Application of a U-Net Segmentation Model in Land Cover Classification for Use in Automated Data Prefiltering Onboard Nanosatellites

Ramiel Deticio
Gokongwei College of Engineering
De La Salle University
Manila, Philippines
ramiel_g_deticio@dlsu.edu.ph

Argel Bandala
Gokongwei College of Engineering
De La Salle University
Manila, Philippines
argel.bandala@dlsu.edu.ph

John Anthony Jose
Gokongwei College of Engineering
De La Salle University
Manila, Philippines
john.anthony.jose@dlsu.edu.ph

Ronnie Concepcion II
Gokongwei College of Engineering
De La Salle University
Manila, Philippines
ronnie.concepcion@dlsu.edu.ph

Mark Angelo Purio
College of Engineering
Adamson University
Manila, Philippines
markangelo.purio@adamson.edu.ph

Edwin Sybingco
Gokongwei College of Engineering
De La Salle University
Manila, Philippines
edwin.sybingco@dlsu.edu.ph

Richard Josiah Tan Ai
Gokongwei College of Engineering
De La Salle University
Manila, Philippines
richard.tanai@dlsu.edu.ph

***Abstract*—The limited physical constraints of nanosatellites due to their size, hinders their ability to transmit large amounts of image data. Because of this, the use of machine learning methods to filter data onboard has become more prominent to increase the bandwidth efficiency of these devices. By having an AI-based classification system for the images, the bandwidth necessary to transmit all these images and the tradeoff when it comes to storage, can potentially be offloaded through having a system which generates metadata that can indicate the data samples which offer the most usability, thus freeing up more space and bandwidth for these more important samples.**

This study explores the task of land cover classification, by utilizing one of the more prominent image segmentation models, U-Net. The model is implemented and evaluated using Pytorch using the DeepGlobe 2018 land cover classification dataset, achieving an average class IoU score of 0.68. This study seeks to support the viability of such a solution and is intended to support any future work which seeks to implement a fully automated data prefiltering system for satellite imagery.

***Index Terms*—computer vision, edge computing, land cover classification, nanosatellite**

I. Introduction

Remote sensing can be expensive, especially when it involves the remote transmission of any data it needs to process. For example, an observation satellite trying to transmit image data from orbit would require a considerable and limited amount of resources to maintain, but also to work concerning its intended objective [1].

In the field of data gathering, it is important to know which data would provide the most amount of value in a specific project. For example, in a lot of computer vision applications, it takes a large amount of data to train and evaluate models [2], data which takes a significant amount of time and resources to annotate and preprocess before it is ready for use. Therefore, people have taken to developing methods that involve the identification of which data samples would provide the most amount of information, mostly through the compilation of diverse data instead of trying to approach it through quantity.

When it comes to satellite operations, the efficient allocation of the platform's limited resources is important to manage the equipment that is onboard and to consistently keep them operational as much as possible [3]. When it comes to orbital transmissions, the distance that is needed to transmit signals between a satellite and a ground station is often considerably more than transmissions between stations on the Earth's surface, and thus, could take more power. Therefore, it is worth exploring the idea of the onboard processing of data if it means that the power spent processing data and transmitting processed data is less than the power it would take to transmit all the raw data that was collected by the satellite .

The problem we aim to solve is to create the potential for an autonomous data collection system that could discriminate between data files and accurately judge their usability. In this paper, we specifically look into satellite image data to accurately determine land cover and help produce land image data that would contain a substantial amount of agricultural land.

The main contributions of this paper are as follows:

- The preprocessing of existing satellite data for land cover classification.
- The retraining of a UNet-based model for the specific purpose of land cover classification.
- The evaluation of this model on the DeepGlobe 2018

979-8-3503-0219-6/23/$31.00 ©2023 IEEE

challenge dataset on land cover classification.

II. Review of Related Literature

A. Convolutional neural networks

Convolutional neural networks are one of the most prominent systems used in deep learning for various computer vision tasks [4]. Neural networks were modeled from the way that a biological neural network processes information, resembling multiple layers of interconnected neurons. CNNs improve upon this structure by involving multiple layers of convolutional filters. This application improved upon fully connected networks, like multilayer perceptrons, by making use of local connections to be more computationally efficient in feature analysis.

Since the introduction of classic CNNs, such as LeNet [5], AlexNet [2] and VGG [6], CNNs have undergone a long series of developments, with the general trend of making deeper and more complex network to achieve higher accuracies. Although, some more recent developments focused more on its compression and resource efficiency, to the point that CNNs were slowly becoming more accessible for use in applications which require lightweight processing. The introduction of MobileNets in particular showed the capability of neural networks for compression enough to be justified for use in mobile and edge applications [7]. Another such example is the development of EfficientNets, which improved upon the ability to scale networks effectively [8].

B. CNNs in image segmentation

The introduction of fully convolutional networks was an attempt to use convolutional networks to the task of image segmentation [9]. This involved the replacement of the classifier head of a CNN with additional convolutional and upsampling layers in order to transform the final output from a single probability value/vector into a classification heatmap. After this, segmentation networks were derived from this concept until it created a more standard structure which would contain a CNN encoder and a corresponding decoder. Such architectures which would employ such a method include DeconvNet [10], SegNet [11] and U-Net [12].

C. Orbital edge computing

In the process of scaling down satellites, the physical constraints of nanosatellites have made data transmission more difficult [3]. In addition to other factors, such as limits on power consumption and ground station availability, other methods need to be considered for satellite communication. Akin to modern sensor systems, local processing instead of transmitting data for centralized processing is considered. While such methods might yield faster data processing, the capability of a system to work in such conditions is limited by the capacity of its network [1]. The justification for orbital edge computing is necessary to accommodate the limited downlinking capabilities of smaller satellites.

One example of an attempt at remote processing is the IPEX CubeSat mission featuring TextureCam, utilizing a random forest classifier and decision tree model ensemble for classifying pixels in captured images into a few set classes [13]. It also involved the salience analysis of images, which used a pixel-based algorithm to assign a salience score for individual pixels.

D. CNN-based classification for satellite imagery

Besides the IPEX mission, one of the more popular tasks to arise in onboard processing was the use of these machine learning algorithms to detect clouds in any collected image data. The proposed networks of MobU-Net and MobDeconv-Net involved the adaptation of conventional segmentation networks for use in edge computing in CubeSats [14]. Another such example was the CloudScout project which involved the use of the Eyes of Things (EoT) board and a Myriad 2 Vision Processing Unit to run the CloudScout CNN, a cloud detection network designed to assess hyperspectral images for cloud coverage [15].

An Image Classification Unit (ICU) was designed as an onboard processing circuit for CubeSats [16]. The circuit involved the use of an STM32 microcontroller to run a model based on U-Net, which was trained on Landsat 8 images. The U-Net model is tasked to run a cloud detection task, which was evaluated using the SPARCS cloud assessment dataset.

III. Methodology

This section will explain the implementation of the U-Net model for use in land cover classification of satellite imagery.

A. Dataset

This study implements a sample from the dataset in the DeepGlobe 2018 Challenge [17]. The sample involves 803 satellite images containing eight classes of terrain. The dataset has a 90/10 train/validation split. Each image has a corresponding segmentation mask classifying each pixel to one of seven types of terrain, including: urban, agriculture, rangeland, forest, water, barren and unknown. Figure 1 shows three samples from the dataset with a satellite image and its accompanying mask. The mask features a segmented image with the different terrain types in the image highlighted by a difference in color.

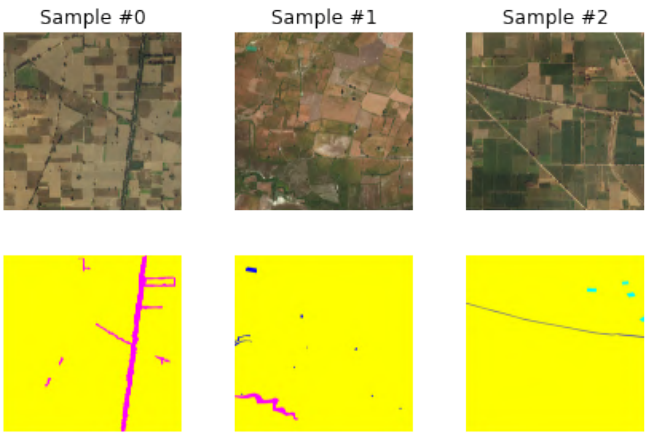

Fig. 1. Samples from the land cover classification dataset in the DeepGlobe 2018 challenge.

B. Model Comparison and Selection

This study focuses on the use of the U-Net segmentation model [12] for this image segmentation task. U-Net has been used in previous literature for similar nanosatellite applications [14], [16], supporting the viability of the model, and this was the main reason that it was used for this specific

application, which is land cover classification. Another reason that this was chosen was due to its ease of implementation, since it was available in the Segmentation Models for Pytorch library [18], which was used in this study.

Although, other models were also put into consideration for potential future versions of the application. One of the two that was considered was SegNet [11], which shared a lot of similarities with the U-Net encoder-decoder structure. Another one was the DeepLab system of segmentation models [19], which offered another alternative with its use of atrous convolution. As part of the scope of the study, the experiments were still eventually limited to using U-Net for the application.

C. Model Architecture and Training

The specific version of the model used is a U-Net segmentation model with a base encoder model of VGG16, which has been pretrained on ImageNet [12]. Figure 2 shows a diagram of the U-Net segmentation architecture. It is characterized by its encoder-decoder structure, where the first component uses convolution and downsample the image, and the second component uses deconvolution and upsamples the features into an output mask.

The model was implemented in PyTorch using the Segmentation Models library for PyTorch [18]. The images were cropped to a size of 1024x1024 and the training images were augmented with horizontal and vertical flips. The model is trained over 5 epochs using Adam as an optimizer.

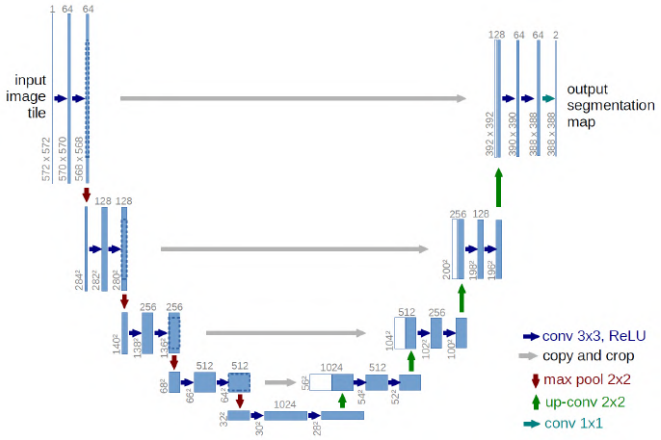

Fig. 2. Diagram of the U-Net architecture.

Dice loss, which can be seen in Eq. 1, is used as the training loss function [20]. Dice loss is equivalent to the F-score between the ground truth and the predicted result.

$$DiceLoss\,(y, \overline{p}) = 1 - \frac{(2y\overline{p} + 1)}{(y + \overline{p} + 1)} \tag{1}$$

IV. Evaluation

To evaluate the model, the Jaccard index [17], [21] will be measured on the validation dataset. The Jaccard index, or IoU, will be used as the accuracy metric for the predicted results. It is a common metric used in segmentation tasks, and is defined as the ratio between the total amount of true positive values over the sum of the true positive, false positive, and false negative values. In an image segmentation task, this is defined as the overlapping area of the predicted and ground truth masks over the total area covered by the predicted and ground truth masks. Evaluation of the created model would involve the use of a pixel-wise Jaccard index (Intersection over Union), which can be seen in Eq. 2,

$$IoU_j = \frac{\sum_{i=1}^{n} TP_{ij}}{\sum_{i=1}^{n} TP_{ij} + \sum_{i=1}^{n} FP_{ij} + \sum_{i=1}^{n} FN_{ij}}, \tag{2}$$

where IoU_j refers to the IoU score for each pixel belonging to class j to a total number of n images. TP_{ij}, FP_{ij}, and FN_{ij} refers to every true positive, false positive and false negative pixels of class j in each image i [17]. The final score is the average IoU among classes, which is expressed in Eq. 3,

$$mIoU = \frac{1}{k} \sum_{j=1}^{k} IoU_j, \tag{3}$$

where k is the total number of classes [17].

V. Results and Discussion

The performance on the U-Net model is shown in this study using the validation set. The model was trained over five epochs and resulted in having a best result after reaching a minimum Dice loss of 0.2604 at epoch four, as shown in Figure 3. In this figure, the Dice loss is tracked over the five epochs of training with the blue curve represents the training loss over time and the red curve representing the validation loss over time. It can be seen that there was a downward trend for the loss, and no significant evidence of overfitting due to the similarity between the training and validation results.

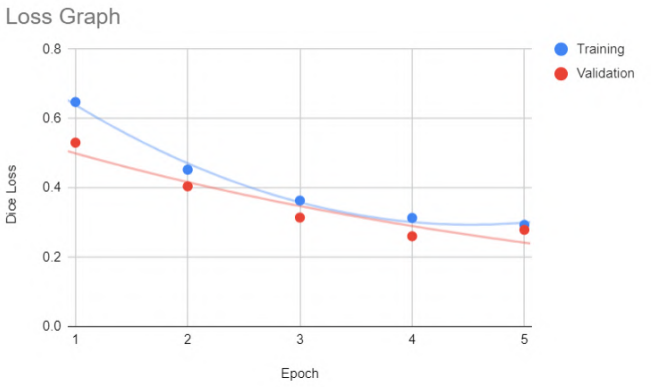

Fig. 3. Graph showing measure of dice loss after each epoch of training.

The model was also shown to achieve a peak validation IoU score of 0.68, which can be seen in Figure 4. This graph shows the recorded score for training and validation over the five training epochs. Again, there is more evidence to support an improvement in performance after training, as the curves have an upward trend.

As for the visual evaluation of the images, Figures 5, 6 and 7 show some sample predictions by the model. In a sample prediction from Figure 5, it can be seen that the model is able to capture correct classes for objects in the image while managing to roughly estimate accurate contours in the images.

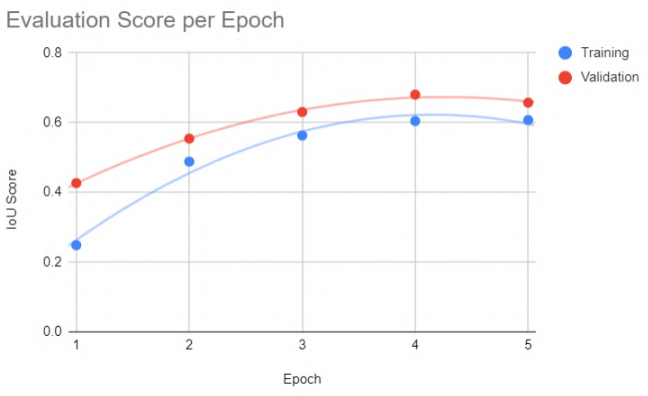

Fig. 4. Graph showing measure of IoU after each epoch of training.

Fig. 5. Sample accurate result from model inference with the actual image (left), the ground truth mask (middle) and the predicted segmentation map (right)

Although, this is not always the case, such as in Figure 6, where while the model manages to predict the shapes of the terrain, it could misclassify the type of terrain it actually is. This could indicate an error in the classification abilities of the model. There might be an error in there having similar results between some classes that should be noted. Both in Figure 5 and 7, the model still makes mistakes in accurately judging the quantity of pixels belonging to a class. This might also be a similar problem where the contours of different classes are blending together due to their similarities

Fig. 6. Sample result with misclassification with the actual image (left), the ground truth mask (middle) and the predicted segmentation map (right)

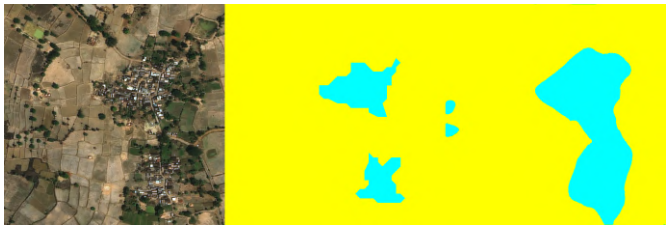

Fig. 7. Relatively accurate sample result with size error with the actual image (left), the ground truth mask (middle) and the predicted segmentation map (right)

in features. It might be important to note that this problem might be less of an issue if certain classes are merged in order to suit a more specific application (e.g. urban and non-urban).

VI. Conclusion

This study explores the use of a U-Net model for use in land classification in nanosatellites. The dataset from the DeepGlobe challenge was used to accurately represent the model in potential space applications in an edge device. Overall, the model was functional and was able to produce usable results. Although, as seen from a few samples in the results, the model still has limitations in either accurately identifying the right pixels belonging to a certain class (misjudging the size of the area of a certain terrain type) or identifying the right class for a certain group of pixels (misidentifying the terrain type of a certain area).

VII. Recommendation

For future studies, it is recommended that other segmentation models could be used to improve the accuracy of a proposed system. It is also potentially worthwhile to attempt to make the model lighter and more efficient for use in orbital edge devices. These orbital devices carry less processing power and total memory storage, so exploring different techniques to compress models to be able to run on smaller processing devices will be ideal when transitioning to more practical applications.

On the data side, other datasets could be used, especially ones that are more representative of the image capturing capabilities of a nanosatellite. Some more complex issues can be introduced, such as lower resolutions and cloud coverage, to train the system to be more robust for more complex inputs. The classification system of the model can also be expanded to include more terrain types or more varied locations.

Finally, an integrated control system could potentially be proposed using the model in this study to create an entirely functional system that takes in input images, and a storage system that adapts itself depending on the usability of current stored images. Additional evaluation metrics could be explored for classification, such as the presence of interesting subjects, quality and clarity of the image, or the total diversity of classes in an image.

References

[1] B. Denby and B. Lucia, "Orbital Edge Computing: Nanosatellite Constellations as a New Class of Computer System," in Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems, New York, NY, USA, Mar. 2020, pp. 939–954. doi: 10.1145/3373376.3378473

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in Advances in Neural Information Processing Systems, 2012, vol. 25.

[3] G. Furano et al., "Towards the Use of Artificial Intelligence on the Edge in Space Systems: Challenges and Opportunities," IEEE Aerospace and Electronic Systems Magazine, vol. 35, no. 12, pp. 44–56, Dec. 2020, doi: 10.1109/MAES.2020.3008468.

[4] Z. Li, W. Yang, S. Peng, and F. Liu, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects." arXiv, Apr. 01, 2020. doi: 10.48550/arXiv.2004.02806.

[5] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, doi: 10.1109/5.726791.

[6] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition." arXiv, Apr. 10, 2015. doi: 10.48550/arXiv.1409.1556.

[7] A. G. Howard et al., “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.” arXiv, Apr. 16, 2017. doi: 10.48550/arXiv.1704.04861.
[8] M. Tan and Q. V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.” arXiv, Sep. 11, 2020. doi: 10.48550/arXiv.1905.11946.
[9] J. Long, E. Shelhamer, and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation.” arXiv, Mar. 08, 2015. doi: 10.48550/arXiv.1411.4038.
[10] H. Noh, S. Hong, and B. Han, “Learning Deconvolution Network for Semantic Segmentation,” presented at the Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1520–1528.
[11] V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 12, pp. 2481–2495, Dec. 2017, doi: 10.1109/TPAMI.2016.2644615.
[12] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation.” arXiv, May 18, 2015. doi: 10.48550/arXiv.1505.04597.
[13] S. A. Chien et al., “Onboard Autonomy on the Intelligent Payload EXperiment (IPEX) Cubesat Mission: A pathfinder for the proposed HyspIRI Mission Intelligent Payload Module,” 2014.
[14] Z. Zhang, G. Xu, and J. Song, “CubeSat cloud detection based on JPEG2000 compression and deep learning,” Advances in Mechanical Engineering, vol. 10, no. 10, p. 1687814018808178, Oct. 2018, doi: 10.1177/1687814018808178.
[15] G. Giuffrida et al., “CloudScout: A Deep Neural Network for On-Board Cloud Detection on Hyperspectral Images,” Remote Sensing, vol. 12, no. 14, Art. no. 14, Jan. 2020, doi: 10.3390/rs12142205.
[16] T. I. Leong, Y. M. O. Abbas, M. A. C. Purio, and H. A. Elmegharbel, “Image Classification Unit: A U-Net Convolutional Neural Network for On-Orbit Cloud Detection Aboard CubeSats,” in 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, Jul. 2021, pp. 2807–2810. doi: 10.1109/IGARSS47720.2021.9553156.
[17] I. Demir et al., “DeepGlobe 2018: A Challenge to Parse the Earth through Satellite Images,” in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Jun. 2018, pp. 172–17209. doi: 10.1109/CVPRW.2018.00031.
[18] P. Iakubovskii, “Segmentation Models Pytorch,” GitHub repository. GitHub, 2019. [Online]. Available: https://github.com/qubvel/segmentation_models.pytorch
[19] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs.” arXiv, Jun. 07, 2016. doi: 10.48550/arXiv.1412.7062.
[20] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. Jorge Cardoso, “Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations,” in Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, Cham, 2017, pp. 240–248. doi: 10.1007/978-3-319-67558-9_28.
[21] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, “Image Segmentation Using Deep Learning: A Survey,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 7, pp. 3523–3542, Jul. 2022, doi: 10.1109/TPAMI.2021.3059968.