NASA MODIS IMAGERY CLOUD IDENTIFICATION

Dongwei Fu
University of Colorado, Boulder
Boulder, CO 80309
dofu3785@colorado.edu

ABSTRACT
Shallow clouds play an important role in moderating the Earth's climate. Satellite imagery, with its advantage of high spatial coverage is one of the most effective methods for studying shallow clouds. In this project, I plan to investigate the problem of classification and segmentation of shallow cloud features in satellite imagery using Convolutional Neural Network (CNN) technique.

Keywords
Segmentation; Satellite imagery; Cloud; U-Net; EfficientNet, ResNet

1. Introduction
Climate change is one of the most pressing challenges human-beings are currently facing. Cloud covers approximately 70% of the Earth's surface, playing a critical role in regulating the planet's climate by reflecting sunlight and trapping heat, which affects weather patterns and the global energy balance. Understanding how clouds interact with other elements of the climate system is essential for accurate climate modeling and predictions. Among the various types of clouds, shallow clouds which are often low-altitude clouds, tend cover large portions of the surface, reflect sunlight back into space, which help to cool the Earth's surface. However, they also trap some of the heat radiating from the Earth, which influences the greenhouse effect. Small changes in the behavior of shallow clouds, such as their thickness or coverage, can lead to significant changes in global temperatures. Understanding the structure and organization of shallow clouds is critical for improving shallow cloud parameterization in climate models and predicting future climate behavior.

Satellite imagery is the most effective approach for studying shallow cloud patterns, structures, and formation because it provides extensive spatial coverage and consistent, long-term data across the globe. With the ability to capture cloud behavior over vast areas, satellite sensors can monitor clouds in remote or inaccessible regions. High-resolution satellite imagery also enables detailed observation of cloud morphology and evolution in real time, offering valuable insights into how shallow clouds form and change. This comprehensive view is essential for identifying large-scale cloud patterns, improving climate models, and understanding cloud-climate interactions.

To form a good understanding of shallow clouds would often require a trained professional (e.g., meteorologist) to analyze and interpret a considerable amount of satellite imagery. With fast development of machine learning techniques, it is possible to develop an intelligent automated system to interpret and understand various cloud structures. In this project, a Convolutional Neural Network (CNN) model has been developed to classify cloud structures from satellite images into four categories. This project is meant to explore the capability of using currently available machine learning packages (like the TensorFlow in Python) and perform satellite remote sensing imagery classification.

2. Related work
Satellite imagery segmentation plays a pivotal role in remote sensing, enabling the classification of pixels to extract valuable information such as land cover, water bodies, or urban infrastructure. Image segmentation can be viewed as combining classification and localization. Through image segmentation, it partitions the image into smaller segments, and then tries to understand what is given at a pixel level and identifies the shapes and boundaries of the object. The final output of image segmentation is a mask where each element indicates which class the pixel belongs to.

In the past, sliding window technique [1] has been used for image segmentation. This architecture creates a local path for each pixel, generating separate class labels for each pixel. Yet this architecture has two main drawbacks: First, it generates a lot of overall redundancy due to overlapping patches. Secondly, the training procedure was slow and takes a lot of time and computing resources.

In 2015, Ronneberger et al. [2] developed a convolutional neural network (CNN) architecture called U-Net for their work on biomedical image analysis. This new approach, U-Net, overcomes the drawbacks of the sliding window technique, and outperformed the latter by using fewer images and data augmentation to increase model performance. U-Net gets its name from its architecture, a U-shaped structure consists of two main components: an encoder network and a decoder network. The encoder compresses the input image by downsampling it through convolutional and max-pooling layers, capturing essential features and context. This is often referred to as the contracting path. Through the encoder, it progressively reduces the spatial dimensions of the input image while increasing depth (i.e., number of filters). The decoder, which is often referred to as the expanding path, it upsamples the compressed features and restores the original spatial resolution, making use of skip connections to transfer high-resolution information from the encoder to the decoder. This design ensures that both contextual and detailed information are preserved, making it highly effective for tasks requiring precise localization of objects in images.

Due to its efficient architecture, U-Net is widely applied across various fields beyond biomedical image segmentation, including satellite imagery analysis, autonomous driving, etc. Its ability to work well with limited data and accurately segment objects even in complex environments makes it a preferred choice for many image segmentation tasks. The skip connections in U-Net help retain fine-grained details lost during downsampling, leading to improved segmentation accuracy, especially for tasks that demand high precision. Since then, various variants of U-Net have been developed, and hybrid models combining U-Net with different backbones, such as ResNet [3], to improve accuracy and computational efficiency. These advancements have proven

particularly effective for processing high-resolution satellite data, where boundary precision is crucial for tasks like land-use classification or environmental monitoring.

More recent research has focused on leveraging lightweight, efficient models like EfficientUet [4] in satellite image segmentation. EfficientNet, known for its compound scaling method, has been integrated into UNet-like architectures to create EfficientNet, which delivers high segmentation performance with reduced computational complexity. This is particularly valuable when working with large-scale satellite datasets, as EfficientNet enables fast processing without compromising accuracy. Similarly, ResNet-based models, such as ResNet, leverage residual connections to enhance gradient flow, allowing for deeper networks and better feature extraction. Both EfficientUNet and ResNet have demonstrated robust results in satellite segmentation challenges, excelling in areas requiring detailed analysis, such as agriculture and urban planning.

3. Proposed work

3.1 Dataset Exploratory Data Analysis

The dataset of this project is provided as an online Machine learning competition posted on Kaggle [5]. It was curated by researchers from Max Planck Institute for Meteorology [5][6]. The dataset consists of 5546 (training) and 3698 (testing) RGB satellite imagery with dimensions of 1400 pixels by 2100 pixels taken from the Moderate Resolution Imaging Spectroradiometer (MODIS) onboard NASA's two polar-orbiting satellites, Terra and Aqua.

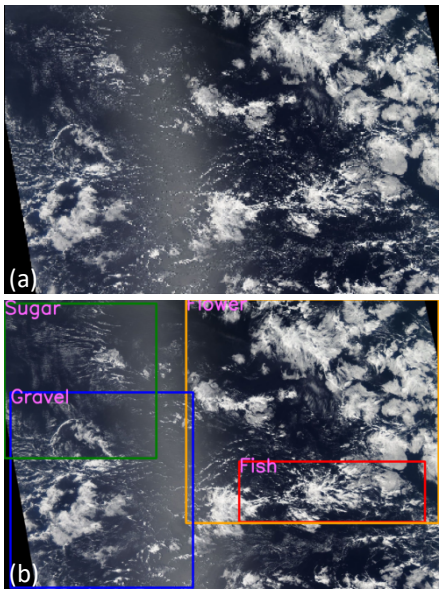

Fig 1. (a) Training Image 015aa06.jpg (MODIS true color RGB Image). (b) bounding box indicating cloud class label(s).

The images contain cloud structures from four class: Fish, Flower, Sugar and Gravel. Each class (label) has its distinct cloud features: For example, "Sugar" categorizes the dusting of very fine clouds with little evidence of self-organization, whereas "Flower" categorizes large-scale stratiform clouds that appears in bouquets with separation between each other; "Fish" refers to large-scale skeletal networks of clouds that are separate from other cloud formations, and finally "Gravel" refers to arcs of randomly interacting cells with granularity [6]. Fig. 1 shows one instance where all four categories appeared in one MODIS imagery. Note to the lower left corner of the imagery there are fill values (black), as there are gaps between different Aqua-MODIS ascending node orbits (daytime orbits) on a daily basis (MODIS has a wide swath of 2330 kilometers, providing near-global coverage every 1-2 days).

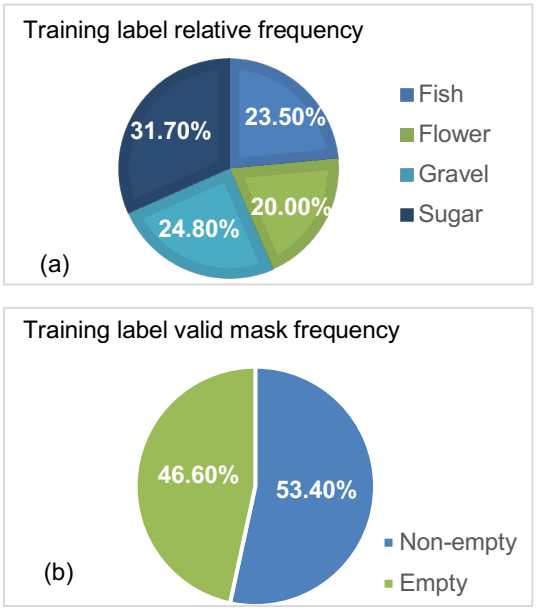

Fig 2. (a) Relative frequency of each label (for valid masks), (b) Percentage of non-empty and empty masks in the training dataset.

Fig. 2 shows the percentage of each class (label) in the training dataset. The Sugar category has the highest percentage at 31.7%, whereas Flower has the lowest percentage at 20.0%. Overall, the four classes are well balanced, and each class makes up nearly equal shares of the dataset. However, it is also worth noting that for the encoded mask (that indicate the class labels) for each image, nearly half (46.6%) of the masks are empty. Finally, the correlation heatmap between each categorical label is provided in Fig. 3. We can see that cross-label correlation among different labels are generally low.

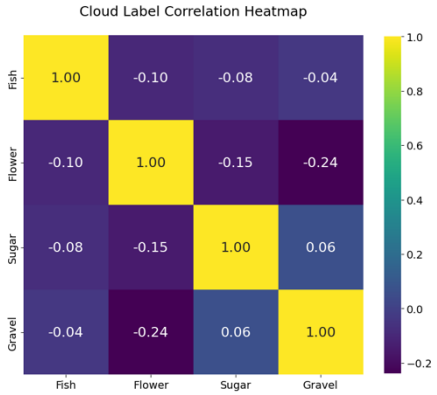

Fig 3. Heatmap of cloud labels from training dataset.

3.2 Data Preprocessing

The images at its original resolution of 1400 x 2100 pixels can be computational heavy for model training. Therefore, data augmentation was performed prior to training. Augmentation

artificially expands the size of the dataset by creating modified data which improves the performance of the model to generalize. Albumentations library has been used for augmentation. This library efficiently implements an abundant variation in image transformation that are performance optimized. The training dataset images were split into batches of 32 and horizontal, vertical flips and random rotation (less than 45 degrees) were performed.

Pixel encoding technique was followed to participate in the submission of the competition since the image sizes were too large for Kaggle submission system [5]. As a result, instead of submitting an exhaustive list of indices for segmentation, pairs of values were submitted, which contained the start position and the run length of the image pixels. For example, a pair value of (1, 3) indicates that the pixel starts at 1 and run 3 pixels. The competition also required a space-delimited list of pairs. The predicted encodings were scaled by 0.25 per side, which scaled down the images of size 1400 × 2100 pixels in both train and test set to 320×480 pixels, hence allowing the scope to achieve reasonable submission evaluation times. This was achieved through run-length encoding of the segmentation mask.

3.3 Evaluation Metric

The evaluation metric used in this paper is the Dice coefficient [5, 7, 8]. It was applied to compare the pixel-wise agreement within a predicted segmentation and corresponding ground truth, using the following equation:

$$\frac{2 * |X \cap Y|}{|X| + |Y|} \qquad (1)$$

Here X defines the set of pixels predicted, whereas Y defines the ground truth of the training set. When X and Y are empty, the dice coefficient is defined as 1. The Kaggle competition Leaderboard score provides the mean of Dice coefficients for each (Image, Label) pair in the test data.

Dice coefficients are slightly different from the more popular evaluation metric: accuracy of a model. They are used to quantify the performance of image segmentation methods. Some ground truth regions are annotated in the images, and then an automated algorithm is allowed to do it. The algorithm is validated by calculating the dice score, which is a measure of how similar the objects are and is calculated by the overlap of the two segmentations divided by the total size of the two objects [8]. Dice coefficient works better in segmentation because of its ease of differentiation, as a result it is preferable over Jaccard's index, another evaluation metric similar to Dice coefficient [9].

The loss function for one image sample is defined as the summary of binary cross entropy loss and dice coefficient loss,

$$Loss = -(Loss_{BCE}(yt, yp) + Loss_{dice}(yt, yp)) \qquad (2)$$

Whereas *yt* is the ground truth, *yp* is the model's prediction.

3.4 Segmentation Model Architecture

As mentioned in Section 2, in this project I will focus on using U-Net model architecture, along with various backbones.

1). EfficientNet

EfficientNet is a family of convolutional neural networks (CNNs) introduced by Google, which revolutionized the approach to scaling deep learning models. The core innovation in EfficientNet is its compound scaling method, which uniformly scales a model's depth, width, and resolution using a simple scaling coefficient. Traditional methods typically scale one dimension at a time, such as increasing network depth or input resolution, often leading to suboptimal results. EfficientNet, however, applies a balance across these three dimensions, allowing for better performance without overburdening computational resources. The baseline EfficientNet-B0 model serves as a foundation, and its variations (B1 through B7) gradually increase in complexity and capability.

EfficientNet models are built using mobile inverted bottleneck convolution (MBConv) blocks and a scaling formula that applies different degrees of scaling to the model's dimensions. EfficientNet-B0 is the baseline model, and higher versions (like EfficientNet-B7) apply the compound scaling formula to progressively increase the network's complexity and performance. Despite its simplicity, EfficientNet has been shown to significantly improve model efficiency, reducing computational cost without sacrificing accuracy. This makes it a popular choice for tasks like image classification, object detection, and segmentation.

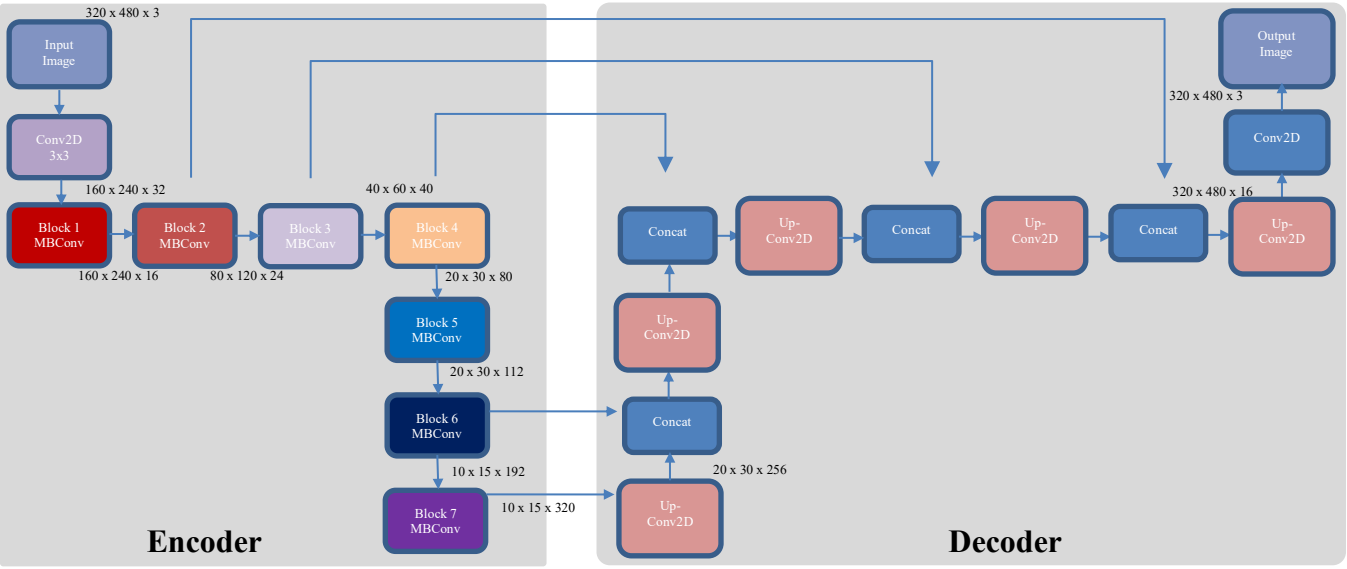

Fig. 4. Architecture of EfficientNet-B0 framework. Adapted from [10].

While UNet is highly effective for tasks like medical image segmentation due to its unique encoder-decoder structure, EfficientNet provides several advantages in terms of performance and efficiency when adapted for similar tasks. One of the major advantages is EfficientNet's ability to handle larger and more complex datasets while maintaining lower computational requirements. The compound scaling strategy allows EfficientNet to improve accuracy without drastically increasing the number of parameters or computational load, which is crucial in resource-constrained environments.

In this project, due to the limited computational resources, I mainly tested using the baseline EfficientNet backbone combined with UNet decoder. Fig. 4 shows a general model architecture of EfficientNet used in this project [10]. The model architecture along with the encoder backbones are provided by Segmentation Models [11].

2). ResNet:

ResNet (Residual Network), is a deep neural network architecture introduced to address the problem of vanishing gradients, which occurs when training very deep networks. ResNet introduced the concept of residual connections, where the network skips layers through shortcut connections, allowing the model to learn identity mappings. This innovation enables the training of very deep networks, sometimes with hundreds or even thousands of layers, without suffering from performance degradation. Residual connections allow gradients to flow more easily through the network, which ensures that even very deep layers can contribute to the learning process. The most popular variant, ResNet-50, is frequently used as a backbone for tasks such as image classification, object detection, and segmentation.

The advantages of ResNet extend beyond its ability to train deep networks. Residual connections enhance the network's ability to generalize to different tasks while maintaining computational efficiency. This adaptability makes ResNet a solid choice as a backbone in many computer vision applications. For example, in segmentation tasks, models like UNetResNet leverage the strengths of both UNet's upsampling architecture and ResNet's feature extraction capabilities. ResNet's architecture, particularly its ability to extract rich, deep features from images, enables it to outperform simpler models on challenging datasets. Its success has paved the way for many subsequent architectures that build on the idea of residual connections, such as DenseNet and ResNeXt, which further improve upon its efficiency and feature learning capabilities.

In both research and practical applications, ResNet continues to be widely used due to its scalability, robustness, and ability to handle complex datasets efficiently, making it a top choice in the field of computer vision. In this project, I will also test with ResNet and compare its performance with EfficientNet for this particular satellite image segmentation task.

3.5 Summary of pipeline

For this project, the datamining pipeline is as follows:

- Using the original training image dataset, 80% of the images are used for training and 20% are used for validation.
- Use Albumentation to perform image augmentation.
- Train baseline model (EfficientUNetB0 architecture) for 30 Epoch.
- Assess Baseline model performance
- Hyper-parameter tuning (adjusting learning rate, changing encoder model).
- Re-assess and finalize model.

4. Evaluation and Discussion

4.1 EfficientNet Results

The first model run was performed using UNet with EfficientNetB0 as the encoder backbone. Learning rate was initialized at 0.0002. Fig. 5 shows the Dice coefficient and the model loss for both training and validation dataset.

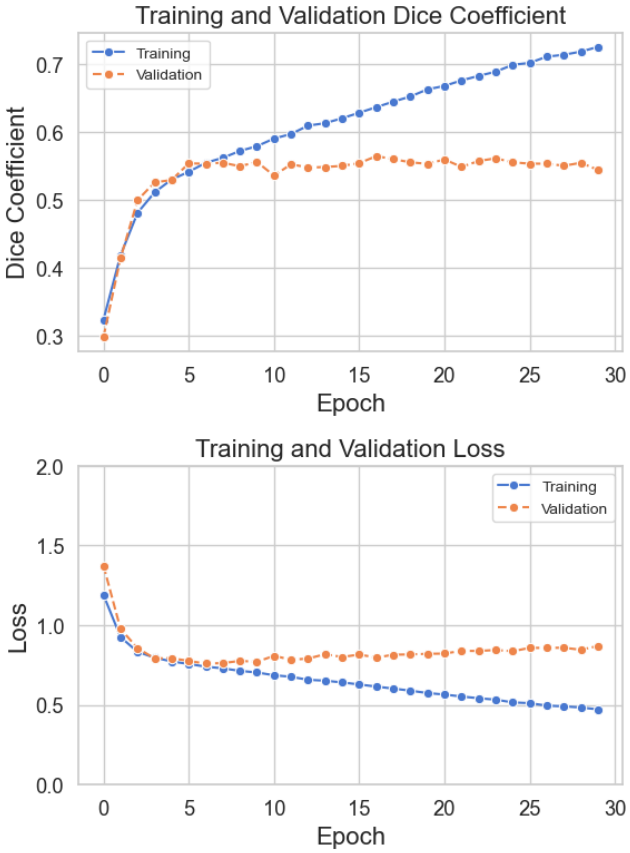

Fig. 5. Training and Validation (a) Dice Coefficient and (b) Loss.

The model's performance, as indicated by the training and validation Dice Coefficient and Loss curves, reveals some key insights into its learning behavior. The training Dice Coefficient steadily improves throughout the 30 epochs, suggesting that the EfficientNetB0 backbone effectively learns features from the training data. Similarly, the training loss decreases progressively, indicating that the model optimizes well on the training set. However, the validation Dice Coefficient plateaus early around epoch 6 and shows minimal improvement afterward, while the validation loss stabilizes without significant reduction after an initial decline. This disparity between training and validation performance suggests that the model is learning well on the training data but struggles to generalize to unseen validation data.

The early plateau in validation Dice Coefficient and the stagnation of validation loss point to potential overfitting, where the model memorizes the training data without effectively generalizing. Possible reasons for this behavior include the model's complexity in relation to the dataset size, class imbalance in the segmentation

task, or the need for further regularization techniques such as dropout or weight decay. Adjusting the learning rate or employing learning rate schedules could also help prevent the model from plateauing too early.

4.2 Learning rate hyperparameter tuning

As seen from the previous section, the baseline run indicated potential overfitting of the training dataset. Using the same EfficientNetB0 architecture, the impact of learning rate was examined. 4 different runs consisting with learning rates from 0.0002 to 0.02 were performed (Fig. 6). Fig. 6 provide clearer insights into how different values affect training and validation performance. As shown, the learning rate of 0.0002 exhibits the most stable and consistent improvement, both in the Dice Coefficient and loss across training and validation sets. The training curve steadily increases, reaching around 0.7 by the 30th epoch, while the validation curve follows more closely compared to the other learning rates. The loss decreases smoothly for both training and validation sets, indicating that the learning rate of 0.0002 allows the model to optimize without oscillating or getting stuck in local minima.

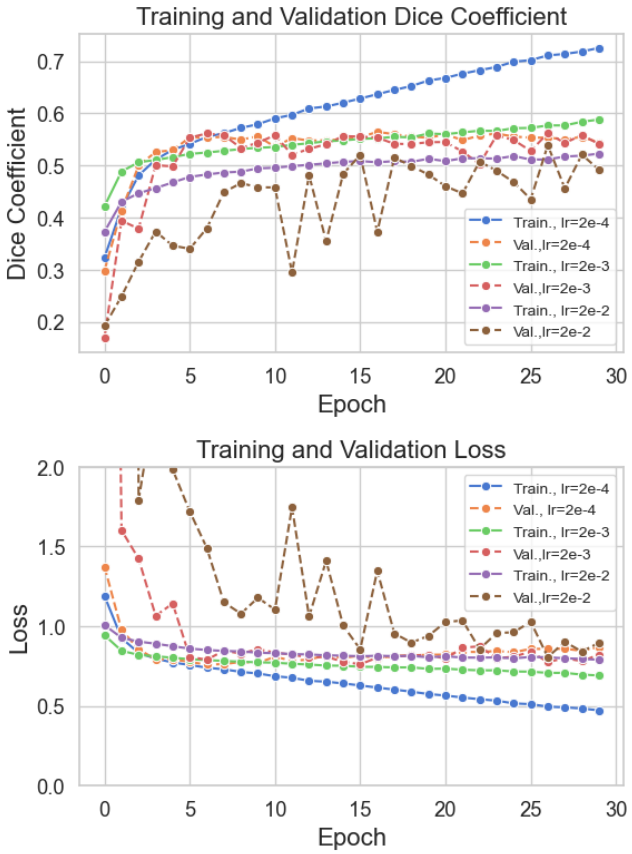

Fig. 6. Training and Validation (a) Dice Coefficient and (b) Loss.

On the other hand, higher learning rates like 0.002 and especially 0.02 lead to more erratic training behavior. The Dice Coefficient curves for the validation sets with these higher learning rates fluctuate significantly, especially for 0.02, where it fails to converge properly. The corresponding loss curves show spikes and inconsistencies, with the validation loss sometimes increasing while the training loss decreases. This suggests that the higher learning rates cause the model to overshoot optimal weights during training, leading to poorer generalization. Overall, the tuning experiment highlights the importance of selecting an appropriate learning rate, as a value like 0.0002 appears to strike the right balance for stable and effective learning.

4.3 Results from ResNet

Fig. 7 presents the model evaluation metrics for the U-Net model architecture using ResNet-34 and ResNet-50 backbone. In general, both models performed very similarly. It is worth noting that both models showed initial increase in loss for the first few epochs. Around 5 epochs, both models reached the peak in model loss, and then the model loss decreases dramatically until becoming stable after ~10 epochs. This corresponds with the initial stall (slow increase) of validation Dice coefficient (~5 epochs), after which validation Dice coefficient increases.

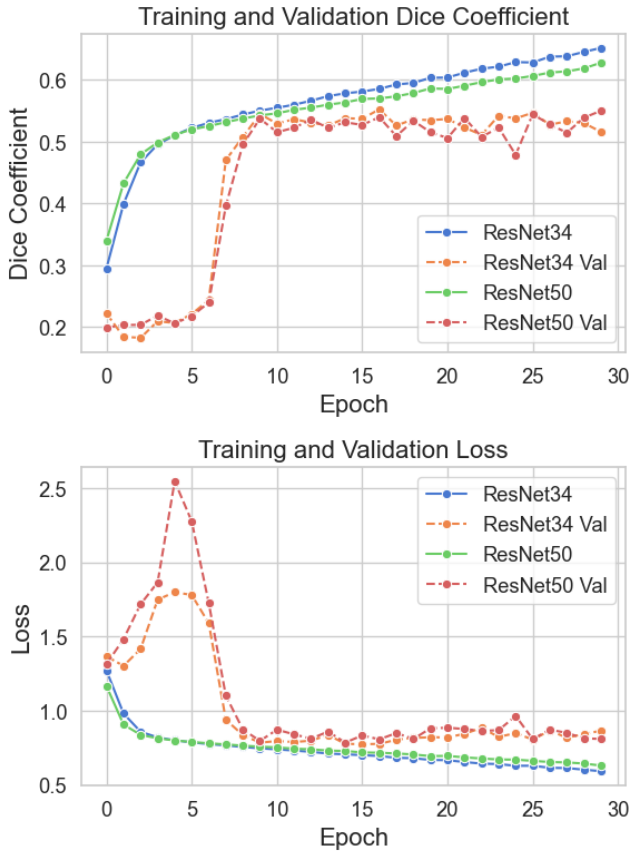

Fig.7. Training and Validation (a) Dice Coefficient and (b) Loss for ResNet-34 and ResNet-50.

Such behavior is quite common for deep convolution network models. In the first few epochs, the model is rapidly adjusting its weights based on training data. Initially the model's predictions can be highly erratic as the weights are far from optimal. The large initial fluctuations in weight updates may result in poorer generalization to the validation set, causing the validation loss to spike. Also, since the validation dataset is not directly involved in the training process. Thus, during the initial epochs, the model may tend to overfit the training data while struggling to generalize to new data from validation dataset.

Nevertheless, in the final run, ResNet-34 achieved slightly higher training DICE score (lower loss), but ResNet-50 performed better for validation dataset (see Table I for the score details). Overall, both ResNet-34 and ResNet-50 models resulted in similar model

performance shows that more complex model architecture and more convolutional layers does not always yield dramatic increase in model performance. Compared to ResNet-34, ResNet-50 has more layers (50 vs. 34) and uses more complex bottleneck residual blocks instead of basic residual blocks. Thus, despite its relative shallowness compared to ResNet-50, ResNet-34 still performed quite and provided a good trade-off between speed and accuracy. model simplicity and faster inference are prioritized over the need for extremely high accuracy.

4.4 Summary of all model runs

Table I shows the final evaluation metrics (after 30 epochs) for all model architectures used in this project. Overall, the baseline EfficientNet-B0 with a learning rate of 0.0002 achieved the highest training Dice score of 0.726 and lowest training loss of 0.472, yet its validation Dice score is slightly lower (and validation loss is slightly higher) then ResNet-50 (with same learning rate). Among the two ResNet runs, while ResNet-34 performed better for the training set, ResNet-50 resulted in better score for the validation set.

On the other hand, when using a higher learning rate of 0.02, EffcientNet-B0 exhibits noticeable drop in performance across all metrics, with Dice Score falling to 0.523 (training) and 0.492 (validation), indicating that model is overfitting and struggles to converge. Overall, these results emphasize the impact of learning rate and model architecture on image segmentation performance. Among EfficientNet-B0 and ResNet, EfficientNet-B0 showed superior adaptability and learning stability when it is properly tuned.

BACKBONE MODEL	TRAIN. DICE	VAL. DICE	TRAIN. LOSS	VAL. LOSS
EFFICIENTNET B0 (LR = 0.02)	0.523	0.492	0.789	0.896
EFFICIENTNET B0 (LR = 0.002)	0.588	0.541	0.691	0.822
EFFICIENTNET B0 (LR = 0.0002)	0.726	0.544	0.472	0.870
RESNET-34	0.652	0.5159	0.592	0.865
RESNET-50	0.627	0.550	0.630	0.810

TABLE I. Training and validation Dice score and loss.

4.5 Test cases with model prediction

Fig. 8 shows two test cases using the final model EfficientNet-B0 with learning rate of 0.0002. Fig. 8(a) shows for test dataset image ID 'db8c669.jpg', three cloud features were identified. Fig. 8(b) shows for test dataset image ID '04326a2.jpg', all four cloud features were identified. Upon close examination of the identified cloud features in both test images, as well as comparing it with the training images (e.g., Fig. 1(b)), the model predictions are reasonable. As can be seen in Fig. 8(a), to the right of the image the "flowers" are correctly identified. In the middle of the image, the mask labeled most pixels "sugar", while also labeling sub-segments within "sugar" bounding box to be "gravel". In Fig. 8(b), again "flowers" in the right were correctly identified. Yet "sugar" and "gravel" in the left showed some overlap. These behaviors are consistent with the training images, and this shows that it is indeed quite challenging to separate cloud formations, especially when some larger-scale features overlap with finer-scale features.

5. Conclusion

In this project, I focused on exploring deep learning models for satellite image segmentation, to identify different cloud features from satellite imagery. Two primary image segmentation model architectures were explored: EfficientNet and ResNet. Throughout this project, different learning rates were tuned to optimize model performance, and the model setup that achieved the highest Dice score was using EfficientNet-B0 architecture with initial learning rate of 0.0002. ResNet also showed reasonable performance when compared with EfficientNet. This project demonstrates the feasibility of using hybrid U-Net setups (combined with efficient encoder) to perform satellite image segmentation.

While the current results demonstrate the effectiveness of EfficientNet for satellite image segmentation, there are several avenues for future improvements. First, running the models on a larger dataset would likely enhance generalization and improve performance on unseen data. The current dataset may have limited the model's ability to fully capture the intricacies of satellite imagery, particularly the wide variety of cloud formations and mesoscale patterns present in real-world scenarios. Expanding the dataset, either through additional satellite imagery or other data augmentation techniques, could lead to more robust models capable of handling more diverse environmental conditions.

In addition to scaling the training data, exploring other architectures beyond EfficientNet and ResNet could further improve segmentation accuracy. Architectures such as EfficientNetV2 or other hybrid models that combine convolutional and transformer-based approaches might offer better feature extraction and generalization capabilities. Future work could explore these alternative architectures and techniques to push the boundaries of segmentation accuracy in satellite imagery tasks.

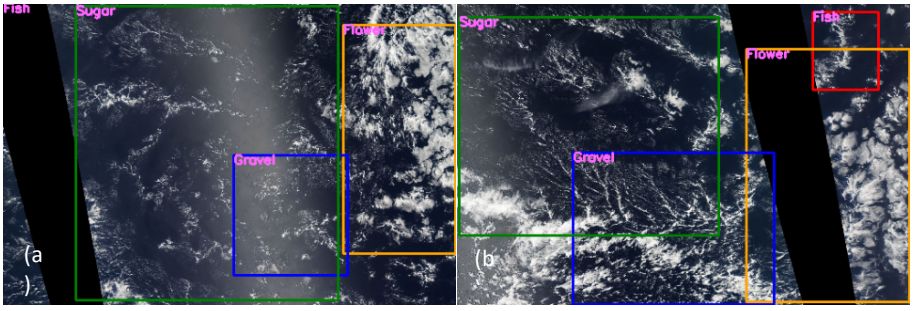

Fig 8. Test cases with model prediction of cloud features for (a) test image 'db8c669.jpg' (b) test image '04326a2.jpg'

6. References

[1] Pappas, T. N. 1992. An adaptive clustering algorithm for image segmentation. IEEE Transactions on Signal Processing, 40(4), 901–914. https://doi.org/10.1109/78.127962

[2] Ronneberger, O., Fischer, P., & Brox, T. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. Lecture Notes in Computer Science, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28

[3] He, K., Zhang, X., Ren, S., & Sun, J. 2016. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2016.90.

[4] Tan, M., and Le, Q. V. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946.

[5] Rasp S., H. Schulz, R. Walter, M. Demkin. 2019. Understanding Clouds from Satellite Images. Kaggle. https://kaggle.com/competitions/understanding_cloud_organization

[6] Rasp, S., H. Schulz, S. Bony, and B. Stevens, 2020. Combining Crowdsourcing and Deep Learning to Explore the Mesoscale Organization of Shallow Convection. Bull. Amer. Meteor. Soc., 101, E1980–E1995, https://doi.org/10.1175/BAMS-D-19-0324.1.

[7] Zou, Kelly H., et al 2004. Statistical Validation of Image Segmentation Quality Based on a Spatial Overlap Index. Academic Radiology, vol. 11, no. 2, 1 Feb. 2004, pp. 178–189, www.ncbi.nlm.nih.gov/pmc/articles/PMC1415224/, https://doi.org/10.1016/S1076-6332(03)00671-8.

[8] Dice, Lee R. 1945, Measures of the Amount of Ecologic Association between Species. Ecology, vol. 26, no. 3, pp. 297–302, https://doi.org/10.2307/1932409.

[9] Bertels, Jeroen, et al., 2019. Optimizing the Dice Score and Jaccard Index for Medical Image Segmentation: Theory and Practice. Lecture Notes in Computer Science, pp. 92–100, https://doi.org/10.1007/978-3-030-32245-8_11.

[10] Ahmed, T., Hossain, N., Sabab., N. 2021. Classification and Understanding of Cloud Structures via Satellite Images with EfficientUNet. SN Computer Science, vol. 3, no. 1, 12 Dec. 2021, https://doi.org/10.1007/s42979-021-00981-2.

[11] Iakubovskii, P. 2019. Segmentation Models. GitHub repository. Retrieved from https://github.com/qubvel/segmentation_models