NORTHWEST NAZARENE UNIVERSITY

Mapping Burn Extent of Large Wildland Fires from Satellite Imagery Using Machine Learning Trained from Localized Hyperspatial Imagery

THESIS
Submitted to the Department of Mathematics and Computer Science
in partial fulfillment of the requirements
for the degree of
BACHELOR OF SCIENCE

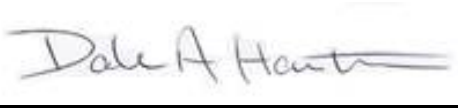
Enoch Levandovsky
2021

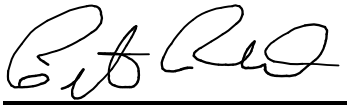
THESIS
Submitted to the Department of Mathematics and Computer Science
in partial fulfillment of the requirements
for the degree of
BACHELOR OF SCIENCE

Enoch Levandovsky
2021

Mapping Burn Extent of Very Large Wildland Fires from Satellite Imagery Using Machine Learning Trained from Localized Hyperspatial Imagery

Author: 
Enoch Levandovsky

Approved: 
Dale Hamilton Ph.D.,
Department of Mathematics and Computer Science, Faculty Advisor

Approved: 
Peter R. Robichaud Ph.D.,
USDA Forest Service Rocky Mountain Research Station, Research Engineer

Approved: 
Barry L. Myers Ph.D., Chair,
Department of Mathematics & Computer Science

Abstract

Mapping Burn Extent of Very Large Wildland Fires from Satellite Imagery Using Machine Learning Trained from Localized Hyperspatial Imagery.

Levandovsky, Enoch (Department of Mathematics and Computer Science)
Hamilton, Dr. Dale (Department of Mathematics and Computer Science)

Many wildland fire researchers are challenged to get an accurate burn acreage estimates of a wildland fire due to technology limitations. This research aids wildland fire researchers in determining the accuracy of mapping wildland fires by using sUAS (small Unmanned Aircraft System) imagery when comparing hyperspatial to sUAS imagery resampled to satellite scale resolution in. This project made the assumption that sUAS burn extent data was accurate. This assumption allowed for the resampled training data using fuzzy logic control as the method for improving satellite resolution data. The results showed that even though the resolution of imagery significantly drops, the accuracy of the burn extent remains relatively high throughout the burned area. Although sUAS imagery is likely more accurate given its resolution, acquiring satellite imagery is much more time efficient, cost efficient, and has a consistent history of clean data which serves as a great asset towards mapping wildland fire compared to sUAS imagery. This thesis compares the accuracy of satellite imagery and sUAS imagery for wildland fire mapping.

Acknowledgments

I would like to acknowledge the support from my family towards my studies. I would also like acknowledge the faculty and students in Northwest Nazarene University Department of Mathematics and Computer Science who have helped with different aspects of this effort, including Dale Hamilton Ph.D. and a former student Nicholas Hamilton. Additionally, I would like to acknowledge Northwest Nazarene University who funded the research efforts.

Table of Contents

1. Overview

In mid-2014, Dale Hamilton Ph.D. studied methods to improve mapping of wildland fire with satellite imagery. He attempted to approach this challenge through utilizing sUAS drone imagery to create training data for a Support Vector Machine (SVM) model to classify burn extent and burn severity. However, his approach did not use raw data, but instead substituted satellite imagery for sUAS imagery resampled to a resolution that of a satellite. The research assessed whether burn extent and severity could be mapped more accurately using hyperspatial imagery acquired without a sUAS, than is possible using 30 m resolution imagery. Although this study successfully proved that sUAS imagery can be used to map wildland fire with satellite imagery with some loss of accuracy, it did not fully complete the process of applying it on actual satellite imagery. Hamilton's presented his findings at IntelliSys in 2017 and published his methods in 2018 (Hamilton, 2018). This research aimed to complete some of his proposed future research direction.

This thesis is attempts to apply using sUAS imagery as training data to map wildland fire area using satellite imagery. The process utilized a combination of different methods and studies from the NNU FireMap research. Most of this research was completed using Python and the C++ languages. To allow for a smooth transition between each of the process, I choose Python as a scripting language. As for the version of python, I choose Python 2.7 as at that time ArcMap 10.7, a soon to be deprecated version, was created in Python 2.7. Many of the functions used in previous projects were not yet fully functorial in ArcMap Pro, the Python 3 version of ArcMap, at that time.

2. Publication

This is a copy of the publication based off this research that was published in November of 2020 by the author, Dale Hamilton Ph.D., and Nicholas Hamilton. This Study describes an overview and the motivation of Dale Hamilton's Ph.D. dissertation and the processes taken to obtain the accuracy of mapping out fire extent and burn severity using drone imagery.

Abstract

Wildfires burn 4–10 million acres annually across the United States and wildland fire related damages and suppression costs have exceeded $13 billion for a single year. High-intensity wildfires contribute to post-fire erosion, degraded wildlife habitat, and loss of timber resources. Accurate and temporally adequate assessment of the effects of wildland fire on the environment is critical to improving the of wildland fire as a tool for restoring ecosystem resilience. Sensor miniaturization and small unmanned aircraft systems (sUAS) provide affordable, on-demand monitoring of wildland fire effects at a much finer spatial resolution than is possible with satellite imagery. The use of sUAS would allow researchers to obtain data with more detail at a much lower initial cost. Unfortunately, current regulatory and technical constraints prohibit the acquisition of imagery using sUAS for the entire extent of large fires. This research examined the use of sUAS imagery to train and validate burn severity and extent mapping of large wildland fires from various satellite images. Despite the lower resolution of the satellite image, the research utilized the advantages of satellite imagery such as global coverage, low cost, temporal stability, and spectral extent while leveraging the higher resolution of hyperspatial sUAS imagery for training and validating the mapping analytics.

Published in Hamilton, D., Levandovsky, E., & Hamilton, N. (2020). Mapping Burn Extent of Large Wildland Fires from Satellite Imagery Using Machine Learning Trained from Localized Hyperspatial Imagery. *Remote Sensing, 12*(24), 4097. https://doi.org/10.3390/rs12244097

2.1. Introduction

This study examines the use of sub-decimeter hyperspatial imagery acquired with a small unmanned aircraft system (sUAS) to train machine learning algorithms to map wildland fire severity and extent from Landsat imagery. This effort increases the accuracy with which severity and extent can be mapped using hyperspatial imagery beyond the study area extent constraints observed due to the current technical and regulatory limitations resulting from the spatial extent of imagery that can be acquired using sUAS. Extending the usefulness of hyperspatial sUAS imagery beyond current flight extent restrictions will provide managers with increased actionable knowledge, leading to improved management decisions and increased ecosystem resilience.

A century of fire suppression and fire prevention communities has led to the current departure of wildlands from fire return intervals typically experienced under pre-European settlement conditions. Wildlands in the western United States (US) are experiencing a much higher incidence of catastrophic fires (Wildland Fire Leadership Council, 2014). Millions of hectares of western United States wildlands are impacted by wildland fire annually, with wildlands burned in some fire seasons exceeding four million hectares (Hoover, 2019), with suppression costs exceeding three billion dollars annually (National Interagency Fire Center, 2020). High-intensity wildland fires contribute to post-fire erosion, degraded wildlife habitat, and loss of timber resources. This loss results in

negative impacts on ecosystem resilience as well as communities in the wildland-urban interface where 25,790 structures were burned in 2018 (Hoover, 2019), Additionally, wildland fires across the US claim more lives than any other type of natural disaster, resulting in the average loss of twenty wildland firefighters per year (NIFC, 2020b; Zhou et al., 2005). The 2018 Camp Fire in northern California alone resulted in 85 fatalities with estimates of insured losses running approximately ten billion dollars (Insurance Information Institute, 2020). Effective management of wildland fire is a critical dimension of maintaining healthy and sustainable wildlands. Actionable knowledge of the relationships between fuel, fire behavior, and the effects on the ecosystem and human development can help land managers develop elegant solutions to wildfire problems. Remotely sensed imagery is commonly relied on in assessing the impact of fire on the ecosystem (Eidenshink et al., 2007). The knowledge gained from remotely sensed data enables land managers to better understand the effects fire has had on the landscape and develop a more effective management response facilitating ecosystem recovery and resiliency.

2.1.1. Background

Although researchers face many challenges in understanding fire behavior and its corresponding effect, we investigate the improvements of what changing data resolution creates when analyzing wildland fires. The improvement with which wildland fire severity and extent can be mapped from medium resolution satellite imagery using machine learning trained from hyperspatial sUAS imagery relied on previously published efforts, including identification of the ecological factors of interest, identification of

current methods of mapping wildland fire extent, and utilization of sUAS as a remote sensing vehicle.

2.1.1.1. Wildland Fire Severity and Extent

The term "wildland fire severity" can refer to many different effects observed through a fire cycle, from determining how intense an active fire had burned to the ecosystem's response to the fire over the subsequent years. This study investigates the direct or immediate effects of a fire, such as biomass consumption, as observed in the days and weeks after the fire is contained (Keeley, 2009). Therefore, this study defines burn severity as the measurement of biomass (or fuel) consumption (Key & Benson, 2006).

"Wildland fire extent" refers to the area in which a wildland fire has consumed organic material. Identification of burned area extent within an image can be achieved by exploiting the spectral separability between burned organic material and unburned vegetation (Hamilton et al., 2017; Lentile et al., 2006). Patchiness examines the fire's spatial completeness within the extent of the fire, examining how much biomass remains unburned within the fire perimeter (Morgan et al., 2001). This study defines burn extent as the area within which a fire has consumed organic materials, omitting the unburned islands of vegetation within the burned area perimeter.

Vector-based digitization of fire perimeters: Burn extent is often recorded as a polygon geospatial feature, recorded by a global navigation satellite system (GNSS) either from a helicopter flying along the perimeter of a fire or by ground-based mapping of the fire-edge with a handheld GNSS receiver. For the pilot, the burned area edge can be difficult to discern, especially when high vegetation canopy closure and shadows

reduce visibility of surface vegetation from above. For ground-based personnel mapping the burned area either on foot or from a motorized vehicle, following the perimeter can be complicated by both rough terrain and the non-uniform manner in which wildland fire burns across the landscape. Most wildland fires contain islands of unburned vegetation dispersed throughout the burned area, ranging in size up to hundreds of hectares. These unburned islands are typically not mapped due to safety concerns as well as the impracticality of traversing the edge of each of these islands either aerially or on the ground. Even an extinguished fire can be a dangerous environment for humans to work in. Structurally unsound trees are known to fall upon and injure or kill unsuspecting people. Additionally, the patchy, convoluted edge of the fire can often be hard to delineate from the ground, let alone from a manned aircraft (Kolden & Weisberg, 2007).

Raster-based mapping of burn extent and severity: The most common metric used for mapping wildland burn severity from medium satellite imagery is the normalized burn ratio (NBR) which is the normalized difference between the near-infrared (NIR) and shortwave infrared (SWIR) bands (Key & Benson, 2006), calculated as:

$$NBR = \frac{(SWIR - \mathrm{NIR})}{(\mathrm{SWIR} + NIR)} \quad (1)$$

While NBR is effectively used for burn severity mapping, differenced NBR (dNBR), calculated as pre-fire NBR (NBRpre) minus post-fire NBR (NBRpost) (Eidenshink et al., 2007), has been found to have a stronger indicator to burn severity than just NBRpost alone (Escuin et al., 2008). Satellite imagery is a useful dataset from which to calculate dNBR due to the ability to obtain pre-burn imagery corresponding to a study area containing an unplanned wildland fire.

The Landsat program has been provided continuous satellite coverage of the earth at 30 m resolution since 1982, providing 38 years of imagery from which to extract fire history data. The US Departments of Agriculture and Interior maintain the Monitoring Trends in Burn Severity (MTBS) program to map burn perimeters and severity across the US starting in 1984. The MTBS program maps wildland fires across the US using dNBR (Eidenshink et al., 2007). Due to a large number of fires across the US, the MTBS program only maps fires exceeding 400 hectares in the western US and exceeding 200 hectares in the eastern US (USDA Forest Service Geospatial Technology and Applications Center (GTAC), 2020). With the offset in the orbits of Landsat 7 and 8, Landsat's 8-day flyover interval, easily allows analysts to access bi-temporal imagery preceding the fire for NBRpre and post-fire for NBRpost. In the event the fire study area was obscured by clouds or smoke during a pass-over, another scene can be used from either the preceding pass-over for NBRpre or in a succeeding pass-over for NBRpost as necessary. While MTBS is extensively used for mapping large fires across the US, it has been shown to overestimate the burn extent by four to sixteen percent due to oversimplification of burned area polygons and not mapping large unburned islands (Sparks et al., 2015).

2.1.1.2. On Origins of Fires Used in This Study

As for the data used in this research, this study utilized a set of fire data from previous similar research efforts. The wildfire burn areas used in this study are a collection of medium to large size fires in southwestern Idaho ranging from xeric sagebrush steppe fires located within the United States Department of Interior Bureau of Land Management Boise District to the mesic upper Payette and Boise River watersheds

in the United States Department of Agriculture Forest Service Boise National Forest. The naming schema for these fires, as displayed in Table 1, are usually determined by a nearby notable landmark such as a creek or a hill. For example, the Hoodoo fire burn area burned near and across the Hoodoo Creek near Idaho City, Idaho. Typically, these names were the official names of these fires assigned by the local jurisdiction and used by dispatchers. The locations of the burned areas of which acquisition flights were conducted are shown in Figure 1.

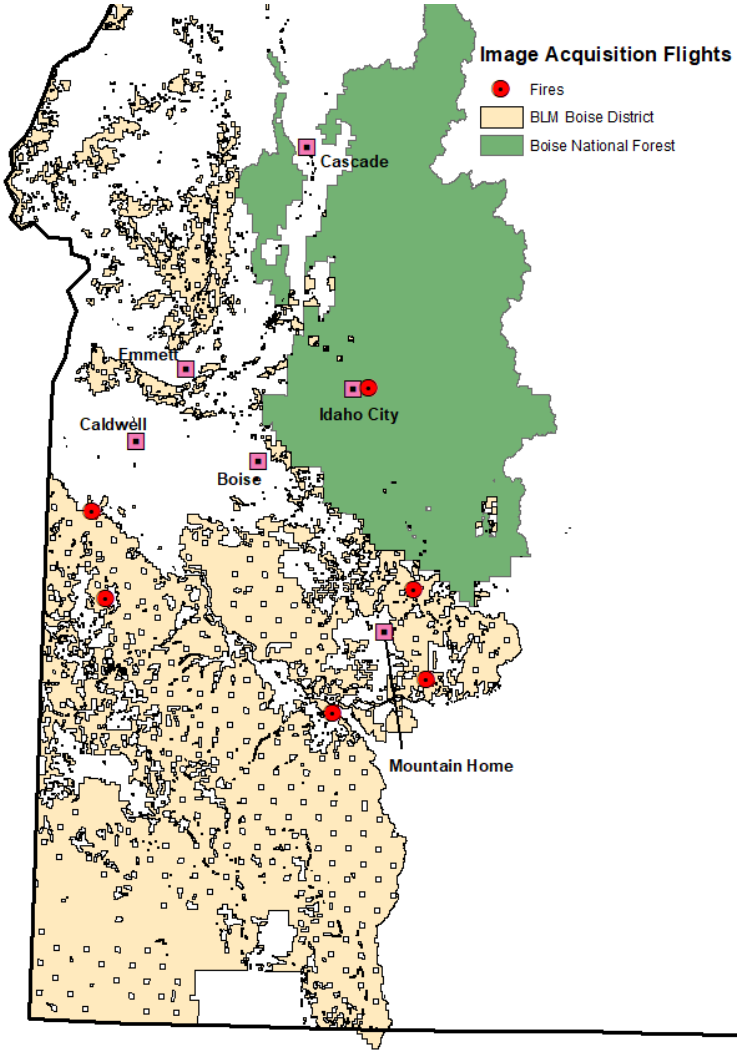


Figure 1. Geographic distribution of wildland fires over which post-suppression image acquisition flights were conducted in southwestern Idaho.

2.1.1.3. Utilization of sUAS for Mapping Wildland Fire

The proliferation of small unmanned aircraft system technology has made the procurement and use of remotely sensed imagery a viable possibility for many organizations that could not afford to obtain such data in the past. A small unmanned aircraft system (sUAS) is a designation given by the US Federal Aviation Administration to UAS that weigh between 0.25 kg and 25 kg (FAA, 2020). Most commercially available sUAS come with an onboard digital camera, a multi-spectral sensor with three bands capturing visible light in the blue, green and, red spectrum ranging from 400 nm to 700 nm (Lebourgeois et al., 2008). Spectral responses in the visible spectra can be used to differentiate between different image features such as white and black ash (D Hamilton et al., 2017; Dale Hamilton et al., 2019; Lentile et al., 2006) and other features of interest to fire managers such as vegetation type (D Hamilton et al., 2017; Rango et al., 2009).

New advances in sUAS capabilities enable imagery acquisition with a spatial resolution of centimeters and temporal resolution of minutes (Laliberte, 2010). The temporal responsiveness of acquiring imagery with a sUAS is significantly increased due to the increased availability of the sUAS being able to be flown at any desired time as opposed to Landsat imagery which can only be acquired when the satellite flies over the scene every 16 days, assuming the scene is not obscured by smoke or clouds during the flyover. However, although a manned aircraft can be a viable alternative, United States governmental agencies update their aerial manned photography through programs such as the Department of Agriculture National Agricultural Imagery Program every few years. However, this is nowhere near the update frequency of satellite imagery as needed in this experiment. In comparison to sUAS, on-demand manned aircraft aerial photography is

far more expensive, with rental costs for manned aircraft running thousands of dollars per hour. High-resolution imagery was only needed on-demand and therefore sUAS imagery was the best temporally responsive and cost-effective solution to the US land management agency regulatory need to acquire post-fire data including mapping burn extent and fire effects within 14 days after fire containment.

Aerial imagery for this project was acquired with a DJI Phantom 4 with a 12-megapixel color camera. The imagery acquired with the sUAS was taken while flying at an altitude of 120 m above ground level (AGL), giving the photos a spatial resolution of 5 cm per pixel (Key & Benson, 2006). Objects that are wider than that pixel resolution will be discernible in the acquired hyperspatial imagery as shown in Figure 2a. The black rectangles in the image are burned areas. Small lines and patches of white within the burned area are white ash from sagebrush which was fully combusted by the fire. The unburned vegetation consists primarily of annual and perennial grasses and forbs, Wyoming big sagebrush (Artemisia tridentata spp. Wyomingensis), and yellow rabbitbrush (Chrysothamnus viscidiflorus). The scene shown in Figure 2 contains two western juniper trees (Juniperus occidentalis spp. occidentalis).

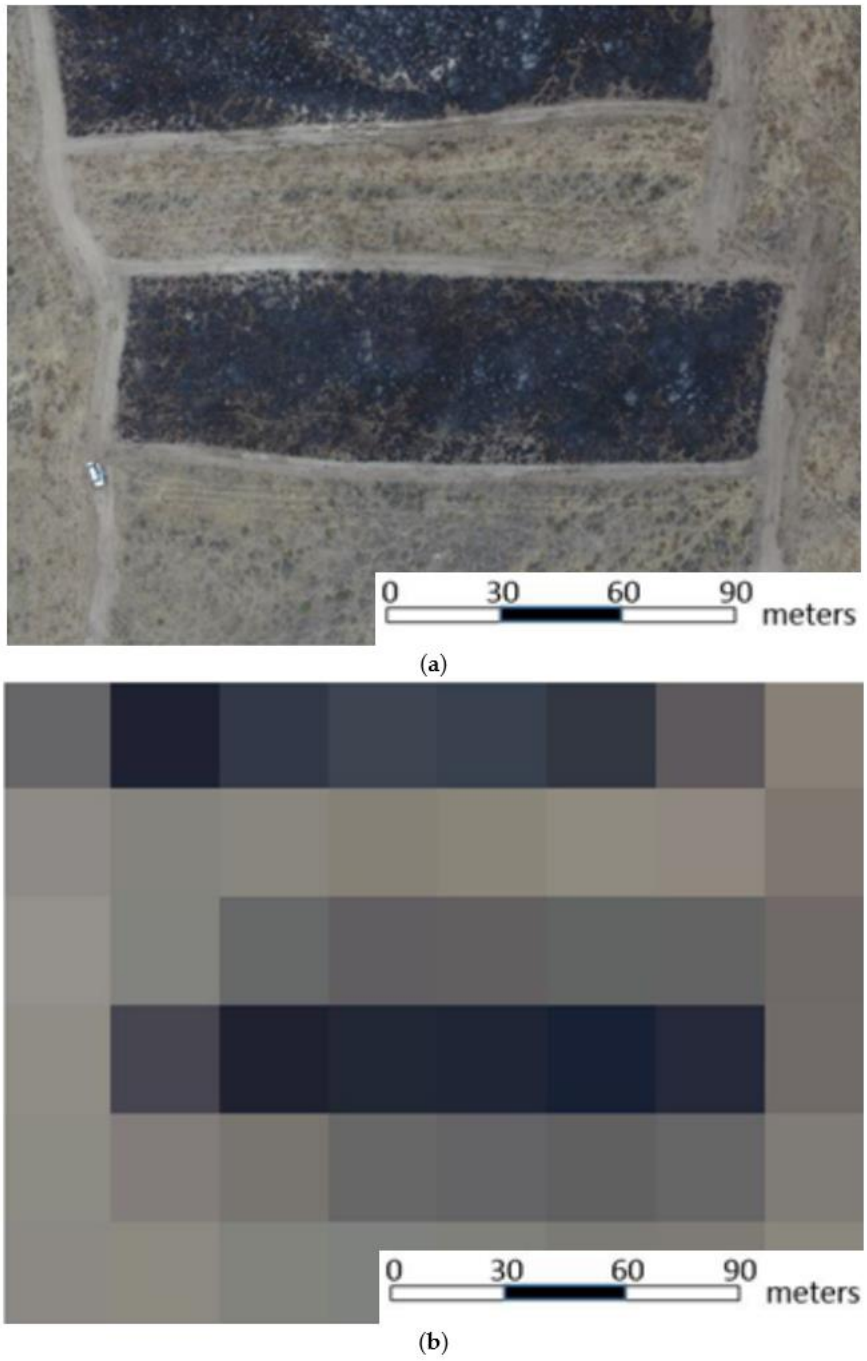


Figure 2. (a) Image of a rangeland study area acquired with a Phantom 4 sUAS flying at 120 m AGL with a spatial resolution of 6 cm per pixel. (b) Same scene resampled to 30 m resolution with six rows and eight columns of pixels (Hamilton, 2018).

Features easily identified in hyperspatial (5 cm) imagery are lost in medium resolution (30 m) Landsat satellite imagery, being aggregated into more dominant neighboring features. Figure 2b shows the same scene as the preceding image but

resampled to 30 m spatial resolution having 48 pixels aligned in six rows by eight columns. Mapping burn severity and extent was found to have significantly lower accuracy when using imagery with 30 m spatial resolution such as Landsat than was found with hyperspatial imagery acquired by flying the sUAS over the burned area (Dale Hamilton et al., 2019). While burn severity and extent can be mapped with much higher accuracy using hyperspatial imagery acquired with a sUAS, this research team found that current technical and regulatory constraints on drone usage realistically will only allow for the acquisition of up to 600 hectares per day (Goodwin & Hamilton, 2019). That same flight extent would be a small part of a single Landsat scene. To effectively map large class F (>400 ha) and G (>5000 ha) fires (NWCG, 2020), a larger and more efficiently acquirable imagery satellite system such as Landsat still needs to be utilized to obtain a large enough analysis extent to map the whole fire.

2.1.1.4. 30-Meter Burn Severity Mapping Analytics Trained with Hyperspatial Classification

In evaluating the effects of spatial resolution on burn severity and extent mapping accuracy, Hamilton (Hamilton et al., 2019) established the following methodology which enabled the use of hyperspatial burn severity classes of unburned vegetation, black ash and white ash as classified using a Support Vector Machine (SVM) (Dale Hamilton, 2018) for training machine learning algorithms to map burn severity from medium resolution (30 m) imagery. This previous effort evaluated only the effect of spatial resolution on mapping accuracy, using 30 m imagery that was resampled from the hyperspatial imagery acquired with the sUAS, thereby removing other variables from consideration which could affect accuracy, such as sensor radiometric resolution, atmospheric influence, and temporal resolution. Machine learning classifiers mapped

burn severity and extent from 30 m imagery separately, and were trained using hyperspatial burn severity and extent classifications using the following methods:

1. Hyperspatial orthomosaics (5 cm) were resampled to have medium resolution of 30 m (30 m), which is an equivalent spatial resolution to Landsat imagery, with the spatial reference for the 30 m imagery being calculated from the spatial reference from the 5 cm orthomosaic.

2. Labeling 30 m training pixels using fuzzy logic by:

    a. Calculating 5 cm burn severity class pixel density within each 30 m pixel, where density is the percentage of 5 cm pixels for a specific class that are found within the containing 30 m pixel.

    b. Applying fuzzification, the post-fire class density is used to establish where fuzzy set membership transitions from 0 to 1 over a range of values. Burn extent transitioned from unburned to burned between 35 and 65 percent of 5 cm pixels being unburned as shown in Figure 3. Burn severity transitioned between low biomass consumption and high biomass consumption between 33% and 50% (Lewis et al., 2008) of burned pixels being classified as white ash as shown in Figure 4.

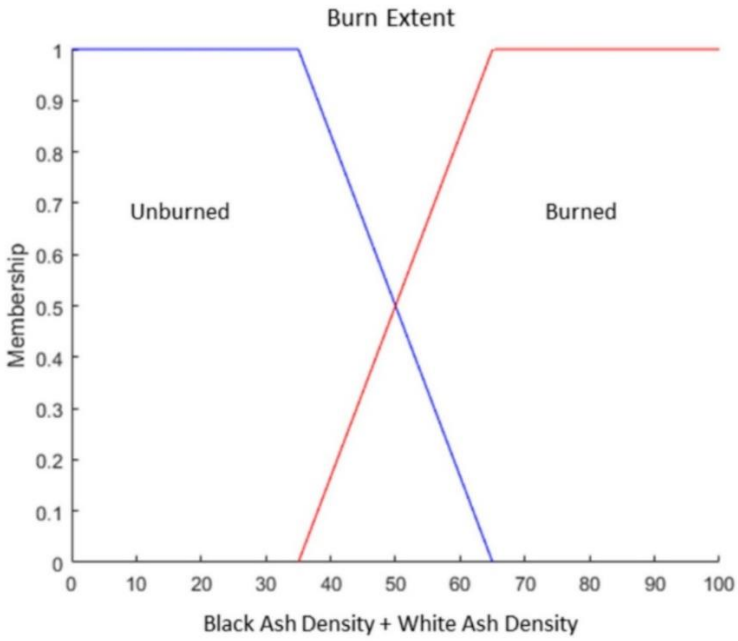


Figure 3. Burn extent fuzzy set showing fuzzy set membership of 30 m pixel transition between Unburned and Burned from 35 to 65 percent of 5 cm pixels being classified as burned.

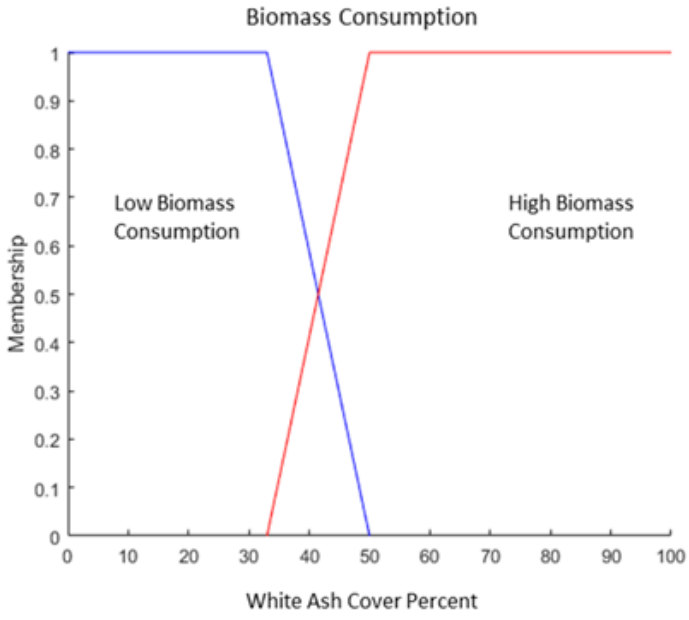


Figure 4. Biomass Consumption fuzzy set showing fuzzy set membership of 30 m pixel transition between Low and High Biomass Consumption from 33 to 50 percent of 5 cm burned pixels being classified as white ash.

c. Activate fuzzy rules by applying fuzzy logic. When evaluating a series of fuzzy if statements, as shown in Figure 5, the data is defuzzified by selecting for activation the action associated with the if expression that has the highest value where a fuzzy AND is evaluated as taking the minimum value of either of the associated operands.

```
If (extent:burned AND combust:high)
    training pixel = White Ash
If (extent:burned AND combust:low)
    training pixel = Black Ash
If (extent:unburned)
    training pixel = Unburned
```

Figure 5. Fuzzy logic algorithm for labeling 30 m pixels from 5 cm burn severity classes.

3. Each of the 30 m pixels was labeled with training data labels with the SVM training on 70 percent of the 30 m training pixels. The remaining labeled 30 m pixels withheld for validation of the SVM.

2.2. Materials and Methods

In order to complete the goal of this effort, a method was developed which mapped and analyzed wildland fire extent using spatial satellite imagery. This was done by converting high-resolution hyperspatial training data to a lower resolution training data using fuzzy logic. This created satellite resolution training data and enabled the burn extent to be determined and analyzed from satellite imagery using an SVM. Although this experiment will be using Landsat 8 imagery in particular, this method is not bound to any specific earth-observing satellite and can be applied to any spatial imagery provided the correct format. The fires used in the research were from the same set of fires used from a previous study (Hamilton, 2018; Hamilton et al., 2019) as mentioned in Section 1.1.2.

Hamilton (Hamilton et al., 2019) mapped wildland fire burn-extent from hyperspatial data with an accuracy of up to 98% on 5 cm pixels. For this experiment, hyperspatial training data will be assumed to be an adequate reference point when determining the accuracy of the SVM on the 30 m spatial imagery. This assumption enables the experiment to use hyperspatial training data resampled to satellite resolution training data using fuzzy logic. This experiment also investigated the effect different spectral bands have on burn extent and severity mapping accuracy.

2.2.1. Assembling Spatial Extent Burn Indicator

As mentioned before, one of the objectives of the application of this experiment was to provide an efficient and a more consistent method of running the experiment so that the results of the experiment were consistent and adequate to analyze using the scientific method. Thus, an application was assembled to evaluate the data in a consistent manner. The application for this experiment is divided into four sections. The first part of the application adjusts the geoposition alignment of the hyperspatial and satellite imagery. The second part utilizes an SVM to create hyperspatial training data from hyperspatial imagery. The third part converts hyperspatial training data into satellite resolution spatial training data using fuzzy logic. The final part of the application runs the SVM on the satellite scene using the newly created training data. This application will provide an effective and efficient way to analyze accuracy impact.

2.2.1.1. Coregistering Hyperspatial and Medium Resolution Images

Before resampling the resolution of the training image, acquired with an sUAS, from hyperspatial resolution to satellite medium resolution, such as is acquired by the Landsat Satellite project, preprocessing and imagery alignment is required to establish

alignment with the smaller pixels and the larger medium resolution image pixels. The first alignment step reprojects the imagery so the hyperspatial image and the medium resolution image have the same spatial reference. In Figure 6, the projection for the medium resolution image (orange) and the hyperspatial sUAS image (blue) have different spatial references, making the pixel boundaries misaligned between the two images.

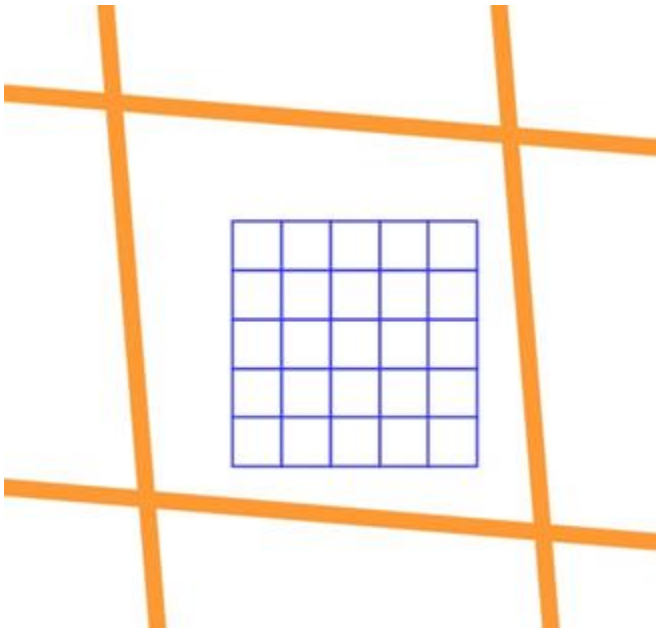


Figure 6. Medium resolution satellite image pixels (orange) and hyperspatial sUAS image pixels (blue) prior to reprojection.

Reprojection to the same spatial reference results in pixel boundaries that are aligned between the images as shown in Figure 7.

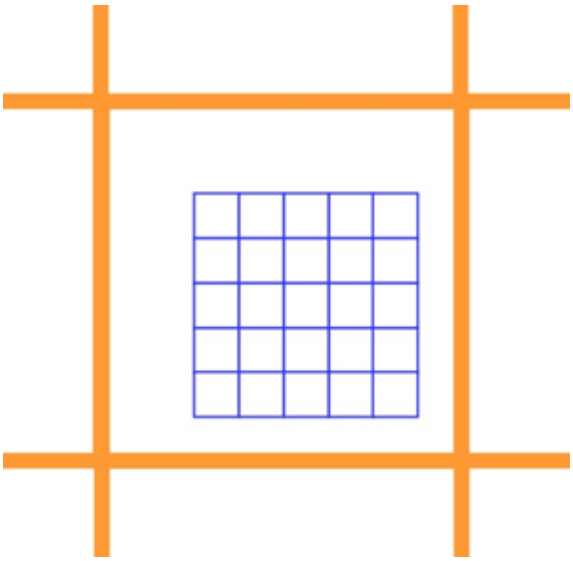


Figure 7. Example of projection before the border is added.

Once the pixel boundaries are aligned between the hyperspatial and medium resolution images, the medium resolution image is cropped so it only contains pixels that overlap the hyperspatial image. At that point, the extent of the hyperspatial image does not correspond to pixel boundaries, as is shown by the white space between the blue pixels and the boundary of the overlapping orange pixel in Figure 6. The extent of the hyperspatial image needs to be expanded out to the edge of the clipped medium resolution image. Additional rows and columns of null pixels are added around the hyperspatial image using OpenCV, expanding the extent of the hyperspatial image until it corresponds to the pixel outer pixel boundaries of the overlapping medium resolution pixels, which are shown in green in Figure 8.

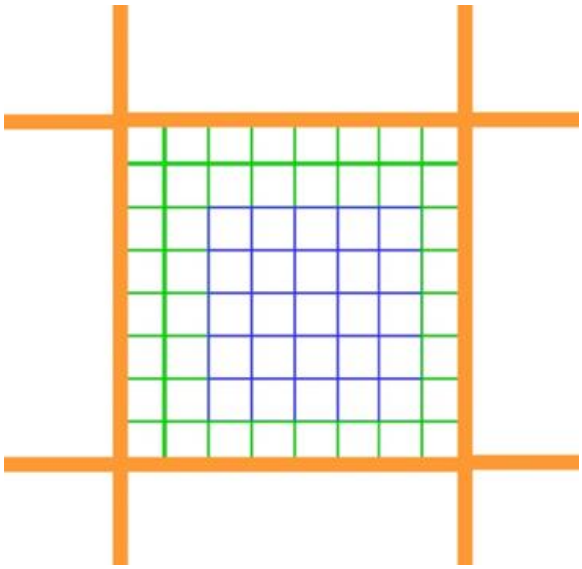


Figure 8. Example of projection after the border is added.

Once the additional rows and columns of null pixels are added to the hyperspatial image, the spatial resolution of the hyperspatial image is adjusted to account for the addition of the new null pixels around the perimeter of the orthomosaic as shown in Figure 9.

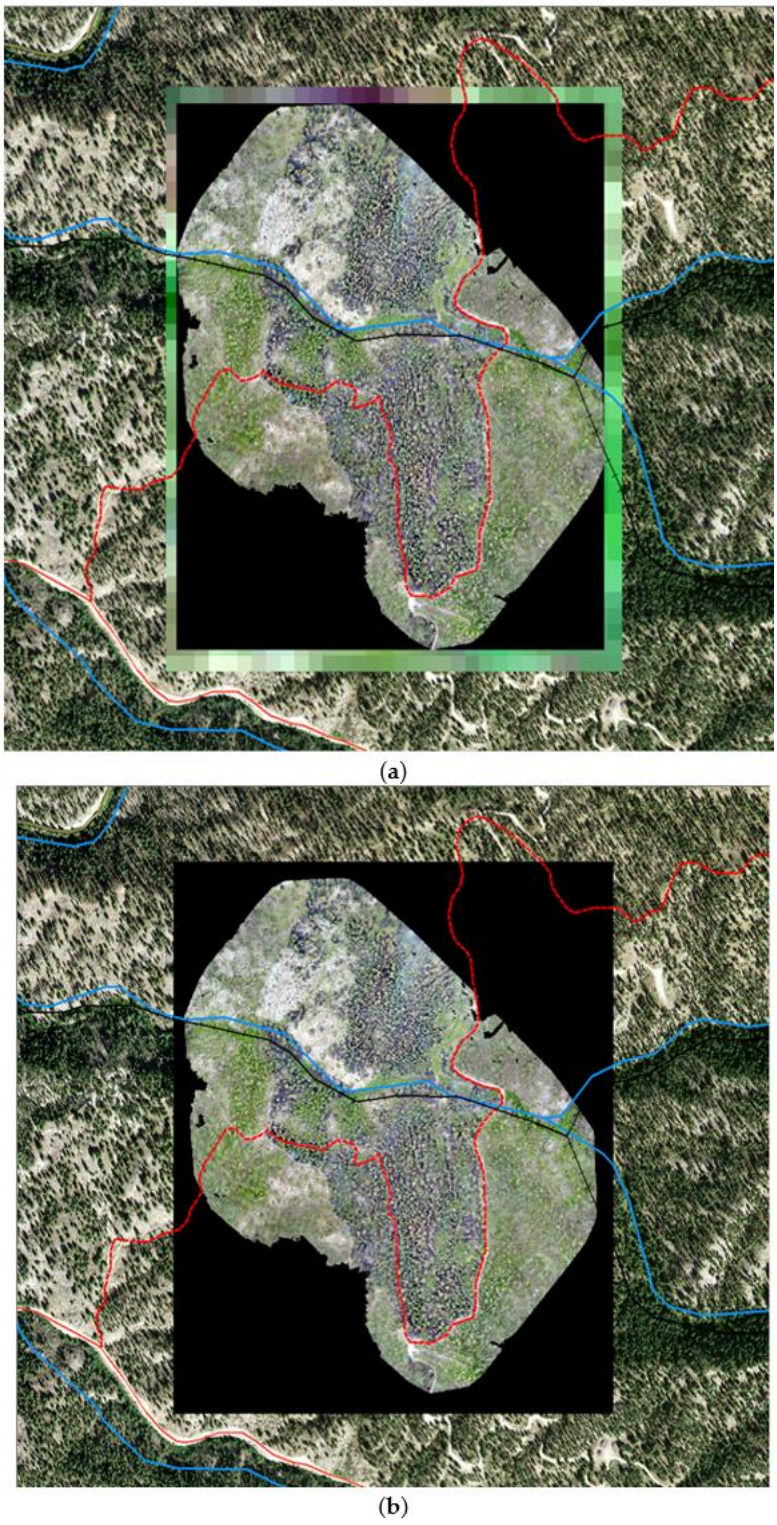


Figure 9. (a) sUAS orthomosaic rendered on top of clipped Landsat image; (b) sUAS orthomosaic with a buffer of null pixels giving it the same extent as the clipped Landsat image. Red polylines represent roads and trails. Blue polylines represent creeks. Black hatched polyline represents a historic railgrade.

2.2.1.2. Creating Hyperspatial Training Data

Once all the spatial imagery was aligned, the hyperspatial imagery burn extent and severity were mapped separately using a support vector machine. The training data used for the hyperspatial imagery was from a data set that had previously been processed while mapping burned areas over multiple years, with accuracy ranging up to 98% (Hamilton, 2018; Hamilton et al., 2019). After burn severity and extent were mapped, the data was denoised using image morphology, reducing sub-object sized noise (Gonzalez, 2008). An example of this noise reduction on the burn extent is shown in Figure 10. The output would then be used as hyperspatial burn extant data resampled to satellite resolution burn extent data used as training data in the next section. A similar process was done with the severity case and the images were combined using the logic mentioned in Section 1.1.4.

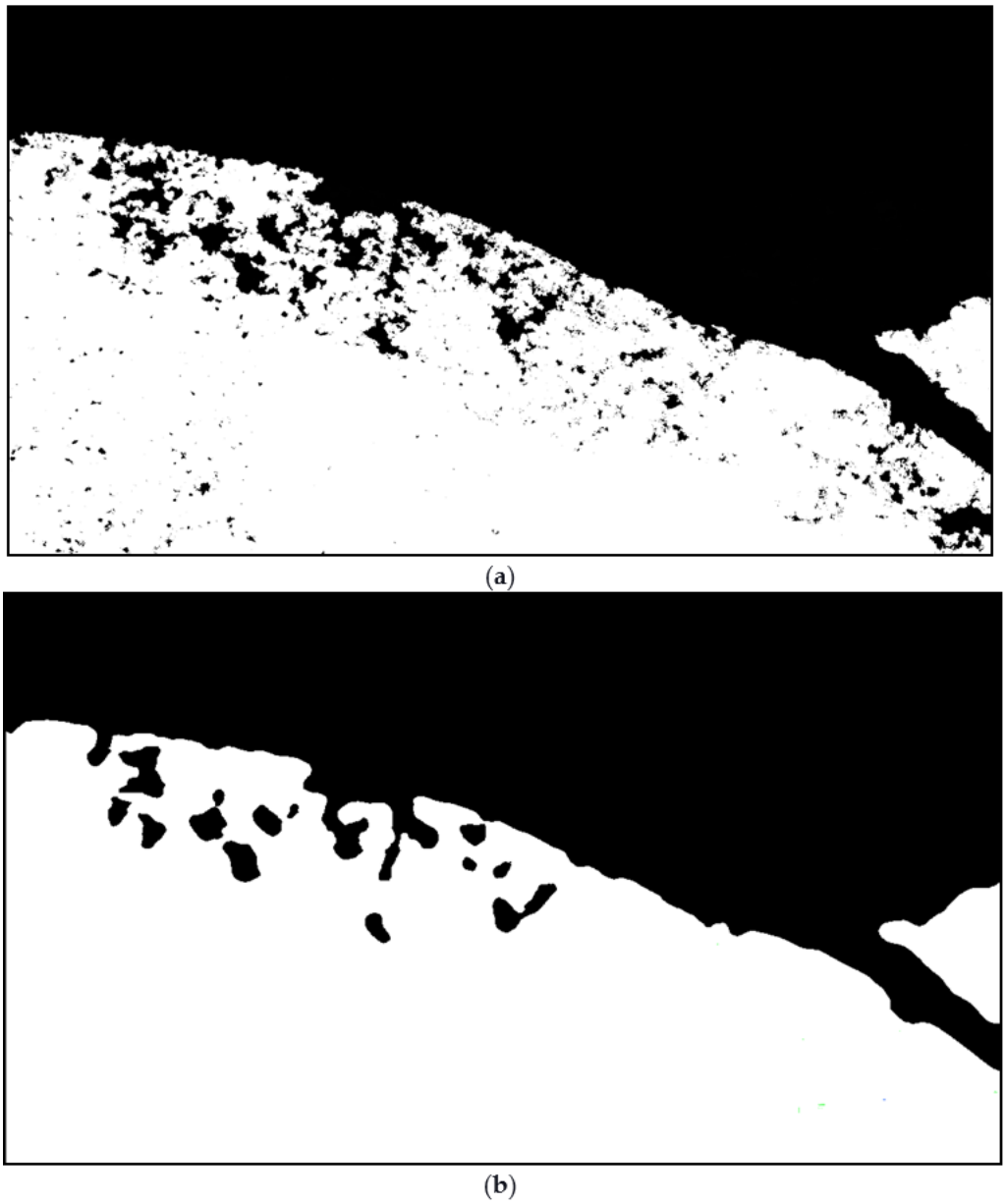


Figure 10. Comparison of Hyperspatial classification with its Denoised Output. White pixels are the burned data and the black pixels are the unburned data. (a) example of an image before denoise tool is applied; (b) example of an image after denoise tool is applied. These images are a post processing view of the bitmap output data with each pixel containing one class value Burned or Unburned.

2.2.1.3. Resampling Training Data to 30 m Using Fuzzy Logic

After generating the hyperspatial burn severity output, fuzzy control was used to label overlapping medium resolution pixels to be used for training an SVM to map burns from satellite imagery. Fuzzy logic allows decisions to be based on imprecise boundaries

rather than relying on precise boundaries that are used by Boolean logic. This use of vagueness allows the expression of how much the data fits given criteria, transitioning from one class to another over a range of values. Fuzzy logic is often more applicable to ecological data than the crisp delineations resulting from Boolean logic, where data will transition from one class to another at a single threshold value.

The fuzzy set theory allows the specification of how well an object satisfies a vague criterion (Russell & Norvig, 2010) with fuzzy logic providing a means for specifying that the transition from one class to another is not demarked at a single value but transitions from one class to another over a range of values (Han, 2012). For example, Hamilton (Hamilton, 2018) defined the transition from low to high biomass consumption as occurring between 33 and 50 percent of burned pixels being classified as white ash as shown in Figure 4. Rather than set membership being expressed as either zero or one, as is the case with Boolean logic, fuzzy logic allows set membership to be specified as a range of membership from 0.0 to 1.0. For example, using these thresholds, a plot with 40 percent white ash cover from the Biomass consumption fuzzy sets shown in Figure 3 would have 0.41 membership in the high biomass consumption set and 0.59 membership in the low biomass consumption set as shown in Figure 11.

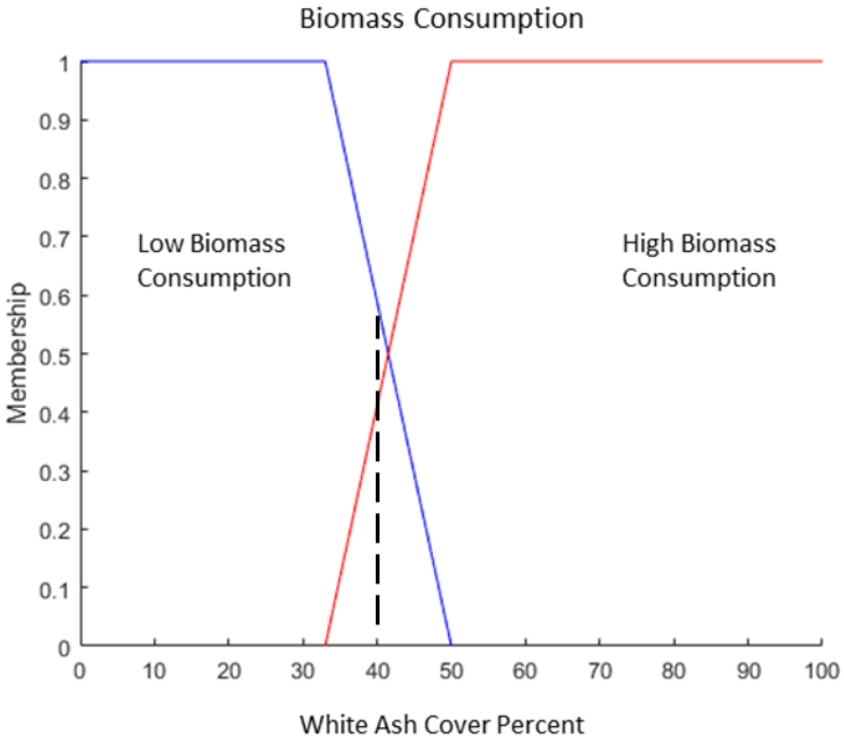


Figure 11. Biomass Consumption Fuzzy Set Membership. With 40 percent White Ash Cover denoted by the dotted line, there is 0.41 membership in High Biomass Consumption and 0.59 membership in Low Biomass Consumption.

Assignment of the post-fire burn extent and severity classes to a 30 m pixel is based on a combination of whether the pixel burned and white ash cover within the pixel. In neither case is Boolean logic appropriate for determining set membership. Increasing the density of a class by a handful of 5 cm pixels and as a result changing a Boolean expression from false to true does not adequately describe the state of the pixel. As mentioned in the background, the first set of data to be evaluated was between the burned and unburned sets, with the burned set consisting of the combination of pixels classified as either burned or unburned. The fuzzy logic transition for these criteria from 5 cm pixels to 30 m was from 35% to 65% (Hamilton, 2018), as previously shown in Figure 3.

An additional set membership was used that measured biomass consumption, which evaluated the relationship between the density of black ash and white ash. The white ash cover was expressed as a percentage of white ash to burned (white and black ash) pixels, transitioning between low and high biomass consumption over a transition from 33% to 50% (Hamilton, 2018; Goodwin, 2019), as previously shown in Figure 4.

Fire biomass consumption needed to be only evaluated on the burn extent, and thus any white ash detection outside of the burned boundaries was disregarded. This filter could be applied using fuzzy logic to fuzzy set membership, a fuzzy AND is expressed as taking the minimum value of the expressions on either side of the AND operator. Likewise, a fuzzy OR is expressed as taking the maximum value of the expressions on either side of the OR operator. When evaluating a series of fuzzy IF statements, as previously shown in Figure 5, the data are defuzzified by selecting for activation the action associated with the IF expression with the highest value.

After the fuzzy logic was applied, the data would be converted into 30 m training data by taking the most dominant set as determined by the fuzzy logic control and was labeled either white ash, black ash, or unburned accordingly. Since the snapping software aligned all the images, the newly created training data from the fuzzy logic training data came out aligned with the associated satellite scene and would be ready to train an SVM using Landsat imagery, which had been labeled using fuzzy logic on the hyperspatial burn extent and severity classifications as described above. After the training data were created, one third of the training data were set aside as validation data to test the classifier’s accuracy.

2.2.1.4. Running the SVM on the Satellite Scene

After the satellite scene training data was created, the satellite scene and the newly created satellite resolution training data was used as the input into the SVM. However, before running the SVM the spatial training data was clipped to balance out the burned and unburned pixels for consistency across fires. This clipping was necessary to balance for the ratio between burned and unburned pixels is normalized between 45–55%. Balancing the burned and unburned pixels was done because most of the selected fires used in this experiment had an imbalance burned and unburned pixel ratio where the unburned pixels numerically dominated the burned pixels. Thus, a solution was developed through an algorithm that repeatedly stripped off each side until the ratio was met. This solution would be sufficient to the imbalance issue because most of the edge pixels were predominantly burned pixels in all the fires that were used. After preprocessing was complete, the SVM would create a TIF image of the fire's burn extent and severity through 30-m resolution.

2.2.1.5. Collecting Results

Once the SVM finished classifying, an output image of the entire satellite scene was produced. A simple testing program was developed to analyze the SVM's accuracy when it mapped burn extent and severity on a satellite. Once the data was normalized, all the pixels were compared with the training data, and the equation shown in Equation (2) was applied to determine the accuracy of the SVM's burn extent.

$$ExtentAccuracy = \frac{|TruePixels|}{|TotalPixels|} \quad (2)$$

However, to determine burn severity accuracy, only the white ash and black ash pixels were evaluated. Thus, all unburned pixels were omitted, and the accuracy

calculation applied accuracy validity to white and black ash pixels, as shown in Equation (3).

$$SeverityAccuracy = \frac{|TrueBlackAsh| + |TrueWhiteAsh|}{|TotalBurnedPixels|} \quad (3)$$

2.3. Results

A set of fires previously used by Hamilton in (Hamilton, 2018; Hamilton et al., 2019) was selected for analysis. The estimated accuracy of mapping out burn severity using color satellite imagery (comprised of red, green, and blue spectral bands) in comparison to the high-resolution results from color imagery, as shown by Hamilton (Hamilton, 2018), is shown in Table 1.

	5 cm Extent	30 m Extent Results	5 cm Severity	30 m Severity Results
Elephant	98.5	58.52	98.67	95.30
MM106	86.58	71.5	98.68	100
Hoodoo	99.29	73.92	95.22	41.50
Immigrant	98.34	77.67	97.97	100
Jack	94.74	57.09	96.97	19.06
Owyhee	87.65	85.40	87.42	100
Mean	93.32	73.116	95.252	72.112

Table 1. Experiment Results Compared with the 5 cm from (Hamilton, 2018; Hamilton et al., 2019).

This experiment also investigated how the accuracy would change if color satellite imagery were replaced for a set of infrared (IR) bands. The IR imagery comprised the following Landsat bands: shortwave infrared 1 (0.156–1.660 μm),

shortwave infrared 2 (2.100–2.300 µm), and near infrared (0.845–0.885µm) (NASA, 2020). The results of this experiment are shown in Table 2.

	30 m Color Extent	30 m IR Extent	30 m Color Severity	30 m IR Severity
Elephant	58.52	55.95	95.30	97.83
MM106	71.5	69.1	100	100
Hoodoo	73.92	71.75	41.50	36.48
Immigrant	77.67	78.19	100	100
Jack	57.09	56.68	19.06	17.40
Owyhee	85.40	56.61	100	N/A [1]
Mean	70.68	64.71	75.98	70.34

Table 2. Color Experiment vs IR Experiment. [1] The fire was too small for the experiment to detect any white ash.

2.4. Discussion

This experiment's results did not converge to a specific accuracy, but instead the accuracy varied across fires. This was expected as many of the fires used in the experiment varied in severity, differed between forest and grass fire, had different atmospherical conditions, and varied in canopy cover. For example, the Elephant fire had a thin layer of cloud haze above it, which skewed off the color balance to be higher, which decreased the SVM's accuracy of detected darker burned pixels. Although each image may have one or multiple of these challenges, these problems were addressed through many various methods during the development of the experiment. However, all could not be met seamlessly. One of these challenges, such as the canopy cover, was addressed by applying the denoise tool, as previously shown in Figure 10. This tool would allow an unburned tree canopy and other smaller unburned objects with burned pixels around it to be marked as noise and replaced in with burnt pixels. Another

challenge pertaining to the atmospherical condition was based on the fact that each satellite image, covering the same spatial area, had slightly different solar lighting, atmospheric density, and moisture. This color imbalance problem was solved by utilizing different hyperspatial training imagery for every fire in the experiment, although the separate training data were all created a consistent way. Ideally, the experiment should not need separate training data for each fire. Without creating separate training data for each fire, the experiments overall accuracy plummeted from color imbalance from atmospheric interference. Although bear in mind that each of the separate training was generated a consistent way as done by Hamilton (Hamilton et al., 2019). However, creating separate training data is necessary because each satellite image has its own unique state. Future work could create a single set of training data that is cross-compatible with all fires, which will account for the ecological, atmospherical, and temporal setting of the fire.

As shown in Table 1, the experiment results for the 30 m burn extent shows that the accuracy of mapping burn extent from a satellite image using hyperspatial training data has a mean of 71% of 30 m extent results accuracy. The accuracy of the burn extent of the 30 m was expected to be lower than the accuracy of the 5 cm burn extent (Hamilton, 2018). Many factors could impair the accuracy level. One of these factors include is the satellite image had a lot of atmospheric dissidence, which caused the sUAS training data to pick up far more unblemished data as previously discussed. Another of these factors includes that most of these fires were considered small fires and included many edge cases as shown in Figure 12. These edge cases would severely decrease the accuracy level of smaller fires. Therefore, accuracy is expected to increase when mapping

larger wildland fires. To increase the overall accuracy of this experiment, future edge case analysis can be done to significantly increase the accuracy of mapping out wildland fires with satellite imagery.

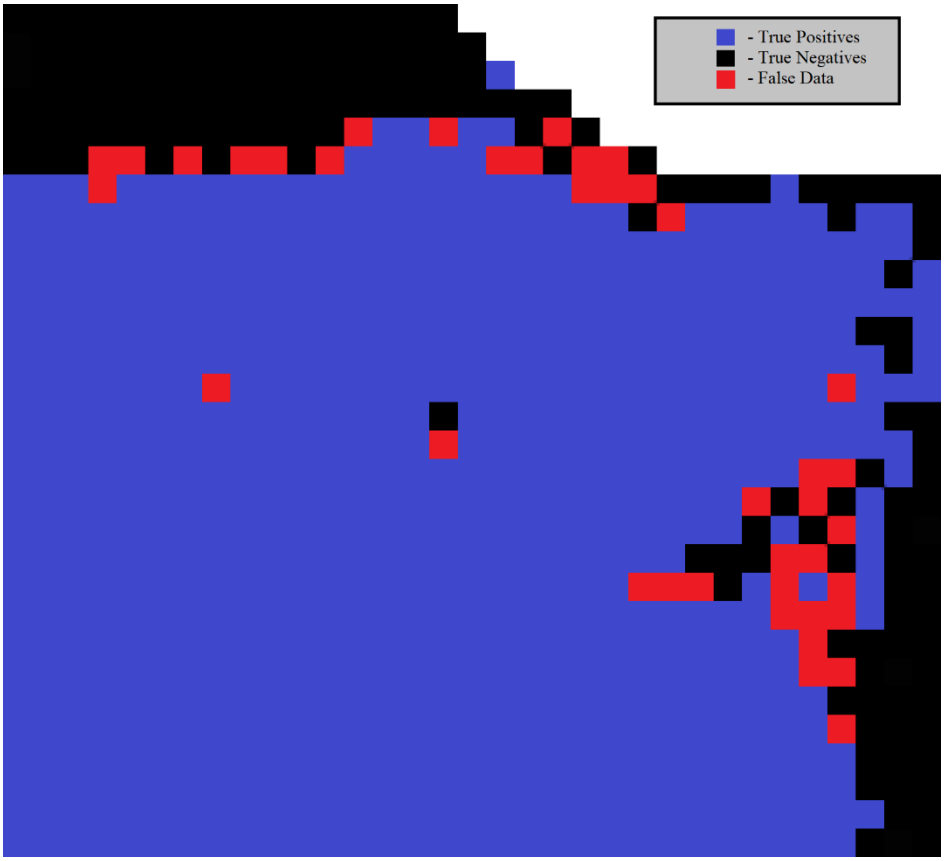


Figure 12. Some error analysis concluded that most False Positives or Negatives, indicated in red, are predominately located at the edge of the burn area.

Observing the IR experiment in compared to the RGB experiment, a decrease in accuracy is observed. However, this is contradictory to what was expected (Key & Benson, 2018). The accuracy decreases, for the most part, is slight, and after further analysis most of the error falls on the SVM edge cases similar to what is shown in Figure 12. A solution for this error is to use IR imagery hyperspatial and spatial imagery in this experiment, which theoretically will decrease error. However, this is left for future work. However, since the experiment was done on parts of the burn area where edge cases are

predominately located, theoretically, the SVM would have better accuracy on a larger fire as the ratio between edge pixels and non-edge pixels is decreased.

2.5. Conclusions

This experiment successfully applied sUAS hyperspatial data to be used as training data to map out the burn extent on a satellite imagery. As shown in Table 2, the experiment went further than just using color imagery to determine accuracy. This experiment has also applied IR spectroscopy imagery in which has been previously proven to improve fire mapping; thus, results were expected to improve (Key & Benson, 2006). However, as previously discussed, the IR experiment results proved contradictory to what was expected and previous work (Key & Benson, 2006). Thus, it was assumed that the issue could be readdressed by using IR hyperspatial training data for the IR imagery as previously discussed.

One of the research goals was to determine the change in accuracy of burn extent mapping between the hyperspatial imagery and satellite imagery. The tool was created and tested on this experiment which showed improved accuracy from 5 cm to 30 m. However, for future work, one could take this research further by determining how the decrease in the resolution would decrease the accuracy of fire extent and severity mapping. This knowledge could enable future researchers to quickly and accurately determine which resolution would best fit their area of research, given their resources and budget constraints.

*Source code developed as well as data used in this effort are available for download at https://firemap.nnu.edu/satellite-burn-mapping.

Implications for Local Management

Improved mapping of fire effects resulting from the development of methods, analytic tools, and metrics resulting in increased fire effects mapping accuracy will improve fire management. These improvements affect post-fire recovery planning and other management operations, which are driven by the extent of the fire and how much biomass was consumed by the fire. Development of post-fire recovery plans includes outlining the restoration activities that determine management response to the fire, facilitating ecosystem recovery and resiliency (Eidenshink, 2007). In order to reduce risks to affected resources, managers will also need increased capacity to determine potential effects on neighboring resources such as hydrologic features, infrastructure, and wildlife habitat.

Existing data about vegetation within the perimeter of a fire are rendered obsolete because of a disturbance such as a wildland fire. Increasingly accurate burn extent and biomass consumption mapping will improve efforts to update geospatial vegetation layers such as existing vegetation type, cover and height to accurately reflect fire fuel data following a wildland fire (Eidenshink, 2007).

Improved mapping will also result in more complete fire history data, facilitating the inclusion of the spatial extent and biomass consumption of small fires which are commonly omitted from fire history data (Hamilton, 2018). The inclusion of knowledge about small fires will reduce the omission of the most ecologically diverse areas burned within a landscape (Hamilton & Hann, 2014). Additionally, calculations will be improved for the departure of current fire frequency from historical fire frequency, a key metric for determining ecosystem resilience (Bowerman & Bowerman, 2014).

2.6. Future Work

This research was unique in a way that the initial intent was to pave a path for future research. As discussed, prior, each satellite scene has its own unique solar lighting, atmospheric density, and moisture effect. This variance across each fire would cause a shift in the pixel values for each of the classes used in this experiment. For example, a burned pixel from a satellite image in early June may have a SWIR value of 1000, but in late August, the SWIR value of a burned pixel in a satellite image may have a value of 700. This could be caused by the solar position during the time of year, the lack of moisture towards the end of the year, and random atmospheric concentration. To solve this problem, one could calibrate the experiment by utilizing ground objects known to remain consistent throughout the year, such as pavement or lakes. Taking those base values to calibrate each satellite image could fix the problem and avoid utilizing different training data for each fire experiment. However, due to complexity, alternatively, a consistent set of hyperspatial data was used to create the satellite training data instead, as shown in Section 2.2.2.

Another challenge that occurred was the color images showing better accuracy than the IR images. As mentioned before, this is contradictory to previous work (Key & Benson, 2006). However, as mentioned previously, most of the inaccuracy occurred at the edge cases, as shown in Figure 12. Thus, to improve accuracy, more work on edge case analysis can be done, which could significantly improve the experiments accuracy. However, due to data and time constraints, this was left off for future work.

3. Epilogue

Since the publication, I have acquired a lot of skills in which will facilitate future analysis. For example, when running tests, the whole process could take 10 minutes to complete and any error could result a lengthy debugging process. Utilizing Jupiter Notebook scripting allows you to save data value between run in which would significantly shorten the process for debugging and extensive testing.

As mentioned in the future work sections, incorporating IR satellite imagery to map fire extent will be useful. Although I created a method in which one could using a combination of up to five different spectral bands to keep the results of the research focused, I only posted the results of RGB imagery and briefly discussed the results of utilizing IR imagery. Although the process is setup and ready to use different spectral bands and resolutions, which will some of the future work can include using different bands, make the process more user friendly, and testing accuracy increase on different satellite resolution. Some of these studies I have begun, but I quickly realized that it could be a whole study of its own in which led me to decide to leave it off for future work.

4. References

Eidenshink, J.C.; Schwind, B.; Brewer, K.; Zhu, Z.-L.; Quayle, B.; Howard, S.M. A project for monitoring trends in burn severity. *Fire Ecol.* **2007**, *3*, 3–21.

Escuin, S.; Navarro, R.; Fernandez, P. Fire severity assessment by using NBR (Normalized Burn Ratio) and NDVI (Normalized Difference Vegetation Index) derived from LANDSAT TM/ETM images. *Int. J. Remote Sens.* **2008**, *29*, 1053–1073.

Federal Aviation Administration (FAA). Frequently Asked Questions. *Unmanned Aircr. Syst. Freq. Asked Quest.* **2020**. Available online: https://www.faa.gov/uas/resources/faqs/ (accessed on 20 May 2020).

Gonzalez, R.; Woods, R. Digital Image Processing: Pearson Prentice Hall. Pearson Prentice Hall: Upper New Jersey, NJ, USA, 2008.

Goodwin, J.; Hamilton, D. *Archaeological Imagery Acquisition and Mapping Analytics Development*; Boise National Forest: Boise, ID, USA, 2019.

Hamilton, D.; Bowerman, M.; Collwel, J.; Donahoe, G.; Myers, B. A Spectroscopic Analysis for Mapping Wildland Fire Effects from Remotely Sensed Imagery. *J. Unmanned Veh. Syst.* **2017**, doi:10.1139/juvs-2016-0019.

Hamilton, D.; Hamilton, N.; Myers, B. Evaluation of Image Spatial Resolution for Machine Learning Mapping of Wildland Fire Effects. In Proceedings of SAI Intelligent Systems Conference, London, UK, 5–6 September 2019; pp. 400–415.

Hamilton, D.; Hann, W. Mapping landscape fire frequency for fire regime condition class. In Proceedings of the Large Fire Conference, Missoula, MT, USA, 19–23 May 2014.

Hamilton, D. *Improving Mapping Accuracy of Wildland Fire Effects from Hyperspatial Imagery Using Machine Learning*; The University of Idaho: Moscow, ID, USA, 2018.

Hamilton, D., Levandovsky, E., & Hamilton, N. (2020). Mapping Burn Extent of Large Wildland Fires from Satellite Imagery Using Machine Learning Trained from Localized Hyperspatial Imagery. Remote Sensing, 12(24), 4097. https://doi.org/10.3390/rs12244097

Han, J.; Kamber, M.; Pei, J. *Data Mining: Concepts and Techniques*, 3rd ed.; Morgan Kaufmann: Boston, MA, USA; 2012.

Hoover, K.; Hanson, L.A. Wildfire Statistics. Congressional Research Service, Oct. 2019. Available online: https://crsreports.congress.gov/product/pdf/IF/IF10244 (accessed on 14 May 2020).

Insurance Information Institute. Facts + Statistics: Wildfires | III. 18 May 2020. Available online: https://www.iii.org/fact-statistic/facts-statistics-wildfires (accessed 18 May 2020).

Keeley, J.E. Fire intensity, fire severity and burn severity: A brief review and suggested usage. *Int. J. Wildland Fire* **2009**, *18*, 116–126.

Key, C.H.; Benson, N.C. Landscape Assessment (LA). In *FIREMON: Fire effects monitoring and inventory system. Gen. Tech. Rep. RMRS-GTR-164-CD*; U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station: Fort Collins, CO, USA, 2016; Volume 164, p. 155.

Kolden, C.A.; Weisberg, P.J. Assessing Accuracy of Manually-mapped Wildfire Perimeters in Topographically Dissected Areas. *Fire Ecol.* **2007**, *3*, 22–31, doi:10.4996/fireecology.0301022.

Lebourgeois, V.; Bégué, A.; Labbé, S.; Mallavan, B.; Prévot, L.; Roux, B. Can commercial digital cameras be used as multispectral sensors? A crop monitoring test. *Sensors* **2008**, *8*, 7300–7322.

Laliberte, A.S.; Herrick, J.E.; Rango, A.; Winters, C. Acquisition, orthorectification, and object-based classification of unmanned aerial vehicle (UAV) imagery for rangeland monitoring. *Photogramm. Eng. Remote Sens.* **2010**, *76*, 661–672.

Lentile, L.B.; Holden, Z.A.; Smith, A.M.; Falkowski, M.J.; Hudak, A.T.; Morgan, P.; Benson, N.C. Remote sensing techniques to assess active fire characteristics and post-fire effects. *Int. J. Wildland Fire* **2006**, *15*, doi:10.1071/WF05097.

Lewis, S.; Robichaud, P.; Frazier, B.; Wu, J.; Laes, D. Using hyperspectral imagery to predict post-wildfire soil water repellency. *Geomorphology* **2008**, *95*, 192–205, doi:10.1016/j.geomorph.2007.06.002.

Morgan, P.; Hardy, C.; Swetnam, T.; Rollins, M.; Long, D. Mapping fire regimes across time and space: Understanding coarse and fine-scale fire patterns. *Int. J. Wildland Fire* **2001**, *10*, 329–342, doi:10.1071/WF01032.

National Aeronautics and Space Administration (NASA). Landsat 8 Bands. 21 July 2020. Available online: https://landsat.gsfc.nasa.gov/landsat-8/landsat-8-bands/ (accessed on 22 July 2020).

National Interagency Fire Center (NIFC). Suppression Costs. 2020. Available online: https://www.nifc.gov/fireInfo/fireInfo_documents/SuppCosts.pdf (accessed on 18 May 2020).

National Interagency Fire Center (NIFC). Wildland Fire Fatalities by Year. National Interagency Fire Center, 2020. Available online: https://www.nifc.gov/safety/safety_documents/Fatalities-by-Year.pdf (accessed on 18 May 2020).

National Wildfire Coordinating Group (NWCG). Size Class of Fire. 2020. Available online: www.nwcg.gov/term/glossary/size-class-of-fire (accessed on 20 May 2020).

Rango, A.; Laliberte, A.; Herrick, J.E.; Winters, C.; Havstad, K.; Steele, C.; Browning, D. Unmanned aerial vehicle-based remote sensing for rangeland assessment, monitoring, and management. *J. Appl. Remote Sens.* **2009**, *3*, 033542, doi:10.1117/1.3216822.

Russell, S.; Norvig, P. *Artificial Intelligence: A Modern Approach*, 3rd ed.; Prentice Hall: Upper Saddle River, NJ, USA, 2010.

Sparks, A.M.; Boschetti, L.; Smith, A.M.; Tinkham, W.T.; Lannom, K.O.; Newingham, B.A. An accuracy assessment of the MTBS burned area product for shrub–steppe fires in the northern Great Basin, United States. *Int. J. Wildland Fire* **2015**, *24*, 70–78.

USDA Forest Service Geospatial Technolgy and Applications Center (GTAC). Monitoring Trends in Burn Severity. *Monit. Trends Burn Sev.* **2020**. Available online: https://www.mtbs.gov/ (accessed on 20 May 2020).

Wildland Fire Leadership Council. CSPhaseIIINationalStrategyApr2014.pdf. Available online:*h*ttps://www.forestsandrangelands.gov/documents/strategy/strategy/CSPhaseIIINationalStrategyApr2014.pdf (accessed on30 July 2020).

Zhou, G.; Li, C.; Cheng, P. Unmanned aerial vehicle (UAV) real-time video registration for forest fire monitoring. In Proceedings of the 2005 IEEE International Geoscience and Remote Sensing Symposium, Seoul, South Korea, 29 July 2005; doi:10.1109/IGARSS.2005.1526355.

5. Code

```
import os
import sys
import re
import arcpy
import shutil
from os import path
from arcpy import env
from arcpy import management
from arcpy.sa import *
import csv
import random
```

```
```

```
```

```
```

```
```

```
############################INPUT################################
outFolder = "C:\Users\Enochlev\Documents\Output"
exePath = "C:\Users\Enochlev\Documents\SourceFolder"
fireName = "Immigrant"
hypImg = "C:\Users\Enochlev\Documents\ArcGIS 10.7\IR Immigrant Test\immigrant_clip_svm_cons.tif" #Future work will make this hard coded to  "\"" + exePath + "\"HyperTrainImg.tif\"" with image normalizer for the landsat
SataliteScene = "C:\Users\Enochlev\Documents\ArcGIS 10.7\IR Immigrant Test\Dont Test\\4.tif"
#hypClass = "C:\Work\Enochs_Inputs\\5_Output.tif" Should be created within this experiement, unless found inside exePath
bandslocation="C:\Users\Enochlev\Documents\ArcGIS 10.7\IR Immigrant test"
SkipMainPictureTraining = True#if true then output_5.tif will be found in exepath folder... will save 4-5 minutes of proccessing time
```

```
```

```
############################INPUT################################
```

```
```

```
```

```
#Written By dale from spatialRes.py
def transferSpatialGeoRefDir(source_tif, target_folder):
    #   Transfer spatial reference information and georeference from a a source raster to a set of
    #   other rasters that have the same spatial reference values, but for which that
    #   information is  not defined
```

```
37.    #    Use when spatial resolution is the same
38.
39.    print ("import Spatial Ref & Georeference")
40.
41.    # create world file from source raster
42.    management.ExportRasterWorldFile(source_tif)
43.    src_world_file = re.sub('\.tif$', '.tfw', source_tif)
44.
45.    # get source raster projection
46.    src_tiff_srs = arcpy.Describe(source_tif).spatialReference
47.
48.    for tiff in os.listdir(target_folder):
49.       if re.match('.+\.tif$', tiff):
50.          # rename and copy world file to target tiffs
51.          world_file_name = re.sub('\.tif$', '.tfw', tiff)
52.          new_world_file = path.join(target_folder, world_file_name)
53.          shutil.copy(src_world_file, new_world_file)
54.
55.
56.          # define projection for target tiffs
57.          tiff_path = path.join(target_folder, tiff)
58.          management.DefineProjection(tiff_path, src_tiff_srs)
59.          print("spatial reference and georeference imported")
60.
61.
62. #Written By dale from spatialRes.py
63. #transfer Spatial Reference from and World Files
64. def transferSpatialReferenceWorldFileDir(source_tif, target_folder, src_world_file):
65.    #    Transfer spatial reference information from a a source raster to a set of
66.    #    other rasters that have the same spatial reference values, but for which that
67.    #    information is  not defined.  Load the georeference from a world file.
68.    #    Use when spatial resolution has changed
69.
70.
71.    # create world file from source raster
72.    #management.ExportRasterWorldFile(source_tif)
73.    #src_world_file = re.sub('\.tif$', '.tfw', source_tif)
74.
75.    # get source raster projection
76.    src_tiff_srs = arcpy.Describe(source_tif).spatialReference
77.
78.    for tiff in os.listdir(target_folder):
79.       if re.match('.+\.tif$', tiff):
80.          # rename and copy world file to target tiffs
```

```
            world_file_name = re.sub('\.tif$', '.tfw', tiff)
            new_world_file = path.join(target_folder, world_file_name)
            if src_world_file != new_world_file:
                shutil.copy(src_world_file, new_world_file)

            # define projection for target tiffs
            tiff_path = path.join(target_folder, tiff)
            management.DefineProjection(tiff_path, src_tiff_srs)
            print ("    spatial reference imported")



#Written By Enoch Levandovsky Started on 07/07/2019 Finished on 7/22/2019

def AdjustCSVRatio(valPxLst30m):
    largestX = 0
    LargestY = 0
    SmallestX = 100
            SmallestY = 100
            trues = 0
            false = 0
            with open(valPxLst30m) as csv_file:
                csv_reader = csv.reader(csv_file, delimiter=',')
                line_count = 0
                data = list(csv.reader(open(valPxLst30m)))
                for row in csv_reader:
                    if row[0] == 'Unburned':
                        false+=1
                    elif row[0] == 'WhiteAsh' or row[0] == 'BlackAsh':
                        trues += 1
                    if line_count == 0:
                        line_count = line_count#do nothing
                    elif int(row[2]) < SmallestY:
                        SmallestY = int(row[2])
                    elif int(row[1]) < SmallestX:
                        SmallestX = int(row[1])
                    elif int(row[1]) > largestX:
                        largestX = int(row[2])
                    elif int(row[2]) > LargestY:
                        LargestY = int(row[1])
```

```
125.                 line_count += 1
126.
127.             if trues < 5:
128.                 print ("Data may be too small to determine anything")
129.                 return
130.             else:
131.                 targetfalse = int(trues * 1.05)
132.                 ratio = targetfalse*1.0 / false*1.0
133.                 i = 0
134.                 renamedfile = re.sub('\.csv$', 'New.csv', valPxLst30m)
135.                 with open(renamedfile,'wb')as file:
136.                     writer = csv.writer(file)
137.                     while i < line_count:
138.
139.                         if 'Unburned' == data[i][0] and targetfalse <= false and (data[i][1] ==
    largestX or data[i][1] == SmallestX or data[i][2] == LargestY or data[i][2] == SmallestY )
    and 'Label' != data[i][0]:
140.                             false -= 1
141.                         else:
142.                             writer.writerow(data[i])  # write all non-matching rows
143.                         i +=1
144.
145.         print("Running Spacial Fire Detection and Burn Severity Experiment:\n")
146.
147.
148.
149.
150.         #////////////////////////////////////////////////////////////
151.         #this is input check and folder creation
152.         print('outFolder: '+outFolder)
153.         print('hyperspatial ortho: '+hypImg)
154.         print('Satalite Scene: '+SataliteScene)
155.         print('fireName: '+fireName)
156.         print('exePath: '+exePath)
157.
158.         elevRasterPre = Raster(SataliteScene)
159.         toRes = elevRasterPre.meanCellWidth
160.         print('TargetResolution: '+ str(toRes))
161.
162.         arcpy.CheckOutExtension("Spatial")
163.
164.         if os.path.exists(outFolder):
165.             print('Deleting Output Folder')
166.             shutil.rmtree(outFolder)
```

```
167.        if not os.path.exists(outFolder):
168.            os.mkdir(outFolder)
169.
170.        if not os.path.exists(SataliteScene):
171.            print('Satellite Scene not found')
172.
173.
174.        if not os.path.exists(hypImg):
175.            print('Hyperspatial orthomosaic not found')
176.        #////////////////////////////////////////////////////////
177.
178.
179.        #////////////////////////////////////////////////////////
180.        # This is the Clipper that clips the Landsat
181.        #image Combine Part 1 of 2
182.        print("\n\nClipping LandSat Scene to extent of hyperImagery and snapping
    Hyper Imagery to landsat")
183.        print("Input1: " + hypImg)
184.        print("Input2: " + SataliteScene)
185.        print("Output1: " + outFolder+ "\\clipped\\" + fireName +
    "_landSat_clipped.tif")
186.        print("Output2: " + outFolder+ "\\clipped\\" + fireName +
    "_hyper_Snapped.tif")
187.        print("\tclipping....")
188.        hyperSnapped = outFolder+ "\\clipped\\" + fireName + "_hyper_Snapped.tif"
189.        SataliteSceneClipped = outFolder+ "\\clipped\\" + fireName +
    "_landSat_clipped.tif"
190.
191.        if not os.path.exists(outFolder + "\\clipped"):
192.            os.mkdir(outFolder+ "\\clipped")
193.
194.        elevRaster = Raster(hypImg)
195.        myExtent = elevRaster.extent
196.
197.        myExtentString = str(myExtent.XMin - 80) + " " + str(myExtent.YMin - 80) + " " +
    str(myExtent.XMax + 80) + " " + \
198.                         str(myExtent.YMax + 80)
199.
200.        arcpy.Clip_management(in_raster=SataliteScene, rectangle=myExtentString,
201.                          out_raster=(outFolder+ "\\clipped\\" + fireName +
    "_landSat_clipped.tif"), in_template_dataset="",
202.                          nodata_value="-9999", clipping_geometry="NONE",
    maintain_clipping_extent="NO_MAINTAIN_EXTENT")
203.        print("Complete")
```

```
204.        #////////////////////////////////////////////////////////////
205.
206.
207.        print("\nFound bands #'s")
208.        if not os.path.exists(outFolder + "\\bandsclipped"):
209.          os.mkdir(outFolder+ "\\bandsclipped")
210.        counterX = 1
211.
212.        numberofbands = 0
213.        noBands = True
214.        while numberofbands <= 4:#9 is max amount of bands
215.          locationString = bandslocation +'\\'+ str(counterX) +".tif"
216.          if os.path.isfile(locationString):
217.            noBands = False
218.            print(counterX)
219.
220.            arcpy.Clip_management(in_raster=bandslocation+ '\\' + str(counterX) +
    ".tif", rectangle=myExtentString,
221.                        out_raster=(outFolder + "\\bandsclipped\\" + fireName +
    "_Band_"+ str(counterX) + "_landSat_clipped.tif"),
222.                        in_template_dataset="",
223.                        nodata_value="-9999", clipping_geometry="NONE",
224.                        maintain_clipping_extent="NO_MAINTAIN_EXTENT")
225.            numberofbands = numberofbands + 1
226.
227.          counterX = counterX + 1
228.          if counterX == 9:
229.            break
230.
231.        if numberofbands == 5:
232.          print ("Maximum Number of Bands reached")
233.        if noBands == True:
234.          print("Could not find any bands... If this is a mistake... name the bands
    1.tif,4.tif,.... and so on")
235.        else:
236.          print("Finished Finding Bands")
237.        #////////////////////////////////////////////////////////////
238.
239.
240.
241.        # Snaps the Hyper imagery to be the same size as the clipped landsat image
242.        #image Combine Part 2 of 2
243.        snapcmd = exePath + "\\Snapper\\x64\\Release\\Snapper.exe \"" + hypImg +
    "\" \"" + (outFolder + "\\clipped\\" + \
```

```
244.                fireName + "_landSat_clipped.tif") + "\" \""+ (outFolder + "\\clipped\\" +
    fireName + "_hyper\"")#+ "_Snapped.tif\"")
245.
246.        print("\tsnapping...")
247.        errorcode = os.system(snapcmd)
248.
249.        arcpy.DefineProjection_management(in_dataset=(outFolder + "\\clipped\\" +
    fireName + "_hyper_Snapped.tif"),
250.
    coor_system="PROJCS['WGS_1984_UTM_Zone_11N',GEOGCS['GCS_WGS_1984',DATUM
    ['D_WGS_1984',SPHEROID['WGS_1984',6378137.0,298.257223563]],PRIMEM['Greenwic
    h',0.0],UNIT['Degree',0.0174532925199433]],PROJECTION['Transverse_Mercator'],PARA
    METER['False_Easting',500000.0],PARAMETER['False_Northing',0.0],PARAMETER['Centr
    al_Meridian',-
    117.0],PARAMETER['Scale_Factor',0.9996],PARAMETER['Latitude_Of_Origin',0.0],UNIT['
    Meter',1.0]]")
251.
252.
253.        # if the file size is too large
254.        if errorcode == -2:
255.            print("error, Image too large, Please crop out the images until it is less then
    around 1GB (NOTE: to check windows explorer should be able to calculate dimension of
    the image, If it donest its too large")
256.            error = True
257.                # This code will Make it run if the file is too large, but the image reduction
    algorithm below will not work
258.                # arcpy.Clip_management(in_raster=hyperImagery,
    rectangle=myExtentString,
259.                #                out_raster=(targetFolder + FileName + "_Snapped.tif"),
    in_template_dataset="",
260.                #                nodata_value="-9999", clipping_geometry="NONE",
    maintain_clipping_extent="NO_MAINTAIN_EXTENT")
261.        print("Complete")
262.        #////////////////////////////////////////////////////////////
263.
264.
265.
266.        if SkipMainPictureTraining == False:#see note line:
267.        #////////////////////////////////////////////////////////////
268.            print("\nRunning SVM on inputed hyper Imagery")
269.            print("Input: " + exePath + "\TrainImg.JPG")
270.            print("Output1: " + outFolder+ "\\clipped\\" + fireName +
    "_svm_Train_con.tif")
```

```
271.            print("Output2: " + outFolder+ "\\clipped\\" + fireName +
    "_svm_Train_ext.tif")
272.
273.            TrainImg = exePath + "\TrainImg.JPG"
274.            TrainConCSV = exePath + "\TrainCon.csv"
275.            TrainExtCSV = exePath + "\TrainExt.csv"
276.
277.            svmExe = exePath +
    '\\SvmBurnClassifier\\x64\\Release\\SvmBurnClassifier.exe'
278.            svmOutCon = outFolder + '\\TrainOutput\\' + fireName + '_svm_Train_con.tif'
279.            svmOutExt = outFolder + '\\TrainOutput\\' + fireName + '_svm_Train_ext.tif'
280.            classKey = exePath + '\\classKey.csv'
281.            hyperSnapped = outFolder+ "\\clipped\\" + fireName + "_hyper_Snapped.tif"
282.
283.            if not os.path.exists(outFolder + "\\TrainOutput"):
284.               os.mkdir(outFolder+ "\\TrainOutput")
285.
286.
287.            print("Running SVM with Extent Training Data...")
288.            svmCmd = svmExe + ' \"' + hyperSnapped + '\" \"' + svmOutExt + '\" \"' +
    classKey + '\" -p -tp \"' + TrainExtCSV + "\" -ti \"" + TrainImg + "\" -chi2   "
289.            os.system(svmCmd)
290.
291.
292.            print("Running SVM with Consumtion Training Data...")
293.            svmCmd = svmExe + ' \"' + hyperSnapped + '\" \"' + svmOutCon + '\" \"' +
    classKey + '\" -p -tp \"' + TrainConCSV + "\" -ti \"" + TrainImg + "\" -chi2"
294.
295.            os.system(svmCmd)
296.            #os.system(svmCmd)  # maby multi thread
297.
298.
299.
300.
301.
302.
303.
304.
305.            transferSpatialGeoRefDir(hyperSnapped, outFolder + '\\TrainOutput')
306.            print("Complete")
307.          #////////////////////////////////////////////////////////////
308.
309.          if SkipMainPictureTraining == False:  # see note line:
310.          #////////////////////////////////////////////////////////////
```

```
311.
312.             #this is because SVM changes the name for con and ext
313.             svmOutCon = outFolder + '\\TrainOutput\\' + fireName +
    '_svm_Train_conCHI2.tif'
314.             svmOutExt = outFolder + '\\TrainOutput\\' + fireName +
    '_svm_Train_extCHI2.tif'
315.
316.
317.             print("Running Denoise on hyper Imagery")
318.             print("Input1: " + svmOutExt)
319.             print("Input2: " + svmOutCon)
320.             print("Output: " + outFolder+ "\\denoise\\")
321.
322.             hypClasImg = outFolder + "\\denoise\\5_Output.tif"
323.             denoiseExe = exePath + '\\denoise\\x64\\Release\\Denoise_v1.4.exe'
324.             denoisecmd = denoiseExe +' \"' + svmOutExt + "\" -pf 1 4000 4000 \"" +
    svmOutCon + "\" -pf"
325.
326.             if not os.path.exists(outFolder + "\\denoise"):
327.                 os.mkdir(outFolder+ "\\denoise")
328.
329.             os.system(denoisecmd)#get RID OF THE OUTPUT MESS to increase speed
330.             transferSpatialGeoRefDir(hyperSnapped, outFolder+ "\\denoise")
331.             print("Complete")
332.         #////////////////////////////////////////////////////////////////
333.
334.
335.         #////////////////////////////////////////////////////////////////
336.         if SkipMainPictureTraining == True:
337.             hypClasImg = exePath + "\\5_Output.tif"
338.         #////////////////////////////////////////////////////////////////
339.
340.
341.         #////////////////////////////////////////////////////////////////
342.         # Calculate 30m density from 5cm classification
343.         if not os.path.exists(outFolder+'\density'):
344.             os.mkdir(outFolder+'\density')
345.         print("")
346.         print("Densifying ...")
347.         print("    hypClass: " + hypClasImg)
348.         print("    densBasename: " + outFolder + '\density\dens ')
349.         print("    toRes: "+ str(toRes))
350.         densityCmd = exePath + '\\density\\x64\Debug\density.exe \"' + hypClasImg +
    '\" \"' + outFolder + '\density\dens\" ' + str(toRes) + ' 255'
```

```
351.        #densityCmd = exePath + '\\density\\x64\Debug\density.exe ' + hypClass + ' ' +
    outFolder + '\density\dens ' + toRes
352.        os.system(densityCmd)
353.        transferSpatialReferenceWorldFileDir(hypImg, outFolder+'\density', outFolder +
    '\density\dens.tfw')
354.
355.        if not os.path.exists(outFolder+'\\fuzzy'):
356.           os.mkdir(outFolder+'\\fuzzy')
357.        fuzzyFolder = outFolder+'\\fuzzy'
358.        env.workspace = fuzzyFolder
359.
360.        if not os.path.exists(outFolder+'\\val'):
361.           os.mkdir(outFolder+'\\val')
362.        valFolder = outFolder+'\\val'
363.        #////////////////////////////////////////////////////////////
364.
365.
366.
367.        #////////////////////////////////////////////////////////////
368.        # Calculate extent & combustion
369.        # BurnExtent = (WhiteAsh + BlackAsh+TreeIsland)/px
370.        densBlackAshPath = outFolder + '\\density\\dens_c2.tif'
371.        densWhiteAshPath = outFolder + '\\density\\dens_c3.tif'
372.        densNoDataPath = outFolder + '\\density\\dens_c255.tif'
373.        densBlack = Raster(densBlackAshPath)
374.        densWhite = Raster(densWhiteAshPath)
375.        densNoData = Raster(densNoDataPath)
376.        ext = densBlack + densWhite
377.
378.        # Combustion = w/(w+b) - White ash cover?
379.        combust = arcpy.sa.Con(ext < 10, 0, densWhite*100/ext)
380.
381.        ext.save(outFolder + "\\fuzzy\\ext.tif")
382.        combust.save(outFolder + "\\fuzzy\\comb.tif")
383.
384.
385.        # Fuzzy Set Membership
386.        print("Fuzzifying ...")
387.        # BurnExtent - Burned (type=linear, min=35, max=65)
388.        burn = arcpy.sa.FuzzyMembership(ext, FuzzyLinear(35,65), "NONE")
389.        # BurnExtent - unburned (type=linear, min=65, max=35)
390.        unburn = arcpy.sa.FuzzyMembership(ext, FuzzyLinear(65,35), "NONE")
```

```
391.        # Combustion - Low (type=linear, min=40, max=15) - Lewis (2010) says transition
    should be 33/50, we are going lower so white ash registers - Lucky peak should get
    higher white ash
392.        lowComb = arcpy.sa.FuzzyMembership(combust, FuzzyLinear(50,33), "NONE")
393.        #lowComb = arcpy.sa.FuzzyMembership(combust, FuzzyLinear(40,15), "NONE")
394.        # Combustion - High (type=linear, min=15, max=40)
395.        hiComb = arcpy.sa.FuzzyMembership(combust, FuzzyLinear(33,50), "NONE")
396.        #hiComb = arcpy.sa.FuzzyMembership(combust, FuzzyLinear(15,40), "NONE")
397.
398.        burn.save(outFolder + "\\fuzzy\\burn.tif")
399.        unburn.save(outFolder + "\\fuzzy\\unburn.tif")
400.        lowComb.save(outFolder + "\\fuzzy\\lowComb.tif")
401.        hiComb.save(outFolder + "\\fuzzy\\hiComb.tif")
402.
403.        # *** DAH: High combustion and low combustion are both very low,  Multiply
    by 100 and look at histogram
404.
405.        # We are not looking at canopy cover because we're working with rangeland.
    There should not be enough hyperspectral pixels to swing 30m pixel to canopy fuels
406.
407.        # Apply Fuzzy Logic and activate fuzzy rules
408.        print("Activate fuzzy logic rules ...")
409.        # If (ext:burned && com:high) -> white ash
410.        whiteAsh = arcpy.sa.FuzzyOverlay([burn, hiComb], "AND")
411.        # If (ext:burned && com:low) -> black ash
412.        blackAsh = arcpy.sa.FuzzyOverlay([burn, lowComb], "AND")
413.        # If (ext:unburned) -> unburned
414.        # Missing this fuzzyOverlay - we can just use the density of unburned. - YES
415.
416.        whiteAsh.save(outFolder + "\\fuzzy\\whiteAsh.tif")
417.        blackAsh.save(outFolder + "\\fuzzy\\blackAsh.tif")
418.
419.        # Get rule w/ highest membership
420.        dom = HighestPosition([unburn,blackAsh,whiteAsh,densNoData])
421.        dom.save(outFolder + "\\fuzzy\\dom.tif")
422.
423.        # Translate rule# to class w/ Reclass
424.        recls30m = Reclassify(dom, "Value", RemapValue([[1, 0], [2, 2], [3, 3], [4, 255]]))
425.        #recls30m.save(outFolder + "\\fuzzy\\domCls30m.tif")
426.        arcpy.CopyRaster_management(recls30m, outFolder +
    "\\fuzzy\\domCls30m.tif",pixel_type="8_BIT_UNSIGNED", format="TIFF")
427.        #////////////////////////////////////////////////////////////
428.
429.
```

```
430.
431.        #////////////////////////////////////////////////////////////
432.        # Segment 30m data into training & validation data
433.        print("\n\nCreate 30m training pixel list")
434.        coordSel30mCmd = exePath + '\\coordSel30m\\x64\Release\coordSel30m.exe
    \"' + outFolder + "\\fuzzy\\domCls30m.tif\" \"" + \
435.                    exePath + "\\classKey.csv\" \"" + outFolder + "\\" + fireName
    +"_cls30m 30\""
436.        # Inputs
437.        #   30m classification derived from 5cm classification
438.        #   Attribute table file
439.        #   train/validation base path.
440.        #   Will append  _trn.csv & _val.csv
441.        #   Withholding rate
442.        os.system(coordSel30mCmd)
443.        #////////////////////////////////////////////////////////////////
444.
445.
446.
447.
448.        #////////////////////////////////////////////////////////////
449.        # Run svm on 30m image - training data derived from 5cm classification
450.        print('SVM running at 30m...')
451.
452.        trnPxLst30m = outFolder + '\\' + fireName +'_cls30m 30_trn.csv'
453.        valPxLst30m = outFolder + '\\' + fireName +'_cls30m 30_val.csv'
454.        svmExe = exePath + '\\SvmBurnClassifier\\x64\\Release\\SvmBurnClassifier.exe'
455.        #svmExe =
    'C:\\NNU\\FireMAP\\ClassifiersOpenCV\\SvmBurnClassifier\\x64\\Release\\SvmBurnCla
    ssifier.exe'
456.        svmOut = valFolder + '\\svm_30m_out.tif'
457.        classKey = exePath + '\\classKey.csv'
458.
459.
460.        #////////////////////////////////////
461.        #MultiBand Proccessor
462.        bandsString = ""
463.
464.        #default with no extentions
465.        svmCmd = svmExe + ' \"' + SataliteSceneClipped + '\" \"' + svmOut + '\" \"' +
    classKey + '\" -p -tp \"' + valPxLst30m + "\" -ti \"" + SataliteSceneClipped + '\"'
466.        #svmCmd = svmExe + ' \"' + SataliteScene + '\" \"' + svmOut + '\" \"' + classKey +
    '\" -p -tp \"' + trnPxLst30m + "\" -ti \"" + SataliteSceneClipped + '\" -vp \"' + valPxLst30m
    + "\""
```

```
467.        #switch these two have final output on clipped or entire scene
468.
469.        counterX = 1
470.        while counterX <= 9:#9 is max amountoutFolder of bands
471.            locationString = bandslocation +'\\'+ str(counterX) +".tif"
472.            if os.path.isfile(locationString):
473.                bandsString = bandsString + " -ct \"" + outFolder + "\\bandsclipped\\" +
    fireName + "_Band_"+ str(counterX) + "_landSat_clipped.tif\" -ci \"" + outFolder +
    "\\bandsclipped\\" + fireName + "_Band_"+ str(counterX) + "_landSat_clipped.tif\""
474.                #bandsString = bandsString + " -ct \"" + outFolder + "\\bandsclipped\\" +
    fireName + "_Band_"+ str(counterX) + "_landSat_clipped.tif\" -ci \"" + bandslocation +
    "\\"+ str(counterX) + ".tif\""
475.                #switch these two have final output on clipped or entire scene
476.
477.            counterX = counterX + 1
478.
479.
480.
481.
482.
483.
484.        #/////////////////////////////////////
485.        svmCmdL = svmCmd + bandsString + " -linear"
486.        os.system(svmCmdL)
487.        svmCmdC = svmCmd + bandsString + " -chi2"
488.        os.system(svmCmdC)
489.        #svmCmdR = svmCmd + bandsString + " -rbf"
490.        #os.system(svmCmdR)
491.
492.        transferSpatialGeoRefDir(SataliteSceneClipped, outFolder+'\\val')
493.
494.
495.        AdjustCSVRatio(trnPxLst30m)
496.
497.        percentageCheckerCMD = exePath + "\\percentage_Checker.exe" + " \"" +
    valFolder + '\\svm_30m_outCHI2.tif\" \"' + re.sub('\.csv$', 'New.csv', trnPxLst30m) + "\""
498.
499.        os.system(percentageCheckerCMD)
500.        print("Done")
501.        #/////////////////////////////////////////
```