IDS 705 Final Report

UAS Semantic Segmentation for Safe Landing

Team Members:
Jabban, Alexander Dany
Lin, Zhanyi
Wu, Yuyou (Pomelo)
Zhu, Yuanjing

Team Number: 02

Abstract

The ethics and safety of UAVs have been called into question for some time, but with so much environmental and social upside, developing safe flying and landing is of the utmost importance. In order to enhance drone safety, our research is concerned with applying state of the art deep learning models to solve semantic segmentation with drone cameras. To evaluate the performance of our models we used several metrics common in semantic segmentation problems including the jaccard index. Our models were able to achieve 0.8 multi class weighted jaccard index, 0.4 unweighted multi class jaccard index, and .85 binary unweighted jaccard index which is on par with other semantic segmentation problems in other domains. However, there remain some shortcomings of our model for specific classes that we hope to address in upcoming research.

Introduction

Unmanned aerial vehicles (UAVs), also known as drones or unmanned Aircraft Systems (UAS), are remotely piloted or fully autonomous aerial vehicles equipped with various sensors, cameras, and other instruments to collect data and perform tasks [1]. Initially developed for military missions in 1916, UAVs have since been extensively utilized for military surveillance, reconnaissance, and targeted strikes. Over time, UAV technology became more advanced, and their size, weight, and cost decreased, making them more accessible to non-military organizations. Later with the development of more sophisticated and versatile drones, UAVs' civilian applications have expanded to include fields such as agriculture, journalism, filmmaking, and even package delivery [3]. In 2013, Amazon's chief executive, Jeff Bezos, announced plans to use UAVs for package delivery [4]. Such development led to the creation of Amazon Prime Air, a delivery system that uses UAVs to deliver packages to customers in 30 minutes or less with the benefits of fast and efficient package delivery and reduction of carbon emissions and traffic congestion on roads.

While UAV delivery offers tremendous benefits and has the potential to revolutionize the logistics field, they face numerous challenges, including ethical concerns and technical problems. One of the most significant challenges is the need to correctly identify safe landing places which requires recognizing objects and scenes with short processing time and negligible landing errors. In order to achieve automatic object detection and recognition, as well as mapping and surveying of large areas, semantic segmentation is applied for efficient and accurate analysis of high-resolution images and videos captured by UAVs. Semantic segmentation is a computer vision technique that involves the labeling of each pixel in an image to the category of the object to which they belong [6]. Through semantic segmentation, therefore, UAVs can accurately identify and classify different elements in the environment, such as roads, buildings, and

vegetation, which allows UAS to navigate through the environment efficiently and safely [7]. If ideally UAVs can achieve 100% successful landing rate, it not only can address technical problems, but also mitigate ethical concerns through differentiating objects, scenes, and humans as well as avoiding airspace crowdedness to maintain safety.

Recent algorithmic development in semantic segmentation has allowed significant improvements in the accuracy of object detection and identification and real-time performance [8]. However, depending on the specific landing scenario and the type of drone, success landing rates vary from 70% to 97% with different heights and dynamic scenes [9]. This wide range of accuracy is extremely problematic as it can lead to increased risks of accidents and damage to the drone and surrounding environment. Unreliable landing performance can also result in significant delays and potentially life-threatening situations. As a result, ensuring robustness to different environmental conditions and generalization to new environments remains a challenging task. Driven by the potential commercial and environmental benefits of drone delivery, therefore in this study, we will focus on applying deep learning techniques for semantic segmentation to help enable UAS to understand objects in a scene, which is crucial for safe and reliable landing navigations. As enthusiasts and practitioners in the field of computer vision, we are motivated to advance state-of-the-art perception tasks as well as improving the accuracy and efficiency of semantic segmentation involving drone images. Ultimately, we hope to foster the development of fully autonomous UAS and overcome the challenges that come with it.

Background

Semantic segmentation partitions an image into meaningful and distinct objects, assigning each pixel of the image a label corresponding to the class of the object it belongs to. Unlike traditional image classification tasks that involve identifying the presence of a particular object in an image, semantic segmentation requires a more granular understanding of the image's content, allowing for a more detailed analysis of the scene. For instance, in autonomous driving (Figure 1), semantic segmentation can assist vehicles in identifying different objects on the road, including pedestrians, cyclists, and other vehicles, enabling them to make better decisions and avoid collisions [10].

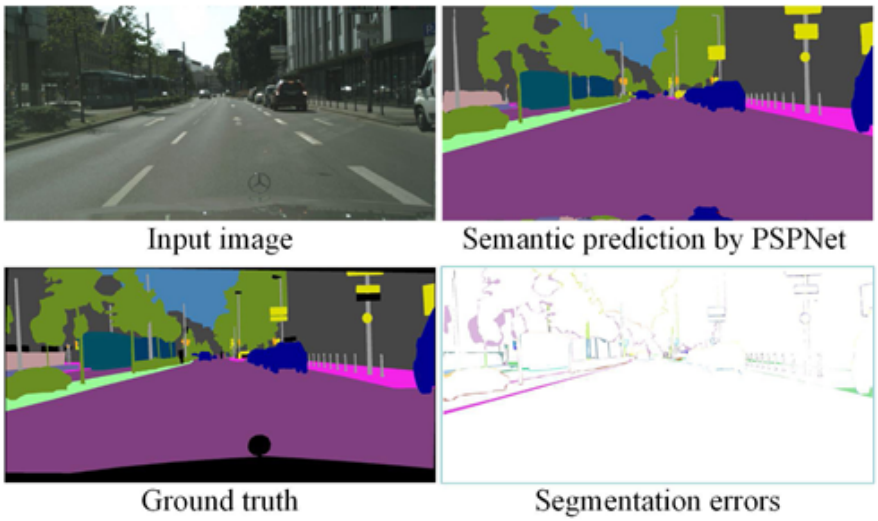

Figure 1 Illustration of semantic segmentation

Several approaches have been proposed to tackle the semantic segmentation problem, ranging from traditional image processing techniques to deep learning-based methods. Early methods involved using hand-crafting features (e.g. edges, texture, and color) and clustering algorithms to segment images. However, these methods often relied heavily on expert knowledge and were limited in capturing complex image structures. In recent years, deep learning-based approaches have achieved remarkable success in semantic segmentation tasks. These methods involve training convolutional neural networks (CNNs) to extract image features and make pixel-level predictions. Building on CNNs, researchers further exploited the architecture advantage of CNNs and created Fully Convolutional Networks (FCNs) [11]. Traditional CNNs utilize a fully connected layer in their final layer, which generates a fixed-size vector indicating the probability of each class for the entire image. In contrast, FCNs replace the fully connected layer with a convolutional layer that produces a spatial map of class predictions for each pixel in the image. This enables FCNs to capture both the local and global context of the image, allowing for more accurate object segmentation.

In 2015, U-net was introduced by researchers at the University of Freiburg and has since become a popular architecture for semantic segmentation analysis [12]. The key feature of the U-Net architecture is the use of skip connections between the encoder and decoder parts of the network, allowing for the preservation of spatial information and the handling of small objects and complex boundaries. In general, UNet improves upon FCNs by incorporating a U-shaped architecture, using skip connections to preserve spatial information, and combining high-level and low-level features to produce more accurate segmentations. SegNet, introduced in 2016 by researchers at the University of Cambridge, is a more computationally efficient architecture, making it a popular choice for real-time segmentation applications [13]. Specifically, the use of max-pooling indices in the encoder, which is used to upsample the feature maps in the decoder, helps to preserve the original spatial information of the input image while allowing the network

to reduce the size of the feature maps. More recent approaches have focused on using attention mechanisms, multi-scale feature fusion, and conditional random fields to expand the receptive field of the network and capture larger contextual information [14-16].

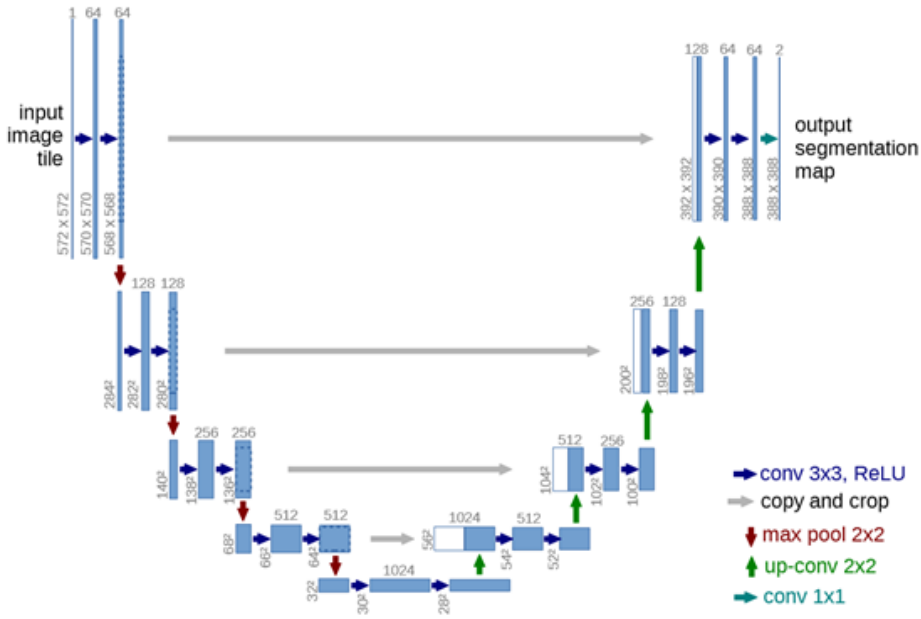

Figure 2. Architecture of U-net for 32×32 pixels in the lowest resolution

We are primarily interested in applying these recent algorithmic innovations to examine the accuracy and efficiency in a newly published comprehensive dataset of aerial residential images and to understand the performance and limitations in these settings. By leveraging existing methods and algorithms, we can identify areas for improvement and explore new approaches to achieve better accuracy in semantic segmentation tasks. Our study will not only contribute to the development of more concrete and accurate semantic segmentation models but also advance our understanding of the performance of the models in new and diverse environments. Ultimately, our research aims to pave the way for more accurate and reliable semantic segmentation in a range of applications.

Data

General Description:

Our dataset is provided by AIcrowd which is believed to be one of the largest image datasets with accompanying full semantic annotation and mono depth estimation ground-truth over a wide range of AGLs and different scenes. This dataset comprises 1787 pairs of images, where each pair consists of an input grayscale image and its corresponding *uint8* image annotation [17]. The make-up of the image subset consists of greyscale birds-eye-view images of residential

neighborhoods ranging from 5 to 25 meters above ground. The annotation subset consists of image-like matrices where each cell contains an integer value corresponding to the class of the respective pixel in the original image.

Both the input and annotation images are stored in *.png* file format, and have the same filename for easy matching. This naming convention also allows for seamless integration with various machine learning pipelines and frameworks, making the dataset highly versatile.

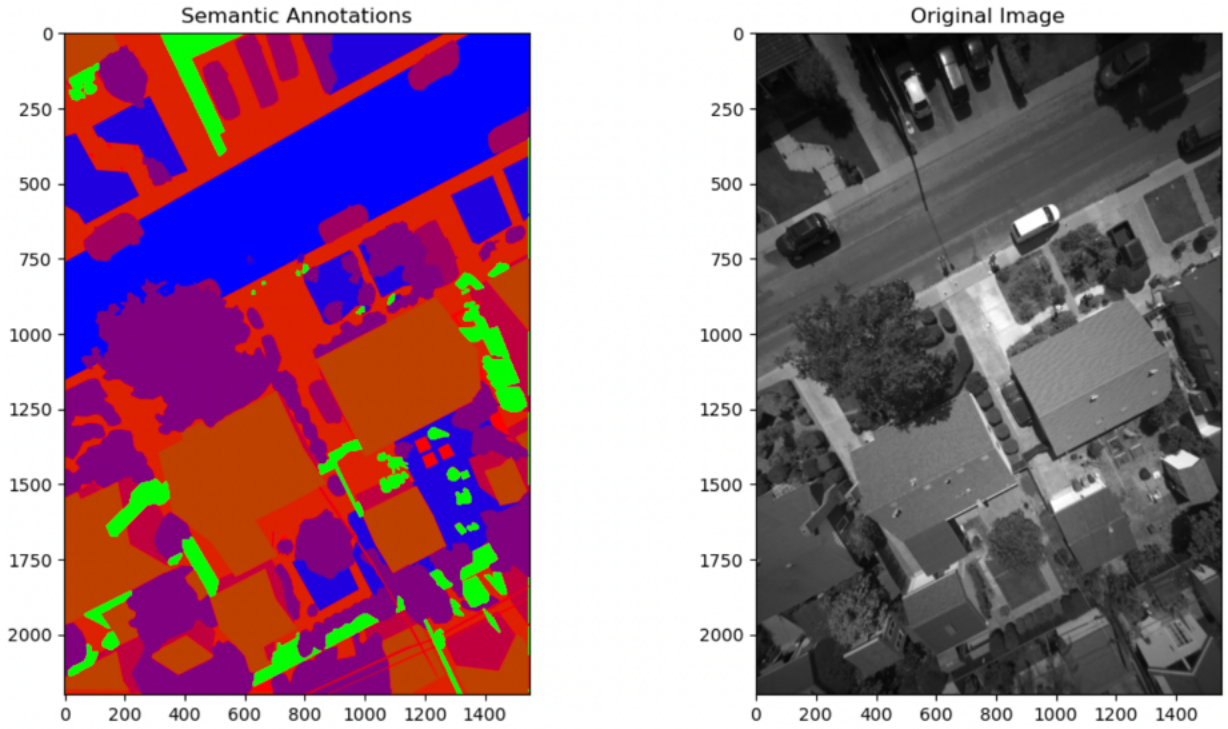

Figure 3. Image from dataset with corresponding semantic annotations

Input file:

The high-resolution ground photo captured by a drone is a crucial component of our drone safety training data. It has a resolution of *2200* x *1550* pixels, providing a detailed view of the terrain around and below the drone, which allows us to assess the safety of landing areas. Our ultimate goal is to train our drones to automatically analyze the captured image and determine whether the landing area is safe or not.

Annotation File:

The annotations are stored as .png files and have the same dimensions, *2200* x *1550* pixels, as our input images. As mentioned previously, each element in the annotation matrix corresponds to the true label of the pixel at the same position of the associated input image. There are 16 distinct labels which take a value between 0 – 15 with an additional label for "unknown" pixels which takes the value 255. The left image in figure 3 depicts a single instance of annotation where each of the classes have a different color corresponding to its unique label value such as bright blue representing all instances of asphalt, dark purple representing all instances of high vegetation, and brown representing all instances of roofs. Table 1 below shows each of the distinct labels and their corresponding values.

Table 1. Annotations of fully labeled images across 17 distinct categories

0	WATER	1	ASPHALT
2	GRASS	3	HUMAN
4	ANIMAL	5	HIGH_VEGETATION
6	GROUND_VEHICLE	7	FAÇADE
8	WIRE	9	GARDEN_FURNITURE
10	CONCRETE	11	ROOF
12	GRAVEL	13	SOIL
14	PRIMEAIR_PATTERN	15	SNOW
255	UNKNOWN		

Potential Issues and Corresponding Solution:

While our dataset of 1787 observations provides a good starting point, it may be limited in size when training complex models or those with many parameters. To address this issue, we have employed the techniques of data augmentation.

By making adjustments to our data in thoughtful ways we will be able to increase the size of our dataset without introducing too much noise, which will ultimately improve the power of our dataset and the robustness of the models that learn from it. A key word used was "thoughtful", since augmentations should be aligned with actual images the drones will see in the real world. Each of the methods are inspected thoroughly to make sure they improve the dataset. The specifics of the data augmentation procedures used are presented in the experiments section.

Experiments

Data Preprocessing and Augmentation:

The initial data exploration and preprocessing indicated that all values in our images, both the original images and their corresponding annotation masks, were within the expected ranges as shown by the histograms in figure 4. There was some variation in image shape in our dataset with widths and heights varying by 100 or so pixels, so we standardized image dimensions by a combination of cropping and pixel interpolation to ensure all images are the shape of 2200 x 1550

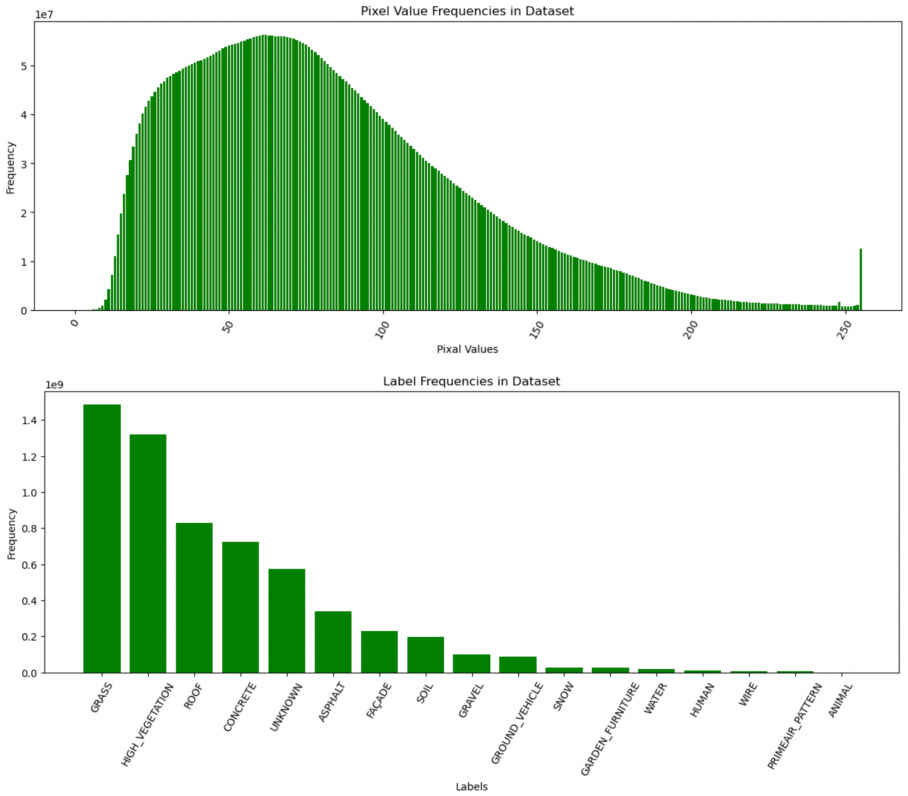

Figure 4. Label and Pixel frequency in dataset

For data augmentation we originally started out with 1786 images. In order to expand our training set we used several data augmentation techniques. We wanted to use techniques that would give us realistic alternative views of our original images that would not distort the images too severely as to create images that are not representative of what exists in the original dataset as well as augmentations that may result in poor learning from our model. Since these images are

high resolution, greyscale, aerial views, the pool of useful transformations consists of rotations and other affine transformations, horizontal and vertical flipping, and cropping. Rotations that are not at multiples of 90 degrees introduce issues of diagonal padding which can make learning harder for the model. Rotations that are not multiples of 180 are also not ideal since they can have very large horizontal padding which is not present in the original images. Vertical and horizontal flips can create equivalent transformations to the remaining rotations. Cropping images is effective but cropping to too small of an image can have problems when trying to get to input image size since the pixel interpolation needed to get back to 2200 x 1550 dimension will result in very blurry images which are not present in our high resolution dataset. Thus we will apply a combination of cropping with large output images with additional vertical and horizontal flips. Since there are only 4 unique combinations of flips and the cropping changes are minimal, we will apply 6 augmentations to each image taking our dataset from 1786 images to 10716 images.

Experimental designs

Figure 5 illustrates the flow of our research. We start by passing the collected dataset to pre-processing and augmentation techniques to enhance the dataset. We then randomly split the images into training, validation, and test sets. Of the 1787 images, 1365 were used for training, 242 for validation, and 179 for testing. We utilized the U-Net architecture for object segmentation, comprising two paths, the encoder and decoder. We employed transfer learning to use the convolution layer of the first step as the encoder, which was then combined with deconvolution layers for the decoder. For the encoders, we experimented with four pre-trained neural network architectures, namely VGG-16, ResNet, MobileNet, and Vision Transformer (ViT). By using these pre-trained neural networks as encoder, the UNet model can leverage the features learned by the pre-trained model on large-scale image classification tasks, which can help the model to achieve better performance on the segmentation task with less data and training time compared to training the entire model from scratch. To ensure that the encoders were functioning properly, we tested them on a well-known sample dataset, CIFAR10, before applying them to our drone dataset. After careful consideration, we eliminated Vision Transformer (ViT) due to its extended training time and lower model accuracy compared to the other three. We then evaluated the segmentation models by comparing the trained model's output segmentation map to the true label mask or ground truth for each object in different test images. Finally, we employed standard evaluation metrics for semantic segmentation, such as variations of the Jaccard Index, confusion matrices and, and variations of the Dice Score on binary and multiclass versions of our models’ outputs.

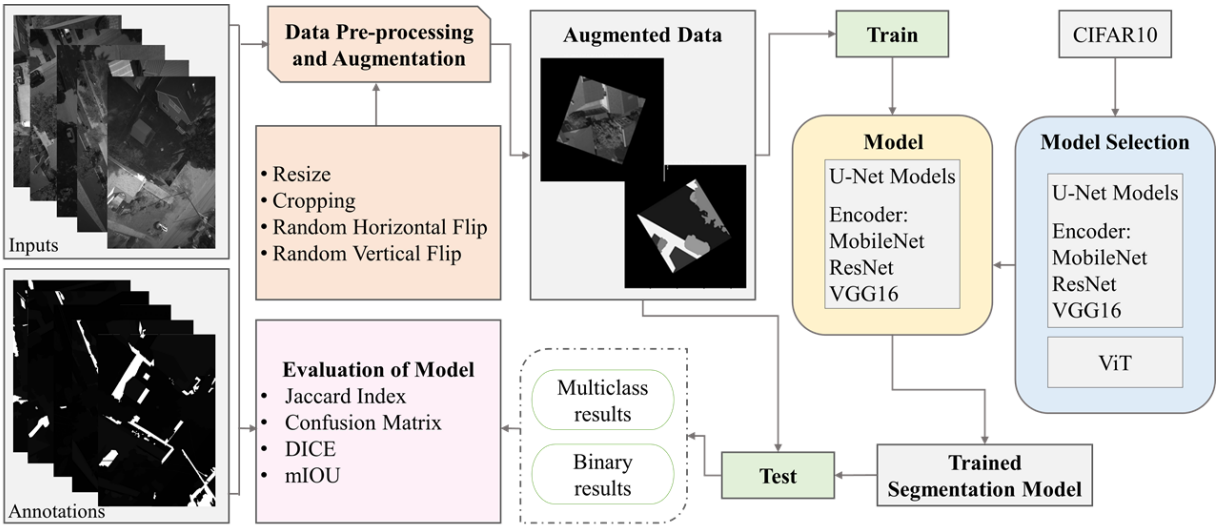

Figure 5. Flowchart showing the overall experimental design

Classification techniques

U-Net-based segmentation model

U-Net architecture is a popular neural network model for semantic segmentation tasks. The architecture consists of an encoder and a decoder path, forming a U-shape, which enables the network to perform both down-sampling and up-sampling operations. The encoder network consists of multiple convolution layers, each of which is followed by a rectified linear unit (ReLU) activation function and a batch normalization layer, which extract feature maps from the input image at progressively lower resolutions. Max pooling layers are used after every two convolution layers to down-sample the feature maps. In this study, the encoder is applied from the three pre-trained models, MobileNetV2, VGG-16 and ResNet (see Figure 6). We retrieved the coefficients of the three models from pre-trained models on ImageNet as initial weights, and the depth of the encoder path is assigned to 5. The decoder path then performs a series of deconvolution and concatenation operations using two Conv2dReLU layers and an Attention layer in each decoder block, which up-sample the feature maps and recover the spatial resolution lost in the encoder path. Skip connections are used between the encoder and decoder paths to help preserve high-level features from the input image while also allowing the network to learn local details. Final output is predicted by applying a Conv2d layer with a 3×3 kernel and 1×1 padding with 17 output channels corresponding to the segmentation task's number of classes.

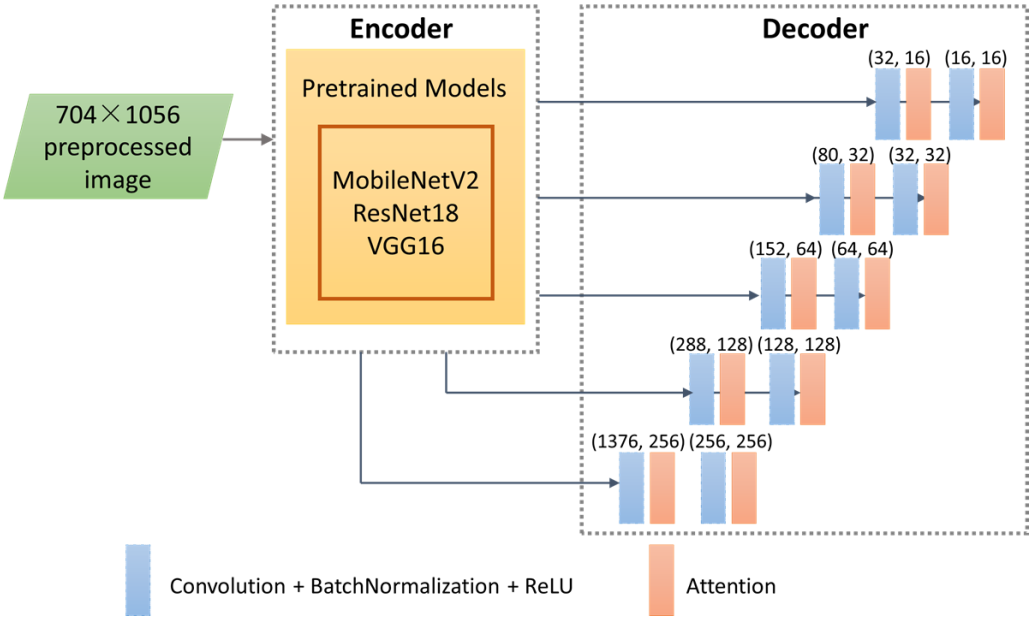

Figure 6. UNet-based neural network architecture

Evaluation Criteria

In order to assess the performance of each of our models, we are using several metrics to gain insight into their strengths and weaknesses. The first metric we are using is the Jaccard index also known as the "intersection over union". We are using three variations which include class weighted average Jaccard index, Unweighted average Jaccard Index, and Jaccard Index per class. The weighted Jaccard Index gives us a single value to easily compare model performance with high class imbalance. This is very much the case in our dataset as seen by figure 4. Jaccard Index per class provides more granular insight into how well individual classes are identified and can indicate which classes are harder to identify than others, and thus where efforts may be needed to improve. The unweighted average Jaccard Index is a common semantic segmentation metric treating all classes equally and is thus useful to have in order to compare to future experiments or papers.

Our next metric, the Dice score, is a similarity metric computed similarly to Jaccard index, but provides a slightly different interpretation and penalizes incorrect classifications less severely. Due to the same reasons mentioned previously for the Jaccard Index, we are using three variations of the Dice score, class weighted Dice, unweighted Dice, and Dice score per class.

The third metric we are using is a multiclass confusion matrix. This provides insight into which groups of classes are most often mistaken for each other. Using the confusion matrix along with the per class Jaccard Index and Dice scores provide a more informative picture of how the models are performing, where the errors are coming from, and the extent of these errors.

Finally, we are applying these metrics to two versions of our results. The first version is the original multiclass prediction masks. The second version takes those multiclass predictions and converts them into binary predictions representing safe landing zones and unsafe landing zones.

Results

After training each of our models using a train-validation split, we had three fitted models that we could apply our unseen test dataset and assess each of our model's generalization performance.

Table 2. Multiclass Model Generalization Performance

	Jaccard Wtd.	Jaccard unWtd.	Dice Wtd.	Dice unWtd.
VGG16	0.7999	0.3895	0.8744	0.4446
ResNet18	0.8007	0.3828	0.8755	0.4368
MobilNet	0.8155	0.3864	0.8842	0.4400
Baseline	N/A	0.4978	N/A	0.6145
Random Guess	0.0213	0.0213	0.0411	0.0411

One thing to note before we explore the results, our Baseline is the performance of an MMSegmentation model trained on the same set of drone images which is provided by AIcrowd and Amazon Prime Air. From our multi class generalization performance table, we see that all of our models perform relatively similarly across the 4 metrics listed. Looking at the random guess performance, we get a sense of just how difficult this task is which is compounded by the total number of classes available to select from. The disparity between the weighted metric performances vs the unweighted metric performances indicates that we are able to determine the larger class components of the image well such as grass or concrete, but struggle with some of the smaller and less frequent classes like humans or wire. This will be explored further when we break down individual class performance. Our results also indicate that we are underperforming the baseline model. This may be a result of minimal hyperparameter tuning given certain time constraints. With such complex models such UNet and Encoder-Decoders it may take a significant amount of time to determine optimal hyperparameters in a relatively large hyperparameter space. Meanwhile, the baseline model was highly fine tuned for this specific task with ample time and resources to determine optimal hyperparameters.

Table 3. Binary Class Model Generalization Performance

	Jaccard Wtd.	Jaccard unWtd.	Dice Wtd.	Dice unWtd.
VGG16	0.8882	0.8481	0.9390	0.9127
ResNet18	0.8909	0.8510	0.9404	0.9143
MobilNet	0.8992	0.8550	0.9454	0.9162
Random Guess	0.3	0.3	0.5	0.5

As a reminder, for the binary class generalization performance, we are combining our 17 distinct classes into a binary set, “safe” or “unsafe” and then applying our metrics to these aggregated groups. This generalization performance indicates that all our models are well above the random guess benchmark and are able to distinguish between safe and unsafe environments quite well, with our main concern being the safety of the public and secondarily the safety of the drone these results are quite promising.

Since all of the models we built had relatively similar performance we will analyze class level metrics only for VGG16 but the findings are almost identical across all three models and we provide the metrics for the other models in the appendix.

water	asphalt	grass	human	animal	high vegitation	ground vehicle	façade	wire	garden furniture	concrete	roof	gravel	soil	primeair pattern	Snow	unknown
0.69	0.81	0.78	0.21	0.00	0.79	0.79	0.61	0.06	0.17	0.75	0.89	0.25	0.22	0.87	0.42	0.54

Figure 7. VGG16 Class Level Jaccard Index

	water	asphalt	grass	human	animal	high vegitation	ground vehicle	façade	wire	garden furniture	concrete	roof	gravel	soil	primeair pattern	Snow	unknown
water	449075	6549	396	0	0	454	741	51	0	423	9959	10	142	5	27	4	2235
asphalt	1606	9100174	139599	60	0	98770	40820	8275	11379	0	187765	5638	2571	9055	0	38	23936
grass	37941	94660	41441871	13213	0	1803606	7700	25510	269	247	269785	70093	109941	856182	6947	25918	812222
human	14	2273	26911	86041	0	10030	7655	373	48	5087	18165	36	1448	1438	584	153	112319
animal	0	0	692	295	0	587	318	0	17	16	476	0	0	0	0	269	1657
high vegitation	1869	79362	1361014	2064	0	35809422	19834	54258	15324	188	170373	103164	77576	148790	0	1544	705220
ground vehicle	141	27430	3937	423	0	26260	2474532	5002	88	86	34207	1904	307	1357	0	563	90945
façade	0	6336	20137	48	0	107874	7564	6003734	21	7327	121573	216893	711	48	0	32	1100996
wire	0	41544	12997	39	0	102053	4185	1436	52837	0	8811	4017	215	829	0	0	41367
garden furniture	17	489	16598	1549	0	11693	19231	6475	0	298748	90904	2316	701	7681	25	296	408256
concrete	16526	536584	361910	3795	0	358090	76484	106862	368	60554	17635168	63275	79225	109237	1869	6454	926402
roof	184	1265	71472	0	0	98248	1360	204674	15	908	35853	25643038	184	298	0	4	572147
gravel	0	6786	211390	380	0	246398	1543	2109	270	0	58890	2703	1890211	78646	163	272	138809
soil	660	31272	1561243	1275	0	601669	2084	12366	476	135	127659	4522	246247	2302911	309	3	529926
primeair pattern	211	0	4746	142	0	300	0	0	0	651	2426	3	7	0	132121	64	491
Snow	14	164	53336	0	0	38852	373	966	0	0	7495	1905	194	0	9	334231	11381
unknown	16536	42635	771385	16435	0	1251671	59995	566572	3325	70963	563920	682200	43981	205542	870	13302	14076863

Figure 8. VGG16 Confusion Matrix

Figure 7, the class level Jaccard indices indicates that our model is able to identify many of the large zones both “safe” and “unsafe” very accurately. For instance, we are able to identify “safe” zones like asphalt, grass, concrete, and roof classes all with over 0.7 jaccard index scores which is well above random guessing. We are also able to identify large “unsafe” zones like water, high vegetation, and ground vehicles with around 0.7 jaccard index scores as well. From Figure 7 we see that Gravel and Soil which are generally safe landing zones have relatively low Jaccard Indices meaning we struggle to identify these areas. However, looking at the confusion matrix for these two classes we see that most of the error comes from confusing gravel with soil and grass, and confusing soil with gravel and grass. All three of these classes are considered “safe” and thus the low performance scores are actually not as problematic as they may first appear. However, one major issue we see is that Figure 7 indicates our models are quite poor at identifying humans, and since human safety is our primary concern this is an area that needs to be improved. From the confusion matrix in Figure 8, we see that humans are most frequently misclassified as grass and unknown. Another worrying sign since grass is considered “safe” and unknown does not have a designation. One solution that we plan to implement in further studies is the introduction of a larger penalty for misclassifying human classes when training the models which will hopefully address these issues. One final concern is the extremely small Jaccard index for the wire class as shown by Figure 7. Figure 8 also indicates that wires are often confused with asphalt which is considered a “safe” zone. Wires, however, are clearly not safe and drones hitting wires have the potential to cause serious damage to power grids, drones, and humans alike. Further investigation into methods to improve wire identification will also be a focus in following studies.

Table 3. Model Training Time

	Vgg-16	ResNet18	MobileNetV3_large
Training Time	220 Minutes	155 Minutes	115 Minutes

In Table 3, we observe the following training times for the models: VGG-16 requires 220 minutes, ResNet18 necessitates 155 minutes, and MobileNetV3_large demands a mere 115 minutes. The training time for MobileNetV3_large is 47.8% shorter than that of VGG-16 and 25.8% shorter than ResNet18. This reduction in training time not only leads to decreased project costs but also enhances the model's scalability.

In real-world applications, especially for computer vision tasks, the size of training sets can be extremely large, often containing millions of images. As the data size increases, the discrepancy in training times is expected to grow substantially. MobileNetV3_large, with its lightweight and efficient architecture, is better equipped to handle larger training sets compared to VGG-16 and ResNet18. Consequently, the model's performance is anticipated to improve as it can more effectively adapt to the large-scale datasets often encountered in practical applications.

Taking into account both cost-effectiveness and potential performance, MobileNetV3_large emerges as the most suitable candidate among the three models under consideration for handling large-scale real-world computer vision tasks.

Table 4. Model Size

	Vgg-16	ResNet18	MobileNetV3_large
Model Size	95 MB	57 MB	27MB

From the model size table, we observe that VGG-16 has a size of 95 MB, ResNet18 is 57 MB, and MobileNetV3_large is only 27 MB. Moreover, as reported by Imran Ahmed, Misbah Ahmad, and Gwanggil Jeon (2021), VGG-16 has approximately 138 million parameters, ResNet has 26 million parameters, and MobileNet has a mere 4.2 million parameters to achieve comparable performance. This indicates that MobileNetV3_large is significantly lighter in terms of both size and number of parameters[18].

The advantages of lighter models, such as MobileNetV3_large, manifest themselves in several aspects. With a reduced number of parameters, the model yields faster inference times, which are essential for real-time applications. Furthermore, the energy efficiency of the model is improved, rendering it suitable for battery-operated devices and contributing to overall power savings in large-scale deployments. In addition, the model's lower storage requirements make it compatible with devices that have limited storage capacity and facilitate distribution across networks with bandwidth limitations.

In summary, the lightweight characteristics of MobileNetV3_large offer benefits in terms of speed, energy efficiency, and storage, positioning it as a preferred choice for a wide array of applications where resource constraints and rapid inference are vital factors. These advantages become particularly apparent in the context of drone segmentation.

Firstly, a brief response time is crucial for optimizing landing behavior, as the time available for landing is limited, and swift decision-making is required. Secondly, the small size of the model is essential, given that the average administrative storage is 2 GB and the pilot system typically occupies 1.3 GB. Consequently, the models should be as compact as possible, considering that additional models, such as image processing algorithms for optimizing photos, must also be accommodated. Thirdly, drones have a limited load capacity, and batteries constitute one of their heaviest components. By decreasing computational costs, energy consumption is reduced, resulting in less battery usage, thereby optimizing drone performance. This optimization allows drones used for package delivery to carry heavier payloads, and those employed for photography and videography to have an extended range and flight time.

Conclusions

Problem we are facing is distinguishing safe and unsafe areas to help facilitate smooth and safe landing. To solve this problem we wanted to use innovations in deep learning with semantic segmentation to help drones map out their environment. We used 3 state-of-the-art algorithms with minor modifications and parameter tuning to learn the semantic segmentation problem for our aerial drone footage of residential neighborhoods. One limitation was that our dataset was somewhat small, in order to remedy this we applied several data augmentations. In order to determine how well our model performed at this task we applied unseen test data to the three models and our results are quite promising. Our models had roughly 0.8 multi class weighted jaccard index, 0.4 unweighted multi class jaccard index, and 0.85 binary unweighted jaccard index. With random guessing being around 1/10th of these respective scores, these techniques certainly provide a promising direction for drone environmental awareness and safe landing procedures. Our models are also able to identify large components of the environment very accurately with water, asphalt, grass, ground vehicles, and roofs all having extremely high individual class jaccard indices of almost 0.8. However, there are still certain challenges we face, those being improved identification of humans and wires. These are two of the most important classes to identify from a societal safety perspective. We plan to look into potential remedies with initial ideas involving weighted penalties for these misclassified classes as well as pretraining parts of the model to be familiar with images of humans and wires so that the

learning process is easier. This provides us a clear direction for us to take our research in the continued effort to improve drone safety.

Roles

Since this is a team project, we want to know what your specific contribution was to this project. Provide detail on your individual role. Each team member should clearly articulate an individual role.

Task	Lead	Support
Data preprocessing and augmentation	Danny	
Model test on sample dataset	Zhanyi Lin, Yuanjing Zhu	Pomelo
Model selection and training	Zhanyi Lin	Pomelo, Yuanjing Zhu
Application on different altitudes	Pomelo	Yuanjing Zhu
Visualization and analysis	Danny	Pomelo
Write-up, presentation	all	

References

[1] Gupta, S. G., Ghonge, D., & Jawandhiya, P. M. (2013). Review of unmanned aircraft system (UAS). *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* Volume, 2.
[2] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3431-3440).
[3] Marcu, A., Costea, D., Licaret, V., & Leordeanu, M. (2019). Towards automatic annotation for semantic segmentation in drone videos. *arXiv preprint arXiv*:1910.10026.
[4] Xiao, X., Zhao, Y., Zhang, F., Luo, B., Yu, L., Chen, B., & Yang, C. (2023). BASeg: Boundary aware semantic segmentation for autonomous driving. *Neural Networks*, 157, 460-470.
[5] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference,* Munich, Germany, October 5-9, 2015, Proceedings, Part III 18 (pp. 234-241). Springer International Publishing.
[6] Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, *39*(12), 2481-2495.

[7] Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., ... & Torr, P. H. (2015). Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 1529-1537).
[8] Oktay, O., Schlemper, J., Folgoc, L. L., Lee, M., Heinrich, M., Misawa, K., ... & Rueckert, D. (2018). Attention u-net: Learning where to look for the pancreas. *arXiv preprint arXiv:1804.03999*.
[9] Wilson, R. L. (2014, May). *Ethical issues with use of drone aircraft - IEEE Xplore*. Retrieved April 14, 2023, from https://ieeexplore.ieee.org/document/6893424
[10] Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2881-2890).
[11] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3431-3440).
[12] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference*, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18 (pp. 234-241). Springer International Publishing.
[13] Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, *39*(12), 2481-2495.
[14] Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., ... & Torr, P. H. (2015). Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 1529-1537).
[15] Oktay, O., Schlemper, J., Folgoc, L. L., Lee, M., Heinrich, M., Misawa, K., ... & Rueckert, D. (2018). Attention u-net: Learning where to look for the pancreas. *arXiv preprint arXiv:1804.03999*.
[16] Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2881-2890).
[17] AIcrowd (2023), Semantic Segmentation Dataset, Retrieved from: https://www.aicrowd.com/challenges/scene-understanding-for-autonomous-drone-delivery-suadd-23#
[18] Ahmed, I., Ahmad, M., & Jeon, G. (2021). A real-time efficient object segmentation system based on U-Net using aerial drone images. Journal of Real-Time Image Processing, 18, 1745-1758.

Appendix

To ensure the proper implementation of various pre-trained encoder models, including MobileNet, VGG16, ResNet18, and ViT, we conducted tests on the CIFAR10 dataset to determine their approximate time consumption, computational resource requirements, and prediction accuracy. We ran 100 epochs and found that MobileNetV3, VGG16, and ResNet18 converged after about four hours, while ViT stopped at 68 epochs due to a runtime limit on Kaggle. The training and test loss of the four models are shown in Figure 7.a, with ResNet18 achieving the lowest loss in both sets (0.4154 and 0.2648). It was followed by VGG16 and MobileNet, while ViT had the highest training and test loss. The accuracy plot (Figure 7.b) also

showed the same trend. ResNet18 achieved the highest accuracy on the test set at 0.9119, even higher than the training accuracy, indicating under-fitting. VGG16 had the second-highest accuracy on the test set at 0.9012, with the training set showing a strong tendency towards overfitting (accuracy of 0.981). MobileNet achieved a test set accuracy of 80.6%, but its curve fluctuated significantly in both the loss and accuracy plots in the initial stages of training. This may be due to a high learning rate, inappropriate initialization, or batch size. Finally, ViT had the lowest accuracy on both training and test sets. By testing these models on a sample dataset, we confirmed their effectiveness and are confident in applying them to our drone dataset.

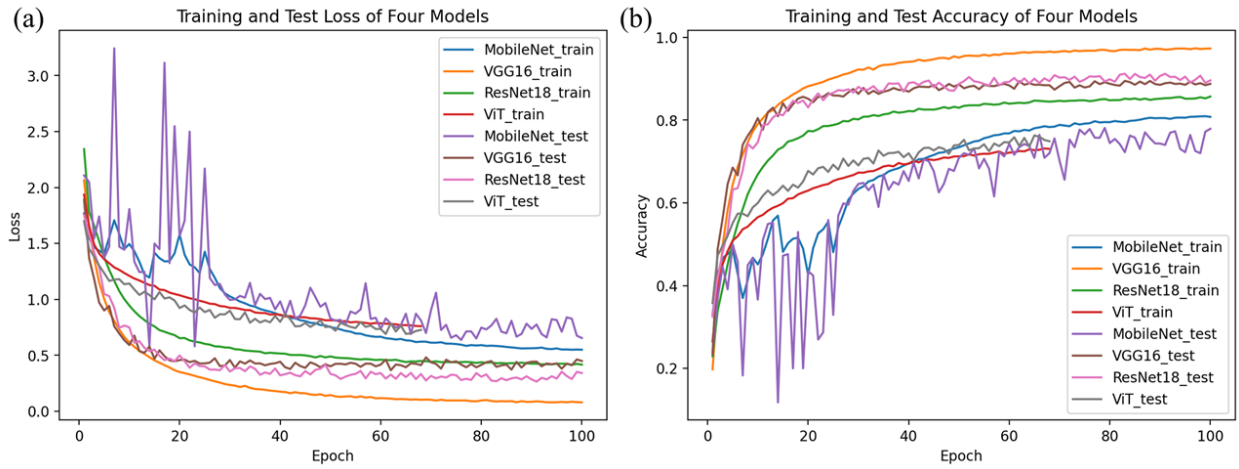

Figure 8. Learning curves of MobileNet, VGG16, ResNet, and ViT models on CIFAR10: (a) Training and test loss, (b) Training and test accuracy.

Table 2. Evaluation metrics of MobileNet, VGG16, ResNet, and ViT models on CIFAR10

	Train Loss	Test Loss	Train Accuracy	Test Accuracy
ViT	0.7598	0.6904	0.7316	0.7628
MobileNet	0.4767	0.5770	0.8346	0.8060
VGG16	0.0577	0.3656	0.9810	0.9012
ResNet18	0.4154	0.2648	0.8568	0.9119

ResNet18 Multi Class Confusion Matrix

	water	asphalt	grass	human	animal	high vegitation	ground vehicle	façade	wire	garden furniture	concrete	roof	gravel	soil	primeair pattern	Snow	unknown
water	470514	558	83	0	0	696	497	0	0	765	11991	16	690	0	0	203	800
asphalt	7218	9055292	144522	29	0	110233	46404	3872	6280	0	219224	6151	1382	26007	0	65	28940
grass	22241	83870	41643213	14975	0	1353337	10027	32747	353	87	361769	55602	116907	922406	6483	33670	776355
human	28	2032	40232	52610	0	7383	11571	612	1	9964	28256	24	1126	2808	1008	515	118467
animal	0	0	826	418	0	599	0	0	0	147	631	0	0	0	0	207	1725
high vegitation	1589	85873	1596284	1089	0	35074626	21560	62178	1589	1819	251970	104163	92309	221539	0	7411	805657
ground vehicle	632	26779	4776	502	0	24039	2452026	14492	0	44	58095	1474	384	3153	0	199	100029
façade	612	6525	24193	38	0	99361	5305	6146130	20	7259	143304	239571	483	1706	0	132	999838
wire	0	47927	14406	2	0	137254	3793	3191	8628	0	8937	3357	229	4806	0	0	41588
garden furniture	61	299	25616	12361	0	13012	18373	6558	0	332198	112765	3096	752	3331	562	2237	325983
concrete	20406	364063	374306	641	0	291514	66541	101397	73	60944	18025710	56843	37456	124535	2356	3811	806187
roof	972	5410	47086	0	0	90425	710	234150	0	0	49580	25660790	256	60	0	54	525035
gravel	41	7220	244313	55	0	190891	264	1181	294	0	70551	2213	1296192	740348	148	2088	145012
soil	132	24853	1306188	917	0	542791	2078	6456	401	260	150268	8290	137340	2795693	100	1195	513413
primeair pattern	53	26	5607	223	0	36	112	0	0	31	2719	0	54	3	133687	127	1591
Snow	28	152	35315	2	0	10198	505	166	0	28	6360	611	1486	0	6	373343	14131
unknown	14803	34234	788811	15908	0	1064625	57173	595339	3340	69037	689549	666518	45450	302762	1218	27147	14123854

ResNet18 Class Level Jaccard Index

water	asphalt	grass	human	animal	high vegitation	ground vehicle	façade	wire	garden furniture	concrete	roof	gravel	soil	primeair pattern	Snow	unknown
0.75	0.79	0.79	0.12	0.00	0.78	0.77	0.61	0.01	0.19	0.76	0.89	0.26	0.26	0.80	0.48	0.54

MobileNet Multi Class Confusion Matrix

	water	asphalt	grass	human	animal	high vegitation	ground vehicle	façade	wire	garden furniture	concrete	roof	gravel	soil	primeair pattern	Snow	unknown
water	454452	2093	2199	31	0	1776	2350	0	0	0	6908	0	0	1832	0	0	2978
asphalt	3470	10798647	83540	281	0	100336	52983	1191	16	0	283680	6781	143	41552	115	3478	11078
grass	1911	82919	41600134	26531	0	1617474	8280	9716	0	606	252094	45502	54333	743970	9289	138182	612591
human	1819	2541	41117	112279	0	11285	8122	572	0	11135	27705	61	2148	4949	0	2133	91088
animal	0	10	829	14	0	91	21	0	0	227	876	0	0	1	0	0	916
high vegitation	2996	64635	1124130	451	0	35863410	6827	49495	0	841	209438	49037	84857	208732	0	44104	592749
ground vehicle	600	32145	9372	134	0	16379	2477684	2851	0	2397	43898	672	1615	2568	0	894	70009
façade	0	1396	10458	5	0	128401	4710	6111337	0	3363	83243	200781	183	65	0	439	803843
wire	65	91166	17794	0	0	61536	2491	3498	21	409	9752	5787	94	994	0	0	15833
garden furniture	214	111	11389	10053	0	16981	4486	4279	0	358239	102813	8190	330	3434	49	3922	308413
concrete	32617	200459	415710	8189	0	346099	99201	95924	0	79084	21774324	60595	32763	56812	8401	11110	771161
roof	10	4151	24441	67	0	107453	1832	224450	0	22	43946	21409249	116	929	0	3073	452723
gravel	615	1769	158729	941	0	242677	1764	1602	0	0	79772	1053	1803812	350441	2	6302	133836
soil	3756	26250	1245364	1317	0	497241	2504	6928	0	998	155689	4453	681704	2651226	108	3068	421677
primeair pattern	0	30	13987	972	0	19219	0	106	0	2356	8193	62	3377	2910	141703	11557	11813
Snow	246	874	3345	353	0	23846	407	1043	0	1717	6217	1280	1210	1648	392	547135	36530
unknown	3847	78412	717191	32713	0	1036305	46921	578138	0	81239	613839	557535	34762	247246	897	13437	12790797

MobileNet Class Level Jaccard Index

water	asphalt	grass	human	animal	high vegitation	ground vehicle	façade	wire	garden furniture	concrete	roof	gravel	soil	primeair pattern	Snow	unknown
0.72	0.82	0.75	0.21	0.00	0.79	0.77	0.60	0.05	0.23	0.76	0.92	0.18	0.13	0.89	0.42	0.53

Table 3. Model Training Time

	Vgg-16	ResNet18	MobileNetV3_large
Training Time	220 Minutes	155 Minutes	115 Minutes

Table 4. Model Size

	Vgg-16	ResNet18	MobileNetV3_large
Model Size	95 MB	57 MB	27MB