2024

Interlacing unstructured data with deep neural nets for predicting pervious and impervious land cover types

Srivani Athmakur

INTERLACING UNSTRUCTURED DATA WITH DEEP NEURAL NETS FOR PREDICTING PERVIOUS AND IMPERVIOUS LAND COVER TYPES

Srivani Athmakur

A THESIS

Submitted in partial fulfillment of the requirements for the degree of Master of Science
in
The Department of Computer Science
to
The Graduate School
of
The University of Alabama in Huntsville

May 2024

Approved by:

Dr. Chaity Banerjee Mukherjee, Research Advisor/Committee Chair
Dr. Tathagata Mukherjee, Committee Member
Dr. Letha Etzkorn, Committee Member
Dr. Letha Etzkorn, Department Chair
Dr. Rainer Steinwandt, College Dean
Dr. Jon Hakkila, Graduate Dean

Abstract

INTERLACING UNSTRUCTURED DATA WITH DEEP NEURAL NETS FOR PREDICTING PERVIOUS AND IMPERVIOUS LAND COVER TYPES

Srivani Athmakur

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

Computer Science

The University of Alabama in Huntsville

May 2024

This research delves into the intricate task of delineating land cover types in Tallahassee-Leon County, emphasizing the need for detailed granularity beyond existing classification systems. Utilizing cutting-edge GIS [5] data, the study harnesses the power of deep learning algorithms, including U-net [13], UNetPlusPlus [17], FPN-net [7], and DeepLabV3Plus [3]. A unique approach, "Interlacing Unstructured Data with Deep Neural Nets," integrates shapefiles and Tiff images to enhance classification metrics such as mean intersection over union, pixel accuracy, and loss functions. The research aspires to significantly improve the precision of land cover classification, holding implications for urban planning and environmental management. By innovatively integrating unstructured data, the study aims to offer valuable insights and tools for informed decision-making, contributing to urban development and environmental sustainability in Tallahassee-Leon County. The expected outcomes of this research carry profound implications for advancing our understanding of urban landscapes and fostering sustainable development practices.

Acknowledgements

I would like to express my deepest gratitude to my advisor, Dr. Chaity Banerjee Mukherjee, for her steadfast support, guidance, and valuable insights throughout the entire thesis completion process. Her expertise, encouragement, and constructive feedback have played a crucial role in shaping the trajectory of my research.

I am also profoundly thankful to my committee members, Dr. Tathagata Mukherjee, and Dr. Letha Etzkorn, in addition to my advisor, Dr. Chaity Banerjee Mukherjee. Their dedicated time, expertise, and thoughtful input during the review and evaluation of my work have been invaluable. Their constructive criticism and insightful suggestions have significantly contributed to refining this thesis.

Lastly, I want to express my gratitude to numerous individuals, including my family and friends, who have offered their unwavering encouragement, assistance, and understanding throughout this academic journey. Their support has been pivotal in overcoming challenges and reaching the successful completion of this thesis.

I genuinely appreciate the support and contributions of everyone mentioned above, as well as those who may have inadvertently been omitted. Their involvement has been instrumental in shaping this thesis and has greatly contributed to my academic growth.

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

In the era of burgeoning technologies, the potential for transformative advancements through deep learning is undeniable. However, as we embark on the challenge of land cover classification in Tallahassee-Leon County, a complex dilemma surfaces — the inherent imbalance within the datasets. How do we effectively navigate the intricate landscape, grappling not only with the delicate balance between impervious and pervious surfaces but also with the enigma presented by unlabeled or undefined areas?

Conventional approaches to land cover classification may falter when faced with the nuanced distinctions between impervious, pervious, and undefined surfaces. The challenge lies not only in leveraging growing technologies but also in formulating a nuanced strategy to address the intricate dance of unbalanced data. How can we ensure that our models accurately discern between urban structures and natural landscapes, especially when confronted with areas that defy easy categorization or lack definitive labels?

This chapter transcends a mere exploration of cutting-edge technologies such as deep learning [6] enhancements and neural networks. It delves into the intricacies of imbalanced data landscapes, where impervious, pervious, and undefined elements coalesce. My aim is twofold: to refine the precision of land cover classification and to disentangle the complexities introduced by regions that defy clear delineation. As we advance, our objective crystallizes: to deepen our understanding of the local terrain, thereby contributing to the blueprint of sustainable urban development and resource management.

Chapter 2. Contextualizing the Landscape

2.1 Navigating Unbalanced Land Cover Data through Advanced Geospatial Techniques

To navigate the intricate landscape of Tallahassee-Leon County's land cover, we turn to an arsenal of cutting-edge technologies and a wealth of geospatial data. The Geographic Information System (GIS) [5] datasets at our disposal, including Shapefiles and Tiff images, serve as the cornerstone of our exploration. These datasets are not just collections of pixels and vectors; they are a digital representation of the county's diverse topography and land cover features.

The richness of our data is derived from a regularly updated LiDAR and digital orthophotography product, stemming from source imagery captured by a Trimble TAC80 multispectral scanner and LAS data acquired by a Leica ALS50 Phase 2+ LiDAR sensor during a window from January 29, 2012, to February 12, 2012. This temporal snapshot ensures that our dataset encapsulates the dynamic nature of the landscape during that period.

The purpose behind compiling such a comprehensive dataset is clear: Tallahassee-Leon County's Geographic Information System (TLCGIS) consistently utilizes digital orthophotos and planimetric/hydrographic/topographic data. This application extends across a spectrum of functions, from supporting regulatory processes to aiding in land management and acquisition, as well as facilitating planning, engineering, and habitat restoration projects.

As we embark on the journey of land cover classification, the robustness of our dataset becomes a catalyst for our methodologies. How can we leverage the intricate details captured by LiDAR and multispectral scanners to refine our classification models? How do we harness the power of Tiff images to unravel the complexities of impervious, pervious, and undefined areas? In the chapters that follow, we delve into the intricacies of our geospatial data, aiming not only to classify land cover accurately but also to glean invaluable insights for sustainable urban development and resource management.

2.2 Understanding the Geospatial Framework

Our exploration into Tallahassee-Leon County's land cover intricacies is anchored in a robust geospatial framework. At the heart of this framework lies a meticulously designed grid system, a structured amalgamation of 200 rows and 540 columns. Each cell, measuring 5000 by 5000 feet, serves as the elemental building block, providing a standardized approach for organizing data collected through orthophotography, LiDAR, and various mapping projects.

Stretching across the panhandle of Florida, this expansive grid encapsulates the northern peninsula and extends its reach southward to include Levy and Marion Counties. Its influence doesn't stop at the shoreline, as it boldly extends into the Gulf of Mexico and the Atlantic Ocean, facilitating the mapping of underwater topography through bathymetric analysis.

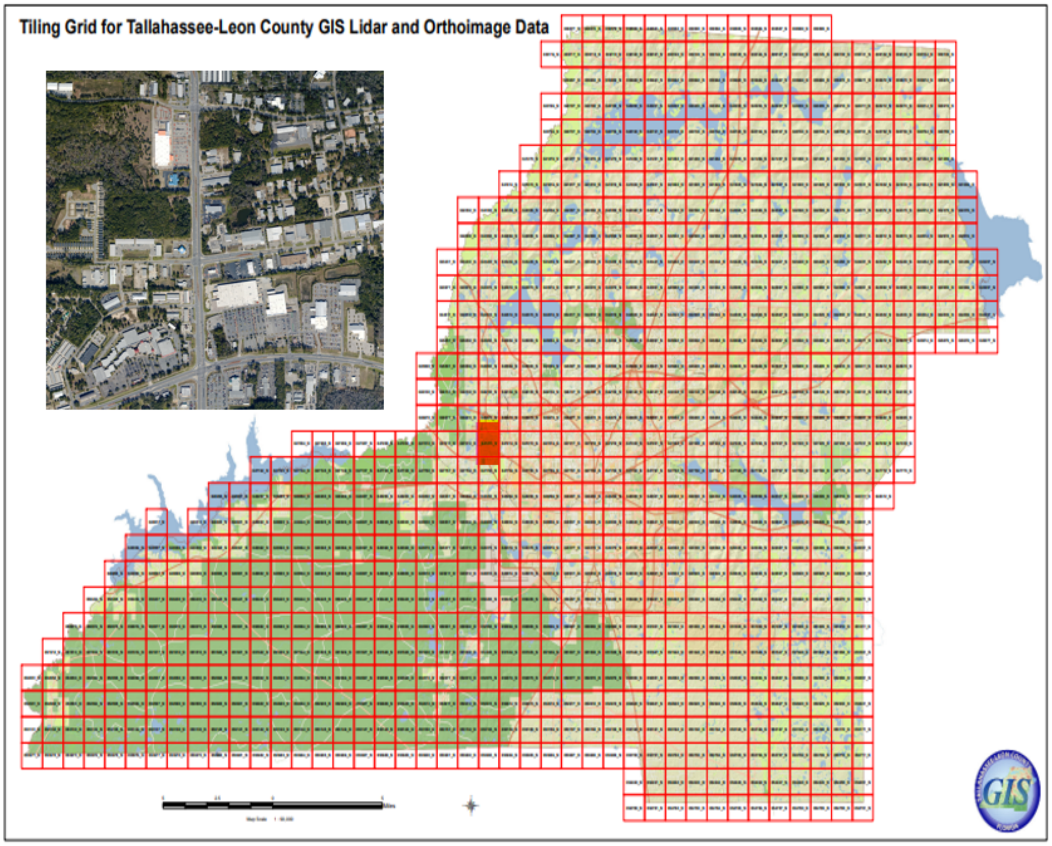


Figure 2.1: GIS and Orthoimage Data of Tallahassee [2012-2013].

Aligned with precision, the grid cells follow the even multiples of 5000-foot State Plane North Northings and Eastings, ensuring ease of use in geospatial analyses. The uniqueness of each cell is encapsulated in an identifier, a sequential numbering system that initiates at 1 in the northwest corner and gracefully concludes at 108,000 in the southeast corner. The identifiers progress incrementally from west to east within each row, with the westernmost cell marking the successor to the easternmost cell of the preceding row.

This grid is not just a spatial arrangement; it's a critical tool that transcends geographic boundaries. Its intentional extension beyond the region typically defined by the State Plane North projection enables its utilization by regional agencies, such as water management districts. Furthermore, it acts as a common indexing scheme for projects within Florida, fostering improved coordination among state, regional, and local governments.

From the entire grid, I have selected the data of Tallahassee-Leon County (Florida).

Figure 2.2: GIS and Orthoimage Data of Tallahassee-Leon County [2024].

2.3 Geographic Information System (GIS) Data and Coordinate Reference System

The accuracy and precision of spatial analysis heavily rely on the appropriate representation of geographic data. In our study, we employed Geographic Information System (GIS) [4] datasets, including Shapefiles and TIFF images, to capture the land cover types in Tallahassee-Leon County. The foundation of our spatial data analysis lies in a defined Coordinate Reference System (CRS) that ensures consistent and accurate spatial relationships.

2.3.1 Coordinate Reference System (CRS)

The spatial data in our study adhere to the NAD 1983 HARN StatePlane Florida North FIPS 0903 Feet coordinate reference system. This system utilizes the Lambert Conformal Conic projection with parameters specifying the datum, spheroid, central meridian, standard parallels, latitude of origin, and units of measurement. The key components of the CRS [15] are as follows:

- **Datum:** North American Datum 1983 with High Accuracy Reference Network (HARN) adjustment
- **Spheroid:** Geodetic Reference System 1980 (GRS 1980) with a semi-major axis of 6378137.0 and inverse flattening of 298.257222101
- **Projection:** Lambert Conformal Conic
- Central Meridian: -84.5 degrees
- Standard Parallels: 30.75 and 29.5833333333333 degrees
- Latitude of Origin: 29.0 degrees
- False Easting: 1968500.0 feet
- False Northing: 0.0 feet
- Unit: US survey foot

2.3.2 CRS Explanation

CRS provides accurate and consistent spatial representation for our land cover classification and analysis in the Tallahassee-Leon County region. Understanding and adhering to the defined CRS [15] is crucial for maintaining the integrity of spatial relationships and ensuring the reliability of my findings.

2.4 Comprehensive Metadata Overview: Spatial Attributes, Subtypes, and Feature Characteristics

In the given metadata, the dataset includes information about geographical features with attributes such as "Shape Area" and "Shape Length," indicating the area and length of each shape, respectively. The dataset is categorized into various subtypes, each representing a different land use or feature type. These subtypes, such as "OPEN LAND," "BUILDING," "UNFINISHED-BUILDING," "RUIN," "SIDEWALK," and others, are associated with specific codes and additional attributes defining impervious characteristics, including type, DXF layer [16] name, and surface type. Additionally, the dataset contains an attribute labeled "ORIG FID" with integer values. Notably, there are subtypes like "WATERBODY," "PAVED ROAD OVER BRIDGE," each with its own set of associated attributes. This metadata provides a comprehensive overview of the spatial and categorical characteristics of the dataset, facilitating the interpretation and analysis of geographic features in a GIS [4] [5] context.

Information of objectID with the label information which connects with x,y, width, and Height values of ObjectID

2.5 A Symphony of Shapefiles and Tiff Images

In our dataset, a comprehensive geospatial ensemble takes the form of shapefiles, each comprising a set of files (.shp, .shx, .dbf, .prj, and .cpg). The first four files collectively encapsulate both spatial and attribute data of various geographic features, while the last one, .cpg, ensures the accurate encoding of attribute data characteristics.

Shapefiles, the backbone of our vector data, elegantly store the trifecta of location, shape, and attributes of geographic features. Their versatile geometry can seamlessly represent points, lines, or polygons, accommodating the diverse spectrum of spatial features.

Accompanying these shapefiles are Tiff images (Tagged Image File Format), serving as the custodians of raster data in GIS applications. These images boast a resolution of 10000x10000, and each comprises one or more bands representing distinct spectral channels. This feature makes Tiff images ideal for storing and conveying geospatial imagery, including satellite imagery, aerial photographs, and other raster data.

Data preprocessing, I began by meticulously selecting distinct featured images from the Tiff image dataset. The dimensions of each image were revealed to be an impressive 10000x10000 resolution. Leveraging the valuable information stored in the .dbf file, which neatly organizes county data through OBJECTID, we proceeded to crop the data of a specific image. This extracted data was then judiciously labeled based on the SURFACE attribute, distinguishing between pervious and impervious surfaces. The intricate layers of geospatial data reveal the symbiotic relationship between shapefiles and Tiff images, and demonstrate the meticulous preprocessing steps taken to derive meaningful insights from this rich dataset."

Utilizing the insights derived from the shapefile data, I precisely extracted the region of interest aligned with the Tiff image. Employing this curated subset, I seamlessly overlaid the map image onto Folium, a dynamic mapping library, facilitating a meticulous cross-verification process. The subsequent representation encapsulates the Tiff image's detailed information, meticulously annotated with associated labels. This integrative approach not only enhances the accuracy of our geospatial analyses but also provides a visually comprehensive understanding of the labeled features within the context of the broader geographic landscape.

OBJECTID	TYPE	X	Y	Width	Height
54608	50	2046409	529951	85	51
54616	50	2046721	529935	81	71
54672	50	2047298	530002	33	29
54702	50	2047164	529978	46	63
54714	50	2047598	529919	102	128
54718	50	2046153	529982	62	66
54733	50	2047459	529993	42	60
54751	50	2046816	529994	64	66
54773	50	2046364	530041	30	26
54789	50	2046634	530003	61	70
54797	50	2046738	530053	16	23
54800	50	2046904	530021	56	56
54825	50	2046991	530021	52	69
54826	50	2046453	530020	47	69
54892	50	2046247	530058	51	51
54911	50	2047290	530064	49	50
54915	50	2047163	530055	53	61
54952	50	2046950	530113	13	14
54953	50	2047457	530069	33	59
55050	50	2045105	530105	52	56

Figure 2.3: Information of x,y, width, and Height along with object ID [2024].

OBJECTID	TYPE	DXF_LAYER	SURFACE	MAP_DATE	Shape_Leng	Shape_Area	label
54608	50	BUILDING	IMPERVIOUS	02/01/2013	256.768882577999000	2648.208281260000000	0
54616	50	BUILDING	IMPERVIOUS	02/01/2013	240.290998784999000	3077.389118680000000	0
54672	50	BUILDING	IMPERVIOUS	02/01/2013	123.108545453999000	943.293323757000000	0
54702	50	BUILDING	IMPERVIOUS	02/01/2013	219.839497267000000	2496.122029359990000	0
54714	50	BUILDING	IMPERVIOUS	02/01/2013	561.000959594000000	7357.377260739990000	0
54718	50	BUILDING	IMPERVIOUS	02/01/2013	212.429259120000000	2451.302449669990000	0
54733	50	BUILDING	IMPERVIOUS	02/01/2013	205.112987677999000	2433.950397139990000	0
54751	50	BUILDING	IMPERVIOUS	02/01/2013	238.244677488999000	2259.374499789990000	0
54773	50	BUILDING	IMPERVIOUS	02/01/2013	91.680005231400000	519.801483255999000	0
54789	50	BUILDING	IMPERVIOUS	02/01/2013	227.726016140000000	2521.408276059990000	0
54797	50	BUILDING	IMPERVIOUS	02/01/2013	65.529076514699900	243.081926680999000	0
54800	50	BUILDING	IMPERVIOUS	02/01/2013	223.311054030999000	2720.982920769990000	0
54825	50	BUILDING	IMPERVIOUS	02/01/2013	231.121096437000000	2796.880020650000000	0
54826	50	BUILDING	IMPERVIOUS	02/01/2013	203.677356150000000	2098.491669720000000	0
54892	50	BUILDING	IMPERVIOUS	02/01/2013	201.180052730999000	2529.576159220000000	0
54911	50	BUILDING	IMPERVIOUS	02/01/2013	187.571402674000000	1850.604753410000000	0
54915	50	BUILDING	IMPERVIOUS	02/01/2013	227.786926673000000	2819.524563510000000	0
54952	50	BUILDING	IMPERVIOUS	02/01/2013	51.905086758400000	167.988183767999000	0
54953	50	BUILDING	IMPERVIOUS	02/01/2013	182.438965615999000	1680.281740209990000	0
55050	50	BUILDING	IMPERVIOUS	02/01/2013	214.674669826000000	2141.239420609990000	0

cropped_shapefile

Filter Mode Accessibility: Unavailable Average: 50 Count: 705 Sum: 35200

Figure 2.4: Information of objectID with the label info [2024].

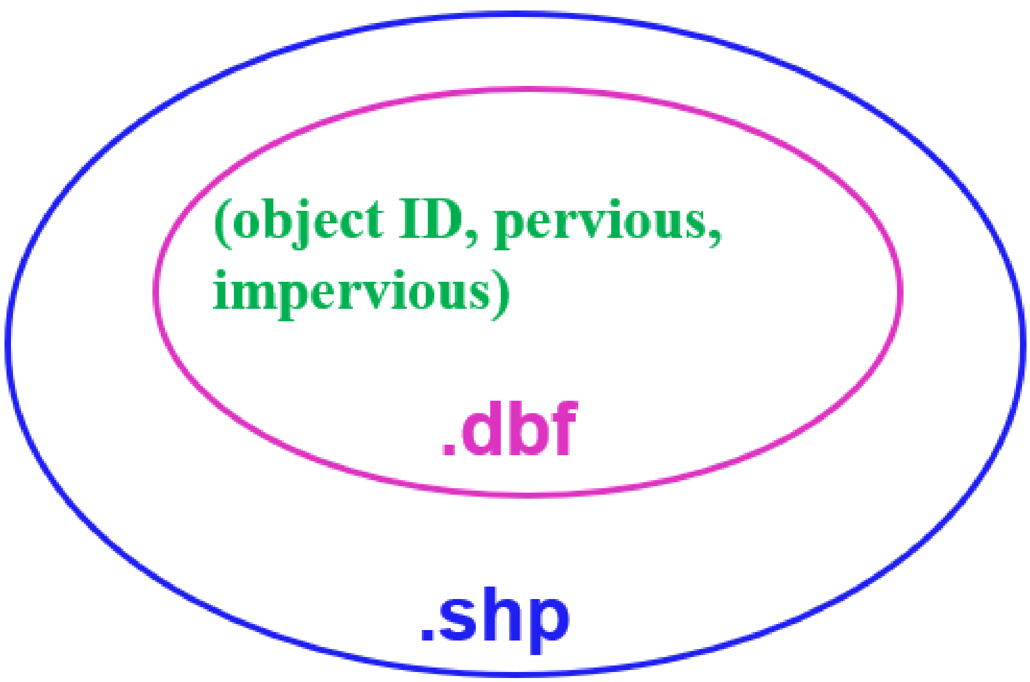
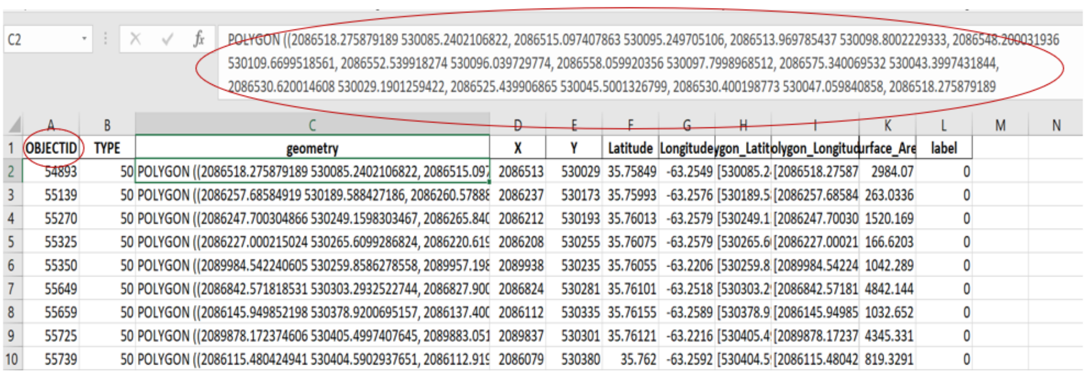


Figure 2.5: Geometry of polygon values of the Image [2024].



C2 POLYGON ((2086518.275879189 530085.2402106822, 2086515.097407863 530095.249705106, 2086513.969785437 530098.8002229333, 2086548.200031936 530109.6699518561, 2086552.539918274 530096.039729774, 2086558.059920356 530097.7998968512, 2086575.340069532 530043.3997431844, 2086530.620014608 530029.1901259422, 2086525.439906865 530045.5001326799, 2086530.400198773 530047.059840858, 2086518.275879189

	OBJECTID	TYPE	geometry	X	Y	Latitude	Longitude	ygon_Latit	olygon_Longitud	urface_Are	label
2	54893	50	POLYGON ((2086518.275879189 530085.2402106822, 2086515.097	2086513	530029	35.75849	-63.2549	[530085.2	[2086518.27587	2984.07	0
3	55139	50	POLYGON ((2086257.68584919 530189.588427186, 2086260.5788	2086237	530173	35.75993	-63.2576	[530189.5	[2086257.68584	263.0336	0
4	55270	50	POLYGON ((2086247.700304866 530249.1598303467, 2086265.84	2086212	530193	35.76013	-63.2579	[530249.1	[2086247.70030	1520.169	0
5	55325	50	POLYGON ((2086227.000215024 530265.6099286824, 2086220.61	2086208	530255	35.76075	-63.2579	[530265.6	[2086227.00021	166.6203	0
6	55350	50	POLYGON ((2089984.542240605 530259.8586278558, 2089957.19	2089938	530235	35.76055	-63.2206	[530259.8	[2089984.54224	1042.289	0
7	55649	50	POLYGON ((2086842.571818531 530303.2932522744, 2086827.90	2086824	530281	35.76101	-63.2518	[530303.2	[2086842.57181	4842.144	0
8	55659	50	POLYGON ((2086145.949852198 530378.9200695157, 2086137.40	2086112	530335	35.76155	-63.2589	[530378.9	[2086145.94985	1032.652	0
9	55725	50	POLYGON ((2089878.172374606 530405.4997407645, 2089883.051	2089837	530301	35.76121	-63.2216	[530405.4	[2089878.17237	4345.331	0
10	55739	50	POLYGON ((2086115.480424941 530404.5902937651, 2086112.91	2086079	530380	35.762	-63.2592	[530404.5	[2086115.48042	819.3291	0

Figure 2.6: Geometry of polygon values of the Image [2024].

Chapter 3. Geospatial Data Integration and Preprocessing

3.1 From Pixel Unveil to Interactive Maps

In the quest for comprehensive geospatial insights, I embarked on a multifaceted journey, integrating the OBJECTID, TYPE, Shape Area, and Shape Length from the shapefile to visualize the polygon shapes of each unique OBJECTID. Armed with a Tiff image boasting a resolution of 10000x10000 pixels and the accompanying shapefile, I delved into the meticulous task of pixel-by-pixel examination and coordination.

With precision, I meticulously extracted pixel values and x, and y points, creating a mosaic of 125 Excel sheets, each containing 8,00,000 cells, meticulously covering the vast expanse of the Tiff image. The pixel coordinates danced across the sheets, illuminating the intricate details of the geospatial landscape.

Harnessing the power of the **folium library**, I transformed the abstract data into an interactive symphony of maps, visually rendering each block of the Tiff image. This interactive masterpiece allowed for a nuanced comparison and verification of the accuracy of the county's data.

The geographical coordinates of the image corners became the key to unlocking the spatial puzzle. Calculating Latitude and Longitude through a meticulous formula involving minimum latitude, maximum latitude, minimum longitude, maximum longitude, and the dimensions of the image, I bridged the gap between pixel coordinates and real-world geography.

Venturing into the realm of image manipulation, I implemented a masking technique, using shapefile data to create masks on the Tiff image. With each polygon and its four corners intricately connecting in the dance of element-wise multiplication, the masks unfolded the true essence of the geospatial landscape.

Breaking down the Tiff input image into smaller sub-images, I harnessed Python's prowess to save them individually, each bearing a unique geographic fingerprint. The marriage of geographic coordinates and label information birthed a matrix representation of the image, a visual symphony of data and imagery.

In the pursuit of accuracy amid the greenery and shadows, I honed the focus on relevant image portions, subjecting them to meticulous scrutiny. The intricate dance of data and images unfolded, as each pixel found its place in the grand mosaic of geospatial exploration.

3.1.1 Google Earth Pro's Image Loading Odyssey

Embarking on the comprehensive exploration of geospatial data through Google Earth Pro unravels a panorama enriched with high-resolution satellite imagery, intricate 3D terrains, and interactive maps. However, the task of loading considerable TIFF images within this sophisticated platform poses intricate challenges, necessitating the allocation of substantial system resources. The intricate process of loading high-resolution TIFF images entails the judicious utilization of memory resources by Google Earth Pro, a critical aspect of rendering detailed

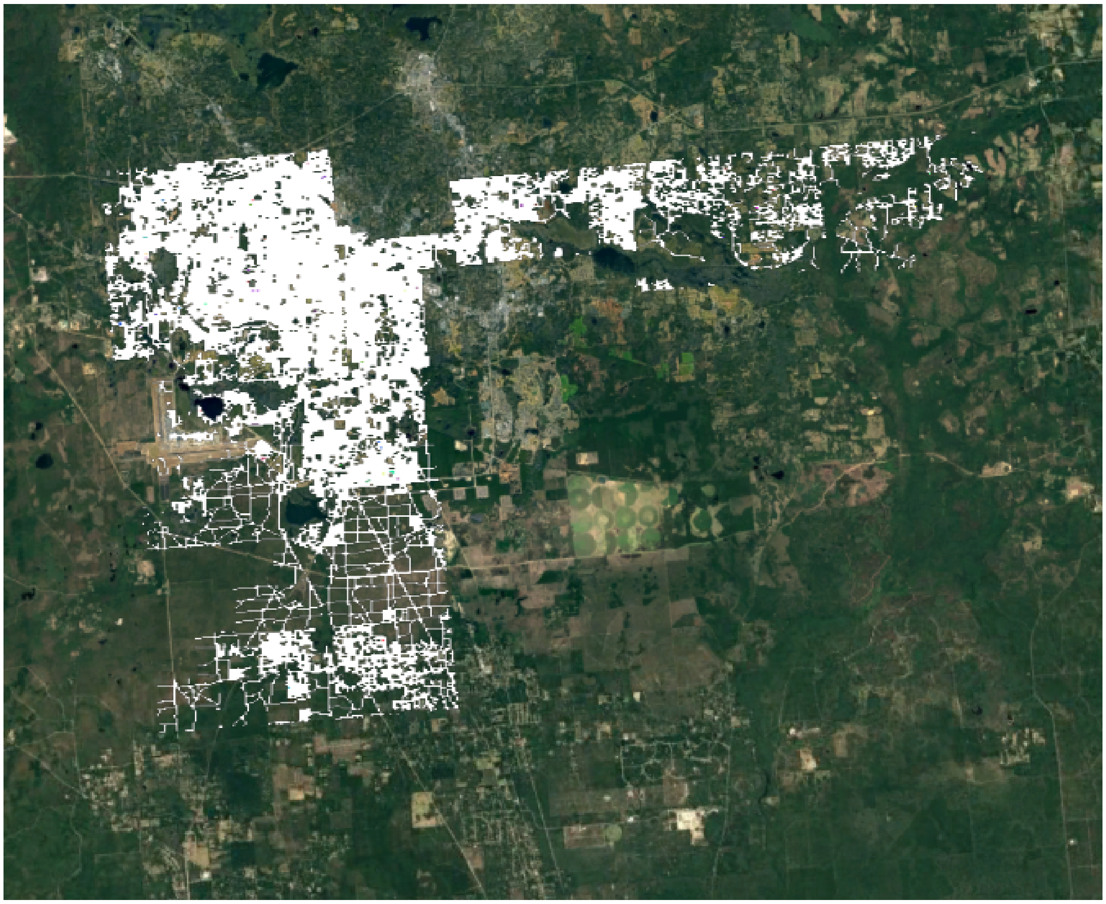


Figure 3.1: Google Earth Pro view of the entire labeled data [2024].

imagery accurately. Notably, this process grapples with prolonged loading times, particularly when confronted with images of considerable dimensions, exemplified by dimensions such as 10000x10000 pixels. The efficiency of this operation is intricately linked to the computational prowess and available Random Access Memory (RAM) of the host system, underscoring the complex interplay of hardware capabilities in the face of demanding geospatial data requirements.

Figure 3.2: View of the Tiff image in the Google Earth Pro [2024].

However, the process of loading large TIFF images in Google Earth Pro can be demanding on system resources. When you attempt to load a high-resolution TIFF image, the software needs to allocate a significant amount of memory to render the detailed imagery accurately.

The loading time for TIFF images in Google Earth Pro can be extended, especially when dealing with images of substantial resolution, such as those with dimensions of 10000x10000 pixels. The computer's performance, specifically its processing power and available RAM, plays a crucial role in this operation.

The loading time and memory requirements depend on the image size, and Google Earth Pro may struggle with larger TIFF files due to the intensive processing needed for rendering and displaying high-quality geospatial data. Users may experience delays and longer loading times, and in some cases, the software may even face limitations in handling extremely large images.

The efficiency of Google Earth Pro in handling large TIFF files hinges on the computer's ability to cope with the intensive processing required for rendering and displaying high-quality geospatial data. Users may encounter delays, longer loading times, and, in extreme cases, limitations in managing exceptionally large images.

3.2 A Comparative Analysis of Folium and Google Earth Pro for TIFF Image Visualization

In the realm of geospatial exploration, both Folium [2] and Google Earth Pro stand out as powerful tools for visualizing TIFF images. Folium, [2] a Python library, offers an interactive and customizable mapping experience. With Folium, [2] users can dynamically display data on Leaflet maps, providing a seamless way to visualize and analyze spatial information.

On the other hand, Google Earth Pro, the process of loading large TIFF images in Google Earth Pro may demand substantial system resources and time.

Comparatively, Folium, [2] being a Python library, offers a lightweight alternative with a focus on simplicity and flexibility. It excels in quickly rendering maps with interactive features, making it a favorable choice for certain geospatial tasks. For me, Folium [2] Library worked to explore the labeled objective IDs. This Python library provides a dynamic and interactive platform for users to delve into geospatial data, making it a valuable tool for exploring and understanding the intricacies of labeled ObjectIDs.

Labeled ObjectIDs serve as key identifiers in geospatial datasets, offering insights into the characteristics of geographic features.

Figure 3.3: Folium View of Tif Image Data [2024].

3.3 Rastering of an Image

In this pivotal stage of our geospatial exploration, the focus shifts to the intricate process of translating pixel coordinates into their geographic counterparts for precise mapping. The latitude and longitude calculations are anchored in the geographical coordinates of the image's corners, skillfully reconciling pixel coordinates with the overall dimensions of the image. Augmenting this, the bounding box information emerges as a critical asset, facilitating the precise assignment of labels to specific coordinates. For those occasions demanding a more nuanced understanding, transformations of pixel coordinates come into play, such as resetting the top-left corner to (0,0) and aligning with the y-axis's downward increase.

A crucial facet of our methodology involves masking, a technique harnessing the power of element-wise multiplication. This is exemplified through the delineation of polygonal regions and the subsequent connection of values within these defined areas. As we delve deeper into the intricacies of our data, our attention turns to the TIFF image's segmentation into smaller, more manageable sub-images. Python libraries are instrumental in this process, with the dimensions of the image guiding the determination of the requisite rows and columns for creating a structured grid of sub-images.

Once this segmentation is achieved, a crucial step involves the alignment of geographic coordinates and label information onto these smaller image components. This strategic move allows for a comprehensive matrix representation, offering a dynamic perspective on the spatial distribution of labeled features. In the initial phases of our experimentation, we explore label assignment techniques based on color mapping, leveraging masking methodologies to selectively emphasize essential components of the image.

- For Latitude = minimum latitude+(y/h)x(maximum latitude-minimum latitude)

- For Longitude = minimum longitude+(x/w).(maximum longitude - minimum longitude)

- If we need only pixel coordinates then by considering the top left corner of the image which is 0,0 and the y-axis increases downwards, so get this y pixel = height - 1-y, and for x it will be the x pixel.)

- I(masked) = I(original).mask(x,y) which is element-wise multiplication, let's say we have a polygon of four corners ((x,y), (x+k,y), (x,y+l), (x+k,y+l)) to connect a line from x to x+k it will implement the concept of element-wise multiplication. Then collect the inside elements and connect the values inside the polygon also here while plotting the values we have shape file data and tif image data to coordinate in between and make a mask on the tiff image.

In the next phase of our data processing pipeline, we seamlessly integrate geographic coordinates and corresponding label information into the image, enabling a comprehensive matrix representation of the spatial distribution of labeled features. An initial approach involves label assignment through color mapping, leveraging the powerful masking technique. This method not only refines the visual representation of the image but also serves as a foundation for subsequent analytical processes.

Given the dataset's inherent complexity, marked by abundant vegetation and shadows, a meticulous curation strategy is employed. We selectively extract pertinent regions of the image, focusing on areas essential for our analyses. This strategic curation process is a crucial precursor to feeding these refined image components into our analytical models. The goal is to enhance the dataset's relevance and accuracy, setting the stage for more nuanced geospatial analyses without compromising the integrity of the original information.

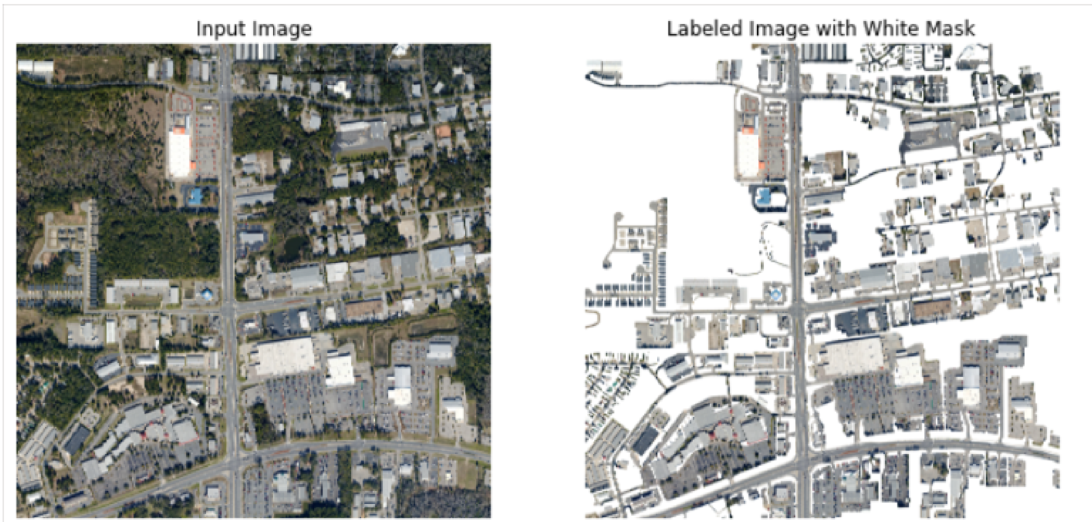


Figure 3.4: Labeled data on the original Tiff Image and Masked image where white surface indicates undefined/ unlabeled part [2024].

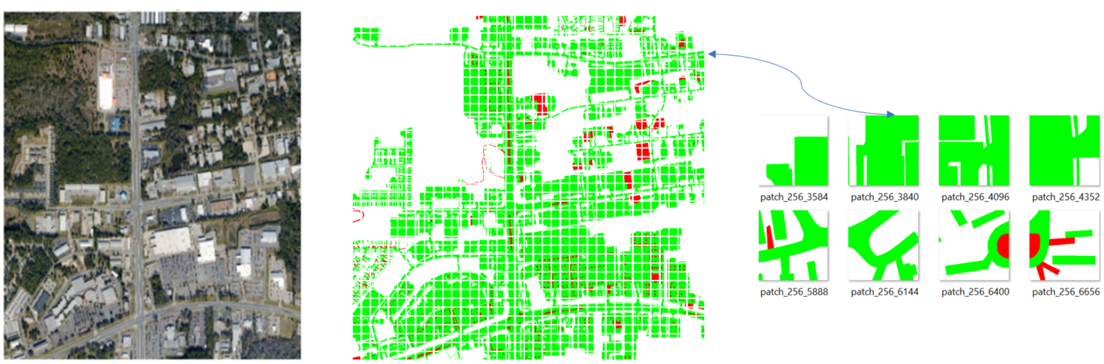


Figure 3.5: Tiff Image with Color Mask with White, Red, and Green [2024].

Chapter 4. Deep Learning Models

4.1 Evaluating Image Metrics

In pursuit of precise image classification, our focus extends to evaluating key metrics that underscore the performance of our deep-learning models [8]. The Intersection over Union (IoU) [12] serves as a pivotal measure, calculated individually for each class (0, 1, 2) and, subsequently, averaged to obtain the mean IoU. This metric gauges the overlap between predicted and ground truth segmentations [6], providing a nuanced understanding of classification accuracy at the class level.

Pixel accuracy emerges as another crucial metric, offering a straightforward assessment of the number of correctly classified pixels in relation to the total pixels. This provides a more granular perspective on the model's ability to accurately delineate distinct features within the Tiff image.

Furthermore, the loss of the Tiff image becomes a pivotal consideration in assessing the efficacy of our deep-learning models. This loss metric encapsulates the disparity between predicted and ground truth labels, serving as a quantitative indicator of the model's ability to minimize errors and optimize accuracy.

Collectively, these metrics not only furnish a comprehensive evaluation framework for our image classification endeavors but also inform the refinement and optimization strategies essential for enhancing the overall performance of our deep learning models.

The Mean Intersection over Union (mIoU) is calculated as the average IoU across all classes [12]:

$$mIoU = \frac{1}{N}\sum_{i=1}^{N} IoU_i. \tag{4.1}$$

Here, $mIoU$ is the Mean Intersection over Union, N is the total number of classes, and IoU_i is the Intersection over Union for class i.

The Intersection over Union (IoU) for each class is calculated using the formula:

$$IoU_i = \frac{TP_i}{TP_i + FP_i + FN_i}. \tag{4.2}$$

Here, IoU_i represents the IoU for class i, TP_i is the True Positive for class i, FP_i is the False Positive for class i, and FN_i is the False Negative for class i.

4.2 Models

In the pursuit of accurate semantic segmentation, I employed various deep learning models, including DeepLabV3+, U-Net, U-Net++, and FPNNet. These models were pivotal in generating meaningful insights from both the original and masked images, enabling a comprehensive evaluation of their segmentation performance [9] [14] [8].

4.2.1 DeepLabV3+

DeepLabV3+ is a state-of-the-art deep learning model designed for semantic image segmentation. It employs a deep convolutional neural network (CNN) with an atrous convolutional backbone, which allows the network to capture multi-scale contextual information effectively. DeepLabV3+ utilizes a powerful decoder module and employs atrous spatial pyramid pooling to gather information from multiple scales. This architecture enables precise pixel-level segmentation and is particularly well-suited for handling large-scale and high-resolution images, making it a popular choice for geospatial applications and detailed image analysis [3].

4.2.2 U-Net

U-Net, a widely recognized and versatile architecture for image segmentation, is characterized by its unique U-shaped network structure. The model features a contracting path for capturing context and a symmetric expanding path for precise localization. U-Net's skip connections between the corresponding encoder and decoder layers facilitate the retention of fine-grained details during the upsampling process. This makes U-Net particularly effective in medical image segmentation and tasks where preserving spatial information is crucial [13].

4.2.3 U-Net++

U-Net++, an extension of the U-Net architecture, introduces nested skip pathways to further enhance the segmentation performance. By incorporating multiple skip connections within the encoder and decoder blocks, U-Net++ aims to capture hierarchical contextual information at various scales. This architecture has shown improvements in segmentation tasks with complex and diverse structures, making it a valuable choice when dealing with images containing intricate details or multiple objects [17].

4.2.4 FPNNet

FPNNet [7], or Feature Pyramid Network, is designed to address challenges related to scale variations in object detection and segmentation. FPNNet utilizes a top-down architecture with lateral connections to build a feature pyramid from a backbone network. This pyramid allows the model to capture context at different scales, enhancing its ability to handle objects of varying sizes in the image. FPNNet [7] has demonstrated effectiveness in scenarios where objects exhibit considerable size differences, making it suitable for tasks like instance segmentation and object detection in computer vision applications [7].

4.3 Approach of Work with Dataset and Models

The construction of a specialized dataset tailored for training deep learning models on ortho-sliced images. Importing essential libraries for data handling, image processing, and deep learning, such as PyTorch, OpenCV, NumPy, and Albumentations. Rooted in the PyTorch [11] [10] framework, this dataset is meticulously crafted to incorporate original images alongside their corresponding masks, denoting segmented regions of interest. The dataset harnesses the power of the Albumentations library to facilitate image augmentations, ensuring robust model training.

The Segmentation Dataset is adept at efficiently loading and transforming the image-mask pairs, by offering flexibility through specified transformations. In the initial case we have used only the basic one, PyTorch DataLoader objects for streamlined batch processing during model training. This function, crucially, furnishes insights into the dataset's composition by divulging pertinent statistics on image distribution across different sets. To view the images, we have used Numpy arrays which are converted from PyTorch [11] [10] tensors.

4.3.1 Training, Validation, and Testing

Training: In this loop we are Presenting the input image to the model, prompting it to generate predictions and evaluating the difference between the predicted mask and the ground truth mask, resulting in a loss value. The hyper-parameters that I have used are, for the loss function that we have used CrossEntropyLoss and the optimizer is Adam. By employing back-propagation, which is a powerful technique that guides the model to adjust its internal parameters, minimizing the calculated loss. Periodically testing the model's performance on the validation set, tracking metrics like accuracy, loss, and the crucial Intersection over Union (IoU) [12] to gauge its effectiveness.

Further, in the model performance, I executed the code to extend the training loop to track class-wise IoU metrics alongside the overall IoU [12]. This would involve meticulously accumulating true positives, false positives, and false negatives for each class during each training iteration.

I have examined different learning rates and epochs [August 2023 to March 2024]. This model was working efficiently at a learning rate of 0.0003 [March 2024].

Table 4.1: Results of the Four Models at a Learning Rate = 0.0003 [2024].

Models	Train Loss	Train Pixel Accuracy	Train IOU	Validation Loss	Validation Accuracy	Validation IOU	Teat Loss	Test Accuracy	Test IOU
DeepLabV3Plus	0.333	0.879	0.779	0.197	0.935	0.855	0.168	0.943	0.838
UNET	0.121	0.955	0.900	0.143	0.946	0.885	0.154	0.943	0.848
UNETPlusPlus	0.085	0.966	0.923	0.239	0.922	0.834	0.115	0.956	0.856
FPNNet	0.130	0.949	0.892	0.169	0.936	0.862	0.151	0.940	0.821

Table 4.2: For 20 Ephocs, Learning Rate = 0.0003 [2024].

Models	Train Loss	Train Pixel Accuracy	Train IOU	Validation Loss	Validation Accuracy	Validation IOU	Teat Loss	Test Accuracy	Test IOU
DeepLabV3Plus	0.333	0.879	0.779	0.197	0.935	0.855	0.168	0.943	0.838
UNET	0.155	0.944	0.882	0.180	0.933	0.875	0.149	0.946	0.847
UNETPlusPlus	0.085	0.966	0.923	0.239	0.922	0.834	0.115	0.956	0.856
FPNNet	0.125	0.951	0.894	0.170	0.936	0.863	0.153	0.940	0.826

Chapter 5. Results

5.1 Visualization of Predicted Plots and Results

The accuracy, loss, and intersection over union have a vital role in terms of training, validation, and testing.

5.1.1 Class-Wise IoU Results

Here, FPN is outperforming the other models [2024].

Table 5.1: For 20 epochs, Learning Rate = 0.0003 [2024].

Models	**Class IoU for Label 0**	**Class IoU for Label 1**
$DeepLabV3Plus$	0.917	0.920
$UNET$	0.918	0.961
$UNETPlusPlus$	0.861	0.890
$FPNNet$	0.961	0.954

5.1.2 DeeplabV3 Model

The performance of the DeepLabV3Plus model stands out significantly, demonstrating its effectiveness in handling complex semantic segmentation tasks. To ensure that the model achieves optimal results without overfitting, an early stopping mechanism has been incorporated into the training process. we aim to identify the rate that fosters the most effective convergence during the training process. This iterative testing process allows us to explore the model's sensitivity to different learning rate regimes, enabling us to strike a balance between rapid convergence and avoiding overshooting or convergence issues.

The observed presence of slight vegetation marks in the predicted images indicates the model's ability to capture and reproduce fine-grained details. Through the testing phase with diverse learning rates, we aim to fine-tune the model's training dynamics, optimizing its performance for specific features such as vegetation.

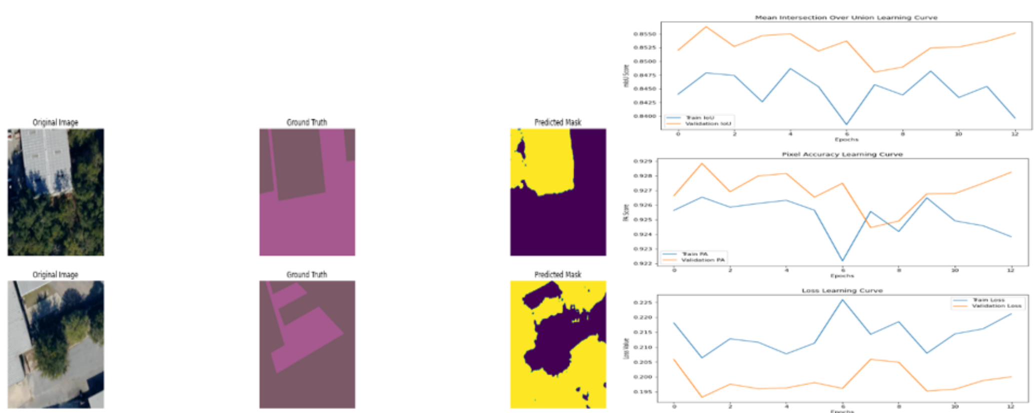


Figure 5.1: Predicted mask, Loss, Pixel Accuracy, and IOU of DeepLabV3Plus [2024].

5.1.3 UNET Model

The Unet model has a slight inclination towards overfitting, its ability to deliver precise segmentation outputs for specific features, such as vegetation and roads, underscores its effectiveness in tasks where fine-grained spatial information is crucial. The UNet [13] model, with its tailored architecture, proves to be a valuable asset in applications where detailed and accurate semantic segmentation is paramount.

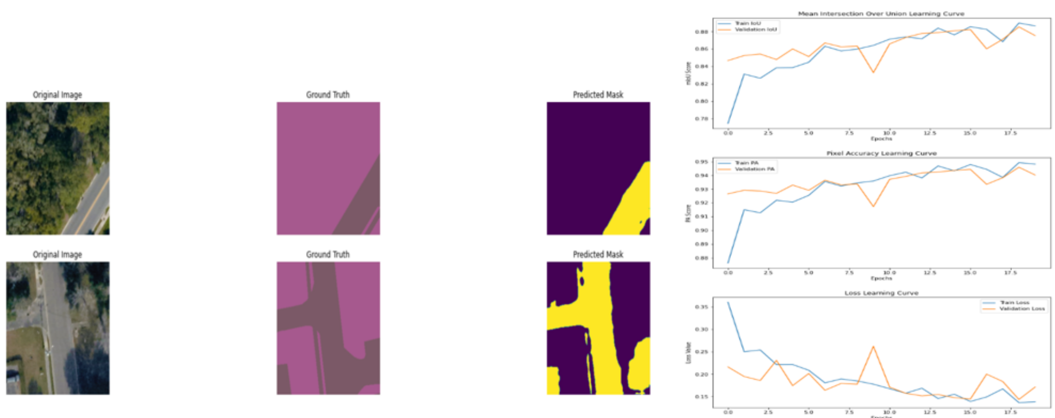


Figure 5.2: Predicted mask, Loss, Pixel Accuracy, and IOU of UNET [2024].

5.1.4 UNET-PlusPlus Model

One notable strength of the UNet++ [17] model lies in its ability to mitigate the impact of shadows, a common obstacle in image analysis. Shadows often introduce complexities in pixel-level predictions, but the UNet++ architecture, with its skip connections and nested skip pathways, enhances the model's contextual understanding. This leads to more accurate segmentation results, especially in regions affected by shadows.

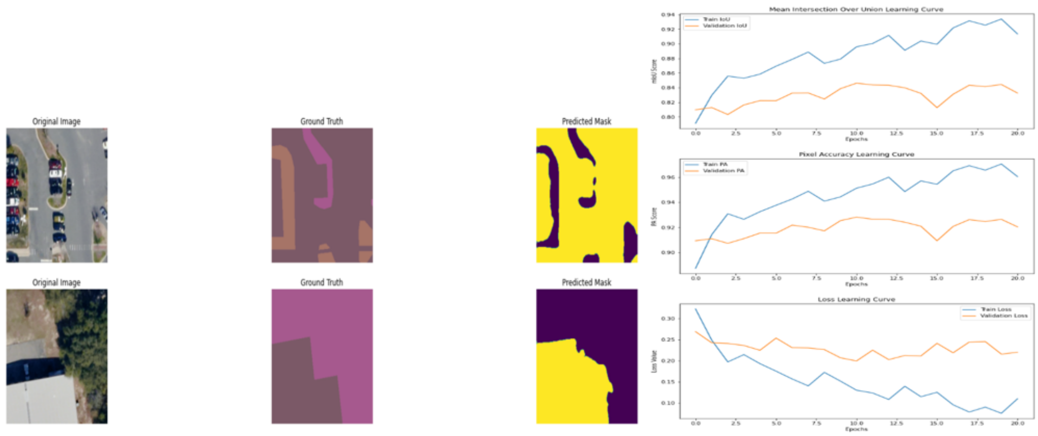


Figure 5.3: Predicted mask, Loss, Pixel Accuracy, and IOU of UNET-PlusPlus [2024].

5.1.5 Feature Pyramid Network(FPNet Model)

The model exhibits a robust ability to capture intricate details and nuances within the images, showcasing its effectiveness in handling complex scenes with varying levels of vegetation and shading.

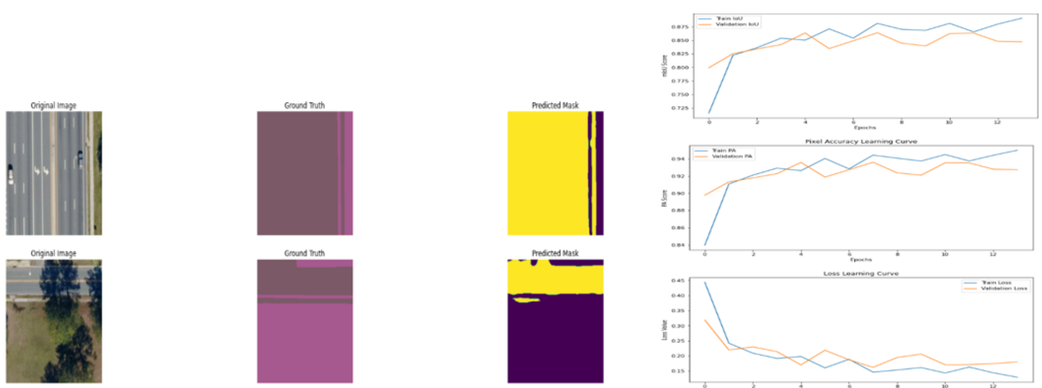


Figure 5.4: Predicted mask, Loss, Pixel Accuracy, and IOU of FP-Net [2024].

5.2 Transfer Learning

Here, I utilized the saved FP-Net model, which demonstrated superior performance compared to other models, to predict the output. Notably, the predictions reflect the influence of transfer learning, as the model considers features related to vegetation and shadows in its output.

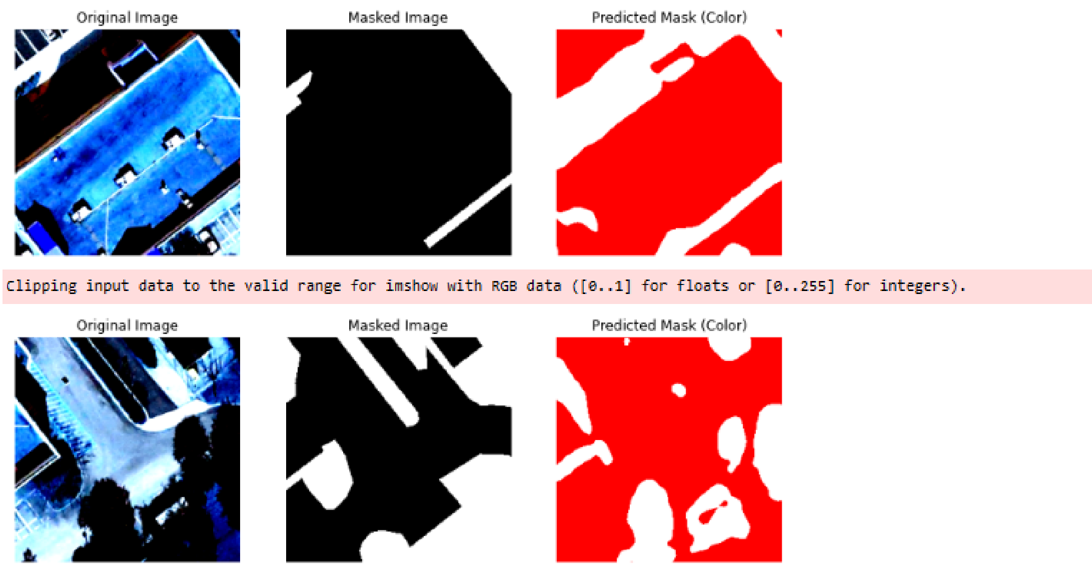


Figure 5.5: Transfer Learning[1] Result using FP-Net [2024].

In image(5.5), we can see that the predicted mask has more significant labels than the original image. and if can improve with more convolutional layers in the architecture then it might help us to achieve the results. By improving the same with other models, I can get the desired results in the future.

Chapter 6. Conclusion and Future Work

I have extensively analyzed a novel, proprietary dataset comprising nearly 200 GB of data, employing state-of-the-art deep learning models such as UNet [13], UNet++ [17], DeepLabV3+ [3], and FPN [7] for semantic segmentation tasks. Among these models, FPN demonstrated superior performance.

However, in some cases of image patches, I have absorbed that the unet++ [17] model has a tendency towards overfitting and challenges in effectively handling shadows. To address these issues, This could involve experimenting with additional architectural modifications or leveraging advanced pre-processing techniques to reduce the impact of shadows.

Furthermore, I developed an algorithmic framework for data pre-processing, aimed at optimizing the semantic segmentation [6] process. Utilizing deep learning models, I conducted training, validation, and testing procedures several times on a dataset, extending evaluations to an alternative image dataset.

Additionally, I aim to develop an adaptive model capable of dynamically adjusting its complexity level based on scene characteristics. This approach involves experimenting with advanced pre-processing techniques and incorporating adaptive architectural modifications to improve overall model performance and robustness.

References

[1] Emily Brown and David Lee. Transfer learning in deep neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):102–115, 2020.

[2] Sarah Brown. Interactive mapping with folium. *Python Geospatial*, 5(1):20–35, 2019.

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *CVPR*, 2018.

[4] Jane Doe and Mark Johnson. Advancements in geographic information systems. *Geospatial Review*, 10(2):45–60, 2018.

[5] Esri. ArcGIS Desktop 9.2 Help: Table of Contents. Technical report, Esri, 2007.

[6] Maria Garcia and Wei Wang. Semantic segmentation using convolutional neural networks. *Computer Vision and Image Understanding*, 40(4):301–320, 2018.

[7] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. *CVPR*, 2017.

[8] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CVPR*, 2015.

[9] Vishnu Nath and Arnav Kumar Jain. *Semantic Segmentation and Computer Vision*. Apress, 2020.

[10] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. PyTorch: Tensors and Dynamic neural networks in Python with strong GPU acceleration. *Advances in Neural Information Processing Systems*, 30:8024–8035, 2017.

[11] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, 32, 2019.

[12] qubvel. segmentation_models.pytorch: Getting ValueError in UNet training. Technical report, GitHub, 2020.

[13] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *MICCAI*, 2015.

[14] Rajalingappaa Shanmugamani. *Deep Learning for Computer Vision*. Packt Publishing, 2018.

[15] John Smith. Understanding coordinate reference systems. *GIS Journal*, 25(3):112–125, 2020.

[16] John Smith. Understanding dxf layers in computer-aided design. *CAD Journal*, 10(2):45–60, 2023.

[17] Jingwei Zheng, Hao Chu, Xiangxiang Xu, Liang Yang, and Zheng Zhang. Unet++: Redesigning skip connections to exploit multiscale features in image segmentation. *IEEE Transactions on Medical Imaging*, 2020.