AERIAL IMAGE SEGMENTATION FOR VISION-BASED AUTONOMOUS DRONE FLIGHT AND LANDING ACTIVITIES

OJEKANMI, OLUWATOBI OLAMIDE

PROJECT SUMMARY

Background: The usability and performance of an autonomous drone, also known as unmanned aerial vehicle (UAV) are determined by its ability to fly, navigate and land precisely. Current self-navigation systems in most low-altitude autonomous drones rely heavily on the global positioning system (GPS) and sensor information which are of low intelligence and are prone to errors due to signal failures. This can be attributed to the fact that our environment is complex, dynamic, and was designed for human use; hence, they can sometimes be difficult to process without vision. This shortcoming has subsequently amplified the use of computer vision techniques for full drone navigation and landing autonomy. Therefore, this project aims to segment real life aerial images collected by UAVs using state-of-art deep learning image segmentation architectures. This project could serve as a basis for more advanced research work in autonomous drone vision.

Method: 3,269 pre-labelled aerial images shot at an altitude of about 5 to 50 meters above ground were used to carry out the experiments. The DeepLabV3+ and U-Net segmentation architectures were initially trained and evaluated using the MobileNetV2 model as their encoder/backbone. The model with the superior performance (highest mean intersection-over-union (mIoU)) was then re-trained using additional four encoder/backbone models (DenseNet121, EffecientNetB0, ResNet50 and VGG19) to identify whether the superior computational effectiveness of the MobileNetV2 architecture is also performance-wise justifiable. The segmentation models were trained to classify the aerial images' pixels into 12 classes — person, bike, car, drone, boat, animal, obstacle, construction, vegetation, road, sky and others/background.

Results: The mean intersection-over-union (mIoU) was used as the optimizing metric, while other factors such as number of model parameters, pixel accuracy, precision, and f1-score were used as satisficing metrics. During the initial training, the U-Net emerged with the highest mIoU of 45% compared to the 37% obtained from the DeepLabV3+ model after training both models for 50 epochs. Also, the mIoU obtained from re-training the U-Net model using additional encoder models are 64%, 57%, 64%, and 62% for DenseNet121, EffecientNetB0, ResNet50, and VGG19 respectively. Overall, using the DenseNet121 with U-Net proved to be the best model after 50 epochs because of superior performance and fair parameters size and could be deep-trained and modified to improve performance.

Conclusion: From the results obtained, the U-Net segmentation model seemed to edge the DeepLabV3+ segmentation model while the DenseNet121 classification model seems to outperform other select encoder models as U-Net encoders. However, this cannot be totally relied on because I only ran few epochs because Google Colaboratory only has limited GPU-size and regulated duration. Similarly, improving the model performance to identify and classify all labels properly irrespective of their pixel size using class weighting wasn't carried out during this project because of the same Google Colaboratory limited and capped GPU-size. Overall, finding additional dataset, deep-training and class weighting would be a standard place to improve model performance, while deploying model to test UAV performance in classifying and making decisions would be a great place to continue this project.

Keywords: Drone Vision, Computer vision, Unmanned Aerial Vehicle (UAV), Deep Learning, Image Segmentation, Semantic Segmentation

1. Introduction

The usability and performance of an autonomous drone, also known as unmanned aerial vehicle (UAV) are determined by its ability to fly, navigate and land precisely. Conventionally, drones are piloted by an operator who controls the robot remotely from a base. However, significant progress has been made to make the navigation of these devices autonomous. The current outdoor autonomous navigation systems rely heavily on the Inertial Measurement Units (IMU) and global positioning system (GPS) [3-5] to aid drone autonomous navigation along predefined paths. However, GPS-dependent systems have a lot of weaknesses. Being a radio signal, GPS is prone to error due to signal outages or signal interference from different sources; hence, prompting the need to develop more robust autonomous navigation systems.

To solve the challenges posed by the GPS for safe and complete UAV and vehicle autonomous navigation, many solutions have been put forward in the recent years. Most of these solutions however rely heavily on the Simultaneous Localization and Mapping (SLAM) technique [6 - 8] to create 3D maps of unknown environments using laser range finders (LIDARs) [9], RGB-D sensors [10], or stereo vision [11], and in turn use these maps to find the relative position of the UAV at any instant of time. However, this process is a computationally expensive, and hence fails in most of the real-time scenarios. Besides, the SLAM technique is of low-intelligence, and not sophisticated enough to aid safe autonomous navigation and interactions with people and dynamic and complex environments like urban areas at low altitudes. A simple example would be a delivery drone trying to land in an urban area. While its sensors can easily map the environment and detect the presence of cars, people, plastic bags, rocks and other surrounding objects. The drone's inability to classify static and dynamic objects and Its inability to distinguish the significance of each object could be costly. The sudden movement of a plastic bag towards the drone could force the drone to hover or change its path drastically. This could inadvertently expose the drone, properties or human lives to potential danger. Thus, to achieve complete and safe autonomy, it is essential for drones to fully understand their environments, distinguish between static and dynamic objects, and understand object significance before planning their travel paths. A common way to achieve this goal is for drones to perceive their environment through vision [12].

The object detection and image segmentation are two popular approaches that can be explored to aid vision navigation and landing of UAVs. The object detection aims to localize and classify recognizable objects in an image or video clip, while the image segmentation technique aims to classify each pixel of an image or video into a class/label. However, the focus of this project will be centered on using the semantic segmentation approach (an image segmentation technique that groups pixels of similar objects together) to evaluate how well aerial images can be classified. Ability to achieve excellent object identification and classification performance could help UAVs better understand their environment.

2. Data Preparation

2.1. Data Collection

The dataset for this study were collected and labelled by Ishan Nigam, Chen Huang, and Deva Ramanan for their "Ensemble Knowledge Transfer for Semantic Segmentation" research project. The images which were acquired by a drone at an altitude of about 5 to 50 meters above ground have a resolution of about 720 pixels. 3269 images from the dataset were preprocessed before using them to train our models. The images were labelled to have 12 classes — person, bike, car, drone, boat, animal, obstacle, construction/ building, vegetation, road, sky and others/background.

2.2. Data Preprocessing

2.2.1. Train/Validation/Test Dataset Split

I distributed the images and their segmentation labels from the dataset as 80% for training, while the other 20% was split in the ratio of 60:40 for validation and testing respectively.

2.2.2. Data Loading Pipeline

I used the **tensorflow.data.Dataset** API to create pipelines to load our images and their segmentation labels (masks). The API allows us to build an asynchronous, highly optimized data pipeline to prevent our GPU from data starvation. It loads data from the disk (images and masks), applies optimized transformations, creates batches and sends it to the GPU. Unlike former data pipelines made the GPU, the Dataset API wait for the CPU to load the data, leading to performance issues. The following variables were used to load our dataset (images and their segmentation labels) during training:

Batch size = 32

Buffer size = 5120

Image size = (128,128,3)

Shuffle dataset = True

Cache dataset = True

2.2.3. Data (image and mask) resizing

I resized the images and their segmentation labels (masks) to have the shape of (128,128,3). I used this small size because of the limited memory I worked with.

2.2.4. Data Augmentation

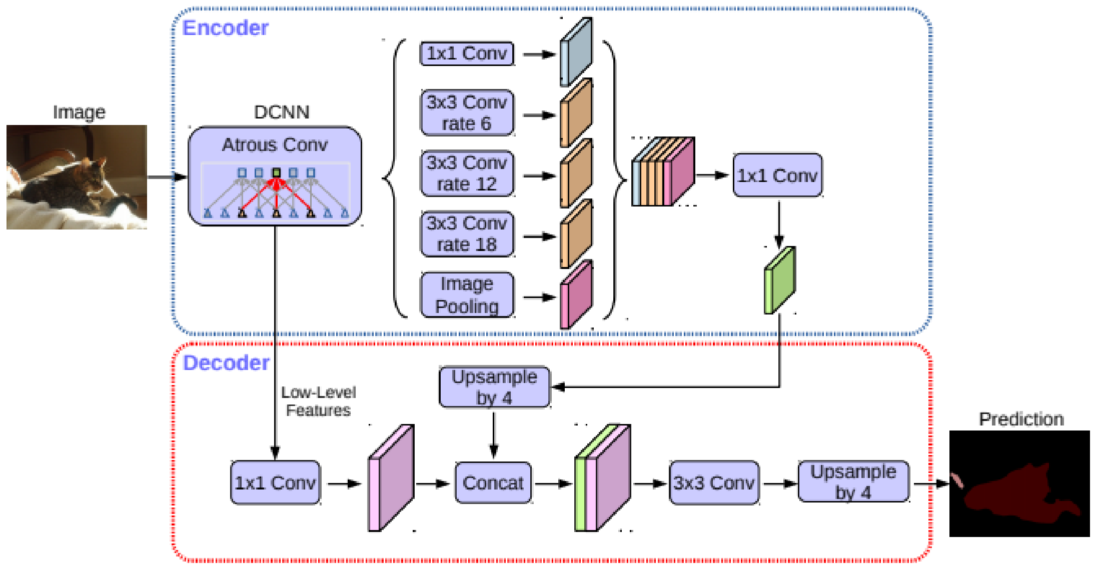
I used the Random Rotate transformation technique to augment the training dataset (training images and their masks) using the TensorFlow ImageDataGenerator Class function. Both the images and their segmentation labels (ground-truth masks) were transformed at the same time and with the same technique to avoid data disparity.

3. Initial Training: Image Segmentation Model Selection, Training, and Evaluation

3.1. Image Segmentation Model Selection

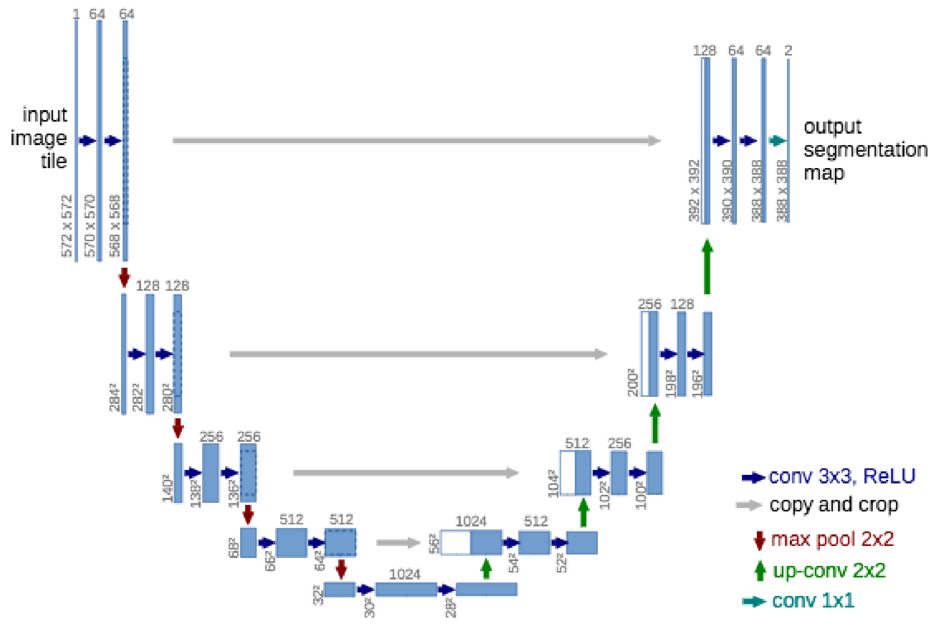
The DeepLabV3+ and U-Net models were initially trained and evaluated to identify the model with the superior performance (model with the highest mean intersection-over-union).

A. **DeepLabV3+**: The DeepLabv3+ is a semantic segmentation architecture created by Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam in 2018 [13]. The model improves upon DeepLabv3 with several improvements, such as adding a simple yet effective decoder module to achieve an encoder-decoder structure. The encoder module processes multiscale contextual information by applying dilated convolution at multiple scales, while the decoder module refines the segmentation results along object boundaries.



DeepLabv3+ framework: an encoder with a backbone CNN and an ASPP produces feature representations to feed a decoder with 3x3 convolutions producing the final predicted image. Source: L.-C. Chen et al. (2018)

B. **U-Net:** The U-Net model was created by Olaf Ronneberger, Philipp Fischer, and Thomas Brox for biomedical image segmentation tasks in 2015 [14]. The model which is now a very popular choice for most semantic segmentation tasks across all domains was named for its U-shape architecture. The U-Net builds on a previous architecture called the Fully Convolutional Network, or FCN, which replaces the dense layers found in a typical CNN with a transposed convolution layer that upsamples the feature map back to the size of the original input image, while preserving the spatial information. This is necessary because the dense layers destroy spatial information (the "where" of the image), which is an essential part of image segmentation tasks. An added bonus of using transpose convolutions is that the input size no longer needs to be fixed, as it does when dense layers are used. The first half of the model which is the downsampling or contracting part has an FCN-like architecture extracting features with 3x3 convolutions, while the other half or the upsampling or expanding part uses up-convolution (or deconvolution) reducing the number of feature maps while increasing their height and width.



Architecture of the U-net for a given input image. The blue boxes correspond to feature maps blocks with their denoted shapes. The white boxes correspond to the copied and cropped feature maps.
Source: [O. Ronneberger et al. (2015)]

3.2. Model Modification

Since I have a relatively small dataset, there's a high probability that our model may struggle to learn a lot of important features or overfit to the dataset. Hence, in addition to data augmentation, I modified both segmentation models to have the MobileNetV2 model as their encoder (downsampling/contracting part of their architectures). Also, I used the pretrained 'imagenet' weights of the MobileNetV2 model as the starting encoder parameters.

3.3. Hyperparameters Selection & Tuning

The following Hyperparameters were used to train our models.

- optimizer = Adam (with default beta_1 and beta_2 values i.e., beta_1=0.9, beta_2=0.999)
- loss = sparse_categorical_crossentropy
- learning rate = 0.001 (the ReduceLROnPlateau function was used with the callback function to reduce the learning rate by a factor of 0.1 if there's no improvement in the previous 10 'validation_accuracy' epoch values. The minimum achievable learning rate was set to be 0.000005)
- batch size = 64
- num epochs = 50
- dropout = 0

3.4. Model Training

The Google Colab was used to carry our experiments. The platform which was equipped with 12GB GPU was used to preprocess and train our model at an average time of 2 hours per model.

3.5. Model Evaluation

To measure and compare the quantitative performance of the select segmentation techniques, different evaluation measures such as precision, sensitivity/recall, dice coefficient/F1-score, and Intersection over Union (IoU)/Jaccard similarity (JS) were measured. These metrics were computed by identifying the variables true positive (TP), true negative (TN), false positive (FP), and false-negative (FN) by calculating the confusion matrix between the predicted segmentations and the ground truth segmentations. The expressions for accuracy, precision, recall, TDR, and IoU are defined as:

$$Precision = \frac{TP}{(TP + FP)}$$

$$Recall = \frac{TP}{(TP + FN)}$$

$$IoU = \frac{TP}{(TP + FP + FN)}$$

$$F1\_Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

The intersection over Union (IoU) for each class/label were computed and their average were computed as mean IoU.

3.6. Results and Discussion

Table 1: Overall evaluations of the segmentation models.

Image Segmentation Model	**Encoder Parameters**	**Model Parameters**	**Overall IoU (%)**	**Overall Precision (%)**	**Overall Recall (%)**	**Overall F1Score (%)**
DeepLabV3+	2,257,984	7,626,819	53	70	70	70
U-Net	2,257,984	16,198,531	81	90	90	90

The overall evaluations were computed by computing the true positive (TP), true negative (TN), false positive (FP), and false-negative (FN) across all categories and summing respective variable before computing each metric.

Table 2: Mean evaluations of the segmentation models.

Image Segmentation Model	**Encoder Parameters**	**Model Parameters**	**Overall IoU (%)**	**Overall Precision (%)**	**Overall Recall (%)**	**Overall F1Score (%)**

DeepLabV3+	2,257,984	7,626,819	37	78	43	55
U-Net	2,257,984	16,198,531	45	71	55	62

The average/mean evaluations were computed by computing the true positive (TP), true negative (TN), false positive (FP), and false-negative (FN) across all categories and also computing each metric for each class using these values. The metric values for the classes are summed and their averages were taken.

The result obtained showed that the U-Net architecture achieved higher overall and mean values across all metrics. Hence, I proceeded on finding a more suitable encoder model for the U-Net segmentation model before deep-training the segmentation model with the best computation-efficient encoder model.

4. U-Net Retraining: Encoder Model Selection, Training & Evaluation

4.1. Encoder Model Selection

The MobileNetV2 was initially used to train both the DeepLabV3+ and U-Net architecture for aerial image segmentation model. This choice was governed by the model's excellent computation cost (lower number of parameters compared to other classification models). However, to ensure that this lower number of parameters does not affect performance significantly, I modified and trained the U-Net segmentation model to have additional four classification models as its encoder models. These classification models are:

I. DenseNet121
II. EffecientNetB0
III. ResNet50
IV. VGG19

Similarly, I fine-tuned the pre-trained 'imagenet' weights of each model while training my segmentation models.

4.2. Hyperparameters Selection & Tuning

I used the same parameters for training the U-Net segmentation model with MobileNetV2 encoder to train the U-Net segmentation model with other encoder models

4.3. Model Training

Similarly, I used the Google Colab to carry our additional experiments.

4.4. Model Evaluation

The precision, recall, intersection-over-union (IoU) and F1 Score were used as evaluation metrics.

4.5. Results and Discussion

Table 3: Overall evaluations of the encoder models.

Model Encoder	**Encoder Parameters**	**Model Parameters**	**Overall IoU**	**Overall Precision**	**Overall Recall**	**Overall F1Score**

DenseNet121	7,037,504	38,561,123	92	96	96	96
EfficientNetB0	4,049,571	38,669,766	86	92	92	92
MobileNetV2	2,257,984	36,325,219	81	90	90	90
ResNet50	23,587,712	67,645,091	91	96	96	96
VGG19	20,024,384	47,621,123	91	95	95	95

The overall evaluations were computed by computing the true positive (TP), true negative (TN), false positive (FP), and false-negative (FN) across all categories and summing respective variable before computing each metric.

Table 4: Mean/average evaluations of the encoder models.

Model Encoder	**Encoder Parameters**	**Model Parameters**	**Mean IoU (%)**	**Average Precision (%)**	**Average Recall (%)**	**Average F1-Score (%)**
DenseNet121	7,037,504	19,614,083	64	80	71	75
EfficientNetB0	4,049,571	18,543,078	57	81	65	72
MobileNetV2	2,257,984	16,198,531	45	71	55	62
ResNet50	23,587,712	43,979,459	64	79	72	75
VGG19	20,024,384	31,033,379	62	79	70	74

The average/mean evaluations were computed by computing the true positive (TP), true negative (TN), false positive (FP), and false-negative (FN) across all categories and also computing each metric for each class using these values. The metric values for the classes are summed and their averages were taken.

The result obtained showed that despite have a fairly good overall values across all metrics, the mean values are average. This disaggregation in reports can be attributed to disparity in the pixel values of each class. For example, the road constitutes a larger portion of our images compared to the person pixels; hence, the road has more pixels to learn from compared to person. Thus, a class-wise evaluation revealed that our model successfully learned to classify the road better than people (something the overall evaluation didn't reveal because the person pixels are highly insignificant compared to the road and other classes pixels). The class weighting or oversampling are the two popular techniques that could be used to ensure these models learn to classify all labels/classes equally. (Free Google Colab is extremely frustrating, so I didn't attempt using any of these techniques). Also, collecting more data to increase the number of pixels of the dwarf classes would be a plus in improving model performance in classifying that class/label.

5. PROJECT LIMITATIONS

1. Lack of high-end computing platforms to aid longer training of our model
2. Lack of sufficient dataset to search for a well-defined architecture for aerial image segmentation tasks
3. Inability to merge similar datasets because of lack of sufficient information on the available datasets

6. CONCLUSION

The U-Net Architecture performed decently across the selected encoder model architectures. However, the performance of these models cannot be relied upon based on the following reasons:

1. The inability of the system to effectively classify all labels/classes properly could be costly and should be avoided at all cost
2. Understanding how these models classify each segmentation class isn't clear and can be assumed to be 'guess-work' or the model has been overfit to that dataset.

Future projects can leverage the use of class weighting to improve model performance and explainable AI tools (if any) to understand how the model makes its decisions. However, whether all these are sufficient to build a solid object detection system for autonomous drone navigation and landing is what I'm skeptical of. While I would be exploring to deploy these codes on a drone controller, I would be interested in hearing out other people's opinion. So kindly let me know your thoughts if you have any.

Cheers and have a wonderful day!!

REFERENCES

[1] Giones, F.; Brem, A. From toys to tools: The co-evolution of technological and entrepreneurial developments in the drone industry. Bus. Horiz. 2017, 60, 875–884. [CrossRef]

[2] The Drone Market Report 2020–2025; Technical Report; Drone Industry Insight: 202

[3] Abbott, E., Powell, D., 1999. Land vehicle navigation using GPS. Proceedings of the IEEE87, 145–162.

[4] Hernandez, A., Copot, C., De Keyser, R., Vlas, T., and Nascu I. Identification and path following control of an ar. drone quadrotor. In System Theory, Control and Computing (ICSTCC), 2013 17th International Conference, pages 583–588. IEEE, 2013.

[5] Santana, L. ¸Brandao, A., Sarcinelli-Filho, M., and Carelli, R. A trajectory tracking and 3d positioning controller for the ar. drone quadrotor. In Unmanned Aircraft Systems (ICUAS), 2014 International Conference on, pages 756–767. IEEE, 2014.

[6] Checchin, P., Ge´rossier, F., Blanc, C., Chapuis, R., Trassoudaine, L., 201 Radar scan matching slam using the fourier-mellin transform, in: Field and Service Robotics, Springer. pp. 151–161.

[7] Engel, J., Scho¨ps, T., Cremers, D., 2014. Lsd-slam: Large-scale direct monocular slam, in: European Conference on Computer Vision, Springer. pp. 834–849.

[8] Sibley, G., Mei, C., Newman, P., Reid, I., 201 A system for large-scale mapping in constant-time using stereo. International Journal of Robotics Research.

[9] Ramasamy, S., Sabatini, R., Gardi, A., Liu, J., 2016. LIDAR obstacle warning and avoidance system for unmanned aerial vehicle sense-and- avoid. Aerospace Science and Technology 55, 344–358.Aa

[10] Huang, A., Bachrach, A., Henry, P., Krainin, M., Maturana, D., Fox, D., Roy, N., 2017a. Visual odometry and mapping for autonomous flight using an rgb-d camera, in: Robotics Research. Springer, pp. 235–252.

[11] McGuire, K., de Croon, G., De Wagter, C., Tuyls, K., Kappen, H., 2017. Efficient optical flow and stereo vision for velocity estimation and obstacle avoidance on an autonomous pocket drone. IEEE Robotics and Automation Letters 2, 1070–1076.

[12] Boloor, A., Garimella, K., He, X., Gill, C., Vorobeychik, Y., and Zhang, X. "Attacking vision-based perception in end-to-end autonomous driving models". Journal of Systems Architecture, p. 101766, 202

[13] Liang-Chieh C., Yukun Z., George P., Florian S., and Hartwig A. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation". 2018.

[14] Ronneberger, O., Fischer, P., and Brox¸T. "U-Net: Convolutional Networks for Biomedical Image Segmentation". 2015.