Algorithms to Improve Analysis and Classification for Small Data

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

William Franz Lamberti
Master of Science
George Mason University, 2017
Bachelor of Science
The College of New Jersey, 2015

Director: Dr. Jason Kinser, Professor
Department of Computational and Data Sciences

Fall Semester 2020
George Mason University
Fairfax, VA

Dedication

I dedicate this dissertation to my Mother and Father, to whom I owe everything.

Acknowledgments

I would like to thank the following people and organizations who made this research possible.

The funding for my research came from George Mason University's Office of the Provost in the form of the Presidential Scholar Fellowship. Some of these experiments were run on ARGO, a research computing cluster provided by the Office of Research Computing at George Mason University, VA (`http://orc.gmu.edu`).

I am extremely grateful to Dr. Jason Kinser, my advisor and committee chair. He saw many of the early ideas I had for the thesis. He helped me to hone them into a contribution worthy of being called a thesis. Further, his continual guidance and expertise throughout the research process was invaluable.

I would also like to express my sincere thanks to the other members of my committee. Dr. William G. Kennedy, Dr. Michael Eagle, and Dr. David I. Holmes each provided a unique perspective and insight to help improve my thesis. They all helped me to push the boundaries of my thesis's contributions to the scientific community.

Josef Lamberti, my brother, has been instrumental as my editor and writing tutor. His efforts have improved my writing to communicate my messages more effectively.

John Schuler and Jon Murphy both provided a valuable sound board for many of my ideas in my thesis. Even though they are from the Economics Department, our experiences as graduate students were similar in many respects. Our fellowship has helped me through my time as a graduate student.

Karen Underwood and Natalie Lapidot-Croitoru, our department's Academic Programs Manager and Human Resources/Fiscal Services Specialist, respectively, both helped me navigate numerous questions, forms, and unexpected problems. Their assistance was vitally important to have a smoother experience as a graduate student.

In a very special way, I would like to thank Dr. Elizabeth Johnson and Dr. David Holmes. They both initially welcomed me to the Fairfax area when I started my graduate career at George Mason University. Their hospitality and kindness continued throughout my time in Fairfax and helped make the graduate experience significantly more enjoyable.

My high school statistics teacher from Bethlehem Catholic High School, Mrs. Janice Little, inspired me to pursue a career related to Statistics and Data Science. I am indebted to her contribution to my professional career path.

At the time of this writing, our world is undergoing the COVID-19 crisis. Due to COVID-19, I decided to make an unexpected and quick move back to my immediate family's house in New Jersey. I am currently living with all of my siblings and parents during this unique time. My younger siblings, Josef Lamberti, Emma Tilly Lamberti, and Vincent Lamberti, and my parents, Caroline Lamberti and William Anthony Lamberti, have helped me throughout my graduate career more than they will ever know. However, during the COVID-19 crisis, they have been particularly supportive as my graduate studies have come

to a close. Their continual love has made this trying time significantly easier. While I am looking forward to the end of COVID-19, I will miss their daily presence in my life.

Table of Contents

List of Tables

List of Figures

Abstract

ALGORITHMS TO IMPROVE ANALYSIS AND CLASSIFICATION FOR SMALL DATA
William Franz Lamberti, PhD
George Mason University, 2020
Dissertation Director: Dr. Jason Kinser

Binary 2D images can be analyzed with an image operator algorithm based on theoretical shape proportions and encircled image-histograms (SPEIs). These images can be classified with an algorithm that reduces complex classification problems into a series of simpler ones using decision trees with automatic model generation (DAMG). DAMG provides exceptional classification rates for small data problems when using the results of SPIEs as variables alongside other shape metrics. SPEIs describe shapes using two metrics: shape proportions (SPs) and encircled image histograms (EIs). SP and EI values are useful for classifying and describing shapes. SPEI-based approaches outperform convolutional neural networks in small data problems, where data is limited. DAMG converts any multinomial classification problem into a series of elegant binary classification problems. DAMG is particularly effective in small data scenarios as it is able to convert imbalanced problems into balanced problems. The developed SPEIs and DAMG tools are applied to the global issue of pill shape classification. The final models produced outperform current approaches and are more easily interpreted than many statistical or machine learning algorithms. Further, SPEI and DAMG are applied to a vareity of different data sets to show their wide applicability.

Chapter 1: General Introduction

Many of the image analysis metrics and classification algorithms mentioned in Chapter 1 lack any combination of simplicity, interpretability, or competitiveness. These shortcomings are amplified when the data analyzed is complex or nuanced.

1.1 Shape Metrics Shortcomings

The current shape metrics are unable to accurately discriminate different classes of shapes in a meaningful manner. Each of the metrics provided failed in some manner, as discussed in Chapter 1. Basic metrics like area and perimeter are simple, but they are not powerful enough to provide discriminative measures. This issue is catalyzed for analyzing shapes with differing scales. Circularity provides no meaningful interpretation for the resulting value, and eigenvalues could not discriminate regular polygons with varying scales. Thus, the Image Analysis and Computer Vision communities need a universal shape metric that is intuitive, competitive, and interpretable.

1.2 Classification Algorithms Shortcomings

Many algorithms in the Machine Learning (ML), or Statistical Learning (SL), community have a theoretical basis based on binary outcomes, as discussed in Chapter 1 [28,34,35]. The implementation of these algorithms also require binary outcomes. For instance, the SVM algorithm must convert multinomial outcomes into a series of binary ones [28].

Deep learning approaches are intrinsically complex and do not provide meaningful interpretations [23]. Convolutional neural networks (CNNs) are extremely powerful, but require a large amount of data [23]. Their popularity in the Computer Vision community has consumed researchers' attention and slowed the development of alternative techniques.

Evaluating these algorithms in the case of a multinomial outcome increases the complexity of their interpretations. Choosing a metric which meaningfully describes the model's performance and has a lucid interpretation is vital. This issue is compounded when dealing with small data problems, especially imbalanced data. However, if we are able to convert multinomial and imbalanced data problems into a series of binary and balanced data problems, many of these issues can be circumvented.

1.3 Hypotheses

I am seeking to answer the following research question: is it possible to outperform CNNs by using shape and classification algorithms when data is limited? Four hypotheses expand the research question of this thesis. These hypotheses will use acronyms for the following words: shape proportions and encircled image-histogram (SPEI), shape proportion (SP), encircled image-histogram (EI), and decision tree with automatic model generation (DAMG).

1. Through the use of SPEIs, 2D binary digital image shapes of circles and regular polygons have unique SP values.

2. If classes of digital image shapes have unique and empirically distinct SP values, then the classes' EIs can outperform CNNs in small data scenarios.

3. Digital pill shapes have distinct class shapes which can be effectively discriminated using classification algorithms.

4. Multinomial classification small data problems are convertible into a series of simpler balanced binary classification problems.

The results regarding hypotheses 1 and 2 will be discussed in Chapter 4. Chapter 4 provides a new algorithm we developed called SPEI. SPEI produces both SP and EI values. I will show that SL algorithms using only EIs will outperform CNNs in small data scenarios. The metric used for evaluation is overall accuracy.

The models in Chapters 5 and 6 will provide evidence in favor of hypothesis 3. Chapter 5's model will have human influences, while Chapter 6's model will be completely machine-driven. The metrics used for evaluation of the models are overall and interclass accuracy.

Chapter 6's algorithm, DAMG, will be a general machine-driven solution to multinomial classification problems in the form of a recursive tree algorithm. The DAMG algorithm also converts imbalanced data problems into a series of balanced data problems. This alleviates the issues associated with imbalanced data.

Chapter 2: Review of the Literature

Humanity has been interested in describing shapes for a substantial amount of time. For example, the ancient Greeks have described shapes in terms of the area and perimeter for 2D shapes and surface area for 3D shapes [12]. In modern times, researchers investigate how shapes exist in a variety of ways such as as 2D digital binary images. Substantial work has been done in the classification and analysis of shapes. However, there is little understanding on how given shape metrics exist for a given 2D digital binary image. This prevents the specification of useful and helpful metrics in a variety of areas such as classification of image groups. Furthermore, the classification of shapes is exasperated when data is limited.

In the following sections, we will first discuss various shape metrics from the literature. Then, we will mention various classification algorithms in the machine or statistical learning and deep learning communities. We end with a discussion of classification problems where there are a small number of observations or instances.

2.1 Shape Metrics

We desire to understand of how our abstractions exist in the world by using shape metrics to describe them is desired. For example, area and perimeter are basic metrics which describe shapes, but they are not discriminative metrics [38]. Some useful metrics for the classification of shapes are circularity and eigenvalues.

2.1.1 Circularity

Circularity, γ, is defined as follows:

$$\gamma = \frac{T^2}{4\pi A}, \tag{2.1}$$

where T is the perimeter of the shape and A is the area [38]. Note that others have defined γ slightly differently, but the definition presented here is essentially the same. For example, some do not have the 4π constant in the denominator [68]. Circularity appears to have been first utilized by Rosenfeld [68]. However, it has been studied in mathematics under the name of the ispoerimetric quotient [72]. Further, research into the ispoerimetric inequality appears to be an active area of research [100]. The discussion herein will focus on the application of the ispoerimetric quotient, which will be referred to as circularity, on digital images.

Circularity has the important feature of having a unique value for circles and regular polygons. For a circle, this value is 1. For a regular polygon with n sides, this value is $\frac{n\tan(\frac{\pi}{n})}{\pi}$. This hypothetically allows for the easy classification of digital shapes for these classes. Table 2.1 shows some of the first few circularity values for regular polygons with n sides. While this fraction is established, the complete proof for the value for regular polygons cannot be found in the literature. Thus, it is provided below where T is the perimeter of the shape, A is the area, n is the number of sides composing the regular polygon, o is the base of the $2n$ right triangles composing the shape, a is the height of the $2n$ right triangles, $s = 2o$ is the length of a side of the regular polygon, and r is the radius of the circle which encompasses the regular polygon.

Proof: $\gamma = \frac{n\tan(\frac{\pi}{n})}{\pi}$

$$\gamma = \frac{T^2}{4\pi A} \tag{2.2}$$

$$A = 2n\left(\frac{1}{2}oa\right) \tag{2.3}$$

$$= 2n\left(\frac{1}{2}\left(r\sin\left(\frac{\pi}{n}\right)\right)\left(r\cos\left(\frac{\pi}{n}\right)\right)\right) \tag{2.4}$$

$$= nr^2\sin\left(\frac{\pi}{n}\right)\cos\left(\frac{\pi}{n}\right) \tag{2.5}$$

$$T = ns \tag{2.6}$$

$$= n\left(2o\right) \tag{2.7}$$

$$= 2nr\sin\left(\frac{\pi}{n}\right) \tag{2.8}$$

$$\implies \gamma = \frac{\left(2nr\sin\left(\frac{\pi}{n}\right)\right)^2}{4\pi\left(nr^2\sin\left(\frac{\pi}{n}\right)\cos\left(\frac{\pi}{n}\right)\right)} \tag{2.9}$$

$$= \frac{n\sin\left(\frac{\pi}{n}\right)}{\pi\cos\left(\frac{\pi}{n}\right)} \tag{2.10}$$

$$= \frac{n\tan\left(\frac{\pi}{n}\right)}{\pi} \quad \blacksquare \tag{2.11}$$

The reader might need further clarification to understand the values of o and a. We will use Figure 2.1 to aid in our explanation. For any regular polygon, it can be divided into n triangles. Each triangle can be divided into 2 right triangles. Thus, the regular polygon can be divided into $2n$ right triangles where the base is o and the height is a. The hypotenuse is known to be r. The top angle is $\frac{2\pi}{2n} = \frac{\pi}{n}$. This is derived from the fact that these angles of the right triangles all meet at the center of the triangle. Thus, the summation of all these angles must be 2π. Since they are all partitioned equally and there are $2n$ partitions, each angle will be $\frac{\pi}{n}$. Thus, using trigonometric identities, one can deduce the exact values for o

Table 2.1: Table for the first 8 values of circularity for regular polygons where $n = 3, ..., 10$ are presented.

n	γ
3	1.65
4	1.27
5	1.15
6	1.10
7	1.07
8	1.05
9	1.04
10	1.03

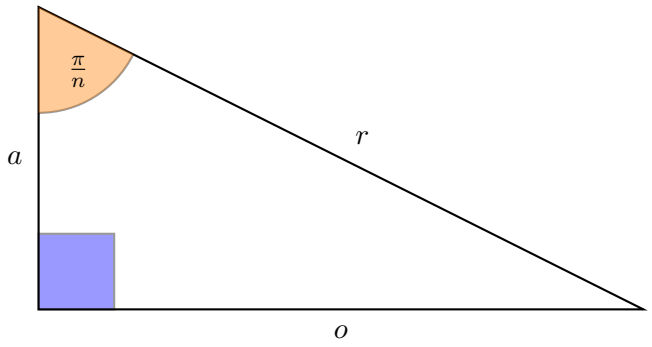


Figure 2.1: Figure of one of the $2n$ right triangles that compose a regular n sided polygon.

and a.

The use of circularity on regular polygons allows for the classification of those polygons. For a given unknown shape, the known circularity regular polygon value that it is closest to is a reasonable guess for its class. However, the inference provided beyond this is limited. For example, circularity does not provide guidance for what to classify a shape with a value of 1.125. It may very well be the case that it is reasonable to observe pentagons with a circularity value of 1.125, however, the metric of circularity is not equipped to answer this question.

Here is an example to make the previous paragraph explicitly clear: suppose the shape

to be analyzed for classification is Figure 2.2. The potential classes are determined to be regular polygons with $n \in \{3, 4, ..., 8\}$, the number of sides, and a circle. This example shape can easily be determined to be a reasonable representation of a square despite having somewhat jagged edges. The total number of pixels, or resolution, of the image is 200×200. Before any metrics are collected, the theoretical value for this shape should be close to 1.27, as seen in Table 2.1. Further, if the observed value does deviate from the theoretical value, there is no basis for the reasonableness of that value. After calculating the area and perimeter of the square in Figure 2.2, the circularity value was determined to be about 1.06. While the difference between the theoretical value and observed value is small, relative to this problem this is a serious issue. If the deviation was disregarded and the shape was classified as one of the potential shapes, it would be reasonable to classify the shape as a circle as that is the closest theoretical circularity value the shape provided. Even at a practical level, circularity is misleading as the theoretical values can provide inaccurate results in applications.

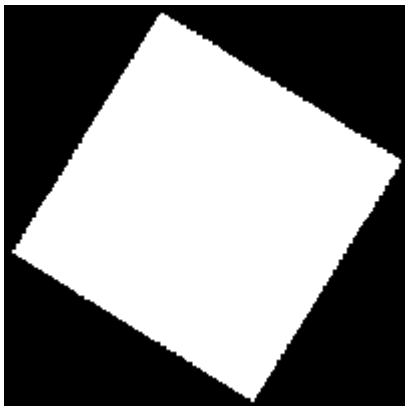


Figure 2.2: Figure shows an example of a created square.

Lastly, circularity does not provide any suggestions on what the exact circularity value is for shapes that does not have a mathematically derived unique value. For example, circularity provides no guidance on what the value should be for a medical pill capsule.

2.1.2 Eigenvalues for Shapes

Shapes can also be described using the eigenvalues of a shape [38]. This approach is particularly useful when the shape has an ill defined perimeter or when the perimeter cannot be accurately determined. Eigenvalues have been used in a variety of problems such as classifying pole like structures like traffic signs and tress from 3D street data [18].

The calculation of the eigenvalues is straightforward. In short, the covariance matrix of a 2D digital shape is calculated. This translates to finding the shapes x and y coordinates and saving them in a matrix. The covariance is calculated on this $2 \times q$ matrix, where q is the number of pairs. From the resulting covariance matrix, the eignevalues are obtained. These two eigenvalues will describe the shape succinctly.

Eigenvalues have some benefits. They are invariant to orientation. This invariation to orientation theoretically allows for the eigenvalues to be used for classification purposes. Further, eigenvalues are used in many fields and are more familiar to technical audiences unfamiliar with image analysis.

However, there are some downsides to using eigenvalues. As we will soon show, the resulting eigenvalues are not always useful for classification. Additionally, the resulting eigenvalues do not have a simple approach for interpretation.

We will collect the eigenvalues of created polygon data. I describe the process of the polygon creation in Section 3.1.1. I will attempt to use these eigenvalues for classification purposes on the six different classes. There are 125 observations per class.

Figure 2.3 summarizes the resulting first two eigenvalues of each of the created regular polygons. As indicated by the plot, there is no clear separation between the classes. All of the observations overlap one another. Thus, eigenvalues are not useful for classification purposes on the created regular polygon data.

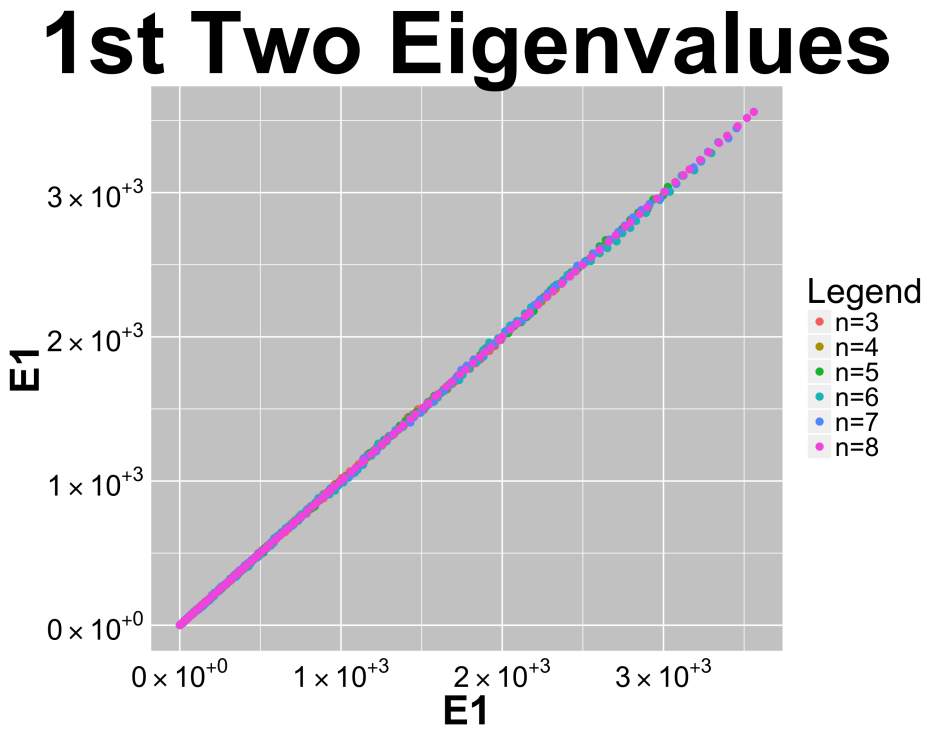


Figure 2.3: Figure showcases the resulting eigenvalues of the regular polygon images. This plot shows that there is no clear separation using the first two eigenvalues.

2.2 Classification Algorithms and Models

Classification problems have motivated the creation of various algorithms or methods for analytically describing and discriminating classes from one another. The Computer Vision community has been very active with developing and using deep learning methods [23]. However, there are a plethora of classification modeling algorithms from the machine or statistical learning communities [28, 35]. Both schools of thought provide valuable tools for attempting to solve many problems encountered in the world today.

2.2.1 Machine or Statistical Learning

Machine or statistical learning, ML or SL for short, is a disciple shared by the fields of Statistics and Computer Science. SL provides a variety of tools to model phenomena from data. This data can take a variety of forms and perform a number of different tasks, but I

will focus on those related to classification problems. Further, I will analyze those capable of modeling 2D image shape data.

An intrinsic assumption of these approaches is that they all need metrics. In other words, they cannot evaluate the image directly. Metrics or features must first be extracted from the image and then fed into the algorithm for analysis.

ML differs from the field of Statistics in that the analyst is not necessarily concerned with statistical significance. In fact, he or she is more concerned with the performance of the model in accurately predicting the phenomena of interest. In other words, the goal of Statistics is to infer the influence of certain variables on an outcome. The goal of SL is to find the variables which predict an outcome effectively, regardless if they are statistically significant or not.

Logistic Regression

Logistic regression, LR, is a type of generalized linear model [28,31,35,55]. The number of tools available for evaluation is large [31]. I will focus on the fundamentals which help to distinguish LR from other classification methods. Let X is a $n \times p$ matrix of n observations and p variables, Y is a $n \times 1$ vector whose contents are $j \in \{1, 2, ..., k\}$ where each j is a label for each unique class, and f is the function which models Y and X where $Y = f(X)$. The model has the form

$$\log \frac{P(C = q|X = x)}{P(C = K|X = x)} = \beta_q^T X, \tag{2.12}$$

where $q \in \{1, 2, ..., K - 1\}$. This series of $K - 1$ equations can be solved via maximum likelihood and the Newton-Raphson algorithm to estimate the parameters of the model [28].

Naïve Bayes

Bayes' Thereom is the foundation for numerous algorithms and techniques [20]. However, I will limit ourselves to naïve Bayes due to its popularity in the literature [28]. Borrowing and inspired by the notation from Laskey and Martignon [45], Bayes theorem is

$$P(D_j|E) = \frac{P(E|D_j)P(D_j)}{\sum_{i=1}^{K} P(D_k)P(E|D_k)} \tag{2.13}$$

where $k \in \{1, 2, ..., K\}$ are the classes, $P(D_j)$ is the probability of belonging to the j^{th} class or the prior, E is the evidence, and $P(E|D_j)$ is the probability of certain evidence given belonging to class j. To obtain naïve Bayes, assume that, all of the evidence is independent conditional on the given class. Thus, Equation 2.13 is now

$$P(D_j|E_1, ..., E_p) = \frac{P(D_j)\prod_{a=1}^{p} P(E_k|D_j)}{\sum_{i=1}^{K} P(D_k)\prod_{k=1}^{p} P(E_k|D_k)} \tag{2.14}$$

where $a \in \{1, 2, ..., p\}$ are the variables. Thus, a given observation belongs to class j if

$$\text{argmax}_{k\in\{1,...,p\}} P(D_j|E_1, ..., E_p). \tag{2.15}$$

Linear and Quadratic Discriminant Analysis

Linear discriminant Analysis, LDA, and Quadratic Dscriminant Analysis, QDA, are based on Bayes theorem. Using the notation and logic from Hastie *et al.*, let $g_k(x)$ be the class-conditional density of X in class $C = j$, and let π_k be the prior probability of class k, with $\sum_{k=1}^{K} \pi_k = 1$ [28]. Using Bayes theorem provides

$$P(C = k|X = x) = \frac{g_k(x)\pi_k}{\sum_{l=1}^{K} g_l(x)\pi_l}. \tag{2.16}$$

Assume that each class has a multivariate Gaussian density. LDA comes about when it is assumed that all of the classes have a common covariance. Conversely, QDA occurs when all of the classes are allowed to have individual covariances. For a given class, the estimated LDA and QDA discriminant function is

$$\hat{\delta_k}(x) = -0.5 \log |\hat{\Sigma_k}| - 0.5(x - \hat{\mu_k})^T {\hat{\Sigma_k}}^{-1} (x - \hat{\mu_k}) + \log \pi_k \tag{2.17}$$

where $\hat{\mu_k}$ is the sample mean of the training data for the k^{th} class, $\hat{\Sigma_k}$ is the sample covariance matrix for the k^{th} class [28]. Thus, an observation is assigned to a class that satisfies

$$\hat{C}(x) = \arg \max_k \hat{\delta_k}(x). \tag{2.18}$$

Support Vector Machines

Support vector machines, SVM, are a popular algorithm for classification problems. SVM essentially creates the best fitting hypersurface that separates the classes from one another. To define SVMs, we will use the notation from James *et al.* and Hastie *et al.* and limit ourselves to the binary case [28, 35]. The case with more than 2 classes will be discussed later.

An SVM can be represented by

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i K(x, x_i), \tag{2.19}$$

where $i \in \{1, 2, ..., n\}$, n is the total number of observations, α_i and β_0 are the parameters to be estimated, and $K(x, x_i)$ is the kernel or inner product defined by the analyst. There are a number of different kernels that are commonly used. Some of the more popular kernels

are the linear, polynomial, and radial kernels, which are, respectively,

$$K(x, x_i) = \sum_{j=1}^{p} x_{ij} x_{i'j}, \tag{2.20}$$

$$K(x, x_i) = (\nu + \gamma \sum_{j=1}^{p} x_{ij} x_{i'j})^d, \tag{2.21}$$

$$K(x, x_i) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2), \tag{2.22}$$

where p is the number of features or variables, d is the degree of the polynomial, γ is a positive constant, ν is a constant, and exp is the exponential function [35].

For the case with more than 2 classes, the solutions are one-versus-all and one-versus-one classifications [35]. The details are omitted from here since in Chapter 6 we provide a method to circumvent the need for SVM with more than 2 classes by using meta-classes, or groups of classes in Chapters 5 and 6.

Trees

James *et al.* describes tree based methods as methods which stratify or segment the "predictor space into a number of simple regions" [35]. While I agree with this definition, we could find no general mathematical definition which encompasses this idea. Instead, I found a series of specific types of tree based algorithms for a variety of problems which I need for Chapter 6. Since I will need this formulation in Chapters 5 and 6, I will create this generalization. Thus, the generalized definition of a tree is

$$f(X) = M_{\mathcal{R}} I_{X \in \mathcal{R}}, \tag{2.23}$$

where $M_{\mathcal{R}}$ is the modeling operation performed on the subspace $\mathcal{R}$ and $I_{X \in \mathcal{R}}$ is the indicator

variable for the data, X that belongs to $\mathcal{R}$. For a regression tree, $M_{\mathcal{R}} = \sum_{m=1}^{M} c_m$ and $I_{X \in \mathcal{R}} = I_{X_i \in \mathcal{R}_m}$ where M is the number of regions and c_m is a response constant m. For a classification tree, $M_{\mathcal{R}} = \sum_{x \in \mathcal{R}_m} \frac{1}{N_m}$ and $I_{X \in \mathcal{R}} = I_{Y_i = k}$ where k is the associated class and N_m is the number of observations in a given region or node, m. The notation for the regression and classification trees are borrowed from Hastie *et al.*[28].

Note that Equation 2.23 generalizes any other modeling approach. For example, if $\mathcal{R}$ was the entirety of the space X occupies and $M_{\mathcal{R}}$ is an SVM model with a linear kernel, we are then simply performing SVM on data X. Another trivial example is to use any of the previous classifications methods without the use of meta-classes. This will result in a tree with a singular node and the number of children equal to the number of classes. This will become important and will be expanded in Chapter 6.

Random Forests

Random forests (RFs) are essentially many trees that are combined together to make a prediction. Once the desired number of trees is built, each tree votes for what the observation's predicted value. The value that receives the most votes is determined to be the RF's prediction for that given observation.

I will describe this algorithm using mathematical notation. Using the notation from Hastie, Tibshirani, and Friedman [28], assume that we have v variables or features and N observations or instances. In other words, we have x_i, y_i for $i = 1, 2, ..., N$ with $x_{i1}, x_{i2}, ..., x_{iv}$. Further, suppose that we have M regions, $R_1, ..., R_M$, that divide our feature space. Our model response is represented by p_{mk} for each region. Then, we have that a given tree, b, is

$$f(x)_b = \sum_{m=1}^{M} p_{mk} I(x \in R_m). \tag{2.24}$$

Note that I represents the indicator variable and $k \in \{1, 2, ..., K\}$, where K is the total

number of classes. We estimate p_{mk} by

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k). \tag{2.25}$$

This is the proportion of class k instances in a given node or region m. A greedy algorithm then considers splits using a given variable and split points. The Gini index is used to grow the tree greedily. For 2 classes, the Gini index is

$$Q_m(T) = 2p(1-p), \tag{2.26}$$

where p is the proportion in the second class. This is repeated until the minimum number of nodes is reached. The RF algorithm repeats this tree building process B times (500 in our case). However, these trees are built using bootstrapped data. However, only t of the v variables are selected. This t is tuned during 10-fold cross-validation (CV). Once all of the trees are built, the majority vote for a given observation determines the class of that observation. The expanded details are provided in Hastie, Tibshirani, and Friedman and Breiman's two papers [4, 5, 28].

Dealing with imbalanced data problems is an area of ongoing research [60]. A common solution is to undersample the majority class [6]. This is often paired with the random forest algorithm [6]. This approach is call balanced random forests (BFR). It is an example of a data level method [78]. These types of approaches fundamentally change the empirical distributions of the data [78].

2.2.2 Deep Learning

Deep learning is used for a variety of image classification problems. While deep learning covers a range of applications, the focus here will be on applying deep learning methods to image problems.

Data Augmentation

In small data problems in Computer Vision, a popular approach to increase the number of observations in the dataset is to use data augmentation. Data augmentation is essentially a series of transformations applied to the images. The types of transformations vary, but include rotations, smoothing, flipping, and scaling [53, 54, 86]. I will use this definition of data augmentation throughout this document. Figure 2.4 shows an example of a rotated character. This figure was taken directly from Miller *et al.*[54]. Data augmentation is a useful technique for increasing the total number of observations in our data set, but the process of data augmentation can be expensive.

Figure 2.4: Figure shows an example of a rotated '4' character. Instead of having a single image of the character '4', we can increase the data size by rotating the original image. This figure was taken directly from Miller *et al.*[54].

Convolutional Neural Networks

Convolutional neural networks, CNNs, are one of the most popular methods for image classification [19]. AlexNet is one such example of a popular CNN [40]. CNNs have been used on a variety of image classification problems such as scene recognition [99], medical pill similarity [86,95], and face recognition [62]. CNNs are popular due to there high performance in prediction. However, they cannot be easily interpreted or explained. Furthermore, CNNs can be costly to train [40, 94].

Low-Shot Learning

An example of a small data learning problem is low-shot learning. Low-shot learning attempts to build a model that can predict well while being trained on a small number of observations [26, 87]. Deep learning communities have a variety of different sub-scenarios [26, 81, 87]. One example is a model that was built using lots of house cat images for the cat class. However, it is then desired to correctly classify an image of a tiger. Figure 2.5 provides a visual description of this problem for the cat class.

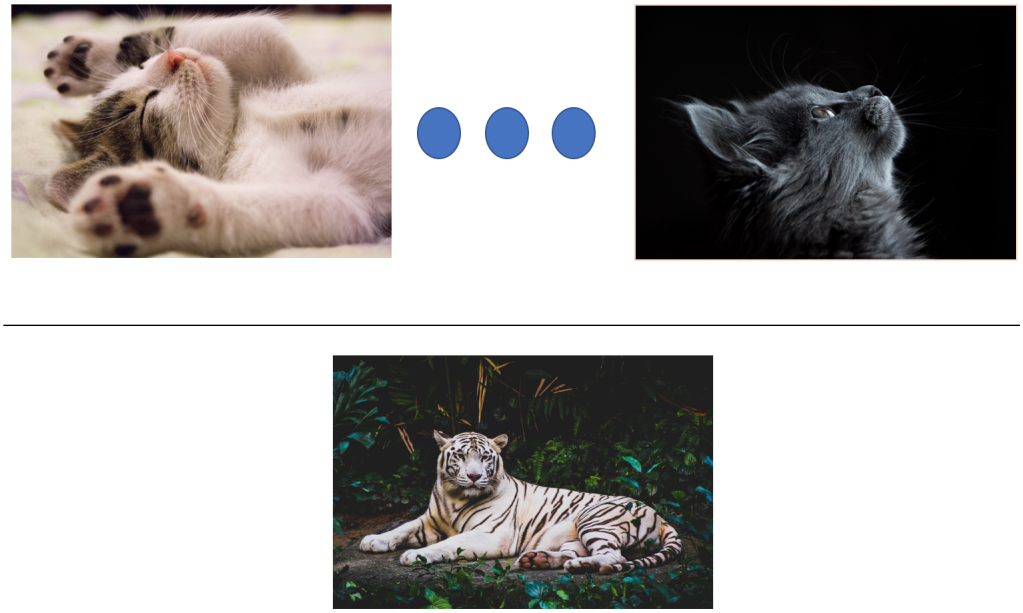


Figure 2.5: Figure shows the class of cats for a imagined low-shot example. The model was trained on the cats on the top half of the figure, with an arbitrarily large n. The challenge is to then correctly classify the single white tiger image without any other training data. This would be considered a zero-shot problem as the Tiger is considered a new and novel class.

Figure 2.6 shows one possible general solution to low-shot learning problems from Hariharan and Girshick [26]. The general solution trains a CNN model on a large number of similar images to the desired small sample data. An example would be to train a CNN model on a large number of cat and dog images to then predict tiger and wolf images.

Deep learning communities have a variety of different sub-scenarios for low-shot learning

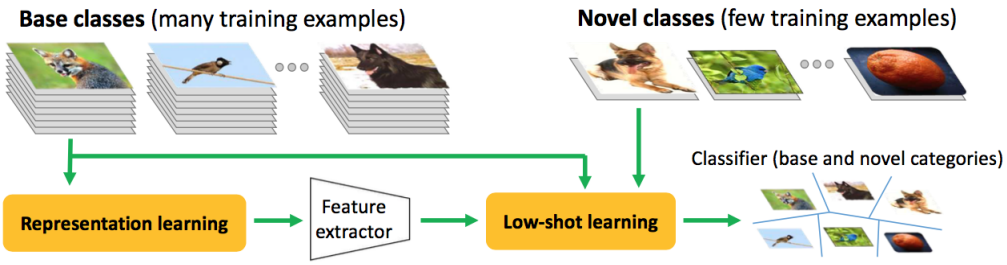


Figure 2.6: Figure shows a basic low-shot overview. Figure is from Hariharan and Girshick [26].

such as one-shot learning [97]. For instance, Buhnia *et al.* uses a one-shot learning CNN model to identify logos [2]. However, this model was still trained on 139104 observations or instances on the training set and 15456 observations or instances on the validation set [2]. To the best of my knowledge, there is no work investigating CNN models built using a much smaller number of instances, such as 5 or 10.

Another low-shot problem is that there exists only a small number of observations and that is all the data. You are limited by doing data augmentation or obtaining additional images due to various costs such as time, money, or human capital.

2.3 Small Data Problems

The definition of small data problems is the following: a phenomenon where data is limited but inferences must be made using this limited data. This encompasses a large number of possible scenarios. For example, the analysts may only have ten observations, but must make infer the population mean from only those instances. Another is that there may in fact have a large number of total observations, but specific cases or classes are lacking. For instance, one class could have 1,000 observations while another only has 5. This specific kind of situation is called an imbalanced data problem. Solutions to these problems vary, and range from using stratification to varying loss functions.

The reason for the existence of small data problems vary but are often related to cost.

The extraction of additional data may be too costly for practical purposes. These costs can take the form of computation, time, money, or human capital. Thus, solutions to alleviate these scenerios are necessary.

2.3.1 Small 'n' Problems

Small n problems are becoming an increasingly important area of research [26,44,66,77]. They lack enough data to make very precise predictions or statement about a given analysis. Many techniques, such as convolutional neural networks, are known to overfit to the training data, or the data used to build the model [19, 40, 76]. To counteract this issue, more data is collected or data augmentation techniques are employed [26, 76, 87] such as rotations, reflections, and noise [38]. Further, data augmentation can be helpful to create synthetic images. For example, one can make an individual with a brown mole on rosy skin a red mole with pale skin by using data augmentation via a neural network [53]. An image classification problem that incorporates the issues of small data learning is pill shape classification.

2.3.2 Imbalanced Data

While data has become increasingly more abundant and plentiful in recent years with the rise of 'big data', there is an increasing need to ensure that smaller classes are accurately modeled. Imbalanced data problems have increased in the literature in recent years [29]. For example, *Nature* recently published an article on substance abuse using social media data [27]. The authors of the article were able to accurately describe a majority of the cases, but were unable to do so on tobacco and prescription/illegal drugs users, or the small class [27]. They stated that they could not find an adequate method to deal with their imbalanced data [27].

Stratification

A possible solution to the small data problem is stratification. Stratification deconstructs the population into a series of distinct but internally homogeneous based on their

corresponding proportions within the overall population. For example, if there are two subpopulations which comprise 75% and 25% of the population, then the data used to build the model should reflect those proportions. This idea is used in a variety of different manners. For example, machine and statistical learning communities use stratification when utilizing cross-validation [39]. Another use of stratification is to build public opinion multilevel models [61]. These two approaches employ post-stratification, where the strata are imposed on the sampling of the data after the data has been collected. However, stratification can also be applied before the data is collected [79]. For example, if a polling company was interested in estimating the national public opinion on a given topic, they could apply stratified random sampling where each state is sampled in proportion to their respective population sizes.

Accuracy and Loss Functions

The use of specific loss functions can help in imbalanced data problems [29]. However, this causes major issues with imbalanced data. What often happens is that the classification method with predict all of the observations to be a member of the larger class [29]. This will provide a large overall accuracy, but completely misclassify the small class [29]. While classification methods have different specific loss functions, they generally are efficient for computing overall accuracy [28, 35].

A popular alternative metric to optimize over is the F measure, or the F1 score [29, 93]. The F measure is the harmonic mean of precision and recall. Recall is calculated as

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}. \tag{2.27}$$

The user interprets recall as the overall classification rate of a given class. The definition of

precision is defined in Equation 2.28.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}. \tag{2.28}$$

While the F-1 score is a popular metric, the final results does not have an easily interpretable meaning. While higher values are indicative a a better value, with 1 indicating perfect classification, and low values are worse, with 0 indicating no classification, a value in-between does not have an interpretable meaning. It does not provide any insight into how to algorithm behaves. Conversely, a value of 75% for overall accuracy means that 75% of the observations were correctly classified.

One of the more popular unique losses in the Computer Vision literature is the triplet loss [73]. Using the notation from Schroff *et al.*, there is a function f where x_i^a is the anchor observation, x_i^p is the positive observation, and x_i^n is a negative observation, where $i \in \{1, 2, ..., N\}$ where N is the number of total observations [73]. The set of all possible triplets in the training set is $\mathcal{T}$ with cardinality N and k being the total number of classes [73]. Thus, the analyst wants

$$||f(x_i^a) - f(x_i^p)||^2 + \alpha < ||f(x_i^a) - f(x_i^n)||^2, \tag{2.29}$$

$$f(x_i^a), f(x_i^p), f(x_i^n) \in \mathcal{T}, \tag{2.30}$$

where α is the margin required between positive and negative pairs [73].

This can be expressed as a loss as

$$\sum_{i=1}^{N} [||f(x_i^a) - f(x_i^p)||^2 - ||f(x_i^a) - f(x_i^n)||^2 + \alpha]_+. \tag{2.31}$$

This loss essentially optimizes the function to have similar observations closer to one another and those that are in different classes to be father apart. However, this is not an efficient

optimization function. This loss is fairly intensive to compute and the anchors must be hand selected [73, 95]. Care must be given to the selection of the anchor observations, and there is no obvious solution [73, 95]. Thus, this loss does not provide a general and elegant solution for imbalanced data problems.

2.4 Medical Pills

A system to identify pills would be useful to global and local communities. Prescription drug use is on the rise in the United States [24, 37]. This increasing trend is not limited to the United States, as the United Kingdom faced a similar increase [96]. In an exploratory study performed in Norway, over half of the thirty patients were given the wrong medication due to poor communication between health care officials [17]. Deaths regarding opioids have also increased in the United States [36]. Developing a system to improve the appropriate utilization and distribution of opioids is needed [36]. This system, a method to identify pills automatically, is desirable by law enforcement agencies, the healthcare industry, and consumers.

The ubiquity of smartphones and affordable, high-quality cameras allows for users to take pictures effortlessly. This allows for pills to be potentially identified by both medical professionals and consumers. Nurses and medical technicians would be able to verify the administration of pills to patients [51]. Multiple research communities have renewed interest in discriminating between fake and real prescription pills [65]. Furthermore, the Food and Drug Administration has advocated for creating a system to monitor patient opioid intake [36]. The National Institute of Health's National Library of Medicine (NLM) hosted a competition in response to some of these issues [58]. Researchers have yet to find a perfect solution for pill identification.

2.4.1 Medical Pill Identification

The goal of the NIH NML competition was to devise a ranking system which provides a high rate of pill identification from a database of possible candidates [58]. In other words, the NIH NML desired to obtain similar pills of an input pill image with the hope that the actual pill in the database is returned from the search as one of the top results. The quick and effortless identification of medical pills could be used in a variety of practical settings.

Pill identification remains a challenging problem. Wong *et al.* created a convolutional neural network (CNN) to identify pills that has a mean overall accuracy of 95.35% [91]. However, they continue to say "From the clinical practicality point of view, [the] accuracy rate... [of our model] is still rather low to allow unsupervised, fully automated pill identification" [91]. The inherent opaqueness of CNNs makes it difficult to diagnose which aspects of the mode work and which fail [23]. Thus, if we built separate models for pill shape classification, pill color identification, and pill text identification that were all interpretable, we could combine the three models into a single interpretable method for pill identification. This approach would allow us to better understand why and how our approaches fail or succeed. My goal for our manuscript is to provide a competitive and interpretable pill shape classification model.

2.4.2 Medical Pill Shape Classification

A solution to classification problems is to create a unique system for the given application. For instance, Maddala *et al.* [51] built a model for classifying medical pills using adaptable rings and a human-machine hybrid (HMH) decision tree. Maddala *et al.* [51] provide two additional models to compare against their proposed model. The first is a neural net using the derived adaptable ring metrics [51]. The second is a logistic regression model using seven Hu moments [51]. Both of these methods are machine driven approaches. Hu moments are popular shape metrics that have desirable theoretical properties such as invariation to orientation [14, 22, 32]. Unfortunately, Hu moments do not appear to provide

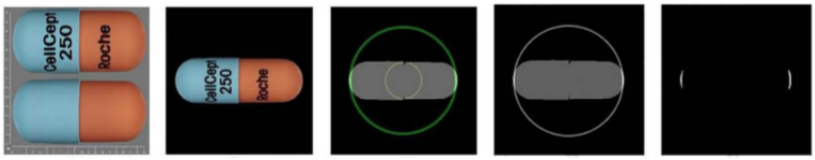


Figure 2.7: Figure shows a general overview of the feature collection process from Maddala *et al.*with the left being the initial input image. They start with the input image. Next, they isolate the pill as presented in the second image. Then, they extract the shape and find the rings in the third image. At this point, they can extract the needed metrics from the third, fourth, and fifth images.

any meaningful insight for discriminating medical pill shapes [51].

While the neural network has a large overall classification rate, it misclassified rectangle, round, oval, and capsule classes [51]. Maddala *et al.*'s approach using Hu moments completely misclassified entire classes [51]. Thus, the medical pill classification problem warranted an improved approach.

Maddala, *et al.*'s third model, the HMH tree, is based on a series of metrics derived from adaptable rings [51]. They used 2,151 pill images with 14 shape classes. They retrieved the data in December 2014. Their approach had very few observations of particular classes at the time of their analysis. For instance, the December 2014 data only had 1 octagon.

This research was partially inspired by the work of Maddala, *et al.*[51]. They provide a rule based system based on a series of metrics derived from adaptable rings. Table 6.1 shows the December 2014 counts and will be discussed in detail in Section 3.1. However, at the time of their analysis, their approach had very few observations of particular classes. For instance, the December 2014 data only had 1 octagon. Figure 2.7 shows a basic pipeline of the image processing and metric collection.

Table 2.2 shows the features collected using adaptable rings by Maddala *et al.*[51]. The first metric is the number of ring overlays, which corresponds to the number of parts of the pill shape that lie on the outer ring. The second metric is the ratio of the area of the outer ring against the inner ring. This feature will have values close to 1 when it is a shape like a circle, as the areas will have similar values. Shapes like capsules will have

Table 2.2: Table provides the six features used in the Maddala *et al.*tree model for pill shape classification.

Feature Number	Description
1	Number of adaptable-ring overlays
2	$\frac{\text{Max adaptable ring overlay area}}{\text{Min adaptable ring overlay area}}$
3	$\frac{\text{Inner ring overlay area}}{\text{Total inner ring area}}$
4	$\frac{\text{Area of pill}}{\text{Area of pill bounding box}}$
5	$\frac{\text{Area of right half of inner ring overlay}}{\text{Area of left half of inner ring overlay}}$
6	$\frac{\text{Max adaptable ring overlay area}}{\text{Area of Pill}}$

larger values as the inner and outer rings have larger differences. The third metric is the inner ring overlap area over the total inner ring area. This provides the proportion of the inner ring of the shape that is completely full. For example, a capsule should have values less than 1 due to the small gaps between the two parts of the capsule. An oval pill should have a value of 1. The fourth metric is the area of the pill divided by the pill's bounding box. This metric will be discussed at length later. The fifth metric is the area of the right half of the inner ring overlay divided by the area of the left half. This value should be equal to 1 for nearly symmetric pill shapes like capsules or ovals, and differ otherwise. The last feature is the adaptable ring overlay area divided by the area of the pill. Thus, shapes like circles or regular polygons with a large number of sides will have larger values, while regular polygons with a smaller number of sides will not.

There are some issues with the metrics they utilized. For example, the fourth metric they used was the proportion of the shape over a bounding box [51]. We interpret this as a normalized geometric moment [32, 38]. While geometric moments are used in a variety

of other metrics such as Hu moments [32], triangularity [69] and other invariant shape measures [13, 14], this fourth measure can provide misleading results. For example, they did not explicitly define the bounding box. If it is presumed that they found the minimum bounding square for a given shape, the fourth feature's value produced by Maddala *et al.* [51] is not invariant to orientation. For instance, for a given square, the ratio they provide will produce different values. If we start with Figure 4.1a, the value of their ratio would be 0.50. However, if one were to rotate the white square by 45°, then their ratio would be 1.00. Thus, distinct shapes would overlap with one another. For instance, while a circle would still be only $\frac{\pi}{4}$, the square would take on values between 0.50 and 1.00. Therefore, the statistic they utilize does not guarantee that a particular value corresponds to a given shape, even in the case that there are the finite kinds of shapes a priori. This warrants a metric which has unique values for many, if not all, of the pill shapes.

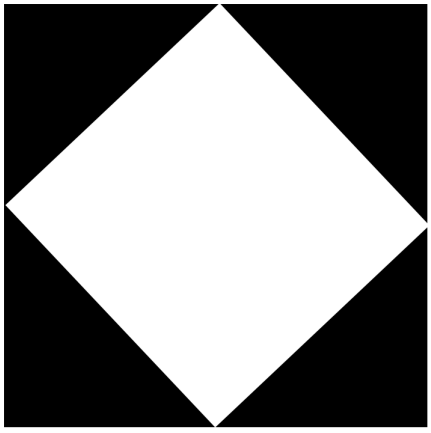


Figure 2.8: Figure is an example image of a square.

Their image processing steps have some issues. Maddala *et al.* treat classes differently during the image processing steps. For example, they center the pill for the oval, capsule, rectangle, and trapezoidal classes using the bounding box center [51]. They calculated a different centroid as the center for the other classes [51]. Results from these two centering

approaches can be seen in Figures 2.9 and 2.10, respectively. These figures are obtained directly from Maddala *et al.*[51]. This is a problem as the classes' features are treated and measured differently.

Figure 2.9: Figure shows an image with adaptable rings using the bounding box center method. The starting image can be seen in Figure 2.10a.

Figure 2.10: Figure provides an example image using the centroid centering method.

Figure 2.11 shows a basic overview of the tree based model used in Maddala *et al.*[51]. For their tree based method, they were ambiguous about the splitting of the data into the training and validation sets. We were able to deduce that all of the Round, Triangle, Square, Pentagon, Hexagon, and Octagon observations were assigned to the validation set. However, they merely stated that the training set was composed of less than 100 observations from

the remaining classes. They did not state if they were randomly chosen or the number used in each class[51]. It is well known that tree based methods tend to overfit to the training data [28, 35, 90]. Note that they did state that they split the data into training and testing sets for the attempted neural network approach they presented [51]. However, the focus of their paper appears to be on the tree based model, as they continually reference it as the approach that performed best.

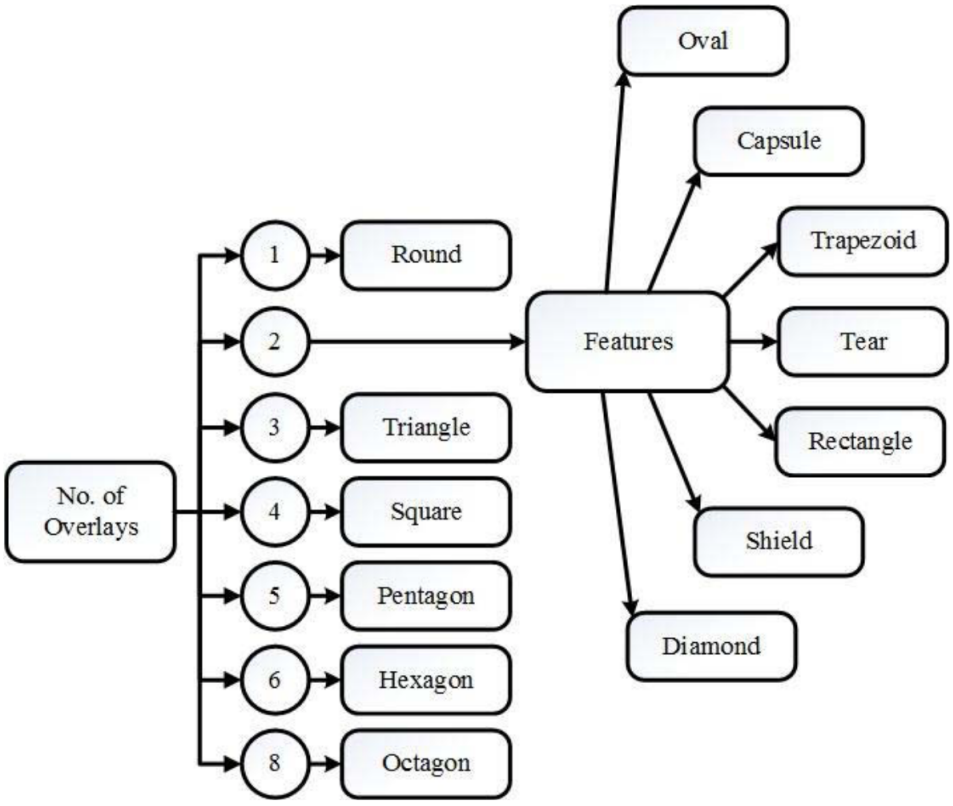


Figure 2.11: Figure shows a broad overview of the tree based model.

Chapter 3: Methods and Materials

In this Chapter, the data, metrics and evaluation methods used in various parts of the thesis are discussed.

3.1 Data

In this section, I will discuss the created polygon, National Library of Medicine (NIH) National Institute of Health (NIH) pill, Hubble Space Telescope galaxy, and MPEG-7 datasets. Each data set is analyzed in Chapter 4, while the NLM NIH data is also analyzed in Chapters 5 and 6.

3.1.1 Created Polygons

The theoretical foundations for shape proportions and encircled image-histograms (SPEIs) are based on regular polygons and circled. Thus, I wanted to analyze created regular polygons as a starting point for our metric and confirm our theories. Further, this data was used to investigate the performance of CNNs.

Image Creation

For a given side length, l, number of sides, n, and image resolution, ζ, the x and y locations, respectively, of the corners of the given polygon was calculated using

$$\vec{x}[i] = l \times \cos\left(2\pi\frac{i}{n} + \frac{360}{n}\right) + \frac{\zeta}{2} \tag{3.1}$$

$$\vec{y}[i] = l \times \sin\left(2\pi\frac{i}{n} + \frac{360}{n}\right) + \frac{\zeta}{2} \tag{3.2}$$

where $i \in \{1, 2, ..., n\}$. Then, lines were drawn from these corners and filled. The Python code to create the image is provided at `https://github.com/billyl320/SPEI-Paper/tree/created_poly`. Figure 3.1 shows examples of $n = 6$ and $n = 8$.

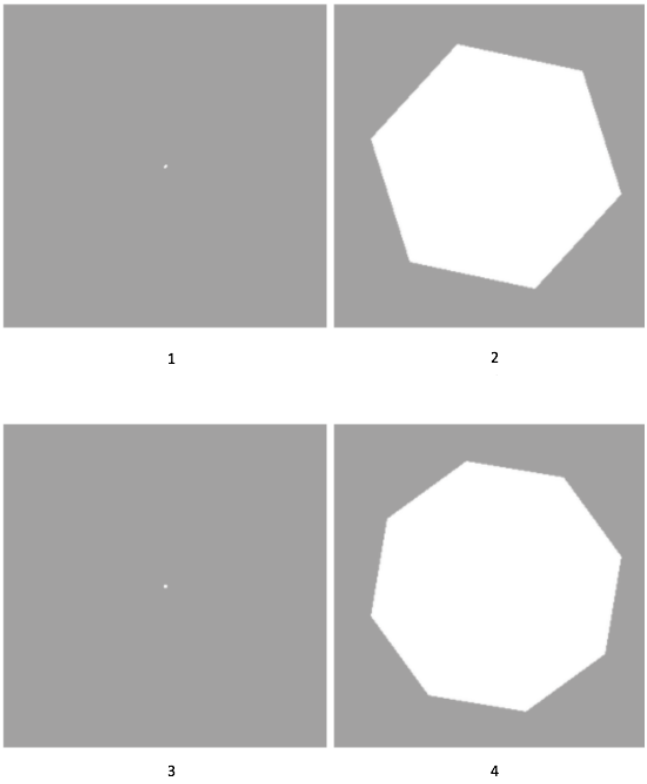


Figure 3.1: Figure examples of the created regular hexagons and octagons. Subplot 1 is the image of the regular hexagon with a side length of 1. Subplot 2 is the image of the regular hexagon with a side length of 125. Subplot 3 is the image of the regular octagon with a side length of 1. Subplot 4 is the image of the regular octagon with a side length of 125. All subplots were edited for presentation purposes. These examples show the range in difference between the initial relative sizes of the polygons.

3.1.2 NLM NIH

I used a subset of images from the National Institute of Health's, NIH's, National Library of Medicine, NLM, competition reference data [58]. There are a total of 1000 unique pills, each with a front and back view taken from the NLM RxIMAGE database. FDA researchers

have indicated pill identification as an important issue that needs a solution [36]. Classifying pill shapes is an important part of achieving a model that can identify pills. I utilized a subset of these pills' classes for this experiment. The counts are provided in Table 6.1. An example from the Triangle class is shown in Figure 3.2. Even though this is considered a triangle, this shape greatly deviates from our conception of a what a triangle should be as it has wavey sides.

Figure 3.2: Figure shows example from the regular triangle class. Note that while it is a triangle, it does deviate from a regular triangle.

Table 6.1 shows the classes alongside their corresponding counts of each of the datasets. Maddala *et al.* added an additional class that is not an officially recognized pill shape by NIH [15]. They split from the "hexagon" class another class called " hexagon (shield)" or "shield" [51]. However, the NIH documentation considers "shields" to be a part of the "Freeform" class [15, 51]. Another convoluted aspect to the December 2014 dataset is that Maddala *et al.* claims that classes such as "double circle" were a part of the "freeform" class. Therefore, there is uncertainty as to what they considered to be a part of the "freeform" class. However, both data sets have similar numbers of observations per overlapping class. This allowed me to perform my analysis on similar footing.

3.1.3 Galaxy

The survey and analysis of galaxies is an important part of the astronomy research community. Governments sponsor programs to collect images of galaxies for analysis [92].

Table 3.1: Table shows the classes and counts of the classes of the NLM NIH reference data and the NIH Pillbox data accessed by Maddala *et al.*in December 2014.

Class	NLM NIH Competition	Maddala, *et al.*Count
Capsule	332	243
Diamond	12	8
Freeform	-	6
Hexagon	8	3
Octagon	-	1
Oval	688	790
Pentagon	12	8
Rectangle	6	4
Round	904	1054
Semi-circle	4	-
Shield	-	5
Square	8	7
Tear	10	9
Trapezoid	4	3
Triangle	12	10

In this experiment, 524 galaxy images selected by Shamir [75] from the Galaxy Zoo project [49] were analyzed and classified. An example image for each class is provided in Figure 3.6. He manually classified the color images into three classes as spiral, elliptical, and edge-on galaxies. The counts for each class are 75, 223, and 225 for the edge, spiral, and ellipse classes, respectively.

3.1.4 MPEG-7

The MPEG-7 data is a commonly used benchmark for experiments in the computer vision and pattern recognition communities [1]. This data contains various orientations, missing parts, and occlusions which test for various conditions shapes may exist [46, 74].

Figure 3.3: Edge example

Figure 3.4: Spiral example.

Figure 3.5: Ellipse example.

Figure 3.6: Figure provides examples of an edge, spiral, and ellipse galaxy, from left to right.

We used a subset of the classes from MPEG-7. The classes used were bird, bone, brick, camel, and cup. Since there are 20 instances per class, the total size of the data I used was 100 observations.

3.2 Metrics

In this section, I describe the shape segmentation algorithms and metric collection applied to the data. The shape segmentation is needed to extract the shapes from the original data, while the metric collection provides the operations performed to collect our features for classification.

3.2.1 Shape Segmentation

I first need to obtain the binary shapes, or a white shape on a black background, of the pills. I passed the entire data set through a single segmentation algorithm. This is better than Maddala *et al.*'s since they required knowledge of the class before the segmentation was performed. I will describe all of the segmentation algorithms using image operator notation as described by Kinser [38].

The shape segmentation algorithm is

$$\mathbf{b}_i[\vec{x}] = \mathcal{I}_{(1)} B \Gamma_{>0} \mathcal{G} \mathcal{L}_L \mathbf{a}_i[\vec{x}], \tag{3.3}$$

where $\mathbf{a}_i[\vec{x}]$ is the input image, $i \in \{1, 2, ..., 2000\}$, $\mathcal{L}_L$ converts the image to grayscale, $\mathcal{G}$ is the gradient operator, Γ is the threshold operator where the intensities greater than 0 are retained, B is the binary fill hole operator, and $\mathcal{I}_{(1)}$ is the isolation operator where only the largest object is retained. I first convert the image to grayscale to reduce the dimension of the images. I then find the gradient so that the edges in the image are retained. I only retain the positive gradient values to binarize the image. I then fill in all of the retained binarized edges to create solid objects. Lastly, I extract the largest object in the image and assume that to be the pill shape. The image segmentation Python code is at our GitHub link: `https://github.com/billyl320/SPEI-Paper`. Examples of $\mathbf{a}_i[\vec{x}]$ and $\mathbf{b}_i[\vec{x}]$ are provided in Figures 5.1 and 5.2, respectively.

3.2.2 Metric Collection

I collected various metrics. I provide the code to collect the metrics at our GitHub link `https://github.com/billyl320/SPEI-Paper`. The first metrics were the shape proportions (SPs) and encircled image-histograms (EIs). I collected these from the shape proportion and encircled image-histogram (SPEI) algorithm from Chapter 4. The other shape metrics were eccentricity [38], circularity [38, 64, 68], and the white and black pixel counts from the minimum bounding box. This results in a total of 7 total metrics used for the HMH decision tree model. Each metric has an intuitive meaning. This makes our resulting model more interpretable.

I collected the encircled image-histograms (EIs) using the algorithm presented in Chapter 4. This algorithm results in a vector $\vec{c}_{EI}$ which contains the white and black pixel counts. These counts are the first two metrics, $\vec{m}_1$ and $\vec{m}_2$, respectively. The shape proportion (SP) value for a given image, i, is merely

$$\vec{m}_{3,i} = \frac{\vec{m}_{1,i}}{\vec{m}_{1,i} + \vec{m}_{2,i}}. \tag{3.4}$$

The SP value is essentially the proportion of white pixels after applying the SPEI algorithm.

This SPEI algorithm puts a shape in its minimum encompassing circle. Then the circle is placed in its minimum encompassing square. I showed SPs to have unique values for regular polygons and circles in Chapter 4. The EIs are the white and black pixel counts after applying SPEIs. The SPs and EIs are a natural fit for a pill shape classification model since they were developed to analyze regular polygons and circles. This is discussed in greater detail in Chapter 4.

Eccentricity, circularity, and the white and black pixel counts from the minimum bounding box had additional image operators performed after $\mathbf{b}_i[\vec{x}]$ was obtained. They were

$$\mathbf{c}_i[\vec{x}] = \triangleleft_{20} \triangleright_{20} \Gamma_{>0.99} \mathcal{S}_{1.5} \mathbf{b}_i[\vec{x}], \tag{3.5}$$

where $\mathcal{S}$ is the Gaussian smoothing operator with a standard deviation of 1.5 and $\triangleright$ and $\triangleleft$ are the erosion and dilation operators, respectively, with a total of 20 iterations each. These operators were performed to obtain more discriminative values.

I calculated eccentricity by finding the ratio of the first and second eigenvalues. To obtain the eigenvalues, we performed

$$\vec{e}_i = \mathbb{E}_{(1,2)} V \mathbf{c}_i[\vec{x}], \tag{3.6}$$

where V collects the covariance matrix of the shape matrix and $\mathbb{E}_{(1,2)}$ calculates the first and second eigenvalues of the resulting covariance matrix. The j^{th} eigenvalue on image i is $\vec{e}_{i,(j)}$. Thus, to obtain $\vec{m}_4$, eccentricity, we perform on a given image, i,

$$\vec{m}_{4,i} = \frac{\vec{e}_{i,(1)}}{\vec{e}_{i,(2)}}. \tag{3.7}$$

It is well-known that the eigenvalues of a covariance matrix correspond to the linear combination in the data which maximizes the variance for their respective dimension [34]. For instance, the first eigenvalue is the linear combination of the data which maximizes the first

eigenvalue [34]. We also know that the linear projections, or eigenvalues, are orthogonal to one another [34]. Thus, the eigenvalues are measures of the major and minor axes of our given shape. Using the ratio of the major and minor axes provide some insight to how a given shape exists as a 2D digital image [38]. A value close to 1 corresponds to a shape with the same major and minor axes' lengths. A value greater than 1 corresponds to the case where the major axis length is larger than the minor axis length. The limit of eccentricity would correspond to the case where the major axis length is infinitely larger than the minor axis length.

The next metrics were the black and white pixel counts from the minimum bounding box. The metrics were collected on image i by

$$\vec{h}_i = \mathbb{H}_2 \mathfrak{B} \mathbf{c}_i[\vec{x}], \tag{3.8}$$

where $\mathfrak{B}$ finds the minimum bounding box of the input image, and $\mathbb{H}_2$ calculates the binary image histogram, or binary intensity histogram, of the bounding box image. The result is a vector of the counts of the white and black pixels, which are represented by $h_{i,w}$ and $h_{i,b}$, respectively. Thus, the metrics m_5 and m_6 (the white and black pixel counts of the image in a minimum bounding box) are:

$$\vec{m}_{5,i} = h_{i,w}, \tag{3.9}$$

$$\vec{m}_{6,i} = h_{i,b}. \tag{3.10}$$

These values describe how rectangular a given shape is. If a given pair has a very large white count, but a very small black count, then this given shape is fairly rectangular.

The last metric, $\vec{m}_7$, is circularity. The metric was collected on image i by

$$\vec{m}_{7,i} = \frac{\sum \mathbf{c}_i[\vec{x}]}{4\pi \times \left(\sum \triangleleft \mathbf{c}_i[\vec{x}] - \sum \mathbf{c}_i[\vec{x}] \right)}, \tag{3.11}$$

Table 3.2: Table provides the metrics used in this analysis on a given image, i. The first column is the q^{th} metric, where $q \in \{1, 2, 3, 4, 5, 6, 7\}$. These variables make our model interpretable.

$\vec{m}_{q,i}$	Metric
1	White EI
2	Black EI
3	SP value
4	Eccentricity
5	White Bounding Box Count
6	Black Bounding Box Count
7	Circularity

where $\sum$ sums the pixel intensity values. The $\sum$ operator will compute the area of the image since we are restricted to binary images. The denominator of this metric is the perimeter of the binary image multiplied by 4π. Circularity provides a measure for how circular a shape is as a 2D digital image [38, 64, 68]. A value of 1 corresponds to a perfect circle [38, 64, 68].

All of the variables used in our analysis have interpretable meanings. This will aid in the interpretation of our models. Table 6.3 provides a summary of the variables or metrics collected for this analysis.

Chapter 4: SPEI: A Tool to Improve the Analysis and Classification for Shapes for Small Data

Shape proportions and encircled image-histograms (SPEIs) is an algorithm that produces simple, intuitive, and competitive measures for shape analysis and classification problems in small data scenarios. Small data problems are a significant challenge in Computer Vision. There are only a few intuitive and mathematically derived shape measures and models. In this chapter, the aim is to develop a measure called shape proportions (SPs). The encircled image-histograms (EIs) from 2D digital binary images are the realizations of SPs. My results show that SPEI-based models outperform CNNs by about 52% on a variety of data sets. SPEI-based models are able to do this since they encode human knowledge about shapes. CNNs are powerful since they can learn without human knowledge, but they are not able to discover human concepts of shapes such as area.

4.1 Introduction

The analysis and classification of shape data has far-reaching influence in many research communities. Researchers in Deep Learning have dedicated a significant amount of effort in shape analysis through some form of a convolutional neural network (CNN). However, CNN models require large amounts of data [23]. There are cases where the amount of data is limited due to a variety of reasons such as time, money, or human capital. An example is in the NIH NLM pill data set, where some classes only have eight observations, or instances [58].

Thus, the plan of this study is to understand how our abstractions of shapes exist as we observe them in the environment. In other words, I hope to explain how the abstraction of 2D shapes in our imaginations exist as 2D digital images. I hypothesize that an approach

which can mathematically describe shapes can aid models to achieve better performance over CNNs in small data problems.

To this end, I developed a mathematical formulation based on mathematical proofs for regular polygons and circles. This mathematical encoding of the abstraction of shapes allows for the conceptualization of SPEI. The plan is to apply SPEI to created regular polygons to confirm the mathematical theory. I then applied SPEI to medical capsule, or pill, data. The complexity of the type of shapes was increased by analyzing non-polygonal shape data in the form of galaxies. I performed an additional analysis on the standard MPEG-7 data which includes shapes with various deformations, occlusions, and orientations. The SPEI-based approach outperforms CNNs by 52% on average.

4.1.1 Outline of Proposed Method: SPEI

I discuss a new and novel approach to analyzing any set of shape classes for any 2D binary shape image, called shape proportions and encircled image-histograms (SPEIs, which is pronounced "spies"). The method is easy to explain mathematically, and the conclusions of the final plot created are effortlessly interpretable. Furthermore, the applications for SPEIs are varied, as SPEIs can be built upon using other methods. For a given application, a researcher can alter the approach to fit the specific problem a researcher is solving.

An analyst can apply SPEIs to any 2D binary shape. A SPEI is particularly powerful when the shape has a unique value for the shape proportion (SP). The SP is the proportion of white pixels resulting from SPEIs. A SP value corresponds to an encircled image-histogram (EI). The EI is the resulting black and white pixel counts. Thus, the SPEI image operator algorithm has two resulting metrics: the SP and EI values.

A newly defined small data problem is the following: the researchers have a small number of observations and are limited in obtaining more data. This scenario is defined as data starved. This means that users cannot perform data augmentation or obtain additional images due to various costs such as time, money, or human capital. A simple example would be that the entire data set consists of four cat pictures and four dog pictures. The

goal would be to correctly classify the cat and dog images. Data starvation is similar to low-shot learning where the analyst has a small amount of data and cannot collect more data or perform data augmentation techniques.

First, the plan is to discuss binary shape images and some properties that regular polygons and circles have when using SPEIs in the Methods and Materials section. Second, I will apply the SPEI algorithm to four data sets and compare the approach to CNNs in the Results section. The first data set is a regular polygon shape data set where the SPs are known. The second is on medical pill data. The third is on a galaxy shape data set where the SPs are not known. The fourth is on a subset of the MPEG-7 data set. This data contains various orientations and occlusions for each of its classes. The Discussion section mentions the major takeaways from all the experiments and future work. The chapter ends with a summary in the Conclusion section.

4.1.2 Contributions

In this paper, I present SPEI, an image processing algorithm which can be applied to any 2D binary image. The SPEI algorithm results in two metrics: the SP values and the EIs. If the analyst can show that the SP values for each class of shapes are distinct, then they can utilize the EIs for classification purposes. I also discuss cases where the analyst cannot prove the SP values for a given shape. Further, SPEI is applied to a newly defined small data problem where the analyst is data starved. The SPEI-based models results are compared to CNNs due to its popularity in the literature. I will show that SPEI-based approaches outperform CNNs by 52% in small data scenarios.

4.2 Methods and Materials

In this section, I will discuss the technical details of SPEIs. Then, I will mention the CNN modeling aspects that will be used for comparison to SPEIs.

4.2.1 SPEIs

In short, SPEIs puts the shape in the minimally encompassing circle. This is then placed inside the minimal encompassing square. Visual representations of this are provided in Figures 4.1a and 4.1b. The circle is placed in a square for convenience, as most digital images are composed of square pixels. In general, the user could apply SPEIs by placing the encompassing circle inside any desired shape, like a hexagon.

I will begin by analyzing how SPEIs behave for regular polygons and circles. I start with this class of shapes as they are more intuitive to conceptualize and understand. I hope that by analyzing this simpler class of shapes, we can begin to understand the more complex nature of other classes of shapes. By applying SPEIs, circles and regular polygons will have unique SP values of

$$p_c = \frac{\pi}{4}, \tag{4.1}$$

$$p_n = \frac{n \sin(360°/n)}{8}, \tag{4.2}$$

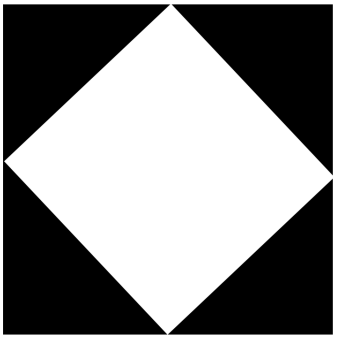
where n = the number of sides of the regular polygon. The proofs are provided in Section 4.2.2. The SP values for a subset of regular polygons are provided in Table 4.1. The implementation of the SPEI algorithm is provided in Section 4.2.3.

Table 4.1: Table shows a subset of the proportions of white pixels for a given n-sided regular polygon.

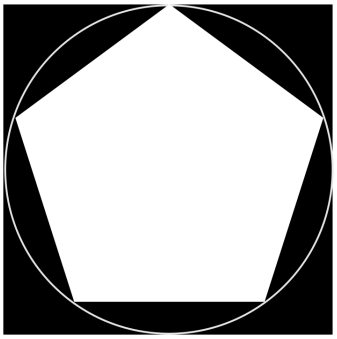
n	3	4	5	6	7	8
p_n	0.3248	0.5000	0.5944	0.6495	0.6841	0.7071

4.2.2 SPEI Proofs

In the following section, a regular polygons and circles can be summarized as a singular value, the SP value. These will be proven. SPEIs is generalizable to any binary shape. If one was able to show that a given group of classes each had unique values for their



(a) Figure is a representation of a regular square whose apothem is the radius of the circle, r. Note that there is no reference circle in this image, unlike in Figure 4.1b. Thus, this is how an actual image of a square is represented after SPEI is applied. In this case, $p_4 = 0.50$.



(b) Figure is a representation of a regular pentagon whose apothem is the radius of the circle, r. Note that in this image, a circle is drawn only for reference, but is not actually part of the image. In this case, $p_5 = 0.5944$.

Figure 4.1: Examples of results of SPEI on square and pentagon, from left to right.

corresponding SP values, then SPEI is still applicable.

Several proofs will utilize notation that will be defined as follows: A_n = the area of a n sided regular polygon, which is cyclic, A_c = the area of a circle bounded by a square whose radius equals half of the square's side, A_r = the minimum bounding square which envelopes a circle whose diameter equals the square's side, $p_n = \frac{A_n}{A_r}$, $p_c = \frac{A_c}{A_r}$, and r = the radius of A_c's corresponding circle.

Proof: $p_c = \frac{\pi}{4}$

$$\begin{aligned} p_c &= \frac{A_c}{A_r} \\ &= \frac{\pi r^2}{(2r)^2} \\ &= \frac{\pi}{4} \quad \blacksquare \end{aligned}$$

Figure 4.2 showcases an example of the scenario we just proved. The figure is representation of a perfect circle whose diameter is the same length of a side of the square. This corresponds to the white pixels having $\frac{\pi}{4}$ of the area of the image.

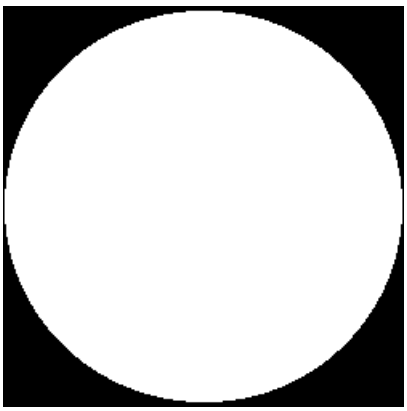


Figure 4.2: Figure is a representation of a perfect circle whose diameter is the same length of a side of the square. This image corresponds to Proof 2.1.

Proof: $p_n = \frac{n \sin(360°/n)}{8}$

For any regular polygon,

$$A_n = \frac{1}{2} n r^2 \sin\left(\frac{360°}{n}\right),$$

where $n =$ the number of sides of the polygon [101]. This formula comes from measuring a circle's radius where that circle perfectly encompasses that given polygon. Thus:

$$\begin{aligned} p_n &= \frac{A_n}{A_r} \\ &= \frac{\frac{1}{2} n r^2 \sin\left(\frac{360°}{n}\right)}{(2r)^2} \\ &= \frac{\frac{1}{2} n r^2 \sin\left(\frac{360°}{n}\right)}{4r^2} \\ &= \frac{n \sin\left(\frac{360°}{n}\right)}{8} \quad \blacksquare \end{aligned}$$

From these two proofs, we have the following theorem:

Theorem 1. *Circles and regular polygons have unique SP values.*

This provides evidence in support for Hypothesis 1 from Chapter 1. From here, we can prove that 2D images that have representations of this shape can have this property as well. For this we need some definitions. Any realization of a shape will have an SP value, $\hat{p}$. Assume that this realization of a shape has a known and unique SP value, p. Then we have that

$$\hat{p} = p + \epsilon, \tag{4.3}$$

where ϵ is some error.

Proof: $\lim_{\epsilon \to 0} \hat{p} = p$

$$\begin{aligned} \lim_{\epsilon \to 0} \hat{p} &= \lim_{\epsilon \to 0} (p + \epsilon) \\ &= \lim_{\epsilon \to 0} p + \lim_{\epsilon \to 0} \epsilon \\ &= p \quad \blacksquare \end{aligned}$$

Thus, we have that

Theorem 2. *The limit of the realization of circles' and regular polygons' SPs is their respective SPs.*

Thus, for 2D images of an object that has an observed shape, if the shape has an adequately small error in its representation, it should have a unique SP value. In other words,

Corollary 2.1. *The limit of 2D binary digital image shapes of circles and regular polygons have unique SP values.*

Theorem 2 and Corollary 2.1 provide evidence in favor of Hypothesis 1 from Chapter 1.

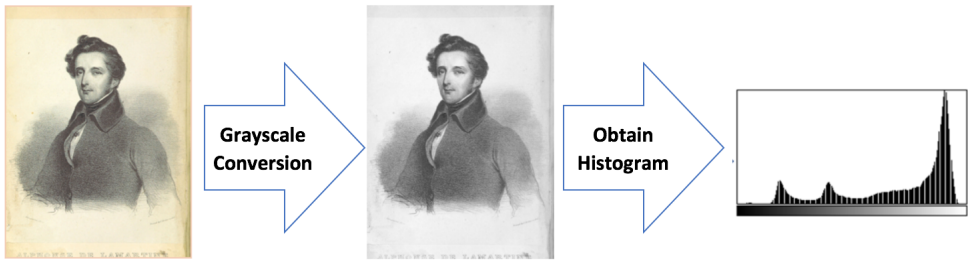


Figure 4.3: Figure shows the original input image, denoted $\mathbf{a}[\vec{p}]$, the grayscale image, and the resulting image histogram, from left to right. The image histogram was produced using ImageJ[82]. This provides a visual representation of Equation 4.4.

4.2.3 SPEI Implementation

Before I discuss the technical details of implementing SPEIs, understanding traditional image histograms, or the pixel intensity counts, is needed to understand some terminology and notation [38, 82]. This will allow us to have some deeper understanding and intuition for the how algorithm behaves. An example of an image histogram on a grayscale image is presented in Figure 4.3. In this case, the image histogram is the resulting 256 grayscale intensities of the middle image. While traditional image analysis considers the plot to be the image histogram, for convenience, we will also refer to the vector intensity counts as an image histogram. In image operator notation [38], this would be

$$\vec{c} = \mathbb{H}_{256}\mathcal{L}_L\mathbf{a}[\vec{p}], \tag{4.4}$$

where $\mathbf{a}[\vec{p}]$ is the original input image from the British Online Library [48], $\mathcal{L}_L$ is the grayscale conversion operator, and $\mathbb{H}_{256}$ is the image histogram operator with the subscript 256 to indicate that 256 intensities are utilized. The result, $\vec{c}$, is a vector containing 256 numeric values, indicating the counts of each grayscale intensity. Figure 4.3 also showcases a general pipeline of these image operators.

I utilized image operator notation from Kinser [38] to describe the algorithm for implementing SPEIs. We provide the Python 3.6 code to perform these operators at `https:`

//github.com/billyl320/SPEI-Paper. The traditional image histogram analysis for a given binary image is as follows:

$$\vec{c} = \mathbb{H}_2 \mathbf{a}[\vec{p}] \tag{4.5}$$

where $\mathbf{a}[\vec{p}]$ is the input binary image, $\mathbb{H}_n$ represents the process of converting the images to its corresponding histogram intensity with the number of bins represented by n, and $\vec{c}$ is a vector with the white and black pixel counts. We assumed that the individual shape is already in a bounding box.

The generalized process we have developed is as follows:

$$c\vec{_{\mathrm{EI}}} = \mathbb{H}_2 \square_{\vec{v_1},\vec{v_2}} \mathbf{d}[\vec{p}], \tag{4.6}$$

where

$$\mathbf{d}[\vec{p}] = D_{\vec{v}} \Gamma_{<128} \mathbf{b}[\vec{p}], \tag{4.7}$$

$$\mathbf{b}[\vec{p}] = U_{\vec{w}} \mathcal{L}_{\mathrm{L}} \mathbf{a}[\vec{p}], \tag{4.8}$$

$$\vec{w} = (2 \times \bigvee Z\mathbf{a}[\vec{p}], 2 \times \bigvee Z\mathbf{a}[\vec{p}]), \tag{4.9}$$

$$\vec{c} = \boxtimes \mathbf{b}[\vec{p}], \tag{4.10}$$

$$\vec{v} = \vec{c} - \vec{w}/2, \tag{4.11}$$

and $c\vec{_{\mathrm{EI}}}$ is the encircled image-histogram white and black pixel counts. $U_{\vec{w}}$ places an image at the center of a larger frame where $\vec{w}$ is the size of the larger frame, $\bigvee$ is the max operator, Z is the dimension operator where the dimensions of an input image are retrieved, $\boxtimes$ is the center of mass operator, $\mathcal{L}_{\mathrm{L}}$ converts the color model to grayscale, $\Gamma_{<128}$ applies a threshold operator of less than 128 for every pixel in the image, and $D_{\vec{v}}$ shifts the image by $\vec{v}$ [38]. $\square_{\vec{v_1},\vec{v_2}}$ represents the window operator which isolates a subimage with opposing corner locations of $\vec{v_1} = (f - r, f - r)$ and $\vec{v_2} = (f + r, f + r)$ where

$$r = \uparrow \bigvee((\mathcal{D}_E \mathbf{o}[\vec{p}]) \times \mathbf{d}[\vec{p}]), \tag{4.12}$$

$$f = \downarrow \left(\frac{\bigvee Z\mathbf{a}[\vec{p}]}{2} \right), \tag{4.13}$$

$\uparrow$ is the ceiling or rounding up math operator, $\downarrow$ is the floor or rounding down math operator, $\mathcal{D}_E$ returns the distance transformation image using Euclidean distance, and $\mathbf{o}[\vec{p}]$ is a matrix of 1s except where the center is 0 and the shape is the same as $\mathbf{b}[\vec{p}]$.

Applying the Theory

Given that each regular polygon can be described by a single number, the SP value, how this translates into a 2D digital image will be calculated by simply multiplying the resolution, or total number of pixels, of the images by p, the SP value. In other words,

$$X = \zeta \times p \tag{4.14}$$

where X = the number of white pixels, p = the SP value, and ζ = the resolution of an image. Note that here we assumed $\zeta = (2r)^2$. For example, if $p = 0.50$ and $r = 50$, then $\zeta = 100^2 = 10000$. Thus, $X = 10000 \times 0.50 = 5000$. In turn, the number of black pixels will be $\zeta - X = 5000$. The combination of X and $\zeta - X$ gives us the theoretical EI.

This formulation provides not only a deeper understanding of the properties of regular polygons, but also an expectation for their behavior as digital images. We can exploit this relationship for classification purposes. I will discuss some simple examples first. While I emphasized regular polygons and circles as the type of shape analyzed so far, this formulation can easily be generalized to any binary shape. I will provide evidence that SPEIs is generalizable to other classes of shapes in Section 4.3.

SPEIs is Invariant to Orientation

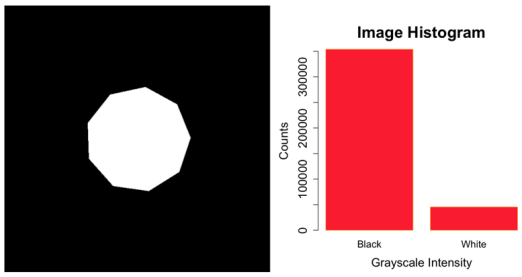
One of the clear properties of this method of describing shape is that after applying SPEIs, the shape is invariant to the orientation. For instance, if the pentagon in Figure 4.1b was rotated by 30°, we would obtain the same minimizing encompassing circle. Thus, the value of p_5, and in general, p_n, remains constant.

Others have used somewhat similar methods. For example, Maddala *et al.* [51] utilized the proportion of the shape over the bounding box as their fourth feature in a rule based system to classify shapes. However, their method is very different than what I am describing. The first difference is that the method we have described is explicit in its execution. Here, I defined the theoretical bounding box explicitly.

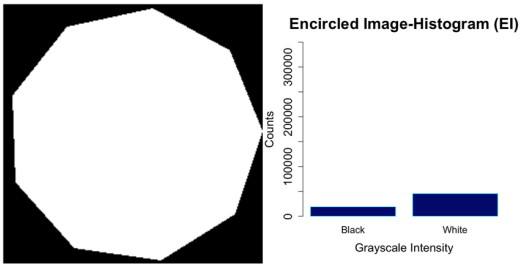
Secondly, if one presumed that they found the minimum bounding square for a given shape, the value that the fourth feature produced by Maddala *et al.* [51] is not invariant to orientation. For instance, for a given square, the ratio they provide will produce different values. If we start with Figure 4.1a, the value of their ratio would in fact still be 0.50. However, if one were to rotate the white square by 45°, then their ratio would be 1.00. Thus, distinct shapes would overlap with one another. For instance, while the circle would still be only $\frac{\pi}{4}$, the square would take on values between 0.50 and 1.00. Thus, the statistic they utilize does not guarantee that a particular value corresponds to a given shape, even in the case that we know the finite kinds of shapes a priori.

4.2.4 SPEIs Adjust Black Pixel Counts

For binary shape images, a traditional image histogram is essentially the first geometric moment [32, 38] alongside the resolution of the image, or the total number of pixels, minus the first geometric moment as shown in Figure 4.4a. After applying SPEI, the resulting image and EI can be seen in Figure 4.4b. After comparing the results of the corresponding pixel intensity counts in Figure 4.4, SPEI is essentially adjusting the black pixel counts for the shape being analyzed as the white pixel counts remain unchanged. Thus, what is occurring during the SPEI process is a readjustment of the shape's black pixel count.



(a) Figure shows the corresponding image histogram on the right of a created regular 9-gon provided on the left. Notice that the shape, the white pixels, composes less pixels than the background, the black pixels.



(b) Figure shows the corresponding encircled-image histogram on the right of the created regular 9-gon provided on the left. Notice that the shape, the white pixels, composes more pixels than the background, the black pixels.

Figure 4.4: Examples of image histogram and EI, from left to right. Notice that the resulting EI is essentially an adjustment of the number of black pixels, while the white pixel counts remain the same.

How to Use SPEIs

The analyst may use SPEIs for different goals. For instance, if the goal is to understand how the shapes exist within reality, the estimation of SPs may be important. However, if the goal of the analysis is classification, then one may only desire to discriminate all of the observations based on the observations' EIs.

One of the benefits of SPEIs is that researchers can use the resulting EI values by a variety of different classification algorithms. For our analysis, quadratic discriminant analysis (QDA), support vector machines (SVM), logistic regression (LR), and trees are examples of classification algorithms used to discriminate the observations based on the EIs. Thus, analysts can use SPEIs in a variety of classification techniques.

4.2.5 CNNs

I built CNNs to compare the models utilizing SPEIs in Python. We provide the code for the experiments here: `https://github.com/billyl320/SPEI-Paper`. Due to the unique aspects of being data starved, I carefully constructed the CNN architecture and obtained guidance from Gu *et al.*[23]. Despite this resource, there appears to be little work done on the

creation of a new CNN architecture for small data. While there was some work on "small" data sets [84], these were significantly larger than ours. Therefore, I attempted different architectures, many of which are not included in this manuscript. For example, I produced a very simple CNN with one convolution layer. However, this produced abysmal results. Thus, I increased the complexity of the architecture by adding more layers. Unfortunately, this did result in some overfitting on the training data, but the validation data did improve despite this issue. Some of the recommendations provided by Gu *et al.*[23] was to use early stopping [63], SpatialDropout [84], and a l_2-norm regularization [30].

Architecture Description and Implementation

The final CNN architecture has a total of nine layers, not including the output layer. There are essentially three triads. Each triad has a convolution, pooling, and dropout layer, in that order. The first convolution layer has 64 nodes, while the remaining convolution layers have 32 nodes. All of the convolution layers have a kernel size of 3 using a ReLU activation [56] and an l_2-norm regularization with $\lambda = 0.001$. The pooling layers all have a batch size of 2×2 and uses the max pooling operation [3]. The dropout layers have a drop rate of 0.20. The output layer used a softmax activation [70]. This architecture is summarized in Figure 4.5.

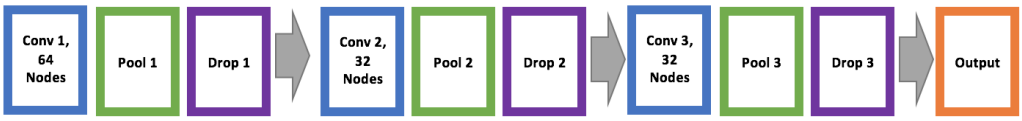


Figure 4.5: Figure shows the general architecture of the CNN used in the data starved setting. It has a total of nine layers excluding the output layer, where each triad has a convolutional, pooling, and dropout layer. They are represented by the colors blue, green, and purple, respectively.

For the early stopping implementation, I permitted a max of 100 epochs. While I measured entropy loss, the minimum change permitted between epochs was 0.001, but I required the stability of this value to remain constant for 10 epochs. I took additional data

preprocessing steps for each experiment performed. The one universal similarity is that the CNN's input images were all binary after the preprocessing was complete.

4.3 Results

To be goal of the experiments was to observe if our SPEI-based approach outperforms CNNs due to their popularity in the literature. I describe four experimental results. The first is on created regular polygon data. The second is on medical pill data. The third is on a galaxy shape data set. The fourth is on the MPEG-7 shape data set. I performed the image processing for SPEIs in Python [38] and the model building for SPEI-based approaches in R [80]. The figures showing the results were also done in R while also utilizing some graphics packages, such as ggplot2 [88] and scales [89]. Implementation guidance for the R models was obtained from James *et al.*[35]. We used various packages for modeling such as e1071 [52] and MASS [85]. We provide the code for all of the experiments at `https://github.com/billyl320/SPEI-Paper`.

4.3.1 Created Regular Polygons

We desire to see how a set of classes with known SP values behave by creating a large number of regular polygons with varying circumradius lengths. I created regular n-polygons where $n \in \{3, 4, ..., 8\}$ with sides ranging from pixel lengths of 1 to 125. This is described in detail in Section 3.1.1.

Models

I first compare image histograms and EIs using the created regular polygon data. Since bounding box algorithms appear the literature in various places such as a part for object detection [98] or as a means to collect metrics [51]. Here, I am focusing on metric collection, as the results of the bounding box are highly dependent on them being properly specified and described.

In our subsequent experiment, I compared CNNs and SPEIs. I contrasted SPEI-based approaches only to a CNN due to the popularity of CNN based approaches [23]. I analyzed the data in six small data learning settings where we were data starved. I randomly split the data into two sets: one to build the model, and another to check model accuracy, which I will call the training and validation sets, respectively. The testing sizes investigated were 3, 4, 5, 6, 7, and 8 per class, where I assigned the remaining observations to the validation set. For example, when the testing size per class was 3, the total number of observations in the testing data set was 18. This corresponds to a validation set size of 109 per class. In the described setting, I ran the experiment 100 times. I collected the overall classification on the training and validation data each time. I report the mean classification rate for the set.

For the collection of the EIs, each input image was processed using Equation 4.6. Then, I built three separate models using the training data. Using a given built model, I utilized the validation data to compute the accuracy of the model. The four possible SPEI-based models used QDA, support vector machines, SVM, using a linear kernel, a tree, and logistic regression, LR. The only variables used for the SPEI-based models was the resulting EIs. I computed all SPEI-based approaches in R.

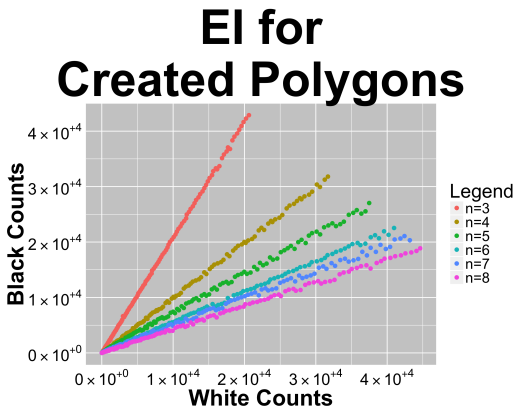
For the CNN approach, the input images were simply the resulting shapes as described in Section 4.3.1. The CNN models are not dependent on SPEIs in any matter whatsoever. The CNN model architecture remained the same from Section 4.2.5.

I added some details of the resulting CNN models and their corresponding layers for the regular polygon data. This provides greater insight into the nature of CNNs in small data problems. This included an experiment by increasing the number of observations for training by means of data augmentation. I performed data augmentation using various rotations to increasingly train the CNN model in five different testing sizes. They were 10, 50, 100, 250, 1000, and 5000, instances per class. I rotated each image by 1° 359 times for a total of 359 images per each original image. This resulted in 44875 images per class to be used for this experiment.

I also confirmed that SPEI-based models are consistent with the theory that they are invariant to orientation. I had the original data rotated by 120° and 240°. This resulted in a new data set of 250 observations per class. None of these observations were ever used to train the model. I had the best SPEI-based model predict the classes on these observations and reported the results.

Comparing Bounding Box and EIs Experiment

(a) Figure shows the counts for the image histograms, or a bounding box, of the shape images. We created a model in an attempt to discriminate the classes, but we were unable to find meaningful results.



(b) Figure shows the counts for the EIs of the shape images. In the Legend, the value of n indicates the number of sides on the regular polygon class. Note that the EIs both follow linear lines. The final resolution ranged from 6^2 to 256^2. Note that clear separation is evident.

Figure 4.6: Examples of image histogram, a bounding box, and EI, from left to right. Notice that the bounding box is not useful for classification, while there is a clear discriminative pattern provided by the EIs.

Figure 4.6a shows the resulting traditional image histogram, or in this case, a bounding box algorithm approach. The white pixels, or the first geometric moment, cannot provide meaningful results for classification even alongside the black pixel counts. Even a normalized version, or the fourth feature from Maddala *et al.*[51], does not provide helpful results. There is no clear visual discrimination available as the points overlap one another substantially in Figure 4.6a. Thus, the realization of the fourth feature, alongside the black pixel counts, will not be useful for discrimination purposes.

Conversely, the results from the EIs on the regular polygons have a clear relationship as seen in Figure 4.6b. Note that each class follows a linear class regardless of the scale and total number of pixels in the image for the shape. Thus, just visually alone, the resulting EIs from the SPEI algorithm is clearly superior to the traditional image histogram.

Comparing CNN and SPEIs Experiment

The results from the formal models are summarized in Table 4.2 and Figure 4.7. The QDA model performs best in terms of achieving a higher predictive accuracy and also providing the most similar predictive performance on the training data. When using the best SPEI model for a given training size per class, this corresponds to an increase of outperforming CNNs by about 128% on the validation data. This provides evidence in favor of Hypothesis 1 from Chapter 1.

SP Value Analysis

If researchers desire to better understand the nature of the shapes as a 2D digital image, they could calculate the empirical SP values. Table 4.3 shows the empirical SP mean and standard deviation, SD, of each polynomial class. Note that these mean SP values are similar to the theoretical SP values presented in Table 4.1. The standard deviations, SDs, do not appear to have any pattern such as monotonicity. However, the SD values do appear fairly stable for the odd n values, while the even n values appear to be steadily increasing.

CNN Investigations

Using the setup described in Section 4.2.5, I investigated a CNN model by observing the layers when various images are computed by it. The CNN model essentially struggles to distinguish shapes that have a large number of sides and small scale. For example, the CNN model had difficulty discriminating heptagons and octagons with a circumradius length of 25. Conversely, it could discriminate those same shapes with a circumradius length of 120. At small scales, a CNN model cannot find the features needed to discriminate similar, but

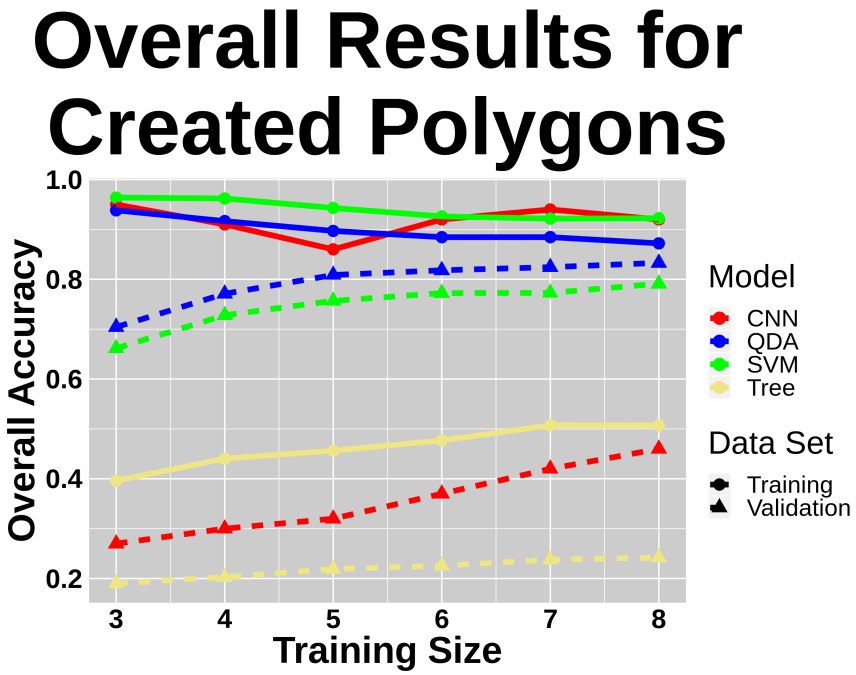


Figure 4.7: Figure shows the overall classification rate over the various settings. For example, the training size of value 3 means that there were 3 observations per class in the training data. The CNN, QDA, SVM, and Tree results are provided in red, blue, green, and yellow, respectively. The training data set results are provided in solid lines with circular points while the validation sets are presented in dashed lines with triangular points. Generally, the training data is on the top half of the figure, while the validation data results are near the middle. Note that the in all training size settings, the SPEI-based QDA model outperforms the CNN model on the validation data.

different, shapes.

We desire to know specifically where the CNN is failing to discriminate the regular polygon shapes. Figure 4.8 shows various shapes that were run through a built CNN model with a training set size of 3. The two circumradius lengths were 25 and 120, and the shapes were triangles, heptagons, and octagons. Notice that at all scales, the shapes are discernible to the human eye. This investigation will shows how the layers are understanding shapes of different scales alongside pairs of shapes that are different from one another, such as triangles and octagons, and shapes that are similar to one another, such as heptagons and octagons.

Table 4.2: Table shows overall mean accuracy results for each of the models on each of the data sets for the regular polygon images. The standard deviation is provided below the mean in parentheses. The number of observations is provided in the column header for the data starved setting. The SPEI QDA model outperforms the CNN model on the validation data.

Model	3	4	5	6	7	8	Data Set
CNN	0.95	0.91	0.86	0.92	0.94	0.92	Training
	(0.054)	(0.062)	(0.072)	(0.046)	(0.036)	(0.035)	
QDA	0.94	0.92	0.90	0.88	0.88	0.87	Training
	(0.059)	(0.058)	(0.063)	(0.052)	(0.046)	(0.048)	
SVM	0.96	0.96	0.94	0.93	0.92	0.92	Training
	(0.081)	(0.057)	(0.048)	(0.047)	(0.059)	(0.043)	
Tree	0.40	0.44	0.46	0.48	0.51	0.51	Training
	(0.081)	(0.073)	(0.074)	(0.058)	(0.060)	(0.052)	
CNN	0.27	0.30	0.32	0.37	0.42	0.46	Validation
	(0.24)	(0.029)	(0.029)	(0.032)	(0.040)	(0.047)	
QDA	**0.69**	**0.77**	**0.80**	**0.82**	**0.83**	**0.83**	Validation
	(0.077)	(0.059)	(0.036)	(0.029)	(0.034)	(0.019)	
SVM	0.66	0.73	0.76	0.77	0.77	0.79	Validation
	(0.081)	(0.049)	(0.056)	(0.059)	(0.071)	(0.059)	
Tree	0.19	0.20	0.22	0.23	0.24	0.24	Validation
	(0.019)	(0.020)	(0.023)	(0.022)	(0.024)	(0.020)	

Table 4.3: Table presenting empirical SP mean and standard deviation, SD, values for each class using all of the observations. Note that these mean values are similar to the theoretical SP values presented in Table 4.1, despite some deviations.

n	Mean	SD
3	0.32	0.01
4	0.51	0.03
5	0.56	0.04
6	0.63	0.03
7	0.64	0.07
8	0.69	0.03

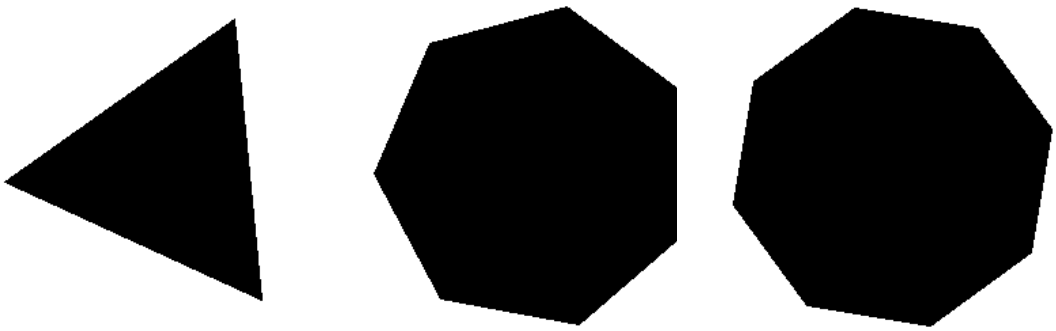
For the shapes with a larger scale, the CNN model had little issues distinguishing the shapes from one another. By the third dropout layer, the CNN was able to adequately capture the three shapes without any issue. These are shown in Figure 4.9.

However, the triad of layers change when we analyze shapes at a small scale. Figure 4.10 shows the third dropout layer for the shapes with a circumradius length of 25. Notice that the CNN is still able to distinguish the triangle from the other shapes. The CNN is unable to capture an adequate number of sides for the heptagon and octagon shapes. Thus, more scale is needed for shapes that are more similar.

(a) Figure is the input triangle with a circumradius length of 25.

(b) Figure is an example of the input heptagon with a circumradius length of 25.

(c) Figure is an example of the input octagon with a circumradius length of 25.

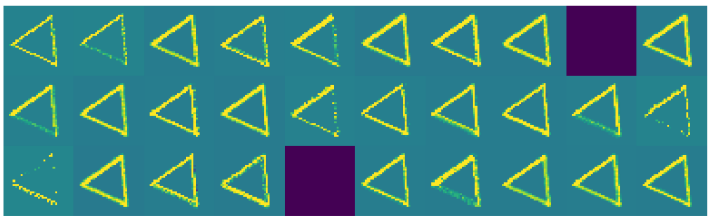


(d) Figure is an example of the input triangle with a circumradius length of 120.

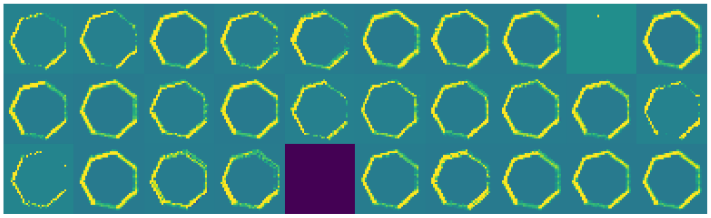
(e) Figure is an example of the input heptagon with a circumradius length of 120.

(f) Figure is an example of the input octagon with a circumradius length of 120.

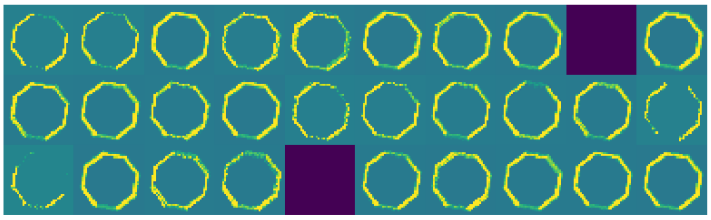
Figure 4.8: Figure presents the example input images to investigate the CNN layers. All of these shapes obvious to the human eye that they belong to their respective classes.



(a) Figure shows the third dropout layer of the triangle with a circumradius length of 120.

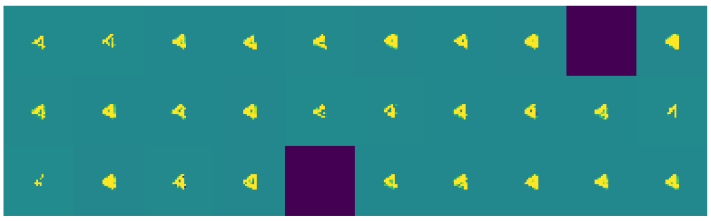


(b) Figure shows the third dropout layer of the heptagon with a circumradius length of 120.

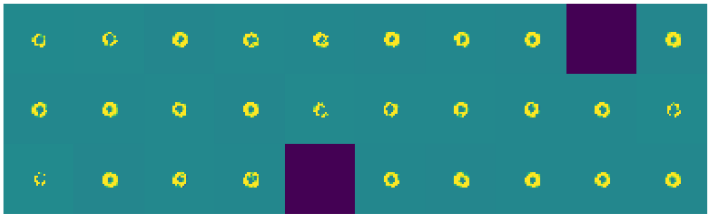


(c) Figure shows the third dropout layer of the octagon with a circumradius length of 120.

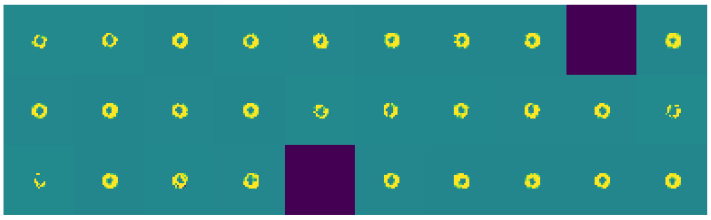
Figure 4.9: Figure shows the third dropout layer for the triangle, heptagon, and octagon with a circumradius length of 120. Notice that with a large scale, the CNN model easily captures all of the sides of each regular polygon.



(a) Figure shows the third dropout layer of the triangle with a circumradius length of 25.



(b) Figure shows the third dropout layer of the heptagon with a circumradius length of 25.



(c) Figure shows the third dropout layer of the octagon with a circumradius length of 25.

Figure 4.10: Figure shows the third dropout layer for the triangle, heptagon, and octagon with a circumradius length of 25. Notice that with a small scale, the CNN model is not able to distinguish heptagons and octagons. Thus, for similar shapes, a large scale is required.

I thought that the problem might be that the CNN model did not have enough observations or instances at a low scale to classify the classes. However, this is not the case. The main point is that the inclusion of additional instances at low and high scale worsened the overall accuracy. Thus, CNN models are ill-equipped to handle cases where some of the classes are similar and instances within classes have a large variation in scale.

Table 4.4 provide a summary of the CNN models built with increasing training set sizes. By increasing the data for training and validation with a large range in scale, the performance of the CNN model decreases. Thus, having adequate scale for classes that are similar must be part of the data collection for CNN models to be viable in training and validation.

Table 4.4: Table provides the overall mean accuracy of CNN size experiment for the training and validation data. These results are lower than the value provided in Table 4.2. Thus, providing additional images at a lower scale for very similar shapes, such as heptagons and octagons, worsened the overall accuracy of the CNN model on both the training and validation data.

Training Set Size	$n = 10$	$n = 50$	$n = 100$	$n = 250$	$n = 1000$	$n = 5000$
Training Accuracy	0.25	0.24	0.17	0.17	0.17	0.17
Validation Accuracy	0.17	0.18	0.17	0.17	0.17	0.17

SPEI Invariation Experiment

A scatterplot of the EIs of the resulting rotated images are provided in Figure 4.11. The results of the model are provided in Table 4.5. The QDA model was still able to perform well and outperform the CNN results provided in Table 4.2. This experiment shows that SPEI models are fairly robust to orientation and still outperform CNNs.

Figure 4.11: Figure shows the counts for the EIs of the shape images that were rotated. In the Legend, the value of n indicates the number of sides on the regular polygon class. Note that the EIs both follow linear lines. The final resolution ranged from 6^2 to 256^2. Note that clear separation is evident, but the lines are closer together than compared in Figure 4.6b.

Table 4.5: Table compares the overall accuracy of the original QDA model validation data against the rotated polygon data. The standard deviation is provided below the means. The models used to predict the classes for the rotated data were the same models presented in Section 4.3.1. In other words, the models never used the rotated images to train the model.

Data Set	$n = 3$	$n = 4$	$n = 5$	$n = 6$	$n = 7$	$n = 8$
Original Validation	0.69	0.77	0.80	0.82	0.83	0.83
	(0.077)	(0.059)	(0.036)	(0.029)	(0.034)	(0.019)
Rotated	0.55	0.62	0.64	0.65	0.66	0.68
	(0.096)	(0.062)	(0.045)	(0.023)	(0.022)	(0.021)

4.3.2 Pill Shapes

In this section, I compare SPEI-based approaches to CNNs in a small data setting. I only used a subset of the classes as these classes were labeled with polygonal names such as "Triangle". This allowed us to increase the complexity of the data classified, analyze shapes that deviated from regular polygons that has an apparent practical use.

Models

A similar approach as described in Section 4.3.1 was employed here. I built and compared the CNN and SPEI approaches. I treated this as a data starved learning setting. The training set size per class ranged from 4 to 6 observations, while I assigned the remaining in each setting to the validation set.

For the collection of the EIs, each input image was processed using Equations 4.6 alongside 4.15. Then, I built three separate models using the training data. Then using the built model, I used the validation data to compute the accuracy of the model. The two models computed in R used SVM using a radial kernel and a tree. Note that I was unable to build a QDA model in this scenario as the algorithm in R failed to compute the boundaries.

For the CNN approach, the input images were computed from Equation 4.16. The CNN architecture and implementation remained unaltered.

Image Operators

As the images are not binary, they must first be converted to grayscale. Next, a Sobel edge detector is applied. Lastly, thresholding is applied to convert the images to binary. After, the SPEIs are computed. This is summarized in image operator notation as

$$\mathbf{a}[\vec{p}] = \Gamma_{>10}\mathcal{S}\mathcal{L}_L\mathbf{r}[\vec{p}], \tag{4.15}$$

where $\mathbf{r}[\vec{p}]$ is the color input image, $\mathcal{S}$ is the Sobel edge detector, and Γ is the threshold operator, where those pixel values greater than 10 are retained. $\mathbf{a}[\vec{p}]$ is then put into Equations 4.9 and 4.10.

For the CNN model, we changed the image operators slightly. From Equation 4.15 we apply

$$\mathbf{o}[\vec{p}] = \Downarrow_{10} \mathbf{a}[\vec{p}], \tag{4.16}$$

where $\Downarrow_{10}$ is the downsample operator. Our input image for the CNN model is the result $\mathbf{o}[\vec{p}]$.

NLM NIH Pill Results

The classification results are summarized in Table 4.6 and Figure 4.12. The CNN and SVM models initially perform similarly. Note the in the data starved setting of 3, the CNN model did outperform the other approaches on the validation data. However, as the training size increase, the performance of the SVM model outpaces the CNN model. However, the SVM model still has a larger gap between the training and validation data as the training size increases. When using the best SPEI model for a given training size per class, this corresponds to a mean increase of outperforming CNNs by about 5.76%. Recall that the SVM model only used the EI values. We opted for a simpler approach to showcase the utility of SPEIs as a simple yet effective method to analyze shapes. This provides evidence in favor of Hypothesis 2 from Chapter 1.

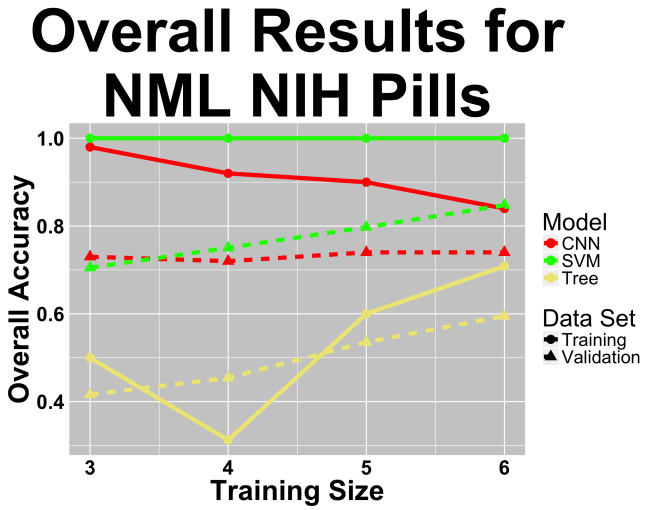


Figure 4.12: Figure shows the overall classification rate over the various settings. For example, the training size of value 3 means that there were 3 observations per class in the training data. The CNN, SVM, and Tree results are provided in red, green, and yellow, respectively. The training data set results are provided in solid lines with circular points while the validation sets are presented in dashed lines with triangular points. Note that the in all training size settings except 3, the SPEI-based SVM model outperforms the CNN model on the validation data.

SP Value Analysis

I also analyzed the SP values for the four classes empirically. The mean and SDs are provided in Table 4.7. Note that these classes are unique shapes from the regular polygons as provided in Tables 4.1 and 4.3.

4.3.3 Galaxy Shapes

In this section, I analyzed the galaxy shape data described in Chapter 3. This data provides a classification problem with irregular classes, unknown SP values, and has a practical application.

Models

I investigated the same data deprived setup we have described in Sections 4.3.1 and 4.3.2. The SVM model used a linear kernel. The training set sizes per class were 3, 4, 5, 7,

Table 4.6: Table shows overall mean accuracy results for each of the models on each of the data sets for the pill data. The number of observations in each of the training data set classes is provided in the column header for each of the data starved settings. Note that the in all training size settings except 3, the SPEI-based SVM model outperforms the CNN model on the validation data.

Model	3	4	5	6	Data Set
CNN	0.98	0.92	0.90	0.84	Training
	(0.045)	(0.066)	(0.076)	(0.010)	
SVM	1.00	1.00	1.00	1.00	Training
	(0.000)	(0.000)	(0.000)	(0.000)	
Tree	0.49	0.53	0.60	0.62	Training
	(0.070)	(0.104)	(0.065)	(0.068)	
CNN	**0.73**	0.72	0.74	0.74	Validation
	(0.105)	(0.118)	(0.112)	(0.126)	
SVM	0.70	**0.75**	**0.80**	**0.85**	Validation
	(0.078)	(0.080)	(0.105)	(0.112)	
Tree	0.42	0.45	0.54	0.59	Validation
	(0.082)	(0.090)	(0.093)	(0.095)	

10, and 20. In each setting, we assigned the remaining observations to the validation set. Here I investigated CNN and SPEI approaches. The CNN architecture and implementation was unaltered from previous experiments. An empirical analysis of the SP values is also provided.

Image Operators

For the models using SPEIs, I performed the following image operators as described in Section 4.2.3 except that $\mathbf{d}[\vec{p}]$ was created by

$$\mathbf{d}[\vec{p}] = \mathcal{I}_{(1)}\Gamma_{\mathcal{O}}\mathcal{L}_L\mathbf{a}[\vec{p}] \tag{4.17}$$

where $\mathcal{L}_L$ converts the image to grayscale, $\Gamma_{\mathcal{O}}$ applies the Otsu threshold [59], and $\mathcal{I}_{(1)}$ isolates each object in the image and returns the largest one. The CNN model simply used the results from Equation 4.17 as the input image for the model building process.

Table 4.7: Table shows the empirical mean and SDs of the SP values for each class of the NLM NIH pill data.

Class	Mean	SD
Triangle	0.54	0.03
Square	0.66	0.02
Pentagon	0.64	0.11
Hexagon	0.68	0.02

Galaxy Results

Figure 4.13 and Table 4.8 provide summaries of the results. In the data starved setup, the CNN model performed worse in terms of accuracy on the validation data while the QDA model performed best on the validation data for higher training sizes. In contrast, for lower training sizes, the SVM model performed best on the validation data. When using the best results from the SPEI-based models, this corresponds to a mean increase in performance over CNNs of about 14.9% on the validation data. This provides evidence in favor of Hypothesis 2 from Chapter 1.

Figure 4.14 provides the EI plot for the galaxy shapes. Notice that the edge class is somewhat linear and is fairly separated from the two remaining classes. While there is substantial separation between the spiral and ellipse classes, there is some overlap. This is reasonable, as some spiral galaxies will ahve shorted arms and look fairly similar an ellipse galaxy.

SP Values Analysis

An analysis regarding the SP values is also presented in Table 4.9. The Edge class is the most unique with a mean SP value of 0.15, while the Spiral and Ellipse classes are more similar. This is reflected in Figure 4.14 as the Spiral and Ellipse classes are closer to one another.

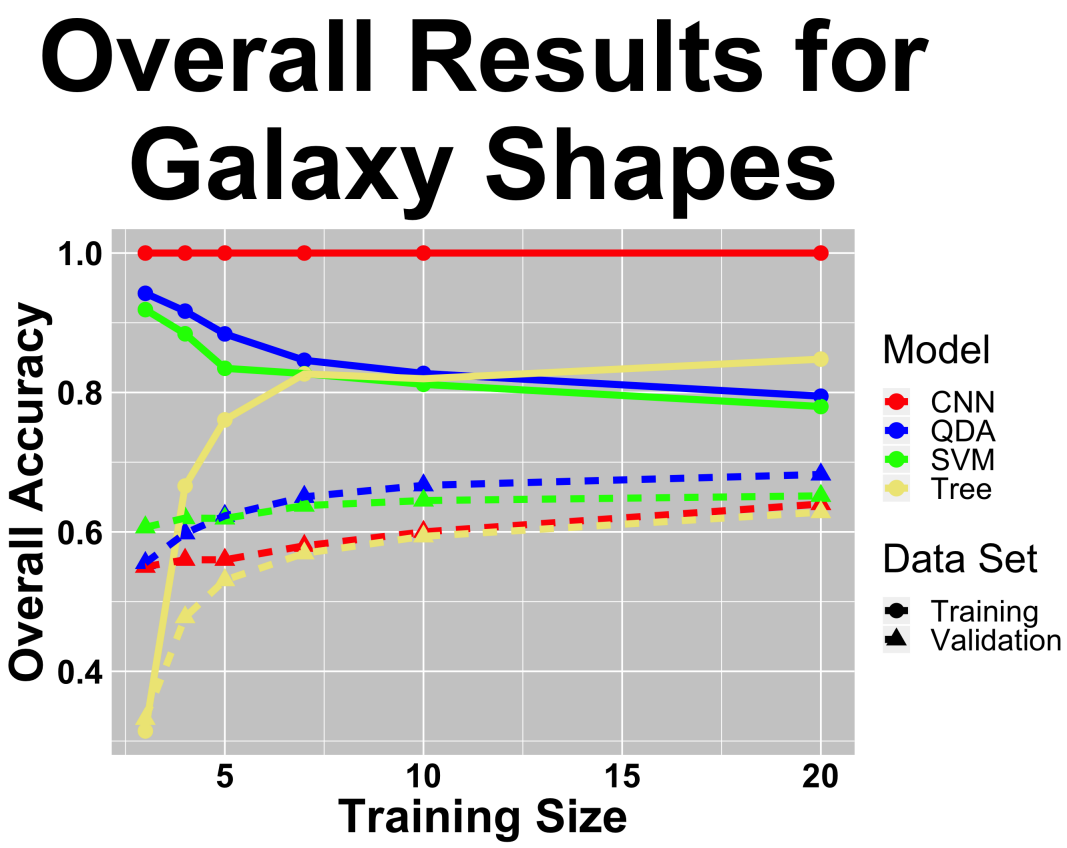


Figure 4.13: Figure shows the overall classification rate over the various settings. For example, the training size of value 3 means that there were 3 observations per class in the training data. The CNN, QDA, SVM, and Tree results are provided in red, blue, green, and yellow, respectively. The training data set results are provided in solid lines with circular points while the validation sets are presented in dashed lines with triangular points. Generally, the training data is on the top half of the figure, while the validation data results are near the middle. This plot shows that the SPEI-based models of SVM and QDA have less overfitting and better predictive capacities than CNN.

Figure 4.14: Figure shows calculated EIs from the Galaxy data set. The Edge and Ellipse classes tend to follow a linear cloud like pattern that are super and sub linear respectively. The Spiral class is a sub-linear cloud that somewhat overlaps with the Ellipse class. Note that despite being non-polygonal shapes, each class follows a somewhat linear cloud-like pattern. This is what we expect as indicated by Figure 4.6b.

Table 4.8: Table shows overall mean accuracy results for each of the models on each of the data sets for the galaxy images. The number of observations is provided in the column header for each of the data starved settings. Note that the in all training size settings, that either the SPEI-based SVM or QDA model outperforms the CNN model on the validation data.

Model	3	4	5	7	10	20	Data Set
CNN	1.00	1.00	0.99	0.95	0.89	0.94	Training
	(0.000)	(0.008)	(0.022)	(0.050)	(0.060)	(0.037)	
QDA	0.94	0.92	0.88	0.85	0.83	0.79	Training
	(0.077)	(0.084)	(0.077)	(0.079)	(0.067)	(0.045)	
SVM	0.92	0.88	0.83	0.83	0.81	0.78	Training
	(0.108)	(0.110)	(0.109)	(0.093)	(0.082)	(0.084)	
Tree	0.31	0.67	0.76	0.83	0.82	0.85	Training
	(0.137)	(0.008)	(0.096)	(0.069)	(0.064)	(0.041)	
CNN	0.53	0.54	0.55	0.56	0.57	0.60	Validation
	(0.036)	(0.034)	(0.043)	(0.034)	(0.039)	(0.031)	
QDA	0.56	0.60	**0.62**	**0.65**	**0.67**	**0.68**	Validation
	(0.091)	(0.063)	(0.060)	(0.044)	(0.035)	(0.017)	
SVM	**0.61**	**0.62**	**0.62**	0.64	0.64	0.65	Validation
	(0.074)	(0.068)	(0.074)	(0.063)	(0.062)	(0.080)	
Tree	0.33	0.48	0.53	0.57	0.59	0.63	Validation
	(0.022)	(0.053)	(0.065)	(0.068)	(0.052)	(0.039)	

Table 4.9: Table shows the empirical SP means and SDs for each of the classes.

Class	Mean	SDs
Edge	0.15	0.04
Spiral	0.41	0.10
Ellipse	0.50	0.07

4.3.4 MPEG-7 Shapes

In this section, I analyzed the MPEG-7 data. This allowed us to analyze shapes with unknown SP values with various imperfections using well known data to the Computer Vision community.

Models

A similar experimental setup is utilized here as was described in Sections 4.3.1, 4.3.2, and 4.3.3. The SVM model used a linear kernel. The training set size per class was only 4. In the setting, I assigned the remaining instances to the validation set. Here I used only CNN and SPEI approaches. The CNN setup remains the same. The SPEI-based models were LDA, SVM, and LR.

MPEG-7 Results

Table 4.10 summaries the results. In the data starved setup where each class had 4 observations for the training data, the CNN outperformed the SPEI-based approach on the validation data by about 2.4%. However, the difference between these results is less than 2 observations. Thus, both approaches preformed similarly on the validation data. Further, the CNN model provided a much larger drop between the training and validation data than the SPEI-based approaches. Thus, the SPEI-based approaches provide a much more realistic model than the CNN model does. This experiment shows that SPEI can perform similarly to CNNs for obscure shapes that have varying deformations such as occlusions and orientations in small data scenarios.

Table 4.10: Table shows overall mean accuracy results for each of the models where each class has 4 observations for the training data for the MPEG-7 shape data. The standard deviations are provided in parentheses. The overall accuracy is provided in each cell. The columns correspond to the overall accuracy for the training or validation data. While the CNN model outperformed the LR model by about 2.4%, this corresponds to a difference of less than 2 observations. Thus, the results are fairly similar for this experiment.

Model	Train	Validation
CNN	1.00 (N/A)	**0.83** (0.044)
LDA	0.87 (0.084)	0.79 (0.048)
SVM	0.97 (0.067)	0.76 (0.073)
LR	0.96 (0.061)	0.81 (0.068)

4.4 Discussion

The first major conclusion is the performance of SPEI-based models over CNN models. Next are some important factors to consider while using SPEIs in an analysis. Third, I will discuss when it is appropriate to use CNNs in small data problems. Lastly, there are broader impacts of SPEIs for the scientific community.

4.4.1 SPEIs Outperform CNNs

Overall, SPEI-based approaches were able to outperform the CNN models fairly consistently. Whether the data was simulated images or non-polygonal galaxy images, the SPEI-based approaches are effective at describing shapes in an intuitive and simple manner. For instance, when the images within each class are fairly similar to one another, interpreting the EIs will be fairly straightforward as seen in Figures 4.6b and 4.14. When using the best results from the SPEI-based models, the created polygon, NLM NIH pill data, and galaxy shape images had an improvement over the CNN models of about 128%, 5.75%, 14.9%, respectively. For the MPEG-7 data, the CNN model outperformed the SPEI-based approach by 2.4%. The overall mean performance improves 52% when using SPEI-based models over CNNs. Thus, SPEI-based approaches are a more effective means for models to classify images than CNNs in data starved settings, particularly when the relative scale of some classes is small. These results provide evidence in favor of Hypotheses 1 and 2 from Chapter 1.

Furthermore, SPEI-based models are much more computationally efficient than CNNs. While we did not perform a formal analysis, computing the SPEI-based models on the NLM NIH pill data for training setting of 3 and 6 took about 5.3 and 18.89 minutes, respectively. The CNN models took longer. For the training settings of 3 and 6, the CNN models took about 551 and 1,167 minutes, respectively. We ran the analyses on a 2011 iMac with a 3.4 Ghz Intel Core i7 processor, 16 GB 1333 MHz DDR3 memory, and a AMD Radeon HD 6970M 1024 MB graphics chip. These results are fairly representative of the other instances and experiments. In short, the CNN models take much longer to compute than all of the

SPEI-based models combined.

4.4.2 Considerations for using SPEIs in an Analysis

There are two separate ways to use SPEIs. The first is for the classification of shapes. The second is for the description of shapes.

One of the benefits of SPEIs is that a modeler can use a variety of different classification methods with it. For example, the QDA and SVM models using the EIs as variables provided substantially better and consistent predictive results than the CNN approach in small data problems. We plan on attempting other classification models in future work. Nevertheless, a researcher can use SPEI-based metrics in a variety of different classification algorithms.

Scientists can use SPEIs to understand how classes of shapes exist as 2D digital images. For instance, I analyzed the SP values in the created polygons, pill shape, and galaxy experiments. Further, we can interpret the EI plots, as seen in Figures 4.6b and 4.14, to visualize the black and white pixel counts for each of the classes. This allows us to understand and observe how classes of shapes behave as 2D digital binary images.

It is important to recognize that if two shapes have similar SP values, then SPEI-based models will struggle to discriminate the classes. We can circumvent this by increasing the resolution or total number of pixels captured for analysis.

Lastly, SPEIs can be used in conjunction with other shape metrics for the classification or description of shapes. While this was not explored in this chapter, this is explored in Chapters 5 and 6.

4.4.3 CNN Suitable for Problems with Large Scale and No Human Knowledge

CNNs are powerful tools. It is best utilized when data is plentiful and similar classes have enough scale to discriminate the shapes. A CNN model can be viable in small data scenarios, but the shapes must be substantially different from one another. If not, using

techniques to increase the amount of data to train the model where the scale within a class greatly differs and any given two classes are similar may actually worsen overall accuracy.

CNNs are more suited to analyzing and classifying shapes when solely machine driven. However, SPEI-based models are able to outperform CNNs if we are permitted to use human constructed features. SPEI encodes human knowledge about shapes, but CNNs are not able to learn human concepts such as area. Thus, SPEI-based models outperform CNNs due to the human knowledge encoded into the algorithm itself.

4.4.4 Broader Impact of SPEIs

Researchers can use SPEIs in conjunction with other shape metrics to solve a specific problem. For example, a biologist could use SPEIs alongside other metrics to understand how non-malignant and malignant cells exist and behave.

Further, SPEIs provides a clear and interpretable encoding of shapes as they exist as 2D images. We were first able to mathematically describe classes from our imaginations. Next, we were able to encode this mathematical formulation in an efficient manner for computational models to interpret. Thus, SPEI's resulting metrics, SP and EI values, are able to provide a practical interpretation of shapes which are realized as 2D digital images.

4.4.5 Future Work

Future work would extend CNNs to incorporate human knowledge of shapes such as area or SP values. This could greatly extend the performance of CNNs in small data scenarios.

Further, a new modeling approach which would capture the slight variations in orientation would allow for a robust SPEI-based model. I suspect that a model that could exploit the linear relationship of EIs would be very effective.

Lastly, I would like to investigate the case where two different classes have the same SP values. I suspect that comparing the statistical distributions of overlapping orientations may provide further insight to discriminate classes.

4.5 Conclusions

SPEIs are a more effective metric for classifying and analyzing shapes than CNNs in small data scenarios. The SP values directly correspond to the EIs and are helpful for classification purposes. The models built using these metrics are competitive in a variety of data starved settings for learning from image shape data. Every example presented provided some benefit over CNNs, whether it was outperforming the accuracy on the validation data, the predictive consistency, or the simplicity of the model, while also providing meaningful insight to how shapes exist as 2D digital images.

Chapter 5: Pill Shape Classification for Small Data with Human-Machine Hybrid Explainable Model

This chapter presents a highly accurate interpretable solution for pill shape classification. I arrived at a human-machine hybrid approach that achieved an overall classification rate of 97.83% and a mean precision of 98.4%. The only misclassifications occurred between ovals and capsules. This corresponds to an average outperformance of 94% compared to the results of other approaches when using mean precision. MY final model used a decision tree where each node classified meta-classes, or groups of classes. Each node used support vector machines with a polynomial kernel. The tree was able to overcome imbalanced data between the classes by using meta-classes. Each node of the decision tree was limited to using only two variables. This made each node interpretable as the final decision boundaries can be plotted on a 2D scatterplot. This chapter provides evidence in favor of Hypotheses 3 and 4 from Chapter 1.

5.1 Introduction

A system to identify pills would be useful to global and local communities. Prescription drug use is on the rise in the United States [24, 37]. This increasing trend is not limited to the United States, as the United Kingdom faced a similar increase [96]. In an exploratory study performed in Norway, over half of the thirty patients were given the wrong medication due to poor communication between health care officials [17]. Deaths regarding opioids have also increased in the United States [36]. Developing a system to improve the appropriate utilization and distribution of opioids is needed [36]. This system, a method to identify pills automatically, is desirable by law enforcement agencies, the health care industry, and consumers.

The ubiquity of smartphones and affordable, high-quality cameras allows for users to take pictures effortlessly. This allows for pills to be potentially identified by both medical professionals and consumers. Nurses and medical technicians would be able to verify the administration of pills to patients [51]. Multiple research communities have renewed interest in discriminating between fake and real prescription pills [65]. Furthermore, the Food and Drug Administration has advocated for creating a system to monitor patient opioid intake [36]. The National Institute of Health's National Library of Medicine (NLM) hosted a competition in response to some of these issues [58]. Researchers have yet to find a perfect solution for pill identification. This paper provides an interpretable and effective model to classify pill shapes - a key part of the pill identification problem.

5.1.1 Outline of Proposed Method

In this chapter, I present a HMH decision tree with a total of 7 interpretable metrics. This model outperforms other approaches. I trained the node using a max of 113 observations for each node in the decision tree. Of these observations, 75 came from the three largest classes: round, capsule and oval. Each of these contributed 25 observations. The remaining classes used half of the total number of observations for the training data. This ranged from 2 to 6 observations for a given class. Each decision node utilized 2 variables with a SVM using a polynomial kernel. This allows users to interpret the results with ease.

First, I will discuss pill shape identification and a general description of our HMH decision tree in the Methods and Materials section. This will show how my approach and metrics are interpretable. Second, I will build the HMH decision tree and report the performance in the Results section. This will show that my model is the best model at present for pill shape classification. Third, I mention the major takeaways from the model in the Discussion section. Last, I end with a summary of the paper in the Conclusion section. These last two sections will explain that our model is competitive and interpretable, the importance of some of the variables used, how our approach improves shape metric collection over previous solutions, and how our approach is a combination of machine and human

learning.

5.1.2 Contributions

In this chapter, I provide a HMH decision tree trained in a small data setting with a total of seven variables. This approach outperforms other modeling approaches [51]. Furthermore, this approach is interpretable and intuitive. This allows the modeler to understand the decisions boundaries with ease as each decision node is restricted to two variables. This chapter will show that digital pill shapes have distinct shapes which can be effectively discriminated using a HMH decision tree model.

5.2 Methods and Materials

I will discuss the technical details of our data, metrics used for our model, and our HMH decision tree. This will show that the metrics and model built make our approach interpretable.

5.2.1 Data

I used the NLM reference data from the recent Pill Image Competition. This data is described in detail in Chapter 3. All of the classes were used.

5.3 Results

I made the figures showing the results in R while also utilizing some graphics packages, such as ggplot2 [11, 35, 57, 85, 88, 89]. I provide the code for the experiments and binary shapes at the provided GitHub link: `https://github.com/billyl320/human_decision_tree_pills`.

5.3.1 Shape Segmentation

I inspected the pills' shapes manually after applying the image operators from Equation 3.3. None of the binary shapes have any distortion or abnormalities. An example of an initial capsule image and its corresponding segmented shape image are provided in Figures 5.1 and 5.2, respectively. Thus, the shapes of the pills are accurate.

Figure 5.1: Figure shows an example of the capsule image before the image segmentation was performed. The resulting shape is shown in Figure 5.2.

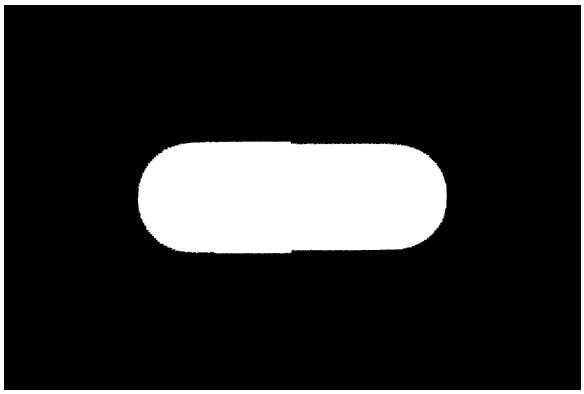


Figure 5.2: Figure shows an example of the capsule image shape of Figure 5.1 after the image segmentation was performed.

5.3.2 Metric Collection

The metrics used in the model were SP values, circularity, EIs, eccentricity, and the minimum bounding box counts. Figures 5.3, 5.4, and 5.5 provide scatterplots of the observations alongside some of the metrics. Upon inspection of the scatterplots, groups of classes are clearly separable. For example, in Figure 5.3, the oval, rectangle, round, and capsule classes are clearly separable from the other remaining classes. Thus, by subdividing the classification task into a series of easier classification tasks, we built an effective pill shape classification model.

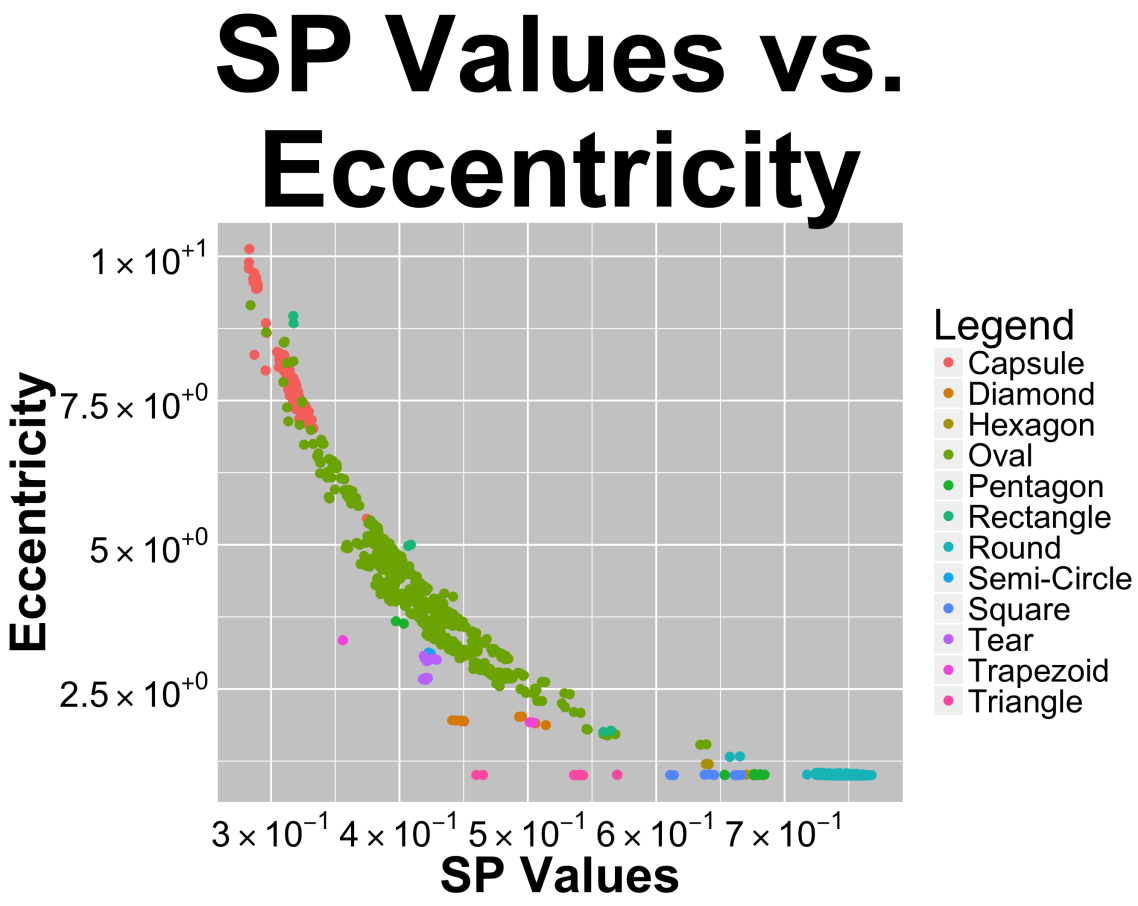


Figure 5.3: Figure shows the SP and Eccentricity values. Note that the capsule, oval, round and rectangle classes are separable from the remaining classes.

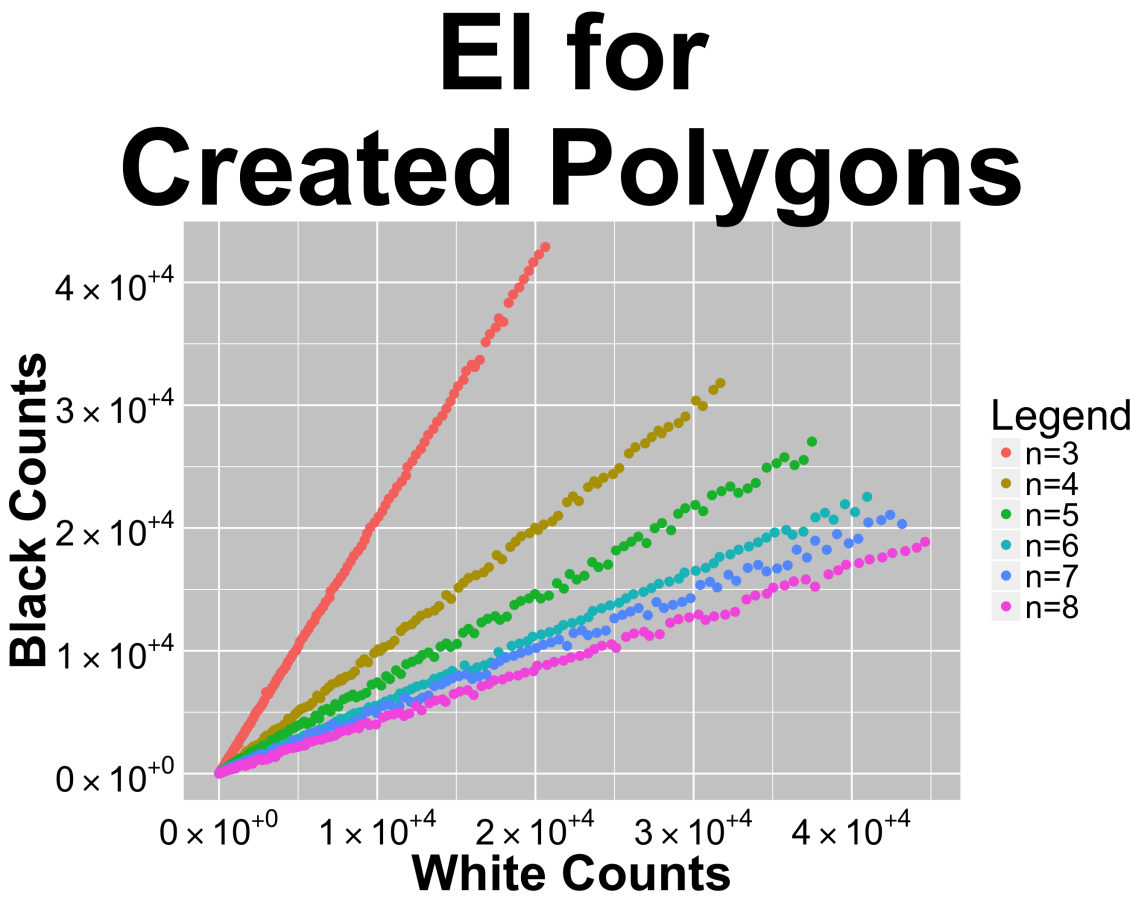


Figure 5.4: Figure shows the EI values of the pill shapes. Some of the classes follow a clear linear pattern, while others do not. Unfortunately, many of the classes overlap with one another. We were able to use these variables once the overlapping classes were separated by using meta-classes.

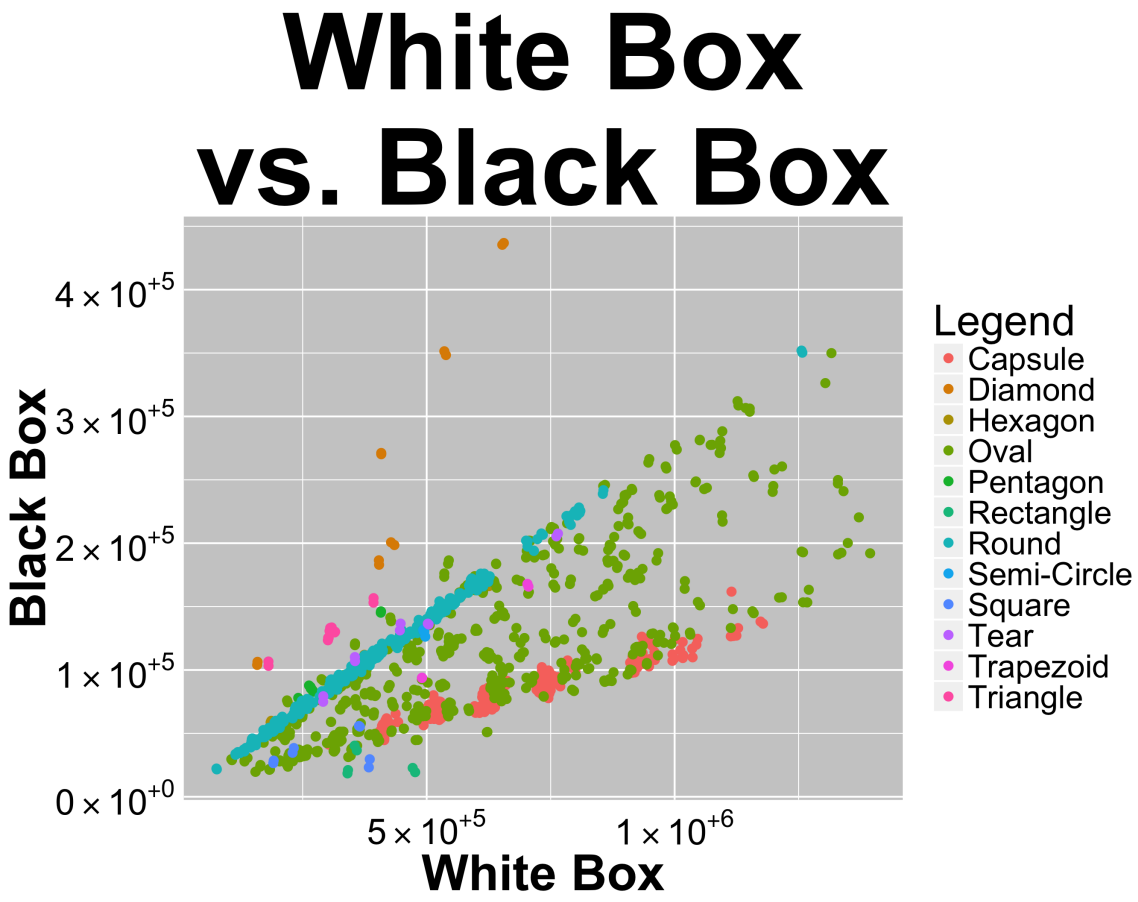


Figure 5.5: Figure shows the minimum bounding box black and white pixel counts for the shape data. This provides a similar perspective to the EI plot in Figure 5.4. However, some of the classes become easier to discriminate using this boxing algorithm instead. This is similar to the case of the EIs in Figure 5.4. Once the overlapping classes were separated through the use of meta-classes, these variables helped to discriminate some of the classes.

Figure 5.6: Figure shows an example of the regular hexagon image before the image segmentation was performed. This stratum of the hexagon class had a total of six observations.

5.3.3 Model

The final model was an HMH decision tree where each decision node used only two variables and an SVM classification algorithm using a polynomial kernel. The parameter values for each node are provided in Table 6.5. This approach provides an interpretable and accurate model.

I utilized stratified random sampling for partitioning the data to the training and validation data [16, 28, 35]. The advantage os using stratified random sampling as opposed to simple random sampling is a reduciton in error for parameter estimation [16]. For our case, I treated each of the classes as individual stratum except for the hexagon strata. I split the hexagon class into two strata. There were two non-regular hexagons and six regular hexagons. Examples of the regular and non-regular hexagon observations are provided in Figures 5.6 and 5.7, respectively. I included one non-regular hexagon and three regular hexagons in the training data set. The final counts of the training and validation sets are provided in Table 5.1.

Figure 6.2 provides an example of the results of the first decision node in the decision tree. I was able to classify the first two meta-classes perfectly using SP and eccentricity. The first meta-class was oval, capsule, rectangle, and round. The second meta-class included the remaining classes. While Figure 6.2 presents only the training data, the node was also able to perfectly classify the validation data as well.

Interpreting the decision boundary in Figure 6.2 is straightforward. The round, capsule, oval, and rectangle classes range from having large SP values with small eccentricity to

Figure 5.7: Figure shows an example of the non-regular hexagon image before the image segmentation was performed. This stratum of the hexagon class had a total of two observations.

Table 5.1: Table shows counts for the training and validation data sets. There were two non-regular hexagons and six regular hexagons. Thus, one non-regular hexagon and three regular hexagons were randomly sampled. The other classes were treated as individual stratum. Those observations in the stratum were randomly assigned to the training data.

Class	Training Count	Validation Count
Capsule	25	307
Diamond	6	6
Hexagon	4	4
Oval	25	661
Pentagon	6	6
Rectangle	3	3
Round	25	881
Semi-circle	2	2
Square	4	4
Tear	5	5
Trapezoid	2	2
Triangle	6	6
Total	113	2000

small SP values with large eccentricity. The second meta-class tends to have smaller SP and eccentricity values. This process of interpreting each node of the tree is repeatable.

I did not provide the plots and interpretations of the decision boundaries for every node in this manuscript for the sake of brevity. However, the code provided in the GitHub

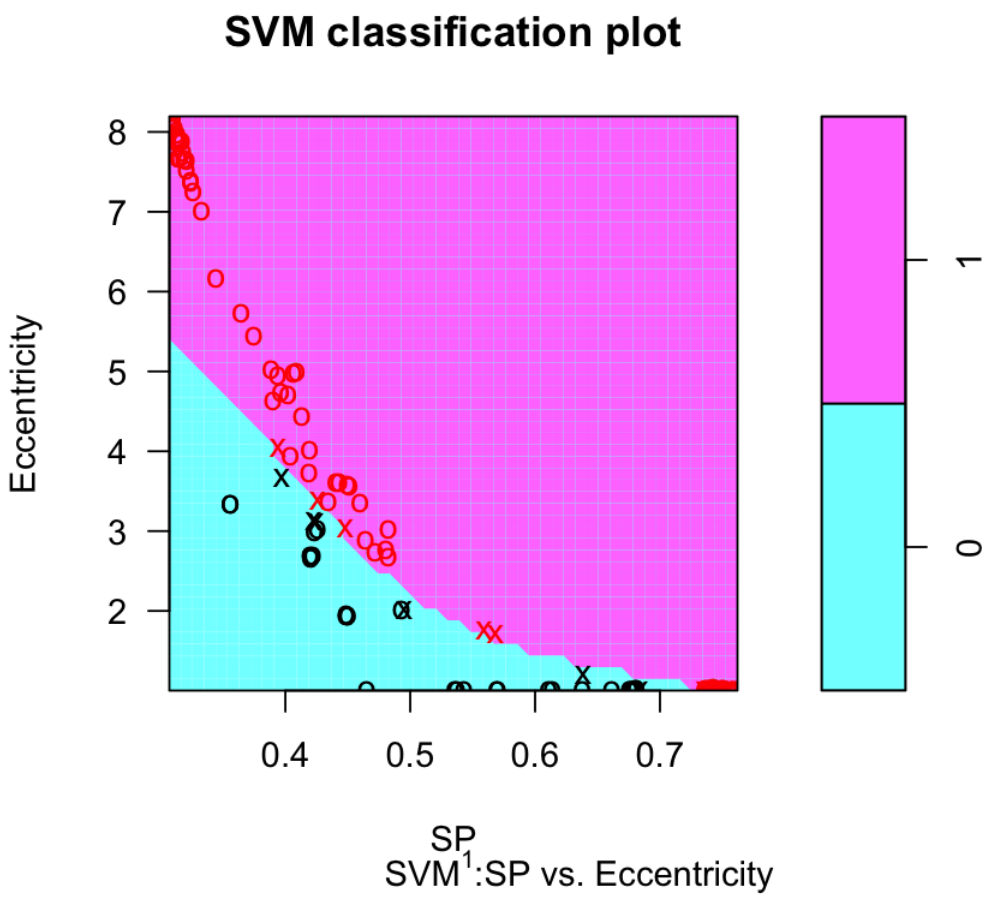


Figure 5.8: Figure shows the decision boundary made using the training data on the first decision node. This model used the SVM algorithm with a polynomial kernel with the associated parameter values of SVM^1 which is found in Table 6.5. The red points are associated with the oval, round, rectangle, and capsule observations. The black points correspond to the other classes. The Xs indicate if the model used the observation as a support vector, while the open circles are not support vectors. The pink area indicates the space where an observation would be classified as a round, capsule, oval, or rectangle. The cyan area indicates the space where an observation would be classified as a diamond, hexagon, pentagon, rectangle, semi-circle, square, tear, trapezoid, or triangle. Each node in the decision tree can have this kind of 2D plot made. Modelers and users can utilize these plots to better understand the decision-making process of the HMH decision tree. Thus, my model is highly interpretable.

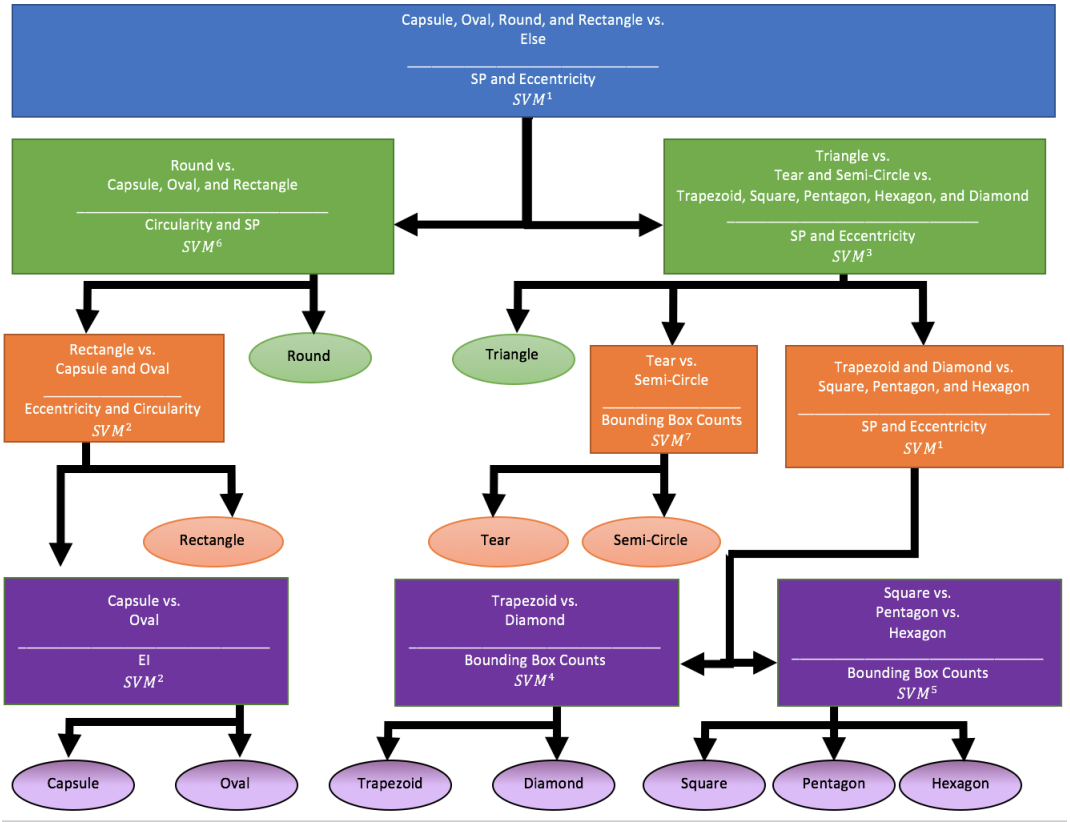


Figure 5.9: Figure shows the resulting decision tree. Each node uses SVM with a polynomial kernel with various parameter values. Table 6.5 summarizes those kernel values. The decision nodes are the rectangles. The leaves are found in the ovals. A leaf's color is determined by the final decision node which classifies the class. The HMH tree correctly classified all of the classes with the exception of discriminating between the capsules and ovals. Thus, my model is interpretable and an effective classifier.

Table 5.2: Table shows five SVM algorithms with there associated polynomial kernel parameter values.

SVM^i	Cost	coef0	Degree
1	1	2	5
2	1	1	2
3	1	1	3
4	1	1	1
5	1	50	2
6	1	1	10
7	1	2	10

link for the experiment produces one of these plots for each node: `https://github.com/billyl320/human_decision_tree_pills`. These plots provide a very interpretable method of understanding the decision tree. The final resulting decision tree is provided in Figure 5.9.

The HMH tree correctly classified all of the classes with the exception of discriminating between the capsules and ovals. It misclassified 1 observation per class on the training data. The HMH tree misclassified 70 oval observations as capsules on the validation data. These misclassifications for the training and validation data are summarized in the confusion matrices provided in Tables 5.3 and 5.4, respectively. These misclassifications only occurred in the node which classified ovals and capsules against one another. This corresponds to an overall misclassification rate on the complete data of 3.6% and a misclassification rate on the validation data of about 3.71%. Our mean precision on the complete data was approximately 98.5%. The MP for our HMH model was 98.4% when constrained to those classes that overlap with Maddala *et al.*'s model. Thus, our model is extremely accurate across all of the classes. These results provide evidence in favor of Hypotheses 3 and 4 from Chapter 1.

Several other machine-driven models were built for comparison. I built three SVM models utilizing a grid search for their parameters. The code for these models is provided

Table 5.3: Table shows the confusion matrix for the training data's capsule and oval classes. The columns correspond to the actual class, while the rows correspond to the predicted class. Note that only two observations were misclassified as ovals. Thus, my model is very accurate on the training data.

Predicted/Truth	Capsule	Oval
Capsule	24	1
Oval	1	24

Table 5.4: Table shows the confusion matrix for the validation data's capsule and oval classes. The columns correspond to the actual class, while the rows correspond to the predicted class. Note that only 70 observations were misclassified as capsules. Thus, my model is very accurate on the validation data.

Predicted/Truth	Capsule	Oval
Capsule	307	70
Oval	0	593

in the GitHub link: `https://github.com/billyl320/human_decision_tree_pills`. The three models each used a different kernel. The kernels were polynomial, raidal, and sigmoid. I also built naïve Bayes and linear discriminant analysis (LDA) models. The mean precision (MP) values for all the models are provided in Table 5.5. This table also includes the MP values for two of Maddala *et al.*'s models and my HMH model. I only considered classes which overlapped with my classes when we calculated the MPs for Maddala *et al.*'s models. My HMH model provided the largest MP value, which indicates that my model performs best across all of the classes. These results provide evidence in favor of Hypotheses 3 and 4 from Chapter 1. The confusion matrices for these comparison models are provided in Tables 5.6 to 5.12.

Table 5.5: Table with the mean precision (MP) values for various models. The first and third rows correspond to the model name. The second and fourth rows correspond to the MP values. The first model is an SVM with a polynomial kernel (SVM - P). The second model is an SVM with a radial kernel (SVM - R). The third model is an SVM with a sigmoid kernel (SVM - S). The fourth model is an NB. The fifth model is an LDA. The sixth model is the HMH adaptable tree built by Maddala *et al.*(Maddala - Tree). The seventh model is the logistic regression (LR) built by Maddala *et al.* using Hu moments (Maddala - LR). The eighth model is my HMH tree described in this manuscript (Lamberti - Tree). Maddala - LR does not have a MP value since it does not predict some classes. Thus, the MP cannot be calculated for this model. Since my approach has the largest MP, my approach performs best across all of the classes. This corresponds to an average outperformance of 101.6%.

Method	SVM - P	SVM - R	SVM - S	NB	LDA
MP	0.355	0.757	0.269	0.623	0.801
Method	Maddala - Tree	Maddala - LR	Lamberti - Tree		
MP	0.897	-	0.984		

Table 5.6: Table shows the confusion matrix for the Hu moments method from Maddala *et al.*. (Maddala - LR).

Capsule	Diamond	Hexagon	Oval	Pentagon	Rectangle	Round	Square	Tear	Trapezoid	Triangle	Octagon
231	0	0	7	0	0	0	0	0	0	0	0
0	4	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	3	0	0	0	0
2	0	0	767	0	0	4	0	0	0	0	0
0	0	0	0	0	0	8	0	0	0	0	0
0	0	0	3	0	1	0	0	0	0	0	0
0	1	0	2	0	0	1050	0	0	0	0	1
0	1	0	1	0	0	5	0	0	0	0	0
0	0	0	0	0	0	0	0	9	0	0	0
0	0	0	0	0	0	0	0	0	2	1	0
0	0	0	0	0	0	1	0	0	0	8	0
0	0	0	0	0	0	1	0	0	0	0	0

Table 5.8: Table shows the confusion matrix for the polynomial SVM (SVM - P) using the same features as the HMH model.

Capsule	Diamond	Hexagon	Oval	Pentagon	Rectangle	Round	Semi-Circle	Square	Tear	Trapezoid	Triangle
332	0	0	46	0	4	0	0	0	0	0	0
0	6	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	6	0	638	2	2	0	4	0	10	4	2
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	8	4	10	0	904	0	8	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	10

Table 5.9: Table shows the confusion matrix for the radial SVM (SVM - R) using the same features as the HMH model.

Capsule	Diamond	Hexagon	Oval	Pentagon	Rectangle	Round	Semi-Circle	Square	Tear	Trapezoid	Triangle
318	0	0	34	0	0	0	0	0	0	0	0
0	10	0	0	0	0	0	0	0	0	0	0
0	0	8	2	0	0	0	0	0	0	0	0
14	2	0	647	4	2	26	0	2	2	2	0
0	0	0	0	8	0	0	0	0	0	0	0
0	0	0	0	0	4	0	0	0	0	0	0
0	0	0	1	0	0	878	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	6	0	0	0
0	0	0	4	0	0	0	4	0	8	0	0
0	0	0	0	0	0	0	0	0	0	2	0
0	0	0	0	0	0	0	0	0	0	0	12

Table 5.10: Table shows the confusion matrix for the sigmoid SVM (SVM - S) using the same features as the HMH model.

Capsule	Diamond	Hexagon	Oval	Pentagon	Rectangle	Round	Semi-Circle	Square	Tear	Trapezoid	Triangle
270	0	0	144	0	4	0	0	0	0	0	0
0	8	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
62	4	0	532	2	2	2	4	0	10	4	2
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	6	12	10	0	880	0	8	0	0	10
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	2	0	0	0	22	0	0	0	0	0

Table 5.11: Table shows the confusion matrix for NB using the same features as the HMH model.

Capsule	Diamond	Hexagon	Oval	Pentagon	Rectangle	Round	Semi-Circle	Square	Tear	Trapezoid	Triangle
195	0	0	32	0	0	0	0	0	0	0	0
0	9	0	0	0	0	0	0	0	0	0	0
0	0	2	0	0	0	0	0	0	0	0	0
0	1	0	264	0	0	0	0	0	0	0	0
0	0	0	0	8	0	0	0	0	0	0	0
0	0	0	2	0	0	0	0	0	0	0	0
0	0	0	0	0	0	845	0	0	0	0	0
7	0	0	17	0	0	2	3	0	0	0	0
0	0	0	0	0	0	0	0	5	0	0	0
0	0	0	0	0	0	0	1	0	7	0	0
130	2	6	369	2	6	57	0	3	3	4	2
0	0	0	4	2	0	0	0	0	0	0	10

Table 5.12: Table shows the confusion matrix for LDA using the same features as the HMH model.

Capsule	Diamond	Hexagon	Oval	Pentagon	Rectangle	Round	Semi-Circle	Square	Tear	Trapezoid	Triangle
332	0	0	47	0	2	0	0	0	0	0	0
0	12	0	0	0	0	0	0	0	0	0	0
0	0	4	0	2	0	0	0	0	0	0	0
0	0	0	628	2	0	0	0	0	0	0	0
0	0	4	4	8	0	0	0	0	0	0	0
0	0	0	4	0	2	0	0	0	0	0	0
0	0	0	0	0	0	904	0	0	0	0	0
0	0	0	0	0	0	0	4	0	3	0	0
0	0	0	0	0	2	0	0	8	0	2	0
0	0	0	5	0	0	0	0	0	7	0	0
0	0	0	0	0	0	0	0	0	0	2	0
0	0	0	0	0	0	0	0	0	0	0	12

5.4 Discussion

My HMH decision tree's outperformance of other approaches is the first major remark. Second, the importance of the SP and eccentricity values for the decision tree was a noteworthy discovery. Third, I will discuss how our image segmentation treated the data better. Lastly, the approach built in this chapter is a hybrid of a human guided model and a machine learning model.

5.4.1 SPEI-Based Decision Tree More Interpretable and Accurate Across All Classes

The CNN models created we created in Chapter 4[1] were only able to achieve a max overall classification rate of 74% on their validation data [43]. I was able to obtain perfect classification. While Maddala *et al.* were able to achieve 98.7% overall classification on their data, my HMH model is able to obtain 96.4%. Furthermore, the difference between achieving these respective rates is 46 observations out of 2,000. Thus, the error rate is fairly comparable. Maddala *et al.* misclassified some of the oval and capsule observations as triangles, trapezoids, and/or diamonds [51]. Our model only misclassified ovals as capsules and vice versa. My approach is more accurate across all of the classes. Its mean precision was 98.4%, while Maddala *et al.*'s was 89.7% on the complete data. This corresponds to a 9.7% outperformance across all of the classes. Additionally, the HMH approach built in this chapter outperforms all other attempted approaches provided in Table 5.5. This corresponds to a mean outperformance rate of 94%. Ultimately, my approach was substantially more interpretable and accurate across all of the classes. This provides evidence in favor of Hypotheses 3 and 4 from Chapter 1.

5.4.2 SP Values and Eccentricity were Essential

The first node in the decision tree used only the SP values [43] and eccentricity [38]. The addition of the SP value proved invaluable. No other pair of metrics was able to provide

[1] on the same data with less classes

the first step to make classification possible. Thus, the SP value and the well-established metric of eccentricity were of paramount importance for making the classification of these observations possible. If these metrics were not used, converting this problem to a large data solution would likely be inevitable. Examples include performing data augmentation or collecting more data [7–9, 40, 53, 54, 76, 86]. These two metrics allowed us to provide a small data solution.

5.4.3 Improved Metric Collection

A major issue with Maddala *et al.*'s solution using adaptable rings is that the image segmentation required the prior knowledge of the classes [51]. Thus, they were essentially measuring two groups of classes in two different manners [51]. The solution presented in this chapter required only one image segmentation algorithm and was able to accurately capture each pill's shape. Thus, I was able to capture the shape of all of the pill shape observations in a uniform and unbiased manner.

5.4.4 Decision Tree is a Hybrid of Human and Machine Learning

This approach requires a large amount of human intervention for determining the meta-classes and variables used. It left the creation of decision boundaries to machine learning (ML) algorithms. Scatterplots were manually inspected to find candidate pairs of variables and potential meta-classes. The training and validation data was used to check if the variables and meta-classes were viable. Thus, this is not a purely ML solution. However, now that we know near perfect classification is obtainable, the ML and Computer Vision communities must be able to meet or exceed this benchmark. A potential solution is to automate the human aspects by the computer. I hope to have the computer determine which two variables to use for a given pair of meta-classes and to build the tree recursively.

5.5 Future Work

Future work would have a model specifically built to discriminate capsule and oval pill shapes perfectly. This may mean extending the number of metrics to more than two. While this may decrease the interpretability of out model, this would potentially allow for a perfect pill shape classification model. Once this was implemented, we would build separate models for pill color and text. Then these three models should be combined into a single method for pill identification.

Outside of pill identification, we would like to use SPEI in other biomedical informatics applications. Investigating the SP values for various types of cells is a clear extension. This is motivated by the SP values' crucial importance for the HMH model we built.

5.6 Conclusions

The HMH decision tree developed in this study provides improved classification and interpretability of pill shapes. Highly accurate classification is achievable using understandable and intuitive metrics. This technique outclasses previously developed approaches. The first node's use of SP values and eccentricity was vitally important for this task. These metrics simplified the complex task of discriminating numerous classes into a series of elegant classification tasks. This approach circumvented issues with optimizing loss functions for overall classification accuracy. This is historically a challenging task for imbalanced data problems. The HMH decision tree developed in this study provides a strong foundation for an interpretable pill identification model.

Chapter 6: DAMG: A Classification Algorithm for Problems with Limited Data

The decision tree with automatic model generation (DAMG) algorithm simplifies multinomial classification problems into a series of binary tasks. Classification problems are an intrinsic part of a variety of fields. Further, deep learning is seemingly capable of achieving high rates of classification. However, deep learning solutions are difficult to interpret and are prone to overfit to the training data. They also fail in small data settings, as shown in Chapter 4. The recursive DAMG uses variable selection and meta-classes to disentangle multinomial classification problems with many variables. While the example shown uses 2D pill shape image data and the standard MPEG-7 shape data, the algorithm is applicable to non-image classification problems as well. The DAMG algorithm was able to outperform other approaches by about 177% for pill shape classification and CNNs by about 156% and 148% on the training and validation data, respectively, for a subset of the MEPG-7 data. This chapter provides evidence in favor of Hypotheses 3 and 4 from Chapter 1.

6.1 Introduction

The level of complexity is high while the general interpretation of a given model is low for many deep and machine learning classification models [23, 35]. These complex models with low interpretation are utilized in a variety of different fields such as image classification [19, 23, 23, 47, 67]. Many of these models are complex due to the large number of features present [28]. Additionally, these models are difficult to explain due to the number of features used to build the model, whether learned or provided [28]. This problem is exacerbated when the number of classes to categorize is large [35].

Thus, I desire to provide an algorithm which can break down complex classification problems into a series of simpler ones. Based on variable selection and the binarization of many classes, classification models can be made substantially simpler and easier to understand. This series of simpler problems can be represented by a decision tree, which eases interpretation. Each node represents a binary classification problem using only two variables. This is done to increase the explainability of the model. The algorithm is dubbed a decision tree with automatic model generation (which is abbreviated as DAMG and pronounced "damage"). The DAMG algorithm was able to outperform other approaches by about 177% for pill shape classification and CNNs by about 156% and 148% on the training and validation data, respectively, for a subset of the MEPG-7 data.

6.1.1 Outline of Proposed Method

I discuss a novel classification algorithm called DAMG. DAMG is a recursive algorithm which first relabels the classes into binary meta-classes, or groups of classes. Then DAMG uses variable selection to select the two or three variables that are most important. I used only two variables per parent node throughout our pill shape experiments. The MPEG-7 data experiments used three variables per node. The binary classification setup then uses the two most important variables to discriminate between the two meta-classes. This is repeated until all of the classes are leaf nodes in the decision tree.

First, I will discuss the DAMG algorithm in the Methods and Materials section. This will show that DAMG simplifies classification problems. Second, we will perform an experiment on 2D pill shape data and a subset of the famous MPEG-7 data in the Results section. This will show that DAMG is applicable to real world problems and benchmark data, respectively. The Discussion and Conclusion sections will explain that DAMG improves the analysis and interpretation of classification problems with many classes.

6.1.2 Contributions

In this chapter, I provide a novel classification algorithm called DAMG. The results from DAMG are interpretable and perform well. Furthermore, this approach is simple and intuitive, which allows the modeler to gain better insight into the model. For my experiments, each parent node in the tree is restricted to two or three variables depending on the data analyzed. This paper shows that multinomial classification problems are convertible into a series of simpler binary classification problems. These results provide evidence in favor of Hypotheses 3 and 4 from Chapter 1.

6.2 Methods and Materials

In this section, we will discuss the technical details of our algorithm, data, metrics used, and the HMH decision tree used for comparison.

6.2.1 Impetus for DAMG

The DAMG algorithm is attempting to break down a single complex classification problem into a series of simpler ones. A single classification problem can have both numerous variables and classes to consider. In computer vision, the need of metrics is lessened as the features are learned from the image itself when a CNN is utilized [19,23,47]. However, both CNNs and data sets with many variables suffer from interpretability issues [23].

A potential solution for a large number of classes is to binarize the classification problem. This involves searching from combinations of the classes to form meta-classes, or groups of classes. In theory, these meta-classes group together similar classes into larger ones. This again reduces the classification problem to a simpler task and increases interpretability by only having a binary outcome.

A potential remedy for a large number of features is variable selection [28, 35]. In general, the goal of variable selection is to reduce the number of features used for modeling [28,35]. This reduces the complexity of the model while also increasing the interpretability

[28,35]. A popular variable selection technique is the least absolute shrinkage and selection operator (LASSO) [71, 83]. The LASSO is the foundation of many other approaches that deal with sparse data [50]. However, the LASSO failed to find meaningful results with limited observations in our experiments.

Thus, DAMG uses meta-classes and variable selection techniques to break down a single classification problem into a series of simpler ones. The result is a recursively built decision tree where each parent node is a simpler classification problem.

6.2.2 Mathematical Representation

Before I discuss the details of the pseducode, it is important to establish the mathematical representation of DAMG. Recall the the generalized definition of a tree from Chapter 1 in Equation 2.23. That is,

$$f(X) = M_{\mathcal{R}} I_{X \in \mathcal{R}}, \tag{6.1}$$

where $M_{\mathcal{R}}$ is the modeling operation performed on the space $\mathcal{R}$ and $I_{X \in \mathcal{R}}$ is the indicator variable for the data, X, that belongs to $\mathcal{R}$. For a regression tree, $M_{\mathcal{R}} = \sum_{m=1}^{G} c_g$ and $I_{X \in \mathcal{R}} = I_{X_i \in \mathcal{R}_g}$ where N is the number of regions and c_g is a response constant for a given g. For a classification tree, $M_{\mathcal{R}} = \sum_{x \in \mathcal{R}_g} \frac{1}{N_g}$ and $I_{X \in \mathcal{R}} = I_{Y_i = k}$ where k is the associated class and N_g is the number of observations in a given region or node, g.

The DAMG algorithm can be represented mathematically using Equation 2.23 by

$$f(x) = M_{\mathcal{R}_g} I_{X_g \in \mathcal{R}_g}, \forall g \in \{1, ..., G\} \tag{6.2}$$

where g is a given node or binarization of the given subspace, G is the total number of subspaces, X_g are the observations or instances that belong to $\mathcal{R}_g$ after the meta-class creation, and $M_{\mathcal{R}_g}$ is the modeling operation performed on the subspace $\mathcal{R}_g$.

Note that Equation 2.23 generalizes any other modeling approach. For example, if $\mathcal{R}$ was the entirety of the space X occupies and $M_{\mathcal{R}}$ is an SVM model with a linear kernel,

we are then simply performing SVM on data X. Another trivial example is to use any classification method without the use of meta-classes. This will result in a tree with a singular node and the number of children equal to the number of classes. These provides evidence in favor of Hypothesis 4 from Chapter 1.

6.2.3 Pseudocode

In this section, I will describe the general DAMG algorithm pseudocode. I will discuss the specific implementation for the analyzed pill shape data set in Section 6.2.6. In general, the DAMG algorithm first finds appropriate candidate meta-classes. Then, for each meta-class, variable selection is performed to determine the selected variables. Then, a classification model is built using the selected variables. If multiple candidate models obtain perfect classification between the meta-classes, then the one which maximizes the absolute difference between the number of instances per meta-class is selected. This encourages the tree to separate imbalanced classes. For instance, if two classes have 100 observations each and another two only have 10 observations, then the preferred meta-classes would have the 100 observation count classes in one meta-class for a total of 200 observations. The other meta-class would have two classes with the 10 observation counts for a total of 20 observations. The pseudocode for the node creation is provided in Algorithm 1 and for the tree creation in Algorithm 2.

Algorithm 1 DAMG Node Creation

```
1: procedure DAMG Node
2:     Determine m combinations of meta-classes
3:     for i ∈ 1 : m do
4:         Perform variable selection to find selected variables
5:         Create classification model using i^th meta-classes and selected
6:     if ∃ multiple models with perfect classification on the testing data then
7:         Find the absolute difference between the meta-classes' counts
8:         Select the one model and selected variables which maximizes this difference
9:     else
10:        Select the one model and selected variables which minimizes the error rate on
       the testing data
```

Algorithm 2 DAMG Tree Creation

```
1: procedure DAMG Tree Creation
2:     Initialize starting node
3:     if Node is Almost Pure then
4:         Create 2 child nodes
5:     else Create 2 child nodes
6:         if Child node 1 is Pure then
7:             Child node 1 becomes a leaf
8:         else Split child node 1
9:         if Child node 2 is Pure then
10:            Child node 2 becomes a leaf
11:        else Split child node 2
```

I will need to define the terms 'pure' and 'almost pure'. 'Pure' means that the observations in a node contain only one class. 'Almost pure' means that the node has no more than two classes. Almost pure nodes will be parent nodes. Pure nodes will always be child nodes.

At each parent node, the classes were grouped by the authors into larger groups or meta-classes. This was done for a number of reasons. The first is a practical one: this helps to counteract imbalanced datasets. Models built using all of the metrics and distinct classes would carry the risk of classifying the smaller classes as observations belonging to one of the larger classes. This is caused by using standard loss functions [29].

The second reason was due to the restriction of using variable selection at each decision node. I typically used only two variables during our experiments. This was motivated by the following example. A common approach to building a model by hand is to plot the data. A common introductory approach is the use of a scatterplot matrix. Using only two variables per node is similar to this visualization technique. Having only two variables and two classes greatly eases interpretation of each parent node and, by extension, the entire model.

The third reason is that this solution is elegant in design. While it may be possible to define a complicated loss function or modeling algorithm, this solution produced by DAMG can be easily explained to a wide technical audience. Further, I argue that this approach can be more easily explained by a wider audience base who would eventually utilize this

Table 6.1: Table shows the classes names, encoding, and counts of the classes of the NML NIH reference data. It also shows counts for the training and validation data sets.

Pill Shape	Encoding	Class Count	Training Count	Validation Count
Capsule	1	332	25	307
Diamond	2	12	6	6
Hexagon	3	8	4	4
Oval	4	686	25	661
Pentagon	5	12	6	6
Rectangle	6	6	3	3
Round	7	906	25	881
Semi-circle	8	4	2	2
Square	9	8	4	4
Tear	10	10	5	5
Trapezoid	11	4	2	2
Triangle	12	12	6	6
Total	-	2000	113	1887

model in their respective fields.

6.2.4 Data

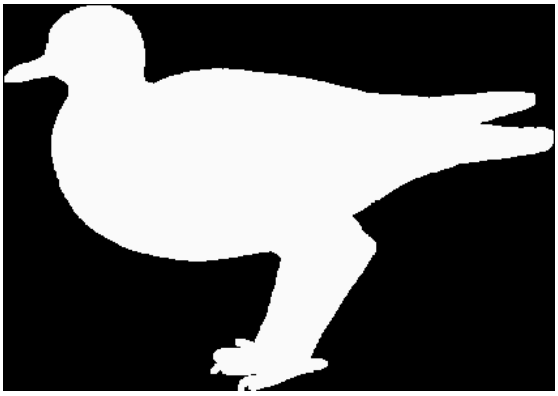
I will use the NLM NIH reference data from the recent Pill Image Competition. Table 6.1 shows the classes alongside their corresponding counts of each of the dataset. The shape segmentation and metric collection are detailed in Chapters 3 and 4. The metrics included are the SP values, EIs, eccentricity[38], circularity[25,38,68], and the white and black pixel counts from the minimal bounding box.

The MPEG-7 data is the second data set used in this manuscript. It is a commonly used benchmark for experiments in the computer vision and pattern recognition communities [46]. This data contains various orientations, missing parts, and occlusions which test for various conditions shapes may exist [46, 74]. We used a subset of the classes from MPEG-7. The classes used were bird, bone, brick, camel, and cup. Since there are 20 instances per class,

Table 6.2: Table shows the classes names, encoding, and counts of the classes of the MPEG-7 data. It also shows counts for the training and validation data sets.

Pill Shape	Encoding	Class Count	Training Count	Validation Count
Bird	1	80	56	24
Bone	2	80	56	24
Brick	3	80	56	24
Camels	4	80	56	24
Cups	5	80	56	24
Total	-	400	280	120

the total size of the original data we used was 100 observations. Additional observations were obtained by performing 3 rotations of 90°, 180°, and 270°. This resulted in a final data set of 400 observations with each class having 80 instances. The counts are summarized in Table 6.2. Figure 6.1 presents two examples of the shape images from the bird class.



(a) Figure shows an example a bird shape from MPEG-7. In this image, the bird shape is fairly accurate.

(b) Figure shows a bird shape from MPEG-7. In this shape, the bird is at an odd orientation and has large holes inside of the shape. Further, some of the feet are missing. We deem this shape to not entirely capture the shape of a bird.

Figure 6.1: Figure shows examples of bird shapes from MPEG-7. Notice that some of this shapes are accurately captured while others are not.

I utilized stratified random sampling for splitting the data to the training and validation data [16]. The basic idea behind stratified random sampling is to reduce the error in our estimation, parameter, or modeling accuracy by partitioning a class into appropriate strata [16]. For the MEPG-7 data, each class was its own strata. Recall that for the pill shape data, I treated each of the classes as individual stratum except for the hexagon class. I split the hexagon class into two strata. There were two non-regular hexagons and six regular hexagons. Examples of the regular and non-regular hexagon observations are provided in Figures 5.6 and 5.7, respectively. I included one non-regular hexagon and three regular hexagons in the training data set. The final counts of the training and validation sets are provided in Table 6.1.

6.2.5 Shape Segmentation and Metric Collection

The details for the shape segmentation and metric collection are provided in Chapters 4 and 5. The image segmentation was mainly obtained by converting the image to grayscale, then finding the edges, performing thresholding, filling in the holes of the binary image, and then isolating the largest resulting shape.

The same segmentation algorithm from Equation 3.3 was applied to the MPEG-7 data for convenience. What differs is that $i \in \{1, ..., 80\}$. To obtain the data augmented shapes for the MPEG-7 experiment, I applied the following image operators:

$$\mathbf{c}_{i+(r-1)\times 80}[\vec{p}] = \Gamma_{>0.50}\mathcal{R}_{\theta_r}\mathbf{b}_i[\vec{p}], \forall i \in \{1, ...80\}, r \in \{1, 2, 3, 4\}, \tag{6.3}$$

$$\vec{\theta} = [0^\circ \ 90^\circ \ 180^\circ \ 270^\circ]'. \tag{6.4}$$

The rotation operator, $\mathcal{R}$, rotates the input image by θ_r for each r. The threshold operator, Γ, ensures that the final rotated image contains values of only 0's and 1's. This results in a total of 400 images.

There were seven metrics included for analysis. The code to collect the metrics is

provided here: `https://github.com/billyl320/DAMG`. They were SPs, EIs, eccentricity, circularity, and the white and black pixel counts from the minimum bounding box. The EIs are the black and white pixel counts after placing the shape in the minimal encompassing circle and then the minimal encompassing square. The SP value measures the number of white pixels divided by the sum of the EIs. This means that SP measure the proportion of white pixels after applying the shape proportion and encircled image-histogram (SPEI) algorithm. Eccentricity measures the relative relationship between the major and minor axes. Circularity measures how circular a given shape is. The white and black pixel counts from the minimal bounding box jointly measure the rectangularity of a shape. The used metrics are summarized in Table 6.3.

Table 6.3: Table provides the metrics used in this analysis on a given image, i. The first column is the q^{th} metric, where $q \in \{1, 2, ..., 10\}$. These variables make our model interpretable.

$\vec{m}_{q,i}$	Metric
1	White EI
2	Black EI
3	SP value
4	Eccentricity
5	White Bounding Box Count
6	Black Bounding Box Count
7	Circularity

6.2.6 Model and Pseudocode Implementation

The DAMG algorithm was used as the model to discriminate the classes. Different approaches were used for each experiment. At each decision node, only two variables were used alongside a support vector machines (SVM) with a polynomial kernel for the pill shape data. The variables were chosen by attempting all possible combinations of pairs. If multiple pairs of variables provide perfect classification, the pair which produces an altered averaged

similarity score within the created meta-classes is utilized. This is explained in more detail in Equation 6.9. Algorithm 1 was implemented using the pseudocode provided in Algorithm 3. Algorithm 2 remained unaltered. The mathematical representation is expressed starting from Equation 6.2 by

$$f(x) = [\beta_0 + \sum_{i=1}^{n_m} \alpha_i (\nu_m \sum_{j=1}^{2} x_{ij} x_{i'j})^{d_m}] I_{X_m \in \mathcal{R}_m}, \forall m \{1, ..., l\} \tag{6.5}$$

where, for a given m, d_m is the degree of the polynomial, γ_m is a positive constant, and ν_m is a constant. The value of l is the maximum number of subspaces.

The MPEG-7 data still used DAMG. However, it used a random forest (RF) at each node with three variables. I utilized a different machine learning (ML) algorithm and number of variables to showcase the flexibility of DAMG. The variables were chosen by attempting all possible combinations of triads. If multiple triads of variables provide perfect classification, the triad which produces an altered averaged similarity score within the created meta-classes is utilized. This is explained in more detail in Equation 6.9. Using the notation from Hastie *et al.* [28], assume that we have v variables or features and N observations or instances. In other words, we have x_i, y_i for $i = 1, 2, ..., N$ with $x_{i1}, x_{i2}, ..., x_{iv}$. Further, suppose that we have M regions, $R_1, ..., R_M$, that divide our feature space. Our model response is represented by p_{ek} for each region. This response corresponds to one of two meta-classes. For the MPEG-7 data, for a given tree, b, Equation 6.2 is,

$$f(x)_b = [\sum_{e=1}^{M} p_{ek} I(x \in R_e)] I_{X_m \in \mathcal{R}_m}, \forall m \{1, ..., l\}. \tag{6.6}$$

Note that I represents the indicator variable and $k \in \{1, 2, ..., K\}$, where K is the total

number of classes. We estimate p_{ek} by

$$\hat{p}_{ek} = \frac{1}{N_e} \sum_{x_i \in R_e} I(y_i = k). \tag{6.7}$$

This is the proportion of class k instances in a given node or region e. A greedy algorithm then considers splits using a given variable and split points. The Gini index is used to grow the tree greedily. For 2 classes, the Gini index is

$$Q_m(T) = 2p(1-p), \tag{6.8}$$

where p is the proportion in the second class. This is repeated until the minimum number of nodes is reached. The RF algorithm repeats this tree building process B times (500 in our case). However, these trees are built using bootstrapped data. Once all of the trees are built, the majority vote for a given observation determines the class of that observation. The expanded details are provided in Hastie *et al.* and Breiman's two papers [4, 5, 28].

Algorithm 3 DAMG SVM Node Creation

procedure DAMG NODE
    Determine all m possible combinations of meta-classes
    for $i \in 1 : m$ **do**
        Determine k combinations of 2 variables
        for $j \in 1 : k$ **do**
            Create the j^{th} SVM model
            Find mean Euclidean distance for 1^{st} class using j^{th} variable pair
            Find mean Euclidean distance for 2^{nd} class using j^{th} variable pair
        Find the candidate model(s), c, that perform best on the training data from all k pairs
        Save results for the i^{th} model from c which minimizes the absolute difference between the Euclidean distances of the classes

    if $\exists$ a Model with perfect classification on testing data **then**
        Calculate Ω for the best models
        Select the one model and variables X_1 and X_2 which minimizes this difference
    else
        Select the one model and variables X_1 and X_2 which minimizes the misclassification error

Since in Chapter 5 I showed that a HMH decision tree can be built with near perfect classification, I imposed additional constraints on the algorithm. For example, the HMH decision tree has perfect classification at every parent node, but not every leaf. Thus, I included that constraint in Algorithm 3 on line 3. We also imposed this for the MPEG-7 data.

We chose to use a custom approach for variable selection. While various other approaches were attempted or developed, none were able to provide satisfactory results. The variable selection process selects the pair or triad of variables which minimizes the absolute distance between the Euclidean distance for a given meta-class. Then, if there are multiple models that provide the same best classification rate on the training data, an additional criterion is used. The value which is then minimized is

$$\Omega = \frac{D_{E,1} + D_{E,2} + \mathfrak{d} \times \lambda}{2} \tag{6.9}$$

where $D_{E,i}$ is the Euclidean distance for the i^{th} meta-class with the given pair or triad of variables, $\mathfrak{d}$ is the absolute difference in the number of counts between the meta-classes, and λ is a positive constant. For our case, $\lambda = 0.41$.

In small data scenarios, many candidate models will provide perfect classification. Thus, additional constraints are needed to select the model from many possible candidates. The constraint, Ω, provides a penalized mean of the average similarity between the two meta-classes. This constraint is useful in small data problems in particular due to $\mathfrak{d} \times \lambda$. This value encourages splits which will balance the meta-class counts. The parameter, λ, is used to ensure that $\mathfrak{d}$ does not overrepresent Ω. Thus, meta-classes which consist of classes with many observations will be encouraged to remain as a meta-class while meta-classes with a smaller number of observations will be encouraged to stay together.

It is important to indicate that the first node's meta-classes for the pill shape data were predetermined. I determined the first meta-classes to be the Capsule, Oval, Rectangle, and Round classes and the remaining classes as the other. The first meta-class includes all

classes that are ovular in nature. The remaining classes are those which are not ovular. Other than this node, none of the other meta-classes were predetermined. The MPEG-7 experiment was completely machine-driven.

I then used an SVM algorithm with a polynomial kernel since we used this algorithm for the HMH decision tree. I then performed a grid search on the parameters they found to be useful for each node of the decision tree. This helped to improve search time as we could search the parameter space where we know a solution exists.

6.2.7 HMH Model Comparison for the Pill Shape Data

While mentioned previously in this manuscript, the DAMG model will be compared to an HMH model used to classify pill shapes. The HMH model was able to provide near perfect classification with a MAR of about 98% [43]. This model achieved perfect classification for all the classes except between ovals and capsules [43]. This HMH model was essentially building a series of models by hand to discriminate potential meta-classes using only two variables per meta-class. Thus, the DAMG algorithm automates this model creation process of the HMH model.

6.2.8 Other Machine or Statistical Learning Approaches for the Pill Shape Data

Other completely automated approaches were attempted throughout the literature, but none were able to achieve a high rate of classification [51]. I provided a CNN for some pill shape classes in Chapter 4, but was only able to achieve 74% on their validation data. HMH models are summarized in Tables 5.5 and 6.4. We included a balanced random forest (BRF) model to compare against other imbalanced data approaches. None of the machine driven models were able to acheive a high level of accuracy when compared to the HMH models. Since Lamberti *et al.* provided the best HMH model, our paper attempts to automate the human influenced parts of their approach and obtain similar rates of accuracy.

Table 6.4: Table with the mean average recall (MAR) values for various models. The first and third rows correspond to the model name. The second and fourth rows correspond to the MAR values. The first model is an NB. The second model is an LDA. The third model is the HMH adaptable tree built by Maddala *et al.* (Maddala - Tree). The forth model is the HMH tree model from the previous chapter (Lamberti - Tree). Many of the other models from Table 5.5 are not presented here since MAR could not be calculated for those models. Maddala - Tree approach has the largest MAR, that approach performs best across all of the classes. This corresponds to an outperformance over Lamberti - Tree of $> 1\%$.

Method	NB	LDA	Maddala - Tree
MAR	0.705	0.766	0.510
Method	Maddala - Tree	Lamberti - Tree	
MAR	0.994	0.990	

6.2.9 CNN Comparison for the MPEG-7 Data

In Chapter 4, I built CNNs to classify the MPEG-7 data in Python. I continued to use this architecture as it seems to provide the most reasonable architecture for small data problems [43]. However, this CNN was also trained using some of the data augmented images.

6.3 Results

The R code uses the `data.tree` [21], `e1071` [52], `randomForest` [10], and `DiagrammeR` [33] packages. Implementation guidance for the models was obtained from James *et al.* [35]. The R code which implemented this is provided at the following GitHub link: `https://github.com/billyl320/DAMG`.

6.3.1 Pill Shapes

Using the resulting shape metrics from the shape segmentation operators, I created a DAMG models for the five random splits. These splits were fairly consistent and provided competitive results by outperforming other approaches by about 177%.

Shape Segmentation and Metric Collection

We used the resulting metrics collected from the binary shapes segmented in previous Chapters. An example of an initial capsule image and its corresponding segmented shape image are provided in Figures 5.1 and 5.2. I found the shapes of the pills were obtained fairly accurately.

Figures of the scatterplots of the observations alongside some of the metrics used by DAMG are provided in Chapter 5. Upon inspection, it is evident that groups of classes are separable. Thus, I am confident that DAMG will also be able to find these classes via meta-classes.

DAMG Model

The resulting model was a DAMG where each decision node used only two variables and an SVM classification algorithm using a polynomial kernel. I created five DAMG models with different training and validation data using random splits to observe the consistency of the structure and performance of the DAMG algorithm. The parameter values for each node's grid search are provided in Table 6.5.

Table 6.5: Table the grid search values for the SVM classification algorithm using a polynomial kernel. Note that only integer values were used. For example, degree used search the grid values of 1, 2, 3, and 5.

SVM_i	Values
γ_m	1
ν_m	1:3, 50
d_m	1:3, 5

A universal similarity between the random splits is that the first node of the DAMG models use the same two variables. Those variables are SP and eccentricity. Using these variables, the DAMG model obtained perfect classification on the training and validation data at this first node. I plotted the decision boundary for this node in Figure 6.2. The

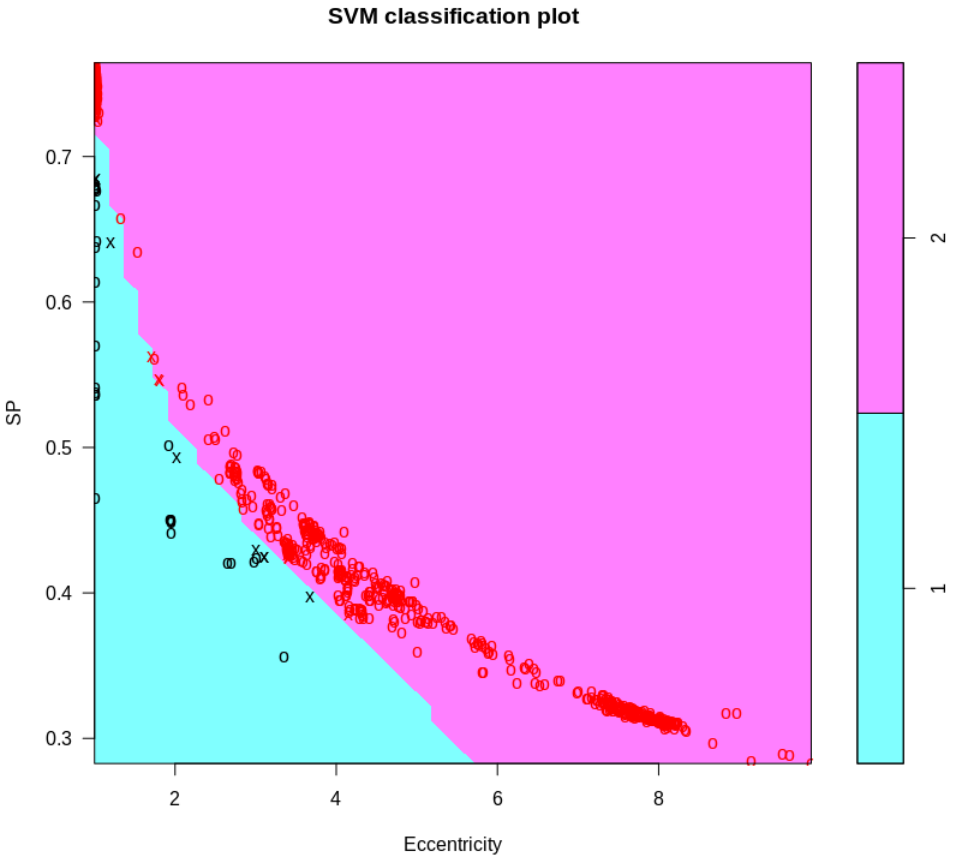


Figure 6.2: Figure shows the decision boundary made using the training data on the first decision node. This DAMG used an SVM with a polynomial kernel. The red points are associated with the oval, round, rectangle, and capsule observations. The black points correspond to the other classes. The Xs indicate if the model used the observation as a support vector, while the open circles are not support vectors. The pink area indicates the space where an observation would be classified as a round, capsule, oval, or rectangle. The cyan area indicates the space where an observation would be classified as a diamond, hexagon, pentagon, rectangle, semi-circle, square, tear, trapezoid, or triangle. Each parent node in DAMG can have this kind of 2D plot made. Modelers and users can utilize these plots to better understand the decision-making process of DAMG. Thus, my model is highly interpretable.

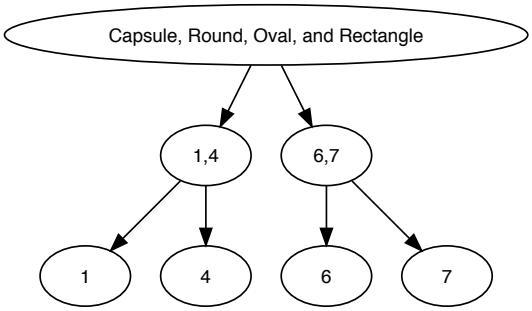
capsule, round, oval, and rectangle meta-class tends to follow a linear pattern between SP and eccentricity. As eccentricity increases, SP decreases at a faster rate. Thus, as the major axis differs more from the minor axis, the proportion of white pixels relative the sum of the EIs decreases. This result is similar to the first node presented in the previous chapter. Further, this plot can be made for any parent node of DAMG.

For the ovular meta-class side of the DAMG trees, the DAMG algorithm generally had the next meta-classes as the Capsule and Oval classes and the Round and Rectangle classes. This structure only deviates once. The non-ovular meta-class remained fairly consistent by reproducing the same meta-class structure in three out of the five models. An example of this structure is provided in Figure 6.3b. While the last two random splits resulted in two unique structures, their initial resulting meta-classes were the same between the two. The mean recall rates for each of the classes are provided in Table 6.6. Recall was calculated in Equation 2.27. We also used precision to measure the accuracy of the model. The traditional definition of precision is defined in Equation 2.28. However, since DAMG sometimes misclassifies a child node with a meta-class which contains more than 1 class, this definition does not work computationally. Thus, we used an pseudo-precision metric which we define as

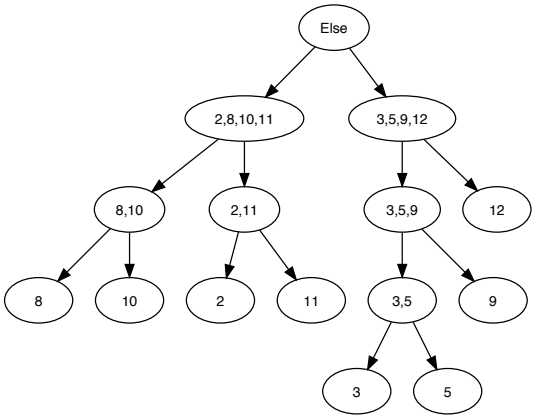
$$\text{Pseudo-Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{Adjusted False Positives}}. \tag{6.10}$$

The adjusted false positives simply divides any false positives for a meta-class equally through the classes it contains. For example, assume that the first meta-class was Triangle and the second meta-class was Hexagon and Pentagon. If one of the Triangle observations were misclassified as belonging to the second meta-class, then the single observations would contribute $\frac{1}{2}$ to each of the Hexagon and Pentagon classes' false positive counts.

When compared to the HMH model from the previous chapter, the DAMG models underperform for the Capsule, Hexagon, Pentagon, Tear, and Trapezoid classes as the HMH model perfectly classified those classes [43]. However, the DAMG model outperformed for



(a) Figure shows the DAMG for the first permutation for the capsule, round, oval, and rectangle pill shape classes. The encodings are provided in Table 6.1.



(b) Figure shows the DAMG for the first permutation for the other pill shape classes. The encodings are provided in Table 6.1.

Figure 6.3: Figure shows the DAMG model for all of the pill shape classes for the first permutation. The first meta-class was manually determined. After, DAMG determined the meta-classes automatically. Each child node results in a pill shape class. The encoding reference for each class is provided in Table 6.1.

the oval class as the HMH model only obtained a classification rate of about 0.938. When using precision to compare the models, the DAMG model had an average outperformance rate of 177%. The adjusted MP value for the DAMG model was 0.985 since I only considered those classes which overlapped all of the models. This provides evidence in favor of Hypotheses 3 and 4 from Chapter 1.

6.3.2 MPEG-7

Using the resulting shape metrics from the shape segmentation operators, we created a DAMG models for the five random splits. These splits were almost identical and provided competitive results by outperforming the CNNs by about 156% and 148% on the training and validation data, respectively.

Table 6.6: Table shows the classes' name, encoding, and mean recall across the five DAMG models. The second and third columns report the mean of their respective statistic across the 5 run. The final row shows the mean of the second and third column values. This shows that the DAMG model provides fairly accurate and consistent results. Note precision was calculated using an adjustment due to the nature of DAMG.

Pill Shape	Encoding	Recall	Adjusted Precision
Capsule	1	0.9892	0.971
Diamond	2	1.000	0.989
Hexagon	3	0.90	0.960
Oval	4	0.9854	0.995
Pentagon	5	0.933	0.971
Rectangle	6	1.000	1.000
Round	7	1.000	1.000
Semi-circle	8	1.000	0.874
Square	9	1.000	1.000
Tear	10	0.840	0.954
Trapezoid	11	0.600	1.000
Triangle	12	1.000	0.989
Mean	-	0.989	0.975

Shape Segmentation and Metric Collection

The MPEG-7 data was run through Equation 3.3 to properly capture the shapes. I found that the shapes were fairly similar. However, some of the original shapes had imperfections on them such as missing holes. The resulting shapes looked fairly better than their initial inputs. Figure 6.4 provides an example of one of the data augmented shapes begotten from Figure 6.1b.

DAMG Model

The resulting DAMG model for the MPEG-7 data used a RF model at each decision node with three variables. Five random splits were performed to check the variability of the results of DAMG. At each node, the number of variables to check for a split for the trees

Figure 6.4: Figure shows an example of one of the data augmented shapes from the bird class. This is an improver version of Figure 6.1b. This is an improvement as many of the holes in the shape have now disappeared while retaining the outline of the bird fairly accurately.

of the RF was tuned via a grid search. The values searched were from 1 to 5. Stratified random sampling was utilized to preserve the proportion of observations per class during the random splits. The split proportions were 70% for training and 30% for validation.

All of the random splits for the MPEG-7 data produced the same structure in terms of separation of meta-classes. Further, the same triad of variables was used at every node for every random partition. Those variables were SP, eccentricity, and circularity. This means that these three variables were key to discriminating these classes. Tables 6.7 and 6.8 provide summary statistics of the bird and bone and brick meta-classes, respectively. Some insight can be gained from these statistics. For instance, the brick and bone meta-class takes on very large values for eccentricity. However, due to the complex nature of these meta-classes, a 3D scatterplot provides more insight. A 3D scatterplot of each of these classes are provided in Figure 6.5. This plot confirms that the bone and brick meta-class does take on very large values for eccentricity, but it is specifically the bone class that does this. The brick class actually takes on smaller values of eccentricity. This plot shows that the bird meta-class is relatively circular in shape, large in context of the nearby area, and

usually has similar major and minor axes' length. The brick and bone meta class can be nicely described using their two classes. The bone class is relatively non-circular in shape, small in context of the nearby area, and usually has a very large major and axis. The brick class is relatively circular in shape, large in context of the nearby area, and usually has a very similar major and minor axes' length. Further, the brick class is fairly consistent, most of the observations are closer to one another. Thus, the DAMG model can be easily interpreted and explained at each child node for a given parent node.

Table 6.7: Table shows the summary of used variables for the bird meta-class.

	SP	Circularity	Eccentricity
Min	0.1627	1.125	2.687
1st Qu.	0.1993	1.985	4.376
Median	0.2243	2.129	5.303
Mean	0.2206	2.376	5.665
3rd Qu.	0.2416	2.472	6.994
Max	0.2754	4.721	9.760

Table 6.8: Table shows the summary of used variables for the brick and bone meta-class.

	SP	Circularity	Eccentricity
Min	0.07634	1.469	7.100
1st Qu.	0.10338	1.602	9.775
Median	0.19837	2.009	21.562
Mean	0.19100	2.957	33.851
3rd Qu.	0.27342	4.680	44.701
Max	0.31814	5.264	156.018

Another similarity is that it correctly classified every class with 100% accuracy except for the camel class. In some of the random partitions, the model misclassified the camel observations in the validation data incorrectly. This is summarized in Table 6.9. The random partitions only differed in the number of variables used to build the trees. Most of the time it was 1 variable, but sometimes it used 3. The accuracy of the DAMG model is provided in Table 6.9.

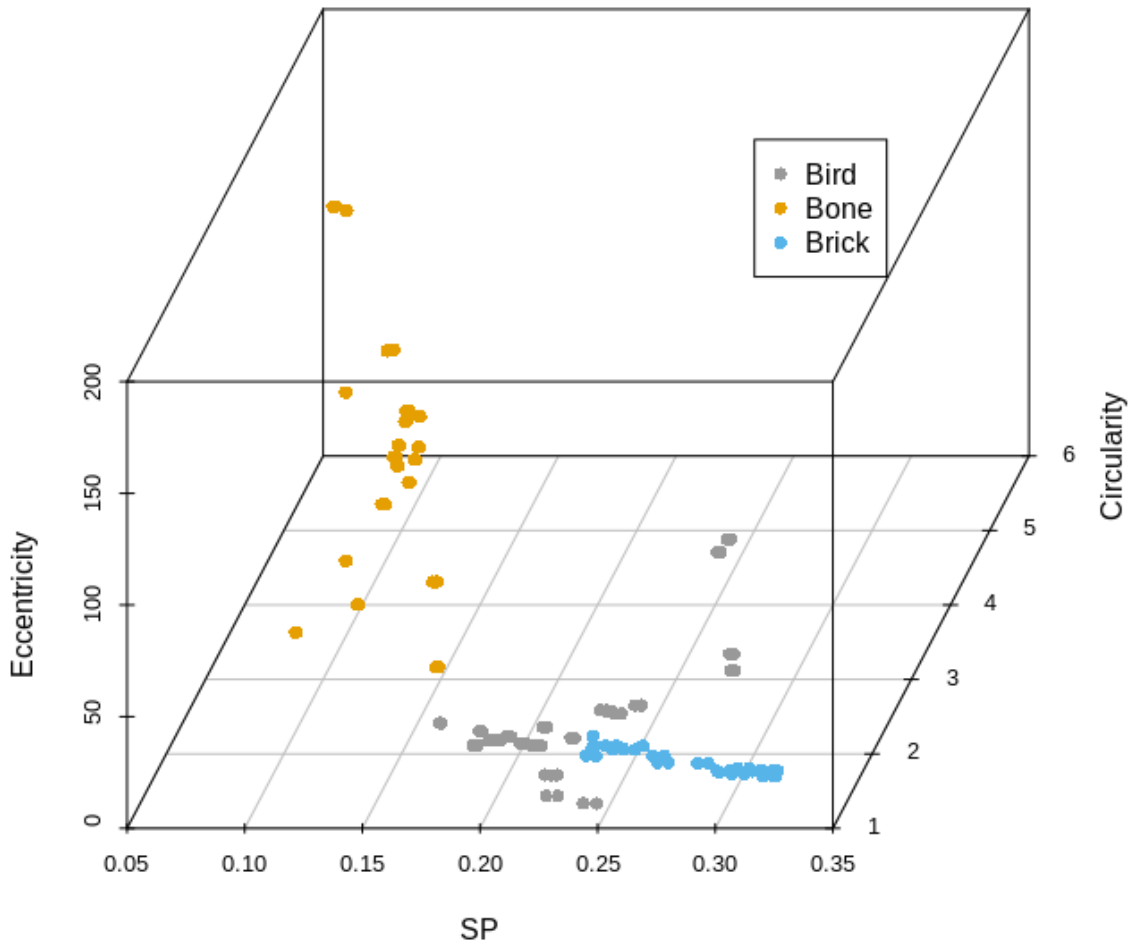


Figure 6.5: Figure shows the 3D scatterplot of the Bird, Bone, and Brick classes. The bird, bone, and brick observations are in gray, orange, and blue, respectively. DAMG classified the bird meta-class from the bone and brick meta-class at one of the parent nodes. At this node, it was able to recognize that the bird observations reside between the other meta-class. The bird meta-class tends to have SP values around 0.22 and small values of eccentricity. However, the bone and brick meta-class had a complex relationship. It either had very large values for eccentricity with a small SP value, or it had larger SP values with smaller eccentricity and circularity.

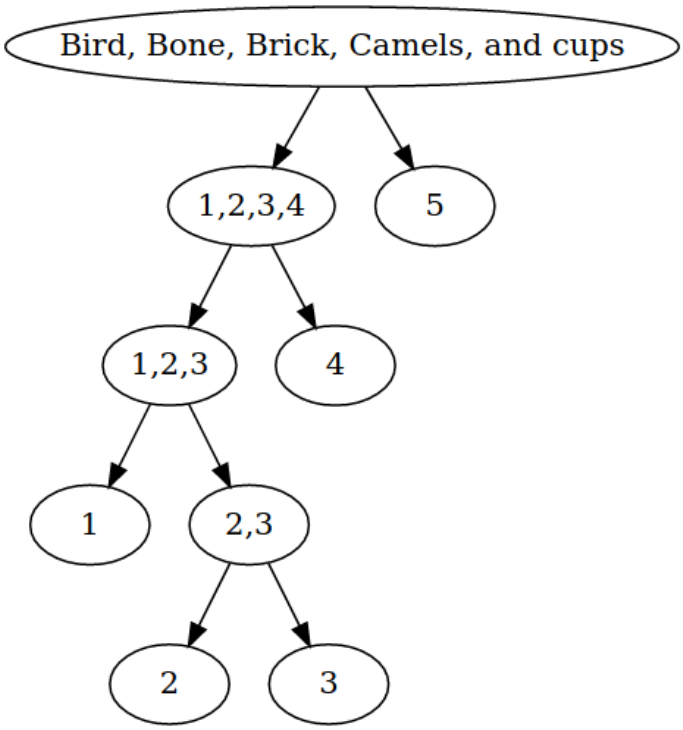


Figure 6.6: Figure shows the resulting DAMG tree for the MPEG-7 data. The structure was uniform throughout the 5 partitions. The encodings for the classes are provided in Table 6.2.

CNN Model

The CNN model was trained under similar conditions to the DAMG model. For instance, both models used data augmentation and the same random partitions. The results for the CNN model are provided in Table 6.9. However, the CNN underperformed the DAMG model. Part of the issue is that CNN models cannot find features that are invariant to orientation. They require a much larger amount of data to train. Thus, the DAMG model outperform the CNN by about 156% and 148% on the training and validation data, respectively. This results provide evidence in favor of Hypothesis 4 from Chapter 1.

Table 6.9: Table shows the classes' name, encoding, and mean recall across the five DAMG and CNN models on the training (Train) and validation (Valid) data. The first column is the numeric encoding for the class, while the remaining columns are the mean recall across the 5 random partitions. This shows that the DAMG model provides fairly accurate and consistent results while the CNN model is unable to distinguish the classes effectively. This corresponds to an outperformance rating of DAMG of about 156% and 148% on the training and validation data, respectively, when using the overall mean recall across all of the random splits.

Class	Encoding	RF Train	RF Valid	CNN Train	CNN Valid
Bird	1	1.00	1.00	0.38	0.34
Bone	2	1.00	1.00	0.56	0.53
Brick	3	1.00	1.00	0.55	0.54
Camels	4	1.00	0.95	0.21	0.30
Cups	5	1.00	1.00	0.24	0.26
Overall	-	1.00	0.99	0.39	0.40

6.4 Discussion

The first conclusion is the outperformance of the DAMG model compared to other results. DAMG's broader implications for modeling are mentioned second. I then discuss DAMG's unique ability to classify imbalanced data sets. I end by discussing the flexibility of the DAMG algorithm.

6.4.1 DAMG Competitively Performs

For the MPEG-7 data, the DAMG model was able to outperform the CNN based approach by 156% and 148% on the training and validation data, respectively, when using the overall mean recall across all of the random splits. This shows that by using interpretable metrics, DAMG is able to classify the shapes better than CNNs. CNNs are not able to capture features of shapes that are invarient to orientation. When this is coupled with very limited data, CNNs are unable to learn the features necessary to discriminate the classes.

For the pill shape data, DAMG was able to outperform other approaches by about 177% on average. The DAMG model outperformed all of the machine driven approaches

and Maddala *et al.*'s HMH tree. Further, the DAMG algorithm performs nearly as well as the HMH decision tree from the previous chapter. This showcases the power of breaking down a complicated problem into a series of simpler ones. This provides competitive results when compared to more complex or costly approaches. This result provide evidence in favor of Hypotheses 3 and 4 from Chapter 1.

6.4.2 DAMG Provides Interpretable and Explainable Solutions

Figures 6.2 and 6.5 and Tables 6.7 and 6.8 provide interpretable and explainable insights for DAMG's results. Given this study's constraints, we were able to provide valuable insights using a variety of methods. Many times, simple statistics will provide an adequate understanding of the meta-classes produced. However, creating 2D or 3D plots may provide further insight to understand how the meta-classes reside in the space they occupy. This allows users to explain the results to a wide range of non-users since we are using features that have very concrete and interpretable meanings.

However, DAMG does heavily depend on the implementation. For example, implementing a DAMG that has a very large number of variables to discriminate meta-classes will drastically reduce the interpretability. Another example is to use a technique that the modeler cannot interpret or explain. This also assumes that the variables used have an interpretable meaning.

6.4.3 DAMG Implies Problems are a Series of Simpler Ones

It is often beneficial to view a given classification task as a series of smaller problems. This process of classifying and finding the important features for a given pair of meta-classes provides two benefits. The first is drastically improving the interpretability of the model. The second benefit is the reduction of the model complexity. This allows users to circumvent the need for intricate models or deep learning. This provides evidence in favor of Hypothesis 4 from Chapter 1.

6.4.4 DAMG Works Well on Imbalanced Data

The DAMG model works well for imbalanced data. This allows modelers to avoid the use of weights on smaller classes through data level methods. DAMG circumvents the need for more complicated approaches. This improves the interpretability of the modeling process.

6.4.5 DAMG is Flexible

The DAMG algorithm can be used with any desired feature selection and classification techniques. We showed this by doing two separate implementations of DAMG. The first was an SVM with a polynomial kernel that used two variables per node. The second implementation was a RF that used three variables per node. This allows the modeler to choose a given set of techniques for any respective problem. For instance, for a given meta-class, they can choose to build a SVM model and a separate Naïve Bayes model. They would then pick the model that performed the best for a given node. Thus, the DAMG algorithm can be designed for a particular problem in a variety of ways, which gives it great power and utility.

6.5 Future Work

Future work would extend implementations to improve the DAMG's selection of models, meta-classes, and variables. For example, allowing DAMG to choose the best model from a series of candidate models allows for greater flexibility. Improving DAMG's selection of candidate meta-classes and variables would make the algorithm faster.

DAMG provides promising results on the presented data. Applying the DAMG to algorithm to a variety of classification tasks in a many practical fields is an exciting prospect for us.

Lastly, I would like to incorporate CNNs into a DAMG implementation. While we recognize that this would greatly reduce the interpretation of the model, we do believe that

this could help CNNs by simplifying the classification task. This could in turn reduce the number of needed features to learn as there would only be a binary classification task. This in turn could allow for a simpler CNN architecture. Visualizing the layers of these CNNs could help with interpreting the results. Further, each child CNN could use transfer learning from the parent CNN to learn the needed features more quickly.

6.6 Conclusions

A new classification algorithm called DAMG is presented. The DAMG algorithm breaks down complicated classification problems into a series of simpler tasks. This then provides nearly identical performance to HMH models for the case of pill shape classification and outperforms CNNs on a subset of the MPEG-7 data. The DAMG model is highly interpretable as each node only requires few variables for classifying two meta-classes. The tree ends with individual leaves for each class. This structure aids as a visual representation.

Chapter 7: General Discussion

In this chapter, I will discuss SPEI's utility, the importance of SPEI for classification models, and the contributions of the human-machine hybrid (HMH) and DAMG models. These will show that the work throughout this thesis provides an elegant and impactful solution to the hypotheses presented in Chapter 1. I will also mention some unexpected results during this research process.

7.1 SPEI-Based Models Outperform CNNs

In Chapter 4, my experiments showed that SPEI-based models outperform CNNs in small data scenarios by 52%. This provides strong evidence that SPEI's resulting EIs are competitive metrics for classification purposes. This provides evidence in favor of Hypothesis 2 from Chapter 1.

7.2 SPEI is Foundational for HMH and DAMG Models

In Chapters 5 and 6, we saw the resulting SP values from SPEI were essential for the initial nodes in the resulting trees. The trees could not have been created without the combination of SP values and eccentricity.

7.3 HMH Model Provides a Standard

The HMH model in Chapter 5 provides a standard for other models to compete against. This model was able to perfectly classify all classes except for capsule and oval pill shapes. Thus, future machine driven models will need to at least obtain the same level of accuracy. This provides evidence in favor of Hypotheses 3 and 4 from Chapter 1.

7.4 DAMG Model Nearly Matches HMH Model

The DAMG model presented in Chapter 6 nearly matches the HMH model in Chapter 5. The DAMG algorithm provided fairly consistent results. It obtained perfect classification up to but excluding the child nodes. The DAMG models were competitive to other pill classification models from the literature. This provides evidence in favor of Hypotheses 3 and 4 from Chapter 1.

7.5 DAMG Simplifies Complex Classification Problems

The DAMG algorithm converts complex multinomial classification problems into a series of simpler ones. This conversion provides interpretable and competitive solutions that work especially well for unbalanced data. This provides evidence in favor of Hypotheses 3 and 4 from Chapter 1.

7.6 Unexpected Changes and Results

During the research process, there were changes and results that were not anticipated at the time of the proposal. There were two changes and two additional results.

7.6.1 Segmentation on Pillbox Data

I underestimated the complexity of segmenting the initially proposed data set, the Pillbox data. This data could not be segmented using a universal algorithm and required the use of the NLM competition data instead. In future, I will try to anticipate the complexity of the data before over-promising results that are unobtainable. Additionally, my appreciation for those individuals who collect data well has increased.

7.6.2 SP Values Important for HMH and DAMG Models

After performing the experiments in Chapter 4, I believed that SP values were not helpful for classification purposes. However, the models presented in Chapters 5 and 6

proved otherwise. Thus, SP values are useful for describing and discriminating shapes. However, for SP values to be useful for classification purposes, the scale of the shape must be adequately large. Future work will explore what this exact scale should be.

7.6.3 DAMG is an Unexpected Discovery

The DAMG algorithm was an unexpected discovery. The process of creating the model in Chapter 5 implied that the human-influenced parts of the model could be automated. This implication resulted in a statistical or machine learning algorithm that greatly improves classification tasks where there are many classes.

7.7 Impacts and Contributions

In the following section, we will discuss the impacts and contributions of this thesis. The contributions are SPEIs, the pill shape classification models, and DAMG.

7.7.1 SPEI

SPEI is an image operator algorithm where a shape is enclosed first by the minimizing encompassing circle and then the minimizing encompassing square. SPEI results in the SP and EI values. These metrics are useful for describing and classifying shapes. SPEI-based models outperform CNNs in small data scenarios by 52%. The SP values are also essential for pill shape classification models.

7.7.2 Pill Shape Classification

I was able to provide an improved pill shape classification model. The HMH model was able to obtain near perfect classification as it only misclassified oval and capsule pill shapes. DAMG was able to obtain results that were nearly the same as those provided by HMH.

7.7.3 DAMG Algorithm

The DAMG algorithm is a new classification technique which simplifies all multinomial classification problems into a series of binary tasks. This approach provides a solution that is elegant and interpretable, greatly increasing the utility of ML or SL techniques. DAMG is particularly effective in small data problems where the data is unbalanced.

7.7.4 Publications

The work of this thesis has already been used to problems outside of the examples presented. I used SP and EI, among other shape metrics, to classify blood cells [42]. The resulting SVM model was able to outperform CNN-based approaches by an average of 5%. Another application used SP and EI, among other metrics, in a RF to classify synthetic aperture radar (SAR) images of icebergs and ships [41]. The RF outperformed CNN-based approaches by about about 7% and 11% on the testing and validation data, respectively.

Further, GMU's Office of Technology Transfer has submitted a patent for the HMH pill shape classification model. Thus, various outside authoritative entities have validated the utility and contribution originally presented in this thesis.

Chapter 8: Summary

This thesis advances science in three areas: shape metrics, pill shape classification, and classification algorithms. The developments presented in this thesis show that shape and classification algorithms are able to outperform. The solutions for these areas are elegant and effective. The SPEI algorithm provides analysts with a new tool that is applicable in a variety of fields. I showed this tool's impact on medical shape analysis, astronomy, and object recognition. The SP and EI values are particularly useful in small data problems. SPEI-based models outperform CNNs in these scenarios by about 52%. The HMH and DAMG models provide a notable improvement to pill shape classification. These models bring the research closer to providing a perfect pill identification system. The DAMG classification algorithm converts complex classification problems into a series of smaller and simpler tasks. This improves solutions to classification problems with many classes by increasing their interpretability and performance, especially for imbalanced data.

Bibliography

Bibliography

[1] MPEG-7:Shape Matching/Retrieval.

[2] A. K. Bhunia, A. K. Bhunia, S. Ghose, A. Das, P. P. Roy, and U. Pal. A deep one-shot network for query-based logo retrieval. *Pattern Recognition*, 96:106965, Dec. 2019.

[3] Y.-L. Boureau, J. Ponce, and Y. LeCun. A Theoretical Analysis of Feature Pooling in Visual Recognition. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 111–118, USA, 2010. Omnipress. event-place: Haifa, Israel.

[4] L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, Oct. 2001.

[5] L. Breiman. Manual On Setting Up, Using, And Understanding Random Forests V3.1, 2002.

[6] C. Chen, A. Liaw, and L. Breiman. Using Random Forest to Learn Imbalanced Data. *Technical Report*, page 12, 2004.

[7] D. Cireşan, U. Meier, and J. Schmidhuber. Multi-column Deep Neural Networks for Image Classification. *arXiv:1202.2745 [cs]*, Feb. 2012. arXiv: 1202.2745.

[8] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber. High-Performance Neural Networks for Visual Object Classification. *arXiv:1102.0183 [cs]*, Feb. 2011. arXiv: 1102.0183.

[9] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le. AutoAugment: Learning Augmentation Strategies From Data. pages 113–123, 2019.

[10] F. o. b. L. B. a. A. Cutler and R. p. b. A. L. a. M. Wiener. randomForest: Breiman and Cutler's Random Forests for Classification and Regression, Mar. 2018.

[11] D. B. Dahl. xtable: Export Tables to LaTeX or HTML, 2016. R package version 1.8-2.

[12] Euclid. *Euclid's Elements*. London, 1728.

[13] J. Flusser and T. Suk. Pattern recognition by affine moment invariants. *Pattern Recognition*, 26(1):167–174, Jan. 1993.

[14] J. Flusser and T. Suk. Affine moment invariants: a new tool for character recognition. *Pattern Recognition Letters*, 15(4):433–436, Apr. 1994.

[15] C. for Drug Evaluation and Research. Data Standards Manual (monographs) - Data Standards Manual (monographs): SPL Shape (Drugs).

[16] J. E. Freund. *Modern Elementary Statistics*. Prentice-Hall mathematics series. Prentice-Hall, Englewood Cliffs : N.J, 4th edition, 1973.

[17] K. Frydenberg and M. Brekke. Poor communication on patients' medication across health care levels leads to potentially harmful medication errors. *Scandinavian Journal of Primary Health Care*, 30(4):234–240, Dec. 2012.

[18] K. Fukano and H. Masuda. Detection and Classification of Pole-Like Objects from mobile Mapping Data. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-3/W5:57–64, Aug. 2015.

[19] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, Apr. 1980.

[20] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2nd edition, 2003.

[21] C. Glur. *data.tree: General Purpose Hierarchical Data Structure*, 2019. R package version 0.7.11.

[22] R. C. Gonzalez, R. E. Woods, and S. L. Eddins. *Digital Image Processing Using MATLAB, 2nd ed. by Rafael C. Gonzalez*. Gatesmark Publishing, S.I., 2nd edition edition, 2009.

[23] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, May 2018.

[24] Q. Gu, C. F. Dillon, and V. L. Burt. Prescription drug use continues to increase: U.S. prescription drug data for 2007-2008. *NCHS data brief*, (42):1–8, Sept. 2010.

[25] R. M. Haralick. A Measure for Circularity of Digital Figures. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-4(4):394–396, July 1974.

[26] B. Hariharan and R. Girshick. Low-Shot Visual Recognition by Shrinking and Hallucinating Features. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3037–3046, Venice, Oct. 2017. IEEE.

[27] S. Hassanpour, N. Tomita, T. DeLise, B. Crosier, and L. A. Marsch. Identifying substance use risk based on deep neural networks and Instagram social media data. *Neuropsychopharmacology*, 44(3):487–494, Feb. 2019.

[28] T. Hastie, T. Robert, and F. Jerome. *Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2nd (corrected 12th printing) edition, Jan. 2017.

[29] H. He and E. A. Garcia. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, Sept. 2009.

[30] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. July 2012.

[31] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant. *Applied Logistic Regression*. John Wiley & Sons, Incorporated, New York, UNITED STATES, 2013.

[32] M.-K. Hu. Visual Pattern Recognition by Moment Invariants. *IRE Transactions on Information Theory*, 8(2):179–187, Feb. 1962.

[33] R. Iannone. DiagrammeR: Graph/Network Visualization, Apr. 2019.

[34] A. J. Izenman. *Modern Multivariate Statistical Techniques Regression, Classification, and Manifold Learning*. Springer Texts in Statistics. Springer New York, New York, NY, 2008.

[35] G. James, D. Witten, T. Hastie, and R. Tibshirani, editors. *An introduction to statistical learning: with applications in R*. Number 103 in Springer texts in statistics. Springer, New York, 2013. OCLC: ocn828488009.

[36] C. M. Jones and J. K. McAninch. Emergency Department Visits and Overdose Deaths From Combined Use of Opioids and Benzodiazepines. *American Journal of Preventive Medicine*, 49(4):493–501, Oct. 2015.

[37] E. D. Kantor, C. D. Rehm, J. S. Haas, A. T. Chan, and E. L. Giovannucci. Trends in Prescription Drug Use Among Adults in the United States From 1999-2012. *JAMA*, 314(17):1818–1831, Nov. 2015.

[38] J. M. Kinser. *Image Operators: Image Processing in Python*. CRC Press, Boca Raton, FL, 1st edition, Oct. 2018.

[39] R. Kohavi. A Study of Cross Validation and Bootstrap for Accuracy Estimation and Model Selection. *International Joint Conference on Articial Intelligence*, page 7, 1995.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems 25*, pages 1097–1105, 2012.

[41] W. F. Lamberti. Classification of Synthetic Aperture Radar Images of Icebergs and Ships Using Random Forests Outperforms Convolutional Neural Networks. In *2020 IEEE Radar Conference (RadarConf)*, Flourance, Italy, Sept. 2020.

[42] W. F. Lamberti. SVM Model for Blood Cell Classification using Interpretable Features Outperforms CNN Based Approaches. In *Symposium on Data Science and Statistics*, Pittsburg, PA, 2020.

[43] W. F. Lamberti, J. Kinser, and M. Eagle. Shape Proportions and Encircled Image-Histograms Improve Analysis and Classification of Shapes for Small Data. *Under Review*.

[44] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-Based Classification for Zero-Shot Visual Object Categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465, Mar. 2014.

[45] K. Laskey and L. Martignon. Comparing Fast and Frugal Trees and Bayesian Networks for Risk Assessment. July 2014.

[46] L. Latecki, R. Lakamper, and T. Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, volume 1, pages 424–429 vol.1, June 2000. ISSN: 1063-6919.

[47] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel. Handwritten Digit Recognition with a Back-Propagation Network. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 396–404. Morgan-Kaufmann, 1990.

[48] T. B. Library. Datasets for image analysis - The British Library, Mar. 2018.

[49] C. J. Lintott, K. Schawinski, A. Slosar, K. Land, S. Bamford, D. Thomas, M. J. Raddick, R. C. Nichol, A. Szalay, D. Andreescu, P. Murray, and J. Vandenberg. Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. *Monthly Notices of the Royal Astronomical Society*, 389(3):1179–1189, Sept. 2008.

[50] C. Liu, C.-T. Zheng, S. Qian, S. Wu, and H.-S. Wong. Encoding sparse and competitive structures among tasks in multi-task learning. *Pattern Recognition*, 88:689–701, Apr. 2019.

[51] K. T. Maddala, R. H. Moss, W. V. Stoecker, J. R. Hagerty, J. G. Cole, N. K. Mishra, and R. J. Stanley. Adaptable Ring for Vision-Based Measurements and Shape Analysis. *IEEE Transactions on Instrumentation and Measurement*, 66(4):746–756, Apr. 2017.

[52] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch. e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien, 2017. R package version 1.6-8.

[53] A. Mikolajczyk and M. Grochowski. Data augmentation for improving deep learning in image classification problem. In *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pages 117–122, May 2018.

[54] E. G. Miller, N. E. Matsakis, and P. A. Viola. Learning from one example through shared densities on transforms. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, volume 1, pages 464–471 vol.1, June 2000.

[55] R. H. Myers, editor. *Generalized linear models: with applications in engineering and the sciences*. Wiley series in probability and statistics. Wiley, Hoboken, N.J, 2nd ed edition, 2010. OCLC: ocn426796752.

[56] V. Nair and G. E. Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 807–814, USA, 2010. Omnipress. event-place: Haifa, Israel.

[57] E. Neuwirth. RColorBrewer: ColorBrewer Palettes, 2014. R package version 1.1-2.

[58] NML. Pill Image Recognition Challenge, 2016.

[59] N. Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, Jan. 1979.

[60] R. O'Brien and H. Ishwaran. A random forests quantile classifier for class imbalanced data. *Pattern Recognition*, 90:232–249, June 2019.

[61] D. K. Park, A. Gelman, and J. Bafumi. Bayesian Multilevel Estimation with Poststratification: State-Level Estimates from National Polls. *Political Analysis*, 12(4):375–385, 2004.

[62] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep Face Recognition. In *Procedings of the British Machine Vision Conference 2015*, pages 41.1–41.12, Swansea, 2015. British Machine Vision Association.

[63] L. Prechelt. Early Stopping — But When? In G. Montavon, G. B. Orr, and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade: Second Edition*, Lecture Notes

in Computer Science, pages 53–67. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[64] D. Proffitt. The measurement of circularity and ellipticity on a digital grid. *Pattern Recognition*, 15(5):383–387, Jan. 1982.

[65] S. Reddy. The Uphill Fight Against Fake Prescription Drugs; Drug companies and law enforcement struggle to stop online sales of counterfeit versions of medications like Xanax laced with fentanyl that can kill unsuspecting users. *Wall Street Journal (Online); New York, N.Y.*, page n/a, Oct. 2018.

[66] B. Romera-Paredes and P. H. S. Torr. An Embarrassingly Simple Approach to Zero-Shot Learning. In R. S. Feris, C. Lampert, and D. Parikh, editors, *Visual Attributes*, pages 11–30. Springer International Publishing, Cham, 2017.

[67] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Lecture Notes in Computer Science, pages 234–241. Springer International Publishing, 2015.

[68] A. Rosenfeld. Compact Figures in Digital Pictures. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-4(2):221–223, Mar. 1974.

[69] P. L. Rosin. Measuring shape: ellipticity, rectangularity, and triangularity. *Machine Vision and Applications*, 14(3):172–184, July 2003.

[70] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, Dec. 2015.

[71] F. Santosa and W. W. Symes. Linear Inversion of Band-Limited Reflection Seismograms. *SIAM Journal on Scientific and Statistical Computing*, 7(4):1307–1330, Oct. 1986.

[72] E. Schmidt. Über das isoperimetrische Problem im Raum von n Dimensionen. *Mathematische Zeitschrift*, 44:689–788, 1939.

[73] F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, June 2015. arXiv: 1503.03832.

[74] T. Sebastian, P. Klein, and B. Kimia. Recognition of shapes by editing their shock graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):550–571, May 2004.

[75] L. Shamir. Ganalyzer: A Tool for Automatic Galaxy Image Analysis. *The Astrophysical Journal*, 736(2):141, 2011.

[76] P. Simard, D. Steinkraus, and J. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, volume 1, pages 958–963, Edinburgh, UK, 2003. IEEE Comput. Soc.

[77] J. Song, C. Shen, Y. Yang, Y. Liu, and M. Song. Transductive Unbiased Embedding for Zero-Shot Learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1024–1033, Salt Lake City, UT, USA, June 2018. IEEE.

[78] J. Stefanowski. Dealing with Data Difficulty Factors While Learning from Imbalanced Data. In S. Matwin and J. Mielniczuk, editors, *Challenges in Computational Statistics and Data Mining*, Studies in Computational Intelligence, pages 333–363. Springer International Publishing, Cham, 2016.

[79] C.-E. Särndal. *Model assisted survey sampling*. Springer series in statistics. Springer-Verlag, New York, 1992.

[80] R. C. Team. R: A Language and Environment for Statistical Computing, 2018.

[81] S. Thrun. Lifelong Learning: A Case Study. Technical report, Defense Technical Information Center, Fort Belvoir, VA, Nov. 1995.

[82] F. Tiago and R. Wayne. ImageJ User Guide - IJ 1.46r, Oct. 2012.

[83] R. Tibshirani. Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

[84] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient Object Localization Using Convolutional Networks. *Computer Vision and Pattern Recognition 2015*, Nov. 2014.

[85] B. Venables and B. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002.

[86] J. Wang, S. Mall, and L. Perez. The Effectiveness of Data Augmentation in Image Classication using Deep Learning. *arXiv:1712.04621*, page 8, Dec. 2017.

[87] Y.-X. Wang, R. Girshick, M. Hebert, and B. Hariharan. Low-Shot Learning from Imaginary Data. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7278–7286, Salt Lake City, UT, June 2018. IEEE.

[88] H. Wickham. ggplot2: Elegant Graphics for Data Analysis, 2009.

[89] H. Wickham. scales: Scale Functions for Visualization, 2017. R package version 0.5.0.

[90] I. H. Witten, E. Frank, and M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques, 3rd Edition*. Morgan Kaufmann, 3 edition, Mar.

[91] Y. F. Wong, H. T. Ng, K. Y. Leung, K. Y. Chan, S. Y. Chan, and C. C. Loy. Development of fine-grained pill identification algorithm using deep convolutional network. *Journal of Biomedical Informatics*, 74:130–136, Oct. 2017.

[92] D. G. York and et al. The Sloan Digital Sky Survey: Technical Summary. *The Astronomical Journal*, 120:1579–1587, Sept. 2000.

[93] R. Yoshihashi, W. Shao, R. Kawakami, S. You, M. Iida, and T. Naemura. Classification-Reconstruction Learning for Open-Set Recognition. page 10, 2019.

[94] M. D. Zeiler and R. Fergus. Visualizing and Understanding Convolutional Networks. *arXiv:1311.2901 [cs]*, Nov. 2013. arXiv: 1311.2901.

[95] X. Zeng, K. Cao, and M. Zhang. MobileDeepPill: A Small-Footprint Mobile Deep Learning System for Recognizing Unconstrained Pill Images. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '17, pages 56–67, New York, NY, USA, 2017. ACM.

[96] F. Zhang, R. Mamtani, F. I. Scott, D. S. Goldberg, K. Haynes, and J. D. Lewis. Increasing use of prescription drugs in the United Kingdom. *Pharmacoepidemiology and Drug Safety*, 25(6):628–636, 2016.

[97] L. Zhang, J. Liu, M. Luo, X. Chang, Q. Zheng, and A. G. Hauptmann. Scheduled sampling for one-shot learning via matching network. *Pattern Recognition*, 96:106962, Dec. 2019.

[98] Y. Zhang, Y. Bai, M. Ding, Y. Li, and B. Ghanem. Weakly-supervised object detection via mining pseudo ground truth bounding-boxes. *Pattern Recognition*, 84:68–81, Dec. 2018.

[99] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 Million Image Database for Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, June 2018.

[100] M. Zwierzyński. The improved isoperimetric inequality and the Wigner caustic of planar ovals. *Journal of Mathematical Analysis and Applications*, 442(2):726–739, Oct. 2016.

[101] D. Zwillinger and Chemical Rubber Company. *CRC standard mathematical tables and formulae*. CRC Press, Boca Raton, Florida, 31st edition, 1996.

Curriculum Vitae

William Franz Lamberti graduated from Bethlehem Catholic High School, Bethlehem, Pennsylvania, in 2011. He received his Bachelor of Arts from The College of New Jersey in 2015. He received his Master of Science from George Mason University in 2017.