OBJECT-BASED CLASSIFICATION OF EARTHQUAKE DAMAGE FROM HIGH-RESOLUTION OPTICAL IMAGERY USING MACHINE LEARNING

James Bialas
Michigan Technological University

Follow this and additional works at: https://digitalcommons.mtu.edu/etds
Part of the Artificial Intelligence and Robotics Commons, and the Remote Sensing Commons

OBJECT-BASED CLASSIFICATION OF EARTHQUAKE DAMAGE FROM HIGH-RESOLUTION OPTICAL IMAGERY USING MACHINE LEARNING

By
James Bialas

A THESIS
Submitted in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE
In Integrated Geospatial Technology

MICHIGAN TECHNOLOGICAL UNIVERSITY
2015

This thesis has been approved in partial fulfillment of the requirements for the Degree of MASTER OF SCIENCE in Integrated Geospatial Technology

School of Technology

Thesis Co-Advisor:	*Dr. Eugene Levin*
Thesis Co-Advisor:	*Dr. Thomas Oommen*
Committee Member:	*Dr. Yushin Ahn*
School Dean:	*Dr. James Frendewey*

Contents

Preface

The core content of this work represents the work and guidance of co-authors Thomas Oommen and Umaa Rebbapragada. Their assistance includes help in algorithm and parameter selection as well as guidance in the data and topics this thesis considers. This work is made possible by their efforts. However, this thesis has been prepared and written be me including all tables, figures, and appendices. Additionally, the production of all classified maps and production of training and validation sets has been carried out by me.

Acknowledgments

This research was partially supported by the National Science Foundation (Program Director: Dr. Dennis E. Wenger) through the Grant No. 1300720.

Abstract

Object-based approaches to the segmentation and supervised classification of remotely-sensed images yield more promising results compared to traditional pixel-based approaches. However, the development of an object-based approach presents challenges in terms of algorithm selection and parameter tuning. Subjective methods and trial and error are often used, but time consuming and yield less than optimal results. Objective methods are warranted, especially for rapid deployment in time sensitive applications such as earthquake induced damage assessment.

Our research takes a systematic approach to evaluating object-based image segmentation and machine learning algorithms for the classification of earthquake damage in remotely-sensed imagery using Trimble's eCognition software. We tested a variety of algorithms and parameters on post-event aerial imagery of the 2011 earthquake in Christchurch, New Zealand. Parameters and methods are adjusted and results compared against manually selected test cases representing different classifications used. In doing so, we can evaluate the effectiveness of the segmentation and classification of buildings, earthquake damage, vegetation, vehicles and paved areas, and compare different levels of multi-step image segmentations. Specific methods and parameters explored include classification hierarchies, object selection strategies, and multilevel segmentation strategies.

This systematic approach to object-based image classification is used to develop a classifier that is then compared against current pixel-based classification methods for post-event imagery of earthquake damage. Our results show a measurable improvement against established pixel-based methods as well as object-based methods for classifying earthquake damage in high resolution, post-event imagery.

1. Introduction[1]

Earthquakes are a major natural disaster that can cause significant loss of life and property damage. The dangers are not limited to the immediate event. Damage to manmade structures can further endanger the public and emergency responders as a result of structural instability that may be intensified by following aftershocks (Coburn & Spence, 2002). Damage to roads and other infrastructure can hamper response by emergency responders as well as evacuation of the public from affected areas (Coburn & Spence, 2002).

A clear and accurate picture of both the intensity and the extent of the damage is an important tool in organizing emergency response to an earthquake (Gamba & Casciati, 1998). Imagery from earth observation platforms has shown much promise in this role (Dong & Shan, 2013). However, there is room for improvement in the accuracy of classifying this data such that damage caused by an earthquake can be cataloged and focused on by emergency responders.

In this thesis, I consider the example of the Christchurch, New Zealand 6.3 magnitude earthquake that occurred February 22nd, 2011 ("New Zealand Earthquake Report – 22 February 2011 at 12:51 p.m. (NZDT)," 2011). 185 lives were lost as a result ("List of deceased," 2011). 9200 buildings were destroyed as well as 146,000 damaged ("Christchurch New Zealand 2011," n.d.). Damages from the event eventually totaled NZ$40 billion ("Christchurch rebuild to cost $10b more," 2013).

Remote sensing technologies including imagery from earth observation systems has a long history of use in identifying and assessing earthquake damage. In regards to two dimensional imagery, classification has been done to quantify, assess, and locate damage within. Traditionally, this classification has been done by human operators or increasingly with pixel-based classifiers applied to multispectral satellite imagery. As satellite sensors increase in resolution and aerial platforms such as drones provide increasingly higher resolution imagery, this presents both the challenge of classification and the benefits of more detailed maps. As spatial resolution increases, often with reduced spectral resolution, traditional pixel-based classifiers become less effective (Gao & Mas, 2008). Our work looks at object-based evaluation of 10cm orthophotos of post event damage from the 2011 Christchurch, New Zealand Earthquake.

[1] The main body of this thesis is intended to be submitted at least in part for later publication in a peer reviewed journal.

2. Objectives

In this thesis, we consider the supervised classification of earthquake damage in remotely sensed imagery, specifically if object-based methods can improve upon established pixel-based methods in image classification. In this study, we consider only the post-event imagery. In addition, the development of an object-based classifier requires many choices in segmentation, feature, and classifier algorithms. We use a systematic approach to evaluate the ramifications of these choices in the development of a classification scheme for a given set of imagery data.

3. Existing work

While there is a vast body of work discussing the theory and application of object-based image analysis, there is limited work in the task of classifying earthquake damage in remotely sensed imagery, particularly when considering only post event imagery without any sort of digital elevation model or height information. In this context, Li et al (2009) has produced the best object-based analysis results published. This study is based on Quickbird panchromatic data of 0.61m and pan sharpened multispectral imagery of the Wenchuan Earthquake of May 12, 2008 in China. Using a watershed based multilevel segmentation, single class SVM classifier, and features such as spectral values and moment invariants they show 60% producer accuracy and 91% user accuracy in building damage detection with 79% overall classification accuracy. However, it is unclear the quality of the segmentation used or the how the accuracy numbers were validated (Li, Xu, Liu, & Guo, 2009).

Taskin Kaya et al (2011) takes a pixel-based approach looking at 0.61m pansharpened multispectral Quickbird imagery of the 2010 Haitian Earthquake. Using a pixel-based method with support vector selection and adaptation, they achieved 81.5% overall accuracy with 63.4% user accuracy and 71.3% producer accuracy for the damage class. Results were generated from an independent validation set (Taskin Kaya, Musaoglu, & Ersoy, 2011).

Numerous studies comparing pixel and object-based results have also been completed using a variety of data for various applications. Some are applicable to our specific study as they incorporate various elements from our study such as comparison of object and pixel-based results using methods different from ours or similar studies in a different application. Kouchi et al (2005) compared pixel-based and object-based methods to data obtained from a visual inspection of pre-event and post-event Quickbird imagery. Data was from the 2003 Boumerdes, Algeria earthquake. Unfortunately, results were not very good with damage producer/user accuracies of 32%/23% for pixel-based and 50%/20% for object-based methods. Maximum likelihood was used for the pixel-based classification and nearest neighbor for the object-based approach (Kouchi & Yamazaki, 2005).

Dong et al (2013) provides a thorough overview of the field of earthquake damage detection with remote sensing by surveying existing work. Studies making use of aerial imagery, satellite imagery, LiDAR data, SAR data, and ancillary data such as building vector data and other GIS maps are considered. Overall, most studies achieve somewhere in the range of 70%-90% accuracy in damage identification using either pixel-based or object-based methods. Studies that consider both pre and post event data as well as studies that make use of multiple data sources tend to provide better results. However, for studies that look at post event imagery only, results tend to improve as spatial resolution increases. (Dong & Shan, 2013) A solution that considers only post-event imagery might be considered more desirable as it only relies on one set of imagery, and can be used even if cloud cover or other technical problems prevent pre-event imagery acquisition or may simply not be available.

Myint et al (2011) perform a thorough investigation of object vs. pixel results within the context of high urban density classification. The study data consists of Quickbird imagery of both Phoenix and Tempe Arizona and looks to classify urban data such as buildings, vegetation, lakes, impervious surfaces and others. The pixel-based classifier is a traditional maximum likelihood classifier. The object-based method is more complex, using a multilevel segmentation along with nearest neighbor or rule based classifiers for different classes. Ultimately, for the test image of Tempe, overall accuracies of 87.8% for pixel-based and 95.2% for object-based classifications were achieved when compared against manually delineated validation data sets created through visual interpretation. In the larger Phoenix image, the results were not quite as good but it is important to consider in our work that the worst case building producer/user classification results were 50%/81.25% for pixel-based and 83.91%/91.25% for object-based (Myint, Gober, Brazel, Grossman-Clarke, & Weng, 2011).

4. Methods

4.1 Input Data

Post event imagery was obtained from Land Information New Zealand. Imagery was obtained on the 24th of February by New Zealand Aerial Mapping Limited at the request of the Christchurch Response Center. The imagery has a 10cm per pixel ground resolution and comprises red, green, and blue layers only. The resulting orthophotos were generated from a pre-existing the digital elevation model (DEM) and were not checked against ground truth to verify if there was any earthquake damage that may not be accounted for in the DEM ((LINZ), n.d.). This work is based on tiles 1-0003-0002 and 1-0003-0003. Tiles were projected to WGS_84_UTM_zone_59S and combined into one image in ArcGIS. The resulting image is 6335x8393 pixels in size.

The projection was done at the suggestion of eCognition technical support to resolve an issue regarding determination of polygon overlap.

Training and sample data was generated manually in ArcGIS. Clearly identifiable examples of different classes were identified as randomly as possibly by the operator. Polygons were used to delineate classes. Generally, polygons were drawn at the edges of visually recognizable objects, such that the polygon enclosed the object as best as possible without including extraneous pixels. This was somewhat challenging with the rubble class as the delineations were not always very clear. It was also impossible with the pavement class as pavement objects are all connected and form a large contiguous recognizable object in the image. As such, sample polygons were drawn only around clear sections of pavement. 5 classes were able to cover all the land cover as seen in the image: building, pavement, vehicle, vegetation, and rubble. A set of polygons was created and a randomly chosen subset split off to form the validation data set. Table 4.1-1 outlines the number of training samples used both in terms of number of pixels as well as number of objects that result from segmentation.

Table 4.1-1 Number of training and validation pixels, as well as objects covered at 75% overlap. Overlap parameters and their impact on results are addressed in chapter 6.

class	training pixels	validation pixels	Training, 75%	validation, 75%
building	1561546	180232	983	129
pavement	304553	165393	114	62
vehicle	12532	9222	19	24
vegetation	42920	69150	13	22
rubble	314096	235490	252	185

4.2 Object-Based

Given the wide ranging objectives, a classification system needs to be able to accommodate numerous variables such as imagery type, classifications needed, feature importance, and training data quality. There are four phases in this system: planning, segmenting, sampling, and classifying. The latter three phases require human interpretation of the results. Based on this interpretation, modifications can be made to improve the results before moving on. Figure 4.2-1 provides a flowchart outlining these phases.

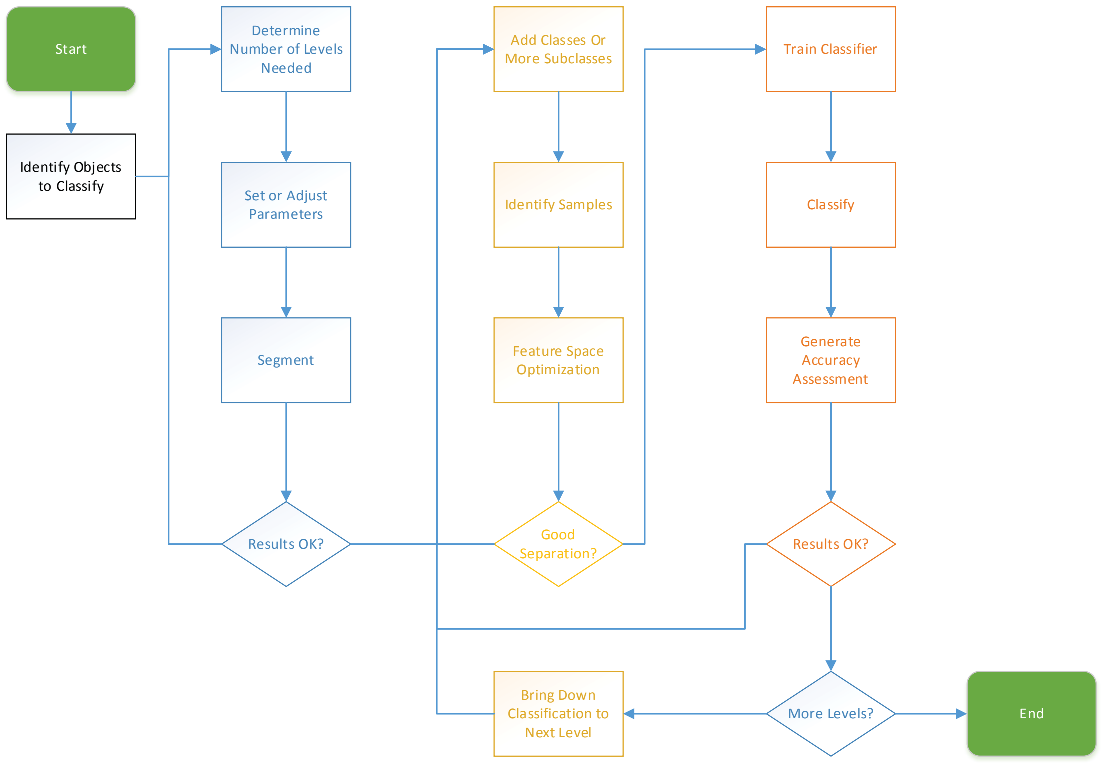
Start
Identify Objects to Classify
Determine Number of Levels Needed
Set or Adjust Parameters
Segment
Results OK?
Add Classes Or More Subclasses
Identify Samples
Feature Space Optimization
Good Separation?
Train Classifier
Classify
Generate Accuracy Assessment
Results OK?
Bring Down Classification to Next Level
More Levels?
End

Figure 4.2-1 Flow chart representing a systematic approach to building an object-based classifier.

In the planning phase, we identify the items in the image we wish to classify. It is important to not only consider the object of interest- rubble in our case, but all readily identifiable objects. By classifying as many objects as possible, we can attempt to achieve more readily distinguishable objects to aid in classification later. It is also important to consider how items differ in spectral, spatial, and texture values. A strategy that aims to classify objects that are most different in these categories will be more successful.

4.2.1 Segmentation

In the segmentation phase, we use eCognition's multiresolution segmentation algorithm to delineate the image into objects for classification. An in depth discussion of the multiresolution segmentation algorithm is presented in Appendix C. The multiresolution segmentation algorithm takes several parameters. First is the scale parameter which sets the average size of the segments for a given level. Next are the composition of homogeneity criterion- shape and compactness. Shape decides how much spatial versus spectral values affect segmentation. Compactness decides how compact in size the resulting segment will be. Finally, the operator must decide on the number of levels needed. Ideally, the items we wish to classify will be perfectly delineated by the object boundaries and comprise a single object. If the objects of interest differ in size to a significant degree, several levels of segmentation may be needed using different parameters at each level. Then, independent classifications can be run at each level. Objects at upper levels are large objects comprised of smaller objects at the next lowest level. A logical relationship between super-objects and sub-objects at lower levels is maintained so classifications can be easily shared between levels. Although several tools have been developed to automate some of the parameter selection (Zhang, Maxwell, Tong, & Dey, 2010), human interpretation of the results and subsequent adjustments to the parameters is still a common practice that yields acceptable results. Suitable segmentation is achieved by adjusting the parameters such that we minimize the number of image objects that comprise a physical item in the image while avoiding objects that span to areas outside of said physical object. Figures 4.2.1-1, 4.2.1-2 and 4.2.1-3 provide examples of segmentation results at different scale parameters.

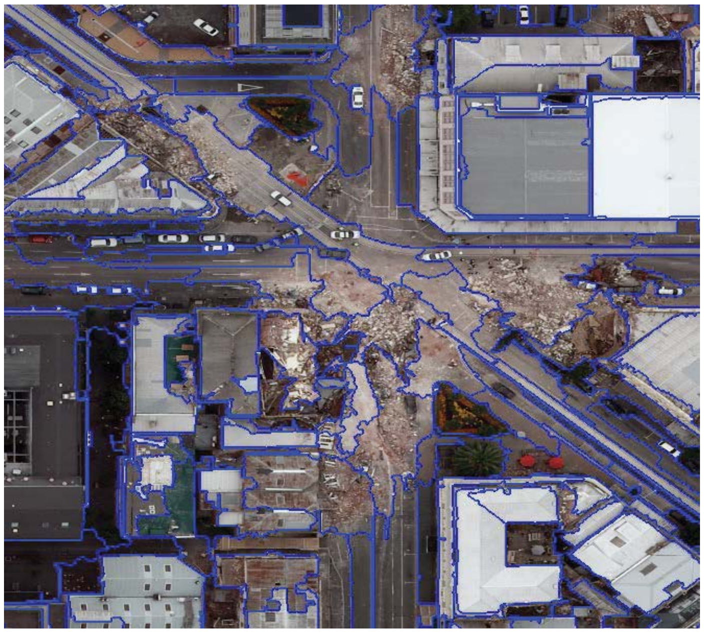
Figure 4.2.1-1 Example of good segmentation showing good coverage of visually recognizable objects by segmented image objects[2]

[2] Original aerial image obtained from Land Information New Zealand (https://data.linz.govt.nz/layer/1932-christchurch-post-earthquake-01m-urban-aerial-photos-24-february-2011/). License for distribution and modification allowed under Creative Commons Attribution 3.0 New Zealand (http://creativecommons.org/licenses/by/3.0/nz/).

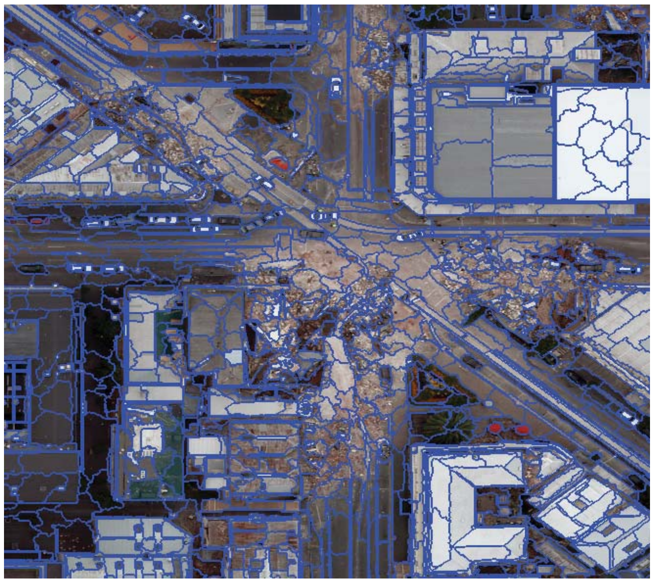
Figure 4.2.1-2 Example of poor segmentation results showing too many image objects per visually recognizable object.[3]

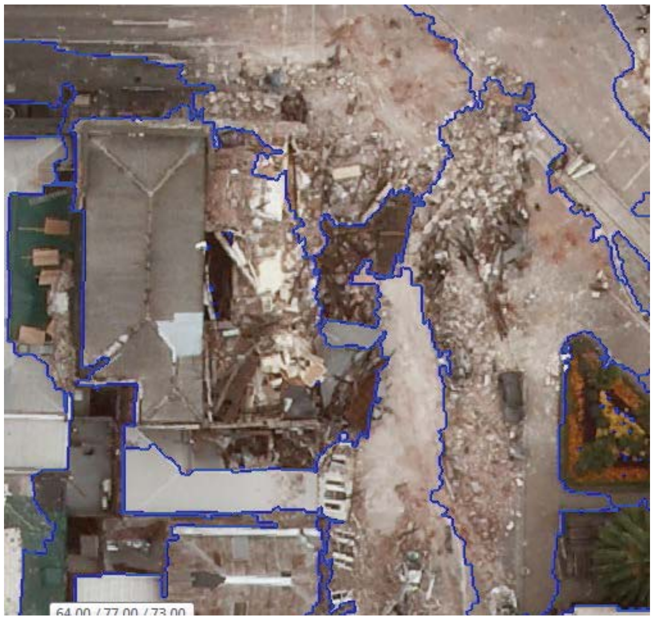

Figure 4.2.1-3 Example of poor segmentation results showing too large of an image object for the visually recognizable objects. [3]

---

[3] Original aerial image obtained from Land Information New Zealand (https://data.linz.govt.nz/layer/1932-christchurch-post-earthquake-01m-urban-aerial-photos-24-february-2011/). License for distribution and modification allowed under Creative Commons Attribution 3.0 New Zealand (http://creativecommons.org/licenses/by/3.0/nz/).

4.2.2 Sampling

In the sampling phase, we identify samples of classes to use in training a supervised classification algorithm. It is important to create training samples for not only the classes we are interested in- rubble and buildings, but to identify training samples of all recognizable object classes within the whole image. The ideal result is a set of classes that are highly separable by some set of features. eCognition provides a Feature Space Optimization (FSO) tool (further described in Appendix D) to perform feature selection. By providing a set of training samples- and feature names, the FSO tool determines which features provide the best class separability. We provided a set of 66 features, comprising spectral, geometrical, and textural values and extract the 10 best features. The 66 features are a subset of pre-configured features available in eCognition. The only preconfigured features omitted are Haralick textures for directions other than “All”. While eCognition offers more features, these require configuration in regards to the spectral bands available or parameters specific to the scene being evaluated. A complete description of the features used is available in the appendix B.

We chose the 10 features that maximized separation distance in the FSO tool. Typically, additional features beyond the top 10 do not add to the discriminative power of the classifier. Based on experience, separation values of greater than or equal to a magnitude provide good classification results. We also observe that it is important to observe a non-decreasing separation curve, which means that the addition of features does not improve the classifier performance. A curve that is not non-decreasing may indicate problems with the sample selection and lead to poor classification results. Typically, adding or removing random samples resolves this issue in one or two tries. If poor separation results are achieved, we return to the start of the sampling phase and build upon the sample selection by adding more classes or subclasses to improve segment separation until a non-decreasing separation curve is observed as shown in Figure 4.2.2-1.

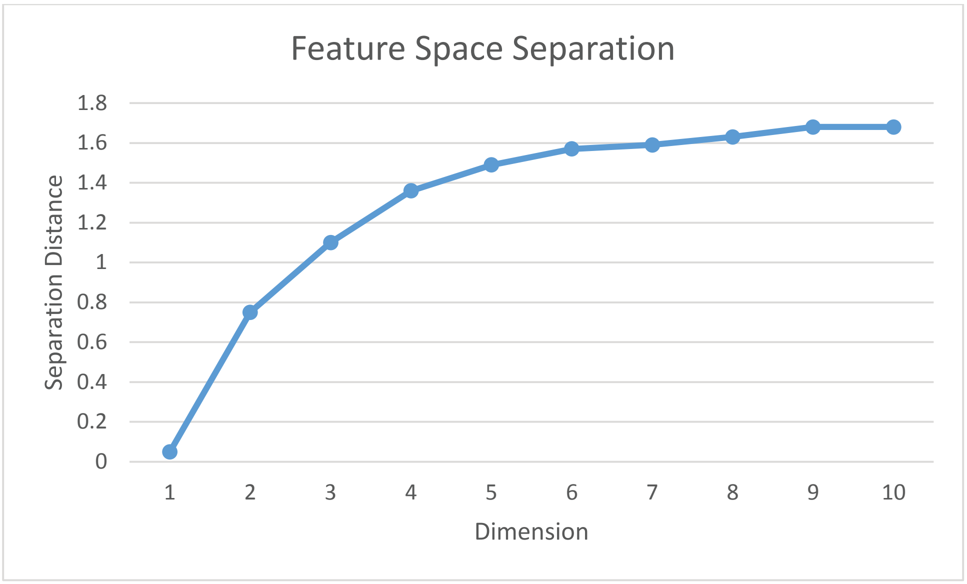

Figure 4.2.2-1 Example of good Feature Space Optimization output graph.

4.2.3 Classifying

In the classifying phase we use the features and samples identified in the previous step to classify objects from the planning phase. Training segments and features are input into a training algorithm. The output classifier is then used to predict the label of new image segments. Different classification algorithms (as defined in appendix A) can be assessed and their parameters adjusted to improve performance. From there, an accuracy assessment can be generated in eCognition against either the existing training samples or a separate validation dataset. If the results are not acceptable, we return to the sample phase to further refine the training inputs or the classifier parameters. If the results are accessible, we apply them to any lower levels in the segmentation hierarchy and start the process over again on the next level below the current level if such exists. An example of a resulting classification map for all levels is shown in Figure 4.2.3-1.

There are still several points in this procedure which require human interpretation of the results. Our work evaluates the effectiveness of different strategies and parameters at these points with an eye toward full automation of these tasks without human intervention. We specifically look at how results are affected by breaking up classes into subclasses, the effects of multilevel segmentations, classifier algorithm and parameter selection, object selection parameters, and labeling of the individual segments.

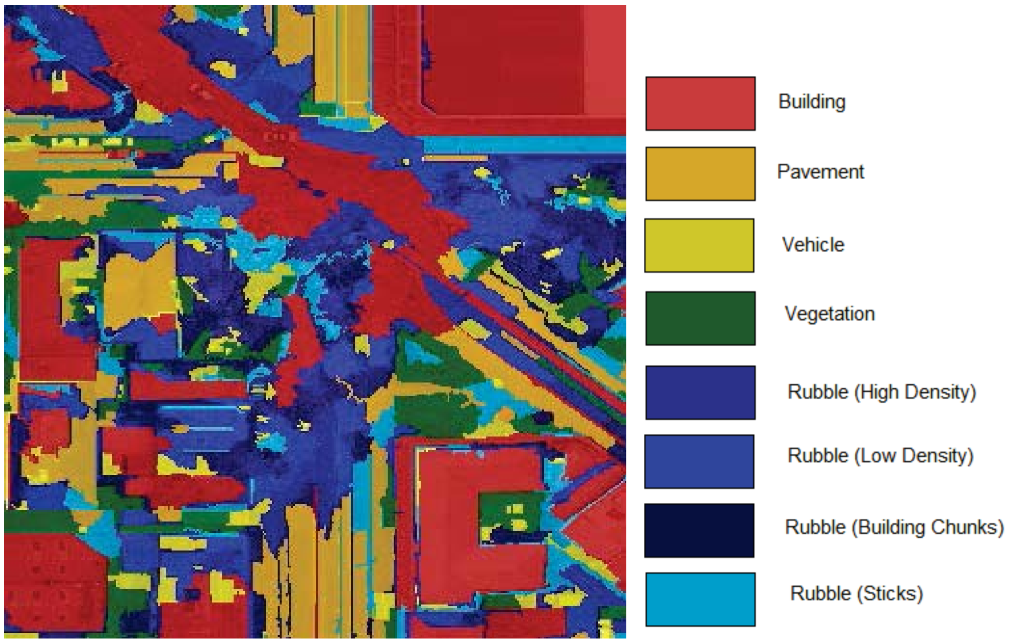

Figure 4.2.3-1 Output results of an object-based classifier delineating the different classes.[4]

4.3 Pixel-Based Approach

All results are compared against a typical pixel-based classification. Pixel-based classification techniques are highly established. Although many variations of pixel-based methods exist, classification of very high resolution imagery based purely on spectral values using an SVM classifier is an established method for similar imagery and situations (Li et al., 2009) (Taskin Kaya et al., 2011). We compare systems using the same training data and classification hierarchies. Final results are compared on a per pixel basis for both object-based and pixel-based methods. Reasoning for this comparison method is further discussed in Chapter 4.4.

We use the Orfeo Toolbox, an open source package to do our pixel-based classification and compute pixel-based accuracy assessments of the final object-based classification results. Figure 4.3-1 shows the output of a pixel-based classifier using the Orfeo Toolbox. The workflow for a pixel-based classification is similar to the steps outlined before, but we ignore the segmentation step. Feature Space Optimization is also not needed as we only consider the spectral values of the pixels.

---

[4] Original aerial image obtained from Land Information New Zealand (https://data.linz.govt.nz/layer/1932-christchurch-post-earthquake-01m-urban-aerial-photos-24-february-2011/). License for distribution and modification allowed under Creative Commons Attribution 3.0 New Zealand (http://creativecommons.org/licenses/by/3.0/nz/).

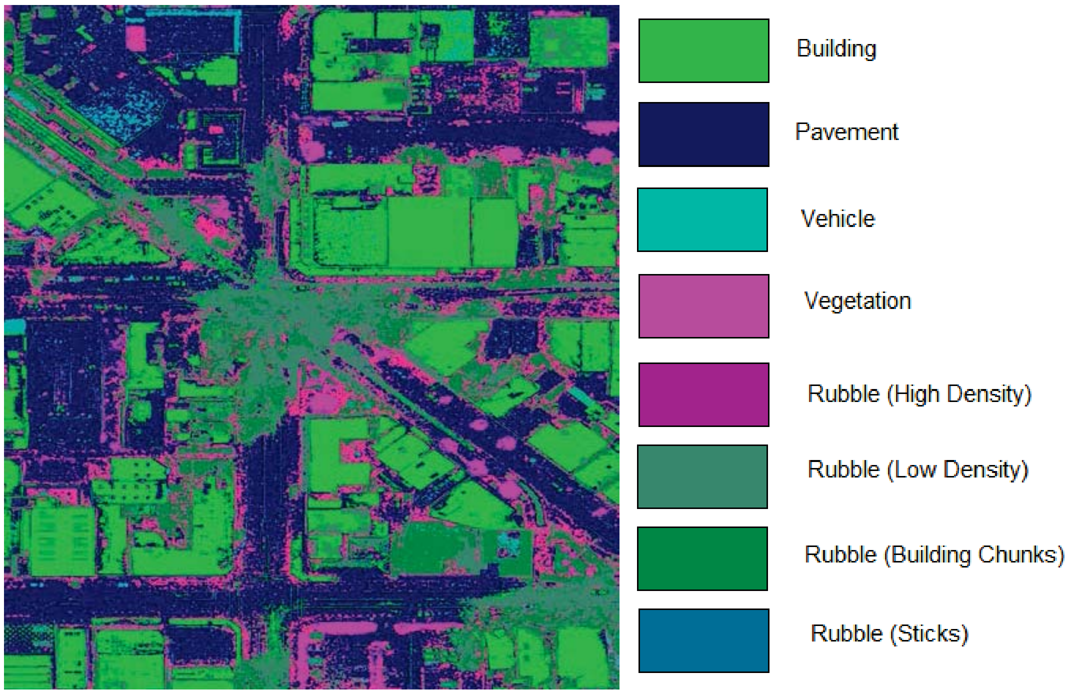

Figure 4.3-1 Output results of a pixel-based classifier delineating the different classes.[5]

4.4 Evaluation of Results

There are two important considerations to be made when comparing object-based and pixel-based results. First, as seen in figure 6-1, image objects may not represent training and validation data the same as pixels, eliminating the possibility of a fair comparison. Pixels considered in a pixel-based evaluation as part of one class, may be considered as a part of a different class in object-based results depending on where the image object boundaries fall. Second, because the image objects differ in the number of pixels they contain, especially in a multilevel object-based classification, they must be weighted to ensure a fair comparison with pixel-based results. The easiest way to address both concerns is to evaluate both pixel-based and object-based results on a per-pixel basis in the final classification maps they produce (Myint et al., 2011).

[5] Original aerial image obtained from Land Information New Zealand (https://data.linz.govt.nz/layer/1932-christchurch-post-earthquake-01m-urban-aerial-photos-24-february-2011/). License for distribution and modification allowed under Creative Commons Attribution 3.0 New Zealand (http://creativecommons.org/licenses/by/3.0/nz/).

5. Results

\While our classification system looked at several classes including building, pavement, vehicle, vegetation and rubble classes, our work is most interested in the classification performance of the rubble classes which are indicators of earthquake damage. When considering post event 10cm RGB orthophotos, our pixel-based classification system produced a 62% overall accuracy and rubble user and producer accuracies of 88% and 62%. Our object-based approach ultimately improved this to 77% overall accuracy with rubble user and producer accuracies of 88% and 94%. Both cases were evaluated at the pixel level using an independent validation data set. Producer and user accuracies for different classes are illustrated in figure 5-1 for object-based classification and figure 5-2 for pixel-based classification.

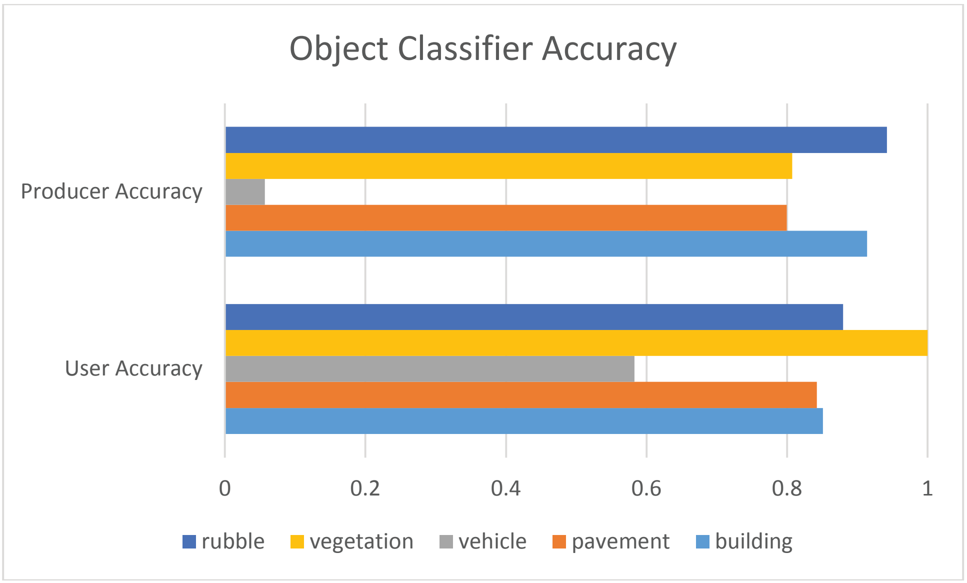

Figure 5-1 Object-based results as compared against validation data for different classes. Confusion matrix is located in Table E-2.

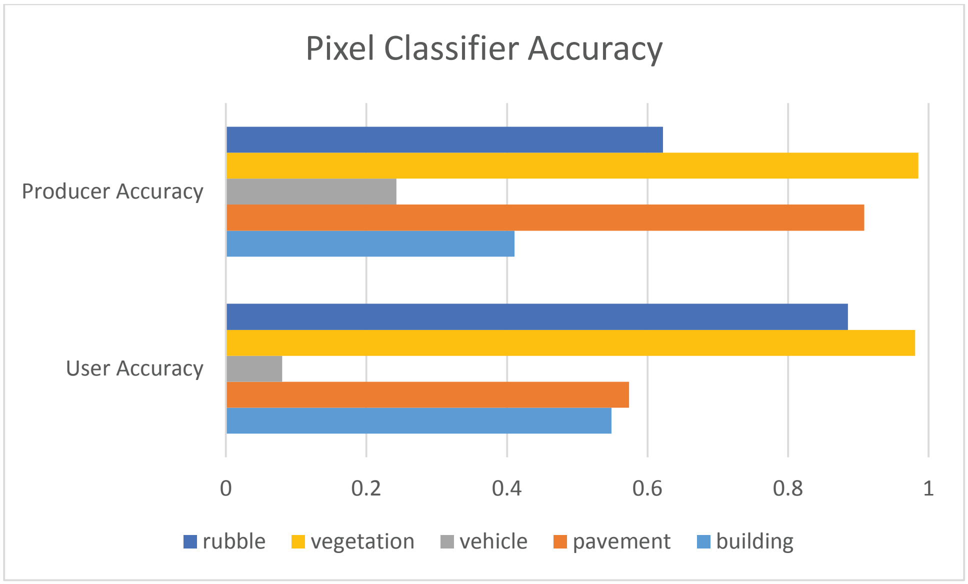

Figure 5-2 Pixel-based results as compared against validation data for different classes. Confusion matrix is located in Table E-1.

Ultimately, this systematic approach (outlined in Chapter 4.2) resulted in a multilevel segmentation comprised of four levels using scale parameters of 20, 50, 100, and 200. All levels used shape and compactness factors of 0.5. Classification was ultimately carried out on the fourth level (scale parameter 200) for buildings, and second level (scale parameter 50) for classes of pavement, rubble, vehicle, and vegetation. A Naïve Bayes classifier was used on level four and a Support Vector Machine on level 2 using the default linear kernel with c parameter of 2. Features used in training the classifier are listed in Table 5-1

Table 5-1 Features returned by the FSO tool used in object-based classifier.

Level Four Features	**Level Two Features**
(Extent) Length/Width	(Layer value) Mean green
(Shape) Compactness	(Layer value) Saturation
(Shape) Main Direction	(Extent) Area
(Based on polygons) Average length of edges	(Shape) Asymmetry
(Based on skeletons) Degree of skeleton branching	(Shape) Main direction
(Based on skeletons) Length of mainline (regarding cycles)	(Shape) Radius of largest enclosed ellipse
(Texture) GLCM contrast	(Based on polygons) Number of edges
(Texture) GLCM correlation	(Texture) GLCM entropy
(Texture) GLDV mean	(Texture) GLCM stddev
(Texture) GLDV contrast	

6. Discussion

The feedback mechanism outlined in Chapter 4.2 provided several opportunities to improve upon our results. The first to consider was object selection. In order to provide a more similar comparison between object and pixel-based results, training samples were taken from polygons in shape files. As shown in Figure 6-1, the resulting objects do not always line up exactly with the polygons and some criteria must be used to decide if an object should be classified as a training object or not. We considered percentage of overlap between the object and the polygon. Classification results are compared when looking at overlap of training polygons with image objects of 0%, 25%, 50%, 75%, and 100%. As we can see in Figures 6-2 and 6-3, different classes benefit differently from the various settings.

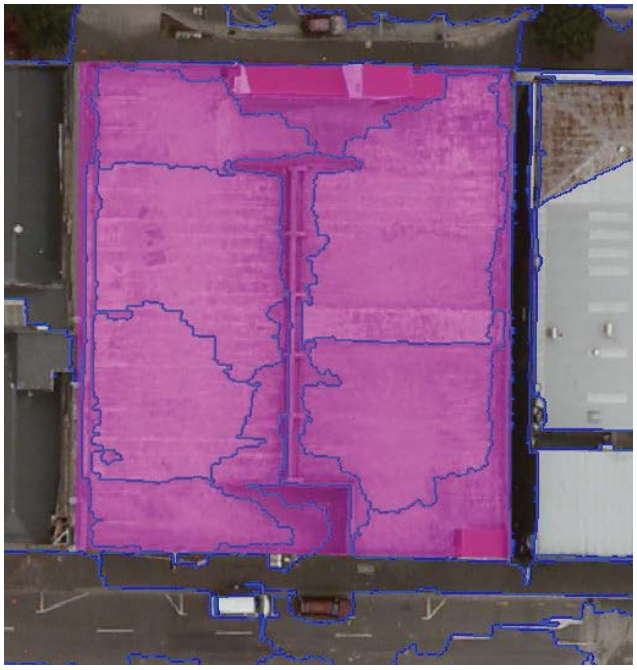
Figure 6-1 Example of a training polygon showing overlap with image objects. [6]

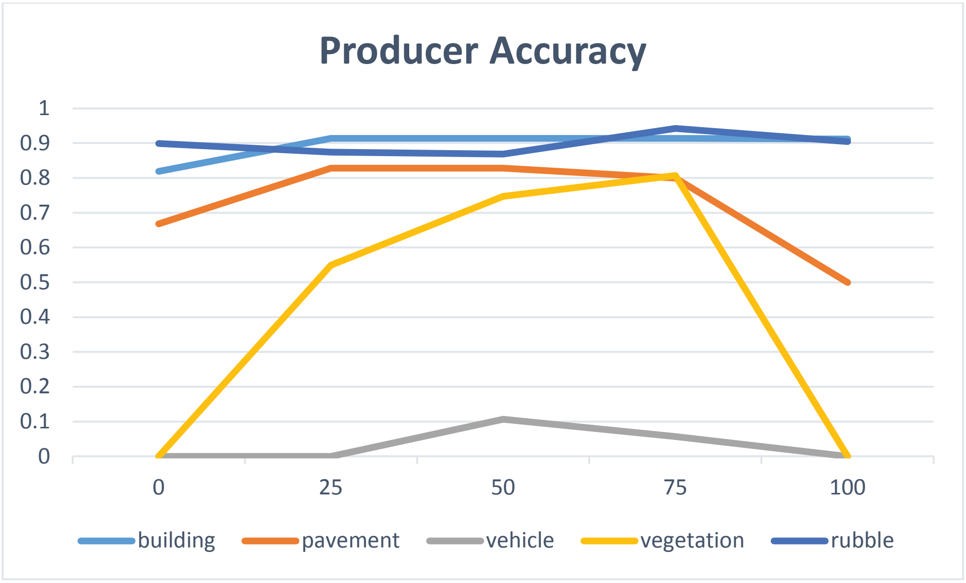

Figure 6-2 Producer accuracies measured at various overlap parameter percentages.

[6] Original aerial image obtained from Land Information New Zealand (https://data.linz.govt.nz/layer/1932-christchurch-post-earthquake-01m-urban-aerial-photos-24-february-2011/). License for distribution and modification allowed under Creative Commons Attribution 3.0 New Zealand (http://creativecommons.org/licenses/by/3.0/nz/).

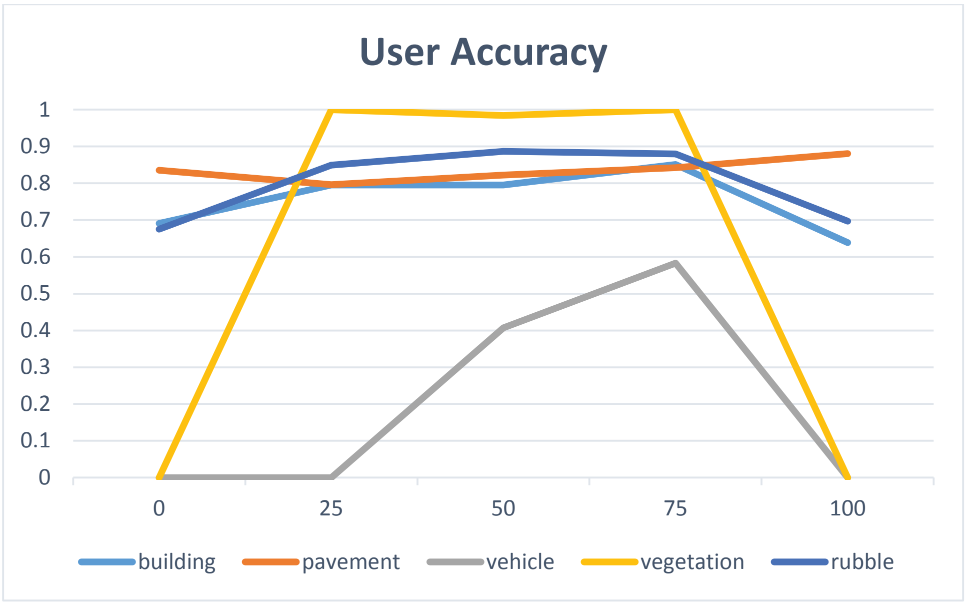

Figure 6-3 User accuracies measured at various overlap parameter percentages.

We also considered multilevel vs single level classification. Instead of classifying buildings on one level and everything else on another level of smaller objects, we classified everything on level 2. Notice how building classification goes from 91% producer accuracy to 32% (as illustrated in figures 5-1 and 6-4, respectively) when classifying them at an inappropriate level.

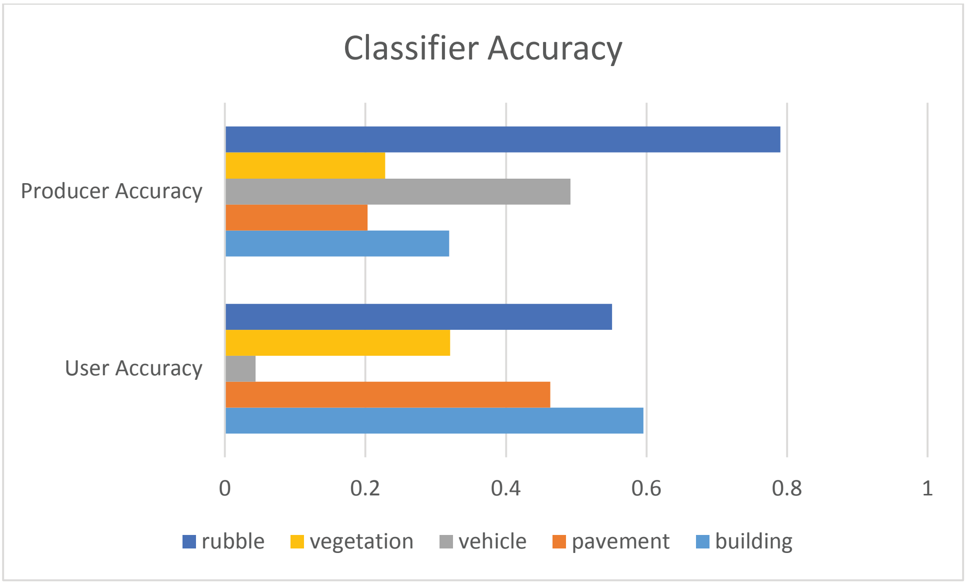

Figure 6-4 Performance of an object-based classifier that only consider objects from one level- Level 2. Confusion matrix is located in Table E-3.

In an attempt to improve class separability and overall classifier performance, we broke down the rubble class into 4 individual subclasses; building chunks- rubble that contained visibly identifiable pieces of building, high density-rubble that contained discernable pieces of debris, low density- a class for rubble with no discernable contents, and sticks- a class used to identify debris containing long structural elements such as steel beams or lumber. Breaking classes up into subclasses offers an encouraging boost in performance when comparing results against the training data. This is likely because a smaller subclass allows greater over fitting. However, the effects of over fitting are not evident when testing on our validation data as illustrated by Figure 6-5.

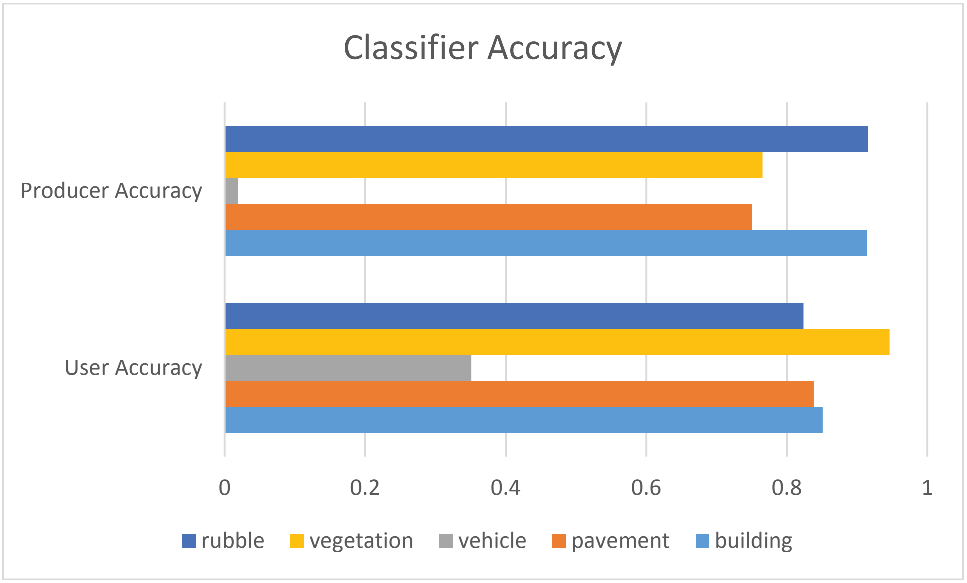

Figure 6-5 Effect of using more specific subclasses of rubble. Confusion matrix is located in Table E-4.

Choice in classifier algorithm and parameters can have a significant impact on results. Techniques for seeking improvements in results revolve around the reduction of over fitting, this is readily apparent when the classifier does very well on the training data, but returns poor results when looking at the validation data. eCognition offers five different classifier training algorithms- Decision Trees, Random Trees, Support Vector Machines, k Nearest Neighbor and Naïve Bayes. We test the classifiers against the validation data both at level 2 and level 4 as well as tuned versions of these algorithms intended to reduce over fitting. To try and reduce over fitting we adjust the following parameters. On decision trees, enable the 1SE rule for pruning. On Random Trees, we increase the minimum sample count to two. For Support Vector Machines, we use a radial basis function kernel with a C parameter of 3. For k Nearest Neighbor, the k parameter is increased to 3. Naïve Bayes has no parameters that can be adjusted. Figures 6-6 and 6-7 illustrate the accuracy of different algorithms.

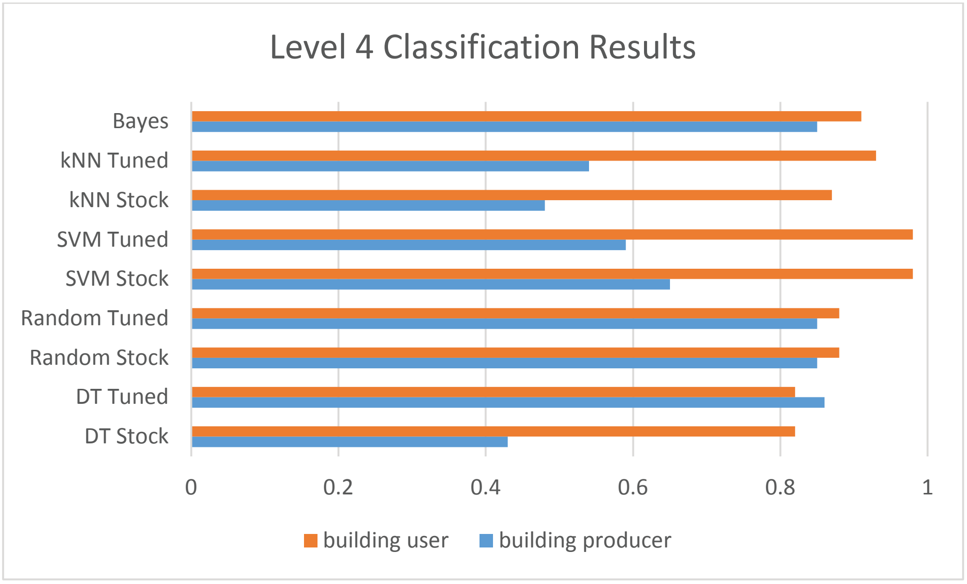

Figure 6-6 Performance of various classifier algorithms on building classification.

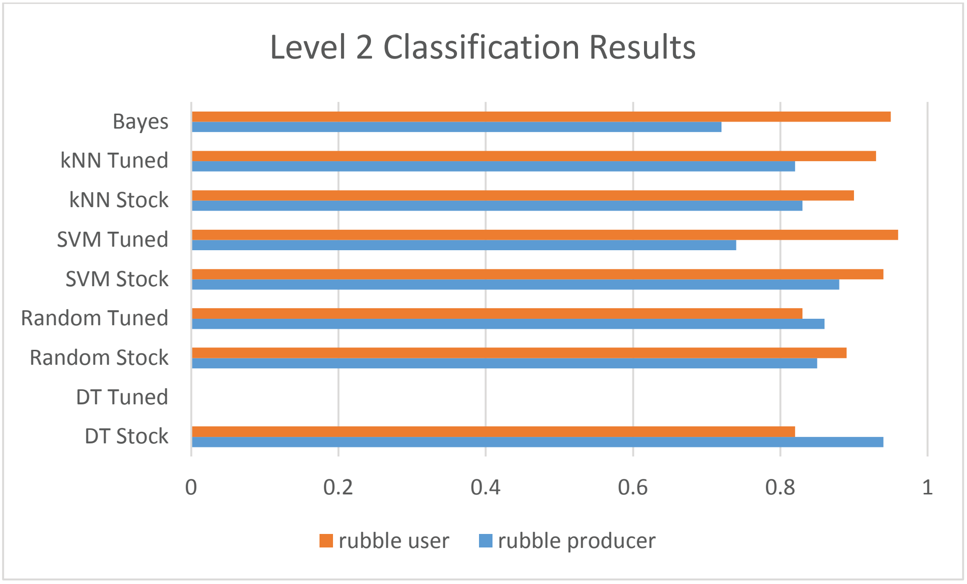

Figure 6-7 Performance of various classifier algorithms on rubble classification.

Often, the choice is not clear as to which classifier is superior. Some may offer excellent producer performance while poor user accuracy performance or vice versa. We attempt to balance these by selecting the classifier which produces the best average user or producer performance by choosing Naïve Bayes for level four and a

Support Vector Machine for level two. However, some applications may favor better performance in certain categories. For examples if we wanted to make sure we classified as much earthquake damage as possible without regard to false positives, we should consider the highest possible producer accuracy without regard to user accuracy.

7. Conclusions and Future Work

As demonstrated, the systematic approach provides feedback to the user that makes an improvement in results from several different choices in classification system parameters. Classifier results are improved by segmentation strategy, class hierarchy, and classifier algorithm choices. As shown by the single layer segmentation results, it's important to consider a segmentation strategy that correctly accounts for the relative size of physical objects. The preceding class hierarchy work strategy also demonstrates the importance of a systematic approach to this hierarchy strategy. And finally, the impact of choice in classifier algorithm is extensively documented above- different classifier algorithms can produce highly variable accuracy results for the same given inputs.

As demonstrated, there can be negative impacts on results if different choices are made. These choices result in an object-based classifier that produces measurable improvements over established pixel-based approaches to classification. Furthermore, the final object-based classification shows an improvement in overall classifier results from 62% for pixel-based to 77% for object-based. Rubble classifier producer accuracy is also improved from 62% to 94% while user accuracy results hold steady at 88%.

Future work looks to employ computational methods to make evaluations regarding such outputs as segmentation, feature selection, and classifier parameters. Choices in these parameters can be automatically made based on these evaluations reducing the need for human interaction and any possible error it can introduce.

Appendix A. Classifiers

A.1 SVM

There are a variety of classification algorithms available in eCognition. Each has various parameters that can also be configured. We first considered the Support Vector Machine. When the data is mapped into an N-dimensional feature vector, the support vector machine will attempt to fit a hyperplane that maximizes the margin distance between examples from each class to this hyperplane. The parameter C, the cost parameter, determines the amount of training data that can be misclassified (e.g., on the wrong side of the hyperplane). A higher C decreases the number of misclassified points, but makes for an overly complex model that may not be as generally applicable to new data (Clarke, Fokou©*, Zhang, & SpringerLink (Online service), 2009). eCognition also allows for the use of a kernel function. A kernel function effectively remaps that data into a different, usually higher-dimensional, basis space. This allows data with complex boundaries to be effectively classified by an SVM. eCognition supports the Radial Basis Function Kernel which is controlled by the gamma parameter. Smaller values of gamma result in a smoother decision boundary while larger values result in a more complex boundary. While a more complex boundary can result in better fitting on the training data, this can result in overfitting on test data (Clarke et al., 2009). Also, SVMs natively work on binary class problems. There are two common approaches for handling multiclass data: 1) a one against many approach that breaks out a single class and compares it with the remainder of that data or 2) a more computationally intensive approach that builds a classifier for every pair of classes and uses an ensemble vote for the final class label (Clarke et al., 2009).

Configurable Parameters:

- C
- Kernel (linear or RBF)
- Gamma (for RBF kernel only)

A.2 Decision Trees

A decision tree is a classification method that classifies a dataset by subjecting it to a series of binary decisions based on feature values. A decision tree can be trained by taking the training data set, partitioning it based on some feature value, and recursively partitioning each subset until the subset of data are all members of the same class or further recursion adds no value to the classification (Alpaydin, 2010). There are several parameters that eCognition allows users to configure. Maximum depth limits the depth of the tree and resulting recursive comparisons. Minimum sample count ensures that any ending branch of the tree has at least this many samples. Both of these prevent overly complex trees that might over fit the data. Max categories specifies the maximum number of categories to cluster data into at any given level. This may result in multiple classes being lumped together at upper levels. A reasonable number reduces processing time while maintaining accuracy. Cross validation folds allows us to compare our model against the training data to assess its accuracy. While cross validation is a useful tool in many different classifier algorithms, it is only supported with Decision Trees by eCognition. The data are broken down into N number of randomly selected folds. One is left out as a validation data set, the rest are used for training. This allows us to select the resulting training model that best fits a given random validation set from the training data (Alpaydin, 2010). The remaining options are for pruning the tree in order to better handle outliers in the training data. The 1SE rule is a preprocessing method that prunes branches as the tree is created. The smallest tree with an error rate less than the observed minimum plus one standard error is pruned (Breiman, 1984). Truncating is a post-pruning method that eliminates unnecessary branches such that the branches that remain cover the largest number of examples possible (Alpaydin, 2010). Finally, the Decision Tree trainer also allows surrogate splits. This allows a node to split not only on its primary feature as normal, but also on a secondary feature that yields a similar split. This allows us to capture objects that may be outliers in a given feature, but inliers in others (Breiman, 1984).

Configurable Parameters:

- Maximum depth
- Minimum sample count
- Max categories
- Cross validation folds
- 1SE rule (yes/no)
- Truncating (yes/no)
- Surrogate splits (yes/no)

A.3 Random Trees

Random Trees, or Random Forests use bootstrap aggregation to create an ensemble of decision trees. At each split in the tree, a random selection of features is used to calculate the split. At the end, the class label is determined by a vote of the forest. Random Forests are better at avoiding over-fitting than decision trees (Breiman, 2001). Random forests can be subject to many of the same parameters as a plain decision tree such as depth, minimum sample count, and maximum categories as well as make use of surrogates. Additional parameters include active variables- the number of randomly selected features to be considered at each tree node, forest accuracy- a target for the desired level of accuracy, and a termination criteria which can be set to the maximum number of trees, forest accuracy, or both.

Configurable Parameters:

- Maximum depth
- Minimum sample count
- Max categories
- Surrogate splits (yes/no)
- Active variables
- Forest accuracy
- Termination criteria (maximum number of trees, forest accuracy, or both)

A.4 Bayes

The Bayesian classifier is a Naive Bayes Classifier that considers each feature independently of the others when deciding on a classification. When presented with an item to classify, it considers the probability of classification in a given class as a product of the probability given each feature as well as the overall probability that an item belongs to a given class (Alpaydin, 2010).

A.5 k Nearest Neighbor

The k Nearest Neighbor classifier is also a simple classifier. Classifications are assigned based on a vote of the k training examples closest to the example to be classified. Classification is based on whichever group holds the majority. This classifier is only configurable by the k parameter (Alpaydin, 2010).

Configurable Parameters:

- k

Appendix B. Object Features

B.1 Layer Values

Layers are features calculated from the different spectral bands in the image. Sometimes they consider only one band, or the relationship of one band to others.

Brightness

This parameter considers the average brightness of the specified levels. We include all 3 RGB layers when calculating this object feature.

w_k^B = brightness weigh tof image layer *k* on a scale of 0 to 1.
K = number of layers *k* used for evaluation

$$w^B = \sum_{k=1}^{K} w_k^B$$

$\bar{c_k}(v)$ = mean intensity of image layer *k*

$$\frac{1}{w^B}\sum_{k=1}^{K} w_k^B \bar{c_k}(v)$$

Mean Layer

This calculates the mean for the given layer.

$\#P_v$= total number of pixels contained in P_v
$c_k(x, y)$ = image layer intensity at pixel (x, y)

$$\frac{1}{\#P_v}\sum_{(x,y)\in P_v} c_k(x, y)$$

Standard Deviation Layer

The standard Deviation for the given layer.

$\#P_v$= total number of pixels contained in P_v
$c_k(x, y)$ = image layer intensity at pixel (x, y)

$$\sqrt{\frac{1}{\#P_v}\left(\sum_{(x,y)\in P_v} c_k^2(x, y) - \frac{1}{\#P_v}\left(\sum_{(x,y)\in P_v} c_k(x, y)\right)^2\right)}$$

Max Diff
This feature is the maximum difference between mean values for different layers.

i, j = image layers
$\bar{c}(v)$ = mean brightness of image layer
$\overline{c_i}(v)$ = mean intensity of image layer *i*
$\overline{c_j}(v)$ = mean intensity of image layer *j*
K^B= image layers of positive brightness and weight

$$\frac{\max\limits_{i,j \varepsilon K^B} |\, \overline{c_i}(v) - \overline{c_j}(v)|}{\bar{c}(v)}$$

Hue, Saturation, Intensity
This is a different method expressing color rather than traditional RGB values.

Given RGB values normalized on a scale of 0,1
max = greatest of the RGB values
min = smallest of the RGB values

if *max* = *R*:

$$H = \frac{60° \times \frac{G - B}{max - min}}{360°}$$

If *max* = *G*:

$$H = \frac{60° \times \frac{B - R}{max - min} + 120°}{360°}$$

If *max* = *B*:

$$H = \frac{60° \times \frac{R - G}{max - min} + 240°}{360°}$$

$$S = 1 - \frac{3}{R + G + B} \times min$$

$$I = {}^{1}\!/_{3}\,(R + G + B)$$

B.2 Texture

Gray Level Co-Occurrence Matrixes are a tool used to measure texture of an object. A GLCM is a matrix that describes the relationship to pixels nearby. Typically, these are computed in a single direction, up, down, across, diagonal, or all directions and a given displacement between pixels. In eCognition, this displacement is one pixel and we consider only the GLCM for all directions. The resulting matrix is defined in the x and y as the possible pixel or gray level values- in an 8 bit image, this would be 0-255. Given a pixel value at *i,j* and it's neighboring value, these would correspond to the x and y values in the GLCM. From the resultant GLCM, different features can be computed to describe the texture of an object. All GLCMs are normalized to allow comparison between objects (Haralick, 1979).
Operation for normalization:

i = row number
j = column number
$P_{i,j}$ = normalized cell
N = number of rows or columns
$V_{i,j}$ = value of cell *i,j* in the matrix

$$P_{i,j} = \frac{V_{i,j}}{\sum_{i,j=0}^{N-1} V_{i,j}}$$

GLCM Homogeneity

The GLCM Homogeneity is high if the GLCM is strongest along the diagonal. A Homogeneous image has little local variability. The value decreases exponentially.

i = row number
j = column number
$P_{i,j}$ = normalized cell
N = number of rows or columns

$$\sum_{i,j=0}^{N-1} \frac{P_{i,j}}{1+(i-j)^2}$$

GLCM Contrast

The GLCM Contrast measures how much local variability there is, in essence- the opposite of homogeneity. This also increases exponentially.

i = row number
j = column number
$P_{i,j}$ = normalized cell
N = number of rows or columns

$$\sum_{i,j=0}^{N-1} P_{i,j}(i-j)^2$$

GLCM Dissimilarity
This is a measure of contrast that increases linearly.

i = row number
j = column number
$P_{i,j}$ = normalized cell
N = number of rows or columns

$$\sum_{i,j=0}^{N-1} P_{i,j}|i-j|$$

GLCM Entropy
This is a measure of how uniformly the elements in the GLCM are distributed.

i = row number
j = column number
$P_{i,j}$ = normalized cell
N = number of rows or columns

$$\sum_{i,j=0}^{N-1} P_{i,j}(-\ln P_{i,j})$$

GLCM Angular 2nd Momentum
A measure of homogeneity, the resulting value is high for larger values and lower for smaller values.

i = row number
j = column number
$P_{i,j}$ = normalized cell
N = number of rows or columns

$$\sum_{i,j=0}^{N-1} (P_{i,j})^2$$

GLCM Mean
The average frequency with which pixels occur in relation to other pixels.

i = row number
j = column number
$P_{i,j}$ = normalized cell
N = number of rows or columns

$$\frac{\sum_{i,j=0}^{N-1} P_{i,j}}{N^2}$$

GLCM Standard Deviation
This a measure of the deviation of pixels surrounding the GLCM Mean.

i = row number
j = column number
$P_{i,j}$ = normalized cell
N = number of rows or columns
$\mu_{i,j}$= GLCM Mean

$$\sqrt{\sum_{i,j=0}^{N-1} P_{i,j}(\, i,j - \mu_{i,j})}$$

GLCM Correlation
This is a measure of the dependency of neighboring pixels

i = row number
j = column number
$P_{i,j}$ = normalized cell
N = number of rows or columns
$\mu_{i,j}$= GLCM Mean
$\sigma_{i,j}$= GLCM standard deviation

$$\sum_{i,j=0}^{N-1} P_{i,j}\left[\frac{(i,\mu_i)(j-\mu_j)}{\sqrt{(\sigma_j^2)(\sigma_i^2)}}\right]$$

GLDV
The grey level difference vector (GLDV) is essentially a sum of the diagnols of the GLCM, it counts up the number of instances of a given different between pixels. There are a few more features that can be calculated from this value as well.

GLDV Angular Second Momentum
A measure of homogenity, the resulting value is high for larger values and lower for smaller values.

N = number of rows or columns
V_k = image object level

$$\sum_{k=0}^{N-1} V_K^2$$

GLDV Entropy
This produces a high value if all elements are smaller, essentially the opposite of the angular second moment.

N = number of rows or columns
V_k = image object level

$$\sum_{k=0}^{N-1} V_K\,(-\ln V_K)$$

GLDV Mean and GLDV Contrast are functionally equivalent to the GLCM Dissimilarity and GLCM Contrast measures, respectively.

B.3 Geometry

These are features that consider the size and the shape of the image object.

B.3.1 Extent

Area

Area is the number of pixels that represent an object. If the ground area per pixel is known, this is multiplied by the number of pixels.

A_V = area of image object V
$\#P_V$ = total number of pixels contained in P_V
u = size of the pixel in system units, u=1 if the unit is a pixel

$$A_V = \#P_V \times u^2$$

Border Length

This is the length of all borders shared with another object or the edge of the scene.

b_v = border length of image object
b_o = length of outside border
b_i = length of inside border

$$b_v = b_o + b_i$$

Length

Length is a function of the total number of pixels and the length to width ratio.

$\#P_V$ = total number of pixels contained in P_V
γ_V = length to width ratio of object V

$$\sqrt{\#P_V \cdot \gamma_V}$$

Length/Thickness
This is exactly what it implies, length of the object divided by thickness.

L = length of the object
T= thickness of the object

$$\frac{L}{T}$$

Length/Width
Length/Width is the smaller of either the ratio of the eigenvalues of the covariance matrix or the value computed from the bounding box.

λ_1, λ_2 eigenvalues
γ_v^{EV} = ratio length of *v* of the eigenvalues
γ_v^{BB} = ratio length *v* of the bounding box to total number of object pixels
γ_V = length to width ratio of object *V*

$$\gamma_v = min\, \gamma_v^{EV}, max\, \gamma_v^{BB}$$

Number of Pixels
This is a straightforward count of the number of pixels contained in the object.

Thickness
This feature only has a valid result for 3-dimensional objects.

Volume
This feature only has a valid result for 3-dimensional objects.

Width
Width is calculated similar to length, it is a function of the total number of pixels and the length to width ratio.

$\#P_V$ = total number of pixels contained in P_V
γ_V = length to width ratio of object *V*

$$\frac{\#P_v}{\gamma_v}$$

B.3.2 Shape

Asymmetry

Given an ellipse that completely encloses the object, this is a measure of the ratio of it major and minor axes. It is similar to length/width but not as accurate.

λ_{min} = minimal eigenvalue
λ_{max} = maximal eigenvalue

$$1 - \sqrt{\frac{\lambda_{min}}{\lambda_{max}}}$$

Border Index

A measure of the border length of the object as compared to the border length of the smallest enclosing rectangle.

b_v = image object border length
l_v = length of image object *v*
w_v = width of image object *v*

$$\frac{b_v}{2(l_v + w_v)}$$

Compactness

Like Border Index, this considers the smallest enclosing rectangle but is a comparison of the number of object pixels to the size of this rectangle.

l_v = length of image object *v*
w_v = width of image object *v*
$\#P_v$= total number of pixels contained in P_v

$$\frac{l_v \times w_v}{\#P_v}$$

Density

Density considers both the size of an enclosing ellipse and the size of an enclosing square. Thus, a perfect square is considered most dense by this measure.

$\sqrt{\#P_v}$ = diameter of a square object with $\#P_v$ pixels

$\sqrt{VarX + VarY}$ = diameter of the elipse

$$\frac{\sqrt{\#P_v}}{1 + \sqrt{VarX + VarY}}$$

Elliptic Fit

Given an ellipse of the same length and width of the object, this is a measure of what falls inside the ellipse versus what falls outside the ellipse

$\#P_v$= total number of pixels contained in P_v

$\varepsilon_v(x, y)$ = the elliptic distance at pixel (x, y)

$$2 \cdot \frac{\#\{(x, y)\varepsilon P_v : \varepsilon_v(x, y) \leq 1\}}{\#P_v} - 1$$

Main Direction

Given a covariance matrix of spatial distribution, this is the larger of two eigenvectors computed from eigenvalues.

λ_1 = eigenvalue

$VarX$ = variance of X

$VarY$ = variance of Y

$$\frac{180°}{\pi} \tan^{-1}(VarXY, \lambda_1 - VarY) + 90°$$

Radius of Largest Enclosed Ellipse

Starting with an ellipse of the same area as the object, the ellipse is scaled down until it is completely enclosed by the object. The ratio of the resulting ellipse to the original ellipse is returned as the value.

$\varepsilon_v(x, y)$ = the elliptic distance at pixel (x, y)

$$\varepsilon_v(x_0, y_0), where(x_0, y_0) = min\ \varepsilon_v(x, y), (x, y) \notin\ P_v$$

Radius of Smallest Enclosed Ellipse

Like the previous feature, this starts with an ellipse of the same area as the object and is increased in size until it completely encloses the object. The ratio of the resulting ellipse to the original ellipse is returned as the value.

$\varepsilon_v(x, y)$ = the elliptic distance at pixel (x, y)

$\varepsilon_v(x_0, y_0), where(x_0, y_0) = max\ \varepsilon_v(x, y), (x, y) \notin\ P_v$

Rectangular Fit

Given a rectangle with equal area to the original object and the same length and width as the object, this is a measure of the area of the image that falls outside the rectangle versus the area inside the rectangle.

$\varepsilon_v(x, y)$ = the elliptic distance at pixel (x, y)

$$\frac{\#\{(x, y)\varepsilon P_v: \varepsilon_v(x, y) \leq 1\}}{\#P_v}$$

Roundness

Given both the smallest enclosing ellipse and the largest enclosed ellipse, this returns the radius of the enclosing ellipse minus the radius of the enclosed ellipse.

ε_v^{max} = radius of largest enclosed ellipse
ε_v^{min} = radius of smallest enclosed ellipse

$\varepsilon_v^{max} - \varepsilon_v^{min}$

Shape Index

The smoothness of the border of an image object, it considers the border length and the number of pixels.

b_v = image object border length
$4\sqrt{\#P_v}$ = border length of square with $\#P_v$ pixels

$$\frac{b_v}{4\sqrt{\#P_v}}$$

B.3.3 Based on Polygons

This is a set of features calculated from a vectorization of the pixels in an object (Trimble, 2014).

Area(excluding inner polygons)

This feature is the area of the polygon subtracting any polygons that may be enclosed by the object.

$(x_i, y_i), i = 0, \cdots, n$ with $x_0 = x_n$ and $y_0 = y_n$ as the given points

$a_i = x_i y_{i+1} - x_{i+1} y_i$

$$\frac{1}{2}\sum_{i=0}^{n-1} a_i$$

Area(including inner polygons)

Same as above, but the area of any enclosed polygons included in the feature.

Average Length of Edges (Polygon)

The average length of all edges of the polygon.

X_i = length of edge *i*

n = total number of edges

$$\frac{\sum_{i=1}^{n} X_i}{n}$$

Compactness (Polygon)

Given a circle with the same perimeter as the polygon, this is a comparison of the area between the two.

a = area

p = perimeter

$$\frac{4\pi a}{p^2}$$

Length of Longest Edge (Polygon)

The length of the longest edge.

Number of Edges (Polygon)

The number of edges the polygon has.

Number of Inner Objects (Polygon)
The number of inner objects completely surrounded by the polygon.

Perimeter (Polygon)
The sum of the length of all edges of the polygon.

Polygon Self-Intersection (Polygon)
This feature isn't really used for classification, but to test for a rare condition where the resulting polygon intersects itself.

Standard Deviation of Length of Edges
How much the lengths of the edges deviate from their average value.

X_i = length of edge *i*
n = total number of edges
$\bar{X}$ = mean value of all lengths

$$\sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n}}$$

B.3.4 Based on Skeletons

Skeletons are based on a Delaunay triangulation (Delaunay, 1934) of the aforementioned polygons. Skeleton branches are determined by the three types of triangles created:
-One neighbor triangles indicate end points of the skeleton.
-Two neighbor triangles indicate connecting triangles
-Three neighbor triangles indicate branch points of the skeleton

A skeleton is a branching line created by midpoints of the above triangles (Trimble, 2014).

Average Branch Length
The average length of all branches in the object's skeleton.

Average Area Represented by Segments
The average area of all the Delaunay triangles.

Curvature/Length (Only Main Line)
A measure of how much the main line of the skeleton changes direction.

Degree of Skeleton Branching
How many degrees to which the skeleton branches in the object.

Length of Main Line (No Cycles)
The length of the mainline ignoring any inner polygons that must be crossed.

Length of Main Line (Regarding Cycles)
The same as above, but not crossing any inner polygons.

Length/Width (Only Main Line)
The ratio of length to width of the main line of the object.

Maximum Branch Length
As measured from the intersection with the mainline to the end of the branch.

Number of Segments
A count of all segments of the main line and branches of an object skeleton.

Standard Deviation Curvature (Only Main Line)
The standard deviation of the changes in direction in the main line.

Standard Deviation of Area Represented by Segments
The standard deviation of the area of the resulting Delaunay triangles in the object.

Width (Only Main Line)
The average height of all the triangles crossed by the main line. In cases where the height is not crossed by a side of the triangle, the nearest side is used to define the height.

Appendix C. Image Segmentation

Although there are many ways to segment an image into objects, we only consider eCognition's multiresolution segmentation algorithm, primarily due to its quality and lack of competitive alternatives in the literature. This segmentation algorithm is a merging algorithm. When starting from an unsegmented image, single pixels are considered first. A merging cost is calculated for each possible merge- this is known as the degree of fitting. If the result is less than the least degree of fitting calculated by the algorithm parameters, a merge is performed. Objects are continually merged until no merges are possible given the initial parameters. For subsequent levels after the base image, the input is the segments from the previous level which are then merged until the given parameters are met.

Segmentation is driven is driven by three main parameters. The most important is the scale parameter- this drives the size of the resulting segments. Scale represents the average size in pixels of the resulting objects. The shape and compactness numbers are given in a scale of 0-1. Shape determines how much influence color versus shape has on the segmentation. A higher value means a lower influence of color. The resulting influence of shape is then further influenced by the Compactness parameter. A higher compactness value results in more compact objects while a lower value results in objects with smoother borders. We primarily stick with the default shape and compactness parameters of 0.1 and 0.5 (Baatz & Schäpe, 2000)

Appendix D. Feature Space Optimization

To decide which of the features to use in classification, we use eCognition's Feature Space Optimization Tool. This tool calculates the Euclidean distance between classes as described by a set of features. Given the set of 66 features as described above, this tool will find the subset of features that provides the best separation distance out to a given dimension. Part of the output includes a graph showing how much separation increases as features are added. This can be a good indication of which features stop adding significant value to the classification and may start inducing some confusion (Trimble, 2014).

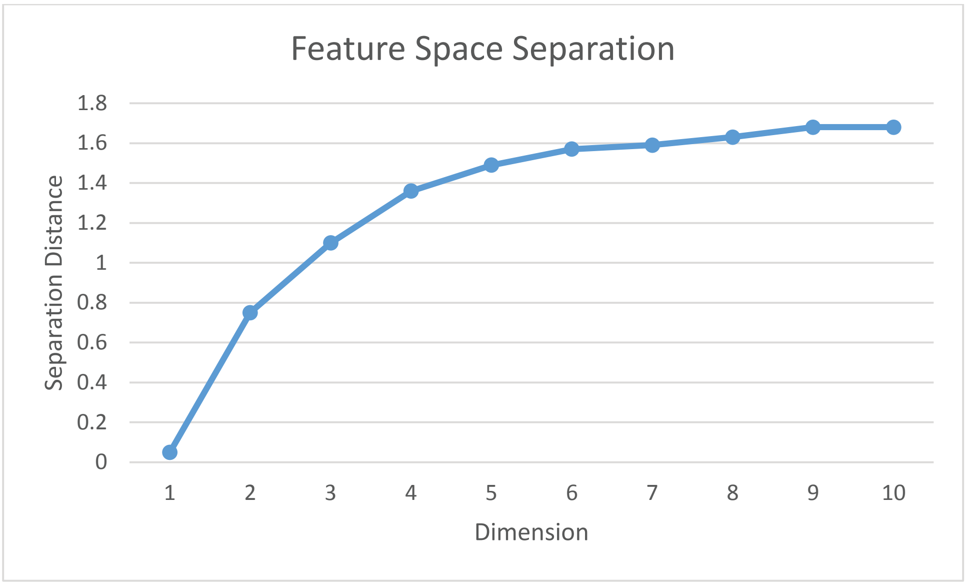

Figure C-1 Example of good Feature Space Optimization output graph.

Appendix E. Confusion Matrixes

Results of a classification process are compared using a confusion matrix. A confusion matrix is a table with actual classification in rows, and the predicted classification in columns. As such, the diagonal of this chart represents correct predictions. Items in the columns that don't match are known ground truth items that have been misclassified by the classifier. We calculate both user and producer accuracy for each class. Producer accuracy is a measure of how many items in the validation data that the classifier successfully classified. User accuracy is a measure of the reliability of the classifier- of all the items in the validation sample classified by the classifier compared to all items in that class identified by the classifier whether part of the validation sample or not. Finally, we calculate the overall accuracy for the entire classification by averaging the producer and user accuracies for all classes (Alpaydin, 2010).

For a given class, the accuracy values can be calculated as such:

$$Producer\ Accuracy = \frac{Correctly\ Classified\ Items}{Sum\ of\ all\ Items\ in\ Validation\ Sample\ (Row)}$$

$$User\ Accuracy = \frac{Correctly\ Classified\ Items}{Sum\ of\ all\ Items\ as\ Classified\ (Column)}$$

Table E-1 Pixel-Based Results using an SVM classifier.

	building	pavement	vehicle	vegetation	rubble	sum
building	76974	79392	18558	735	11885	187544
pavement	6239	156428	2220	8	7355	172250
vehicle	3165	3511	2318	91	472	9557
vegetation	227	753	0	70968	72	72020
rubble	53718	32666	5828	567	152535	245314
sum	140323	272750	28924	72369	172319	
User Accuracy	0.55	0.57	0.08	0.98	0.89	
Producer Accuracy	0.41	0.91	0.24	0.99	0.62	
average	0.62					

Table E-2 Object-based results using a Level 4 Bayes classifier, Level 2 Stock SVM classifier, 75% overlap for training object creation, and combined rubble in a single class.

	building	pavement	vehicle	vegetation	rubble	sum
building	164640	11059	2	0	4504	180205
pavement	20945	132207	0	0	12213	165365
vehicle	2616	1599	524	0	4512	9251
vegetation	0	4141	0	55680	9171	68992
rubble	5319	7980	373	1	221726	235399
sum	193520	156986	899	55681	252126	
User Accuracy	0.85	0.84	0.58	1.00	0.88	
Producer Accuracy	0.91	0.80	0.06	0.81	0.94	
average	0.77					

Table E-3 Object-based results using an SVM classifier on level 2 only, 0% overlap for training object creation, and multiple subclasses for rubble.

	building	pavement	vehicle	vegetation	rubble	sum
building	57534	38827	27468	11511	44865	180205
pavement	6729	33587	38088	19971	66990	165365
vehicle	1203	39	4547	562	2900	9251
vegetation	5176	0	12158	15722	35936	68992
rubble	25958	73	21708	1318	184956	234013
sum	96600	72526	103969	49084	335647	
User Accuracy	0.60	0.46	0.04	0.32	0.55	
Producer Accuracy	0.32	0.20	0.49	0.23	0.79	
average	0.40					

Table E-4 Object-based results using Bayes on Level 4 Bayes, Level 2 stock SVM classifier with multiple rubble, and 75% overlap for training object creation.

	building	pavement	vehicle	vegetation	rubble	sum
building	164640	11486	0	9	4070	180205
pavement	20945	124048	0	0	20372	165365
vehicle	2616	213	178	0	6244	9251
vegetation	0	1022	0	52805	15165	68992
rubble	5279	11212	329	3007	214444	234271
sum	193480	147981	507	55821	260295	
User Accuracy	0.85	0.84	0.35	0.95	0.82	
Producer Accuracy	0.91	0.75	0.02	0.77	0.92	
average	0.72					

Appendix F. Imaging Platforms

There are many earth observation systems that can be used for detecting earthquake damage. Each has several benefits and drawbacks. The first major sensor category is passive systems that detect emitted or reflected spectral data from the earth's surface such as cameras and CCDs. The second is active systems that direct a signal to the earth surfaces and measure the reflected signal such as LiDAR and Synthetic Aperture RADAR systems. Passive systems exist at all levels of spectral and spatial resolution depending on the need and platform capabilities. In addition to capturing 2 dimensional data, 3 dimensional data can be captured when stereo pairs of the same scene are acquired. Active systems are also available in a variety of spatial resolutions and provide height data, but lack the spectral capabilities of passive systems (Bossler, Jensen, McMaster, & Rizos, 2004).

These sensor systems are mounted on a variety of platforms- ground based, aerial, and space based. For our purposes, space based and aerial platforms are most interesting as they can both aid in mapping damage over a large area quickly. Each excels in specific areas though. Space based systems can more easily regularly visit remote areas as their operation depends on no earth bound infrastructure after launch. Typically, their higher altitude also allows a space based sensor to observe a larger area at a given time as well. However, these higher altitudes also provide spatial resolution challenges and space based systems are typically not as high resolution as an aerial based system. Temporal resolution also has some advantages and disadvantages. While satellite orbits are fairly predictable with regular revisits to the same areas on the globe, the time period between these revisits can be lengthy indeed. While some space based systems have the ability to operate off nadir, increasing their temporal resolution, this provides added challenges in contending with the resultant shadows and other factors from oblique imagery. And finally, space based systems have the added challenge of contending with atmospheric conditions. Weather, smoke, dust and other items both man-made and natural can obscure views of the ground (Bossler et al., 2004).

Aerial based platforms including planes and UAVs provide several advantages and disadvantages to space based systems. Spatially, aerial platforms can hold the advantage over space based systems offering higher resolution imagery, although at the cost of coverage area. Still, given the lower costs and flexibility in aerial platforms, multiple vehicles may be deployed simultaneously to increase coverage in a given time period. The flexibility of aerial systems also gives more options for increasing temporal resolution either through using multiple vehicles and multiple flights of one or more vehicles. However, the capabilities of aerial platforms can often make operations in very remote areas challenging or impossible, especially if local infrastructure has seen significant damage. Aerial systems also have some advantage in dealing with atmospheric conditions in that they can fly under conditions that might obscure the view from a satellite, however some of the very same atmospheric conditions that satellites are susceptible to can also make flying an aircraft dangerous or impossible such as dangerous weather conditions or extensive dust and smoke. (Bossler et al., 2004)

As we can see, while satellites can offer a consistent, reliable service when in all areas when atmospheric conditions permit, aerial platforms provide greater flexibility, the potential for better spectral, spatial, and temporal resolution as well as a platform more conducive to non-imaging sensors such as LiDAR.

Alternate forms of data are also useful in assessing earthquake damage. Increasingly, geospatial data is readily available on roads, buildings, and many natural features such as waterways and vegetation cover. Data such as this can be used in assisting the classification process as well as used in comparison with post event damage assessment. The internet and social media have also shown some usefulness as sources for geospatially significant, timely information on natural disasters that also might aid in earthquake damage assessment (Yin, Lampert, Cameron, Robinson, & Power, 2012).

Bibliography

(LINZ), L. I. N. Z. (n.d.). Christchurch Earthquake Imagery. Retrieved 4/13, 2015, from http://www.linz.govt.nz/land/maps/linz-topographic-maps/imagery-orthophotos/christchurch-earthquake-imagery

Alpaydin, E. (2010). *Introduction to machine learning* (2nd ed.). Cambridge, Mass.: MIT Press.

Baatz, M., & Schäpe, A. (2000). Multiresolution segmentation: an optimization approach for high quality multi-scale image segmentation. *Angewandte Geographische Informationsverarbeitung XII*, 12-23.

Bossler, J. D., Jensen, J. R., McMaster, R. B., & Rizos, C. (2004). *Manual of geospatial science and technology*: CRC Press.

Breiman, L. (1984). *Classification and regression trees*. Belmont, Calif.: Wadsworth International Group.

Breiman, L. (2001). Random forests. *Machine learning, 45*(1), 5-32.

Christchurch New Zealand 2011. (n.d.). Retrieved 3/22/2015, 2015, from http://gemecd.org/event/185

Christchurch rebuild to cost $10b more. (2013). Retrieved 3/22/2015, 2015, from http://www.3news.co.nz/politics/christchurch-rebuild-to-cost-10b-more-2013042813

Clarke, B., Fokou©*, E., Zhang, H. H., & SpringerLink (Online service). (2009). *Principles and theory for data mining and machine learning Springer series in statistics* (pp. xv, 781 p.).

Coburn, A., & Spence, R. (2002). *Earthquake Protection* (pp. 436 p.).

Delaunay, B. (1934). Sur la sphere vide. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk, 7*(793-800), 1-2.

Dong, L., & Shan, J. (2013). A comprehensive review of earthquake-induced building damage detection with remote sensing techniques. *ISPRS Journal of Photogrammetry and Remote Sensing, 84*, 85-99. doi: 10.1016/j.isprsjprs.2013.06.011

Gamba, P., & Casciati, F. (1998). GIS and image understanding for near-real-time earthquake damage assessment. *Photogrammetric engineering and remote sensing, 64*, 987–994.

Gao, Y., & Mas, J. F. (2008). A comparison of the performance of pixel-based and object-based classifications over images with various spatial resolutions. *Online journal of earth sciences, 2*(1), 27-35.

Haralick, R. M. (1979). Statistical and structural approaches to texture. *Proceedings of the IEEE, 67*(5), 786-804.

Kouchi, K., & Yamazaki, F. (2005, 2005). *Damage detection based on object-based segmentation and classification from high-resolution satellite images for the 2003 Boumerdes, Algeria earthquake*.

Li, P., Xu, H., Liu, S., & Guo, J. (2009, 2009). *Urban building damage detection from very high resolution imagery using one-class SVM and spatial relations*.

List of deceased. (2011). Retrieved 3/22/2015, 2015, from http://www.police.govt.nz/major-events/previous-major-events/christchurch-earthquake/list-deceased

Myint, S. W., Gober, P., Brazel, A., Grossman-Clarke, S., & Weng, Q. (2011). Per-pixel vs. object-based classification of urban land cover extraction using high spatial resolution imagery. *Remote Sensing of Environment, 115*, 1145-1161. doi: 10.1016/j.rse.2010.12.017

"New Zealand Earthquake Report – 22 February 2011 at 12:51 p.m. (NZDT)". (2011). *GeoNet. Earthquake Commission and GNS Science.* 22 February 2011. Retrieved 22 February 2011, 2011, from http://www.geonet.org.nz/earthquake/quakes/3468575g.html

Taskin Kaya, G., Musaoglu, N., & Ersoy, O. K. (2011). Damage assessment of 2010 Haiti earthquake with post-earthquake satellite image by support vector selection and adaptation. *Photogrammetric Engineering & Remote Sensing, 77*, 1025–1035.

Trimble. (2014). eCognition Developer 9 reference book: Trimble Germany GmbH München, Germany.

Yin, J., Lampert, A., Cameron, M., Robinson, B., & Power, R. (2012). Using social media to enhance emergency situation awareness. *IEEE Intelligent Systems, 27*, 52–59.

Zhang, Y., Maxwell, T., Tong, H., & Dey, V. (2010). *Development of a supervised software tool for automated determination of optimal segmentation parameters for eCognition*: na.