On the algorithmic implementation of multiclass kernel-based vector machines,” *Journal of Machine Learning Research*, vol. 2, pp. 265–292, Mar. 2002.

[39] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *Journal of Artificial Intelligence Research*, vol. 2, pp. 263–286, 1995.

[40] Y. Shiraishi and K. Fukumizu, "Statistical approaches to combining binary classifiers for multi-class classification," *Neurocomputing*, vol. 74, no. 5, pp. 680–688, 2011.

[41] L. Breiman, "Bagging predictors," in *Machine Learning*, 1996.

[42] A. Criminisi, J. Shotton, and E. Konukoglu, "Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning," *Foundations and Trends in Computer Graphics and Vision*, vol. 7, no. 2–3, pp. 81–227, Feb. 2012.

[43] S. Bidhendi, A. Shirazi, N. Fotoohi, and M. Ebadzadeh, "Material classification of hyperspectral images using unsupervised fuzzy clustering methods," in *Third International IEEE Conference on Signal-Image Technologies and Internet-Based System*, Dec. 2007.

[44] B.-C. Kuo, J.-M. Yang, T.-W. Sheu, and S.-W. Yang, "Kernel-based knn and gaussian classifiers for hyperspectral image classification," in *IEEE International Geoscience and Remote Sensing Symposium*, vol. 2, Jul. 2008.

[45] S. Wang, C.-M. Wang, M.-L. Chang, C.-T. Tsai, and C.-I. Chang, "Applications of kalman filtering to single hyperspectral signature analysis," *IEEE Sensors Journal*, vol. 10, no. 3, pp. 547–563, Mar. 2010.

[46] Q. Du and S. Chakrarvarty, "Unsupervised hyperspectral image classification using blind source separation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, Apr. 2003.

[47] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, no. 8, pp. 1778–1790, Aug. 2004.

[48] M. Lennon, G. Mercier, and L. Hubert-Moy, "Classification of hyperspectral images with nonlinear filtering and support vector machines," in *2002 IEEE International Geoscience and Remote Sensing Symposium*, vol. 3, Jun. 2002.

[49] Y. Tarabalka, M. Fauvel, J. Chanussot, and J. Benediktsson, "Svm- and mrf-based method for accurate classification of hyperspectral images," *IEEE Geoscience and Remote Sensing Letters*, vol. 7, no. 4, pp. 736–740, Oct. 2010.

[50] M. Fauvel, J. Chanussot, J. Benediktsson, and J. Sveinsson, "Spectral and spatial classification of hyperspectral data using svms and morphological profiles," in *IEEE International Geoscience and Remote Sensing Symposium, 2007*, Jul. 2007.

[51] B. Demir and S. Erturk, "Hyperspectral image classification using relevance vector machines," *IEEE Geoscience and Remote Sensing Letters*, vol. 4, no. 4, pp. 586–590, Oct. 2007.

[52] J. Ham, Y. Chen, M. Crawford, and J. Ghosh, “Investigation of the random forest framework for classification of hyperspectral data,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 3, pp. 492–501, Mar. 2005.

[53] S. Joelsson, J. Benediktsson, and J. Sveinsson, “Random forest classifiers for hyperspectral data,” in *2005 IEEE International Geoscience and Remote Sensing Symposium Proceedings*, vol. 1, Jul. 2005.

[54] I. Tosic and P. Frossard, “Dictionary learning,” *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, Mar. 2011.

[55] B. A. Olshausen and D. J. Field, “Sparse coding with an overcomplete basis set: A strategy employed by V1?” *Vision Research*, vol. 37, no. 23, pp. 3311–3325, Dec. 1997.

[56] M. Aharon, M. Elad, and A. Bruckstein, “K-svd: an algorithm for designing overcomplete dictionaries for sparse representation,” *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[57] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, “Online learning for matrix factorization and sparse coding,” *The Journal of Machine Learning Research*, vol. 11, pp. 1–43, Aug. 2009.

[58] A. S. Charles, B. A. Olshausen, and C. J. Rozell, “Learning sparse codes for hyperspectral imagery,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 5, pp. 963–978, Sep. 2011.

[59] A. Charles, B. Olshausen, and C. J. Rozell, *Sparse coding for spectral signatures in hyperspectral images.* IEEE, Nov. 2010.

[60] N. Gillis and R. Plemmons, “Sparse nonnegative matrix underapproximation and its application to hyperspectral image analysis,” in *3rd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS), 2011*, Jun. 2011.

[61] M. Möller, E. Esser, S. Osher, G. Sapiro, and J. Xin, “A convex model for matrix factorization and dimensionality reduction on physical space and its application to blind hyperspectral unmixing,” pp. 1–6, 2010.

[62] M.-d. Iordache, J. M. Bioucas-Dias, and A. Plaza, “Sparse unmixing of hyperspectral data,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 6, pp. 2014–2039, Jun. 2011.

[63] Z. Xing, M. Zhou, A. Castrodad, G. Sapiro, and L. Carin, “Dictionary learning for noisy and incomplete hyperspectral images,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. submitted, pp. 1–29, 2011.

[64] J. Mairal, F. Bach, and J. Ponce, “Task-driven dictionary learning,” *ArXiv e-prints*, Sep. 2010.

[65] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, “Robust face recognition via sparse representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[66] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 4, pp. 791–804, Apr. 2012.

[67] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Supervised dictionary learning," p. 15, 2008.

[68] Q. Zhang and B. Li, "Discriminative K-SVD for dictionary learning in face recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2010.

[69] A. Castrodad, Z. Xing, J. Greer, E. Bosch, L. Carin, and G. Sapiro, "Discriminative sparse representations in hyperspectral imagery," in *2010 IEEE International Conference on Image Processing.* IEEE, Sep. 2010.

[70] R. Rubinstein, M. Zibulevsky, and M. Elad, "Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit," 2008.

[71] N. Nasrabadi, "Hyperspectral target detection : An overview of current and future challenges," *IEEE Signal Processing Magazine*, vol. 31, no. 1, pp. 34–44, Jan. 2014.

[72] D. Manolakis, R. Lockwood, T. Cooley, and J. Jacobson, "Is there a best hyperspectral detection algorithm?" *Proceedings of the SPIE*, vol. 7334, pp. 733 402–733 402–16, 2009.

[73] J. Theiler and B. Foy, "Ec-glrt: Detecting weak plumes in non-gaussian hyperspectral clutter using an elliptically-contoured generalized likelihood ratio test," in *IEEE International Geoscience and Remote Sensing Symposium*, vol. 1, Jul. 2008.

[74] Y. Chen, N. M. Nasrabadi, and T. D. Tran, "Sparse representation for target detection in hyperspectral imagery," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 3, pp. 629–640, Jun. 2011.

[75] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proceedings of the 24th International Conference on Machine Learning*, ser. ICML '07. New York: ACM, 2007.

[76] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.