Semantic Map Segmentation: a Comparison of U-Net and Fully Convolutional Architectures

Sophie Zhao
University of Richmond
Richmond, VA

Ginny Zhang
University of Richmond
Richmond, VA

Win Htet Aung
University of Richmond
Richmond, VA

Tolya Evdokimov
University of Richmond
Richmond, VA

I. Abstract

This project explores different convolutional neural network architectures applied to the semantic map segmentation task. We retrieved the map dataset from Kaggle and implemented three CNN models: U-Net, FCN-8, and FCN-32. We conducted quantitative and qualitative analyses which showed that U-Net performs the best on the segmentation task, while the FCN models demonstrate sub-optimal performance likely due to the input image resolution, dataset size, and the specifics of the FCN architecture. We also discuss the limitations and future work related to the map segmentation task.

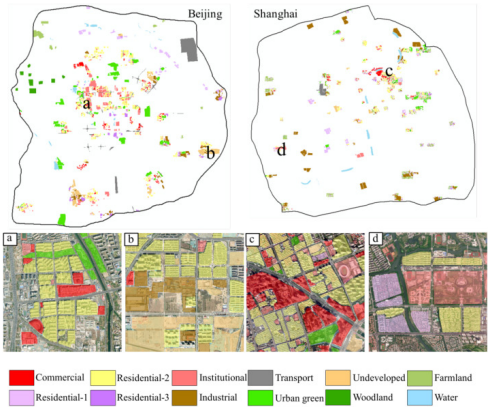

Fig. 1: Urban Planning

II. Introduction

Images come in all forms — shapes, sizes, and colors — each containing various convoluted components. While humans can effortlessly distinguish one component from another, computers require sophisticated methods to achieve similar results. The origin of computer vision can be traced back to a grainy image of a baby, merely 5 centimeters square. This image was captured by computer scientist Russell Kirsch, the pixel inventor, and his colleagues at the National Bureau of Standards in 1957 [1]. Their work laid the foundation for the field of image processing, which has since evolved into more specialized areas such as image segmentation, and more specifically, map segmentation.

Segmentation is the computational process of partitioning different regions in a given media. It involves high-precision analysis of an element's boundary and the ability to differentiate between various segments. This process is critical in applications ranging from tumor detection in medical imaging to defect identification in manufacturing and object classification in autonomous vehicles, where precise segmentation of objects and areas is crucial.

Expanding on this foundation, our project focuses on the use of semantic segmentation for urban planning and architecture. Urban planners rely on precise land use classification to make informed decisions that impact urban development and environmental conservation [Figure 1]. To address this problem, we utilized models such as U-Net, FCN-8, and FCN-32 to identify and differentiate regions of a terrain map from aerial views. The application of semantic segmentation allows us to use the power of AI to transform aerial imagery into actionable insights for urban development and environmental planning. Through this project, we aim to demonstrate our ability to apply AI concepts and implement various complex models to gain insights into real-world problems.

III. Survey of Related Work

Semantic image segmentation is most prominent in the medical field with models segmenting tumors, body parts, internal organs, etc. It was not until the introduction of U-Net that high-performance image segmentation became possible [6]. U-Net is an encoder-decoder-based model that accepts an image as an input and outputs the transformed image with segmented areas colored. The encoder part of the model uses a regular convolutional network that downscales the images and extracts features, while the decoder part upscales the

images, increases the resolution, and provides the final output of the model [7]. The basic U-Net structure has been built upon and improved with the introduction of edge-boosted U-net [11] and LCU-Net [10].

In addition to its widespread use in the medical field, image segmentation is used in urban planning in various terrain classification tasks. The ability to be able to extract roads, detect buildings, and identify landforms using satellite images is crucial in planning for a sustainable urban environment. For example, Singh and Nongmeikapam [8] employed a Deep U-Net, a modified version of the U-Net architecture, for semantic segmentation of satellite images. The key distinction between Deep U-Net and U-Net lies in the incorporation of additional feature layers in the original U-Net model to capture more intricate representations from input images. As a result, Singh and Nongmeikapam achieve 90.6% overall accuracy, outperforming other models using a U-Net-like archotecture. Another approach that involves highly accurate mapping is the incorporation of VHR images and applying a multi-scale semantic segmentation network to predict the output segmented image [2]. Similar to Deep U-Net, multi-scale semantic segmentation exceeds U-net's performance as well due to its superiority in multi-scale contextual features. Although both Deep U-Net and multi-scale semantic segmentation are highly accurate performing models, as their pioneer, U-Net performs very well in basic map segmentation.

Even though U-Net has revolutionized semantic segmentation with high performances, FCN (Fully Convolutional Networks) is another architecture for executing image segmentation. Similar to U-Net, FCN also has an encoder-decoder portion in its architecture with some skip connections [9]. However, unlike U-net, it is often not fully symmetric. FCN contains more convolutional layers and less connected layers. When comparing the performance on map segmentation of the two models, U-Net performed well for both 256x256 and 512x512 images, whereas FCN performed well only for 512x512 images [4]. This result will serve as the foundation for our analysis of the experiment result when we compare the performance of U-Net, FCN-8, and FCN-32 on a 128x128 dataset of images.

IV. Approach

Our dataset, titled "Earth Terrain, Height, and Segmentation Map Images," was obtained from Kaggle [5]. This dataset comprises paired examples of terrain, height, and segmentation maps. In our project, we focused on using the terrain and segmentation maps exclusively. We used Python for the implementation of our project, and used TensorFlow and Keras as our primary backend frameworks for developing and training our models. The analysis was conducted within Jupyter Notebooks. In this project, we utilized several essential Python libraries: Matplotlib for data visualization, NumPy for efficient matrix operations, and OpenCV (cv2) for image resizing. We performed all the training and testing on Spydur.

V. Formulation

The problem is formulated as a pixel-wise classification task within the domain of supervised learning. Specifically, our objective is to accurately classify each pixel in aerial terrain maps into one of seven predefined terrain categories — Water, Grassland, Forest, Hills, Desert, Mountain, and Tundra.

VI. Dataset

Our dataset contains 5,000 examples, each consisting of a pair of images: a terrain map and a corresponding segmentation map. Figure 3 and Figure 4 are examples of a terrain map and segmentation map.

The terrain maps are rich in detail, featuring varied land types differentiated by unique color codes and augmented with relief shading to enhance the depiction of elevation. These images are encoded in the PNG format. Each segmentation map in the dataset serves as a labeled example, with distinct color codes representing different terrain types. This labeled data is essential for training the models, as it provides the ground truth needed to learn the accurate classification of each pixel.

Before the experimentation, we divided the initial dataset into three subsets. The first subset included the first 4,000 images designated for training purposes, the second subset consisted of images numbered from 4,001 to 4,500 intended for validation, and the third subset encompassed images numbered from 4,501 to 5,000 allocated for training. The resulting split is 80-10-10.

Our data processing begins with raw input images, each with dimensions of 512x512 pixels. These images are first resized to 128x128 pixels. Corresponding to each raw image, there is a label image also originally at 512x512 pixels, which is also resized to match the transformed image dimensions of 128x128 pixels. Once resized, the 128x128 label images go through a color-to-class conversion. This involves mapping specific color codes in the images to class identifiers (0-6). This step translates visual information into categorical data. The class identifiers are then converted to a one-hot

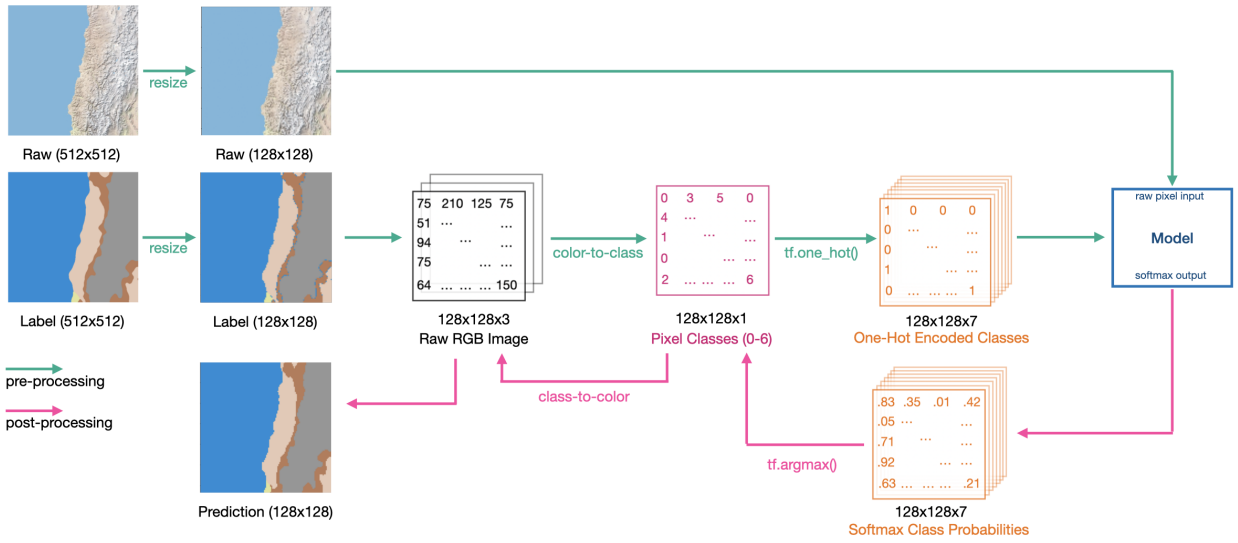

Fig. 2: Data Processing Pipeline

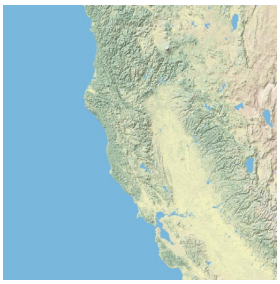

Fig. 3: Example of a terrain map

Fig. 4: Example of a segmentation map

encoded format. In one-hot encoding, each class label is represented as a binary vector with all zeros except for a single high (1) at the index of the class label. For instance, a class label of 3 would be represented as [0, 0, 0, 1, 0, 0, 0]. This transformation is necessary for training of the neural network by providing a way to handle multi-class labels. The one-hot encoded classes and the resized raw images are input into the model. Then the model processes the raw pixel input to produce a softmax output, which represents the probability distributions over the class labels for each pixel. The softmax output from the model gives the probabilities for each class at each pixel, formatted as 128x128x7. This output then undergoes an argmax operation, where the highest probability in each vector is selected, determining the class with the highest likelihood for each pixel. The predicted class data is now reduced to a single class per pixel. Finally, the predicted class data is converted back to color data through a class-to-color conversion process. This allows us to visualize the model's outputs and interpret the predictions. The data processing pipeline is summarized in Figure 2.

Figure 5 provides a comprehensive breakdown of pixel counts for each class that we identified in this experiment. On the x-axis, it displays the six classification classes, while the y-axis quantifies the number of pixels representing each class across the entire dataset. It is comprised of three charts, each depicting the distribution of one of the three different datasets utilized in our experiment. Notice the y-axis is scaled by some power of ten in order to visualize the trend in the pixels of the dataset across the three datasets. Based on the visual interpretation of each chart, all three distributions exhibit a consistent imbalance across the splits, with each class containing a similar proportions of pixels relative to the others within each distribution.

VII. Models

For our experiment, we decided to implement a U-Net architecture to create a model that can process raw images and output segmented images. Our U-Net model consists of both down-sampling and up-sampling with skip connections within each layer of the model which can be seen in the Figure 6. The down-sampling is the initial stage of the model where the convolutional layers are used to reduce the spatial dimensions of the input while increasing the number of feature channels. Max pooling occurs to create the feature maps, which summarizes the features. The lowest layer of the model is called bottleneck: this is where the down-sampling

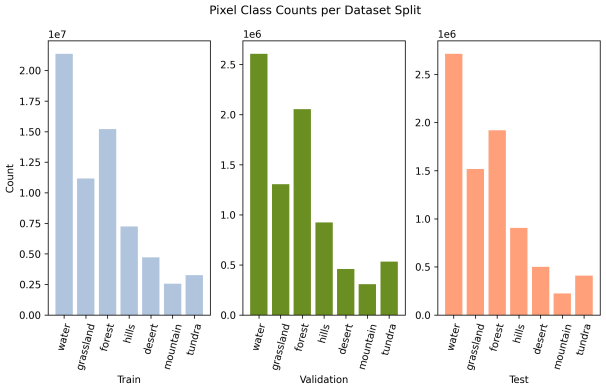

Fig. 5: Pixel Class Counts per Dataset Split

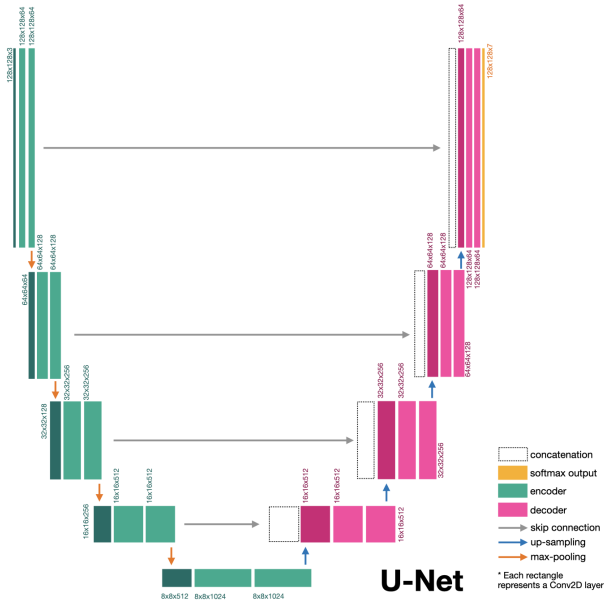

Fig. 6: U-Net Architecture

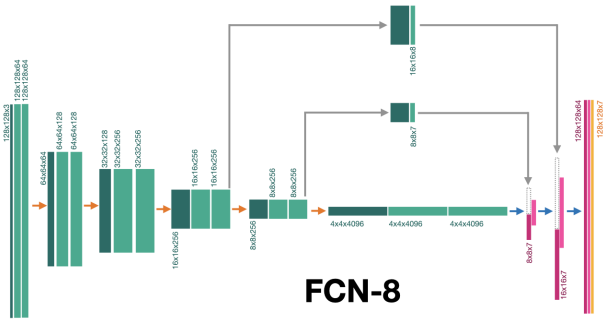

Fig. 7: FCN-8 Architecture

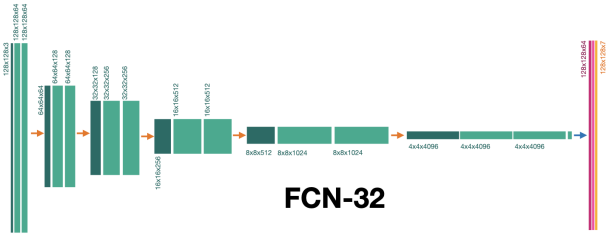

Fig. 8: FCN-32 Architecture

ends and up-sampling begins. Up-sampling is done to gradually restore the spatial resolution of an image. The process at each layer of decoder involves up-sampling the feature map and concatenating it with the feature map from a corresponding encoder module. With the combination of down-sampling, up-sampling, and skip connections, our U-Net architecture aims to effectively process raw images and produce accurate segmented outputs, demonstrating its utility in image map segmentation tasks.

To evaluate our U-Net performance, we have decided to include fully implemented models of FCN-8 and FCN-32 with our modifications [3]. The FCN architecture emphasizes the encoder to extract a rich set of features. Various FCN models incorporate different number of skip connections and connectivity layers to enhance spatial resolution. For instance, in Figure 7, FCN-8 has two connectivity paths utilized for prediction using the feature maps during decoding, which is similar to the up-sampling in U-Net where skip connections act as a bridge for passing information to the decoder. However, FCN-8 lacks a symmetric architecture since it incorporates fewer connectivity paths to the decoder as the main focus of FCN lies in the encoder's feature extraction. Unlike U-Net and FCN-8, FCN-32, shown in Figure 8, has the most straightforward architecture where its sole purpose is to utilize the features extracted from the encoder and predict the output with a single up-sampling operation without any aid from connectivity paths. FCN-32 is computationally less expensive than both FCN-8 and U-Net; however, as a result of this design choice, it may lead to a less accurate segmentation.

Based on varying architectures, we expected that U-Net would have the highest accuracy, compared to FCN-8 and FCN-32, in predicting the true map segmentation of any input map due to having more connectivity paths between the encoder and decoder which will provide a more precise segmentation. We also expect that FCN-8 would perform better than FCN-32 since FCN-32 will solely rely on the features extracted

as opposed to FCN-8, having some connectivity path from the encoder.

VIII. Experiments & Analysis

A. Training

We initially trained our U-Net model on images with dimensions of 512 x 512 pixels. However, during the first epoch, it achieved only 8% validation accuracy, and this declined further in subsequent epochs, which indicates overfitting. To address this issue, we resized our images to 128x128 pixels for all models' training and testing.

First, we trained the U-Net model. We compiled the model with the Adam optimizer, categorical cross-entropy as our loss function, and accuracy as an additional metric computed at each epoch. We configured several callbacks including early stopping with patience of 5 and model checkpoint that saved only the best weights in the H5 format. The number of epochs was configured to be 30,000 (terminate when the model stops improving based on the validation loss monitor).

The U-Net model was trained for 54 epochs with 49 being the best epoch, after which the model started overfitting. The final test accuracy and test loss were 95.11% and 0.139 respectively. The U-Net training process is summarized in Figure 10. Additionally, to see how U-Net is performing over the epochs, we extracted the intermediate weights from the middle epochs to see how the model improved in Figure 9. From the figure, it is clear that the first categories that the model learned were water, grassland, and forest. The hills, mountains, and desert categories were only learned in later epochs (after epoch 7) which is expected due to the class imbalances in the dataset reported in Figure 5, where water, grassland, and forest are disproportionately overrepresented in the data.

Both FCN models were trained with the same training hyper-parameters as the U-Net model, but both only learned for 7 epochs prior to early stopping which indicates severe overfitting, as seen in Figure 11. In fact, the best results for both models were achieved in epoch 1, so most overarching patterns were extracted in the first dataset run. The test accuracy and test loss for FCN-8 were 49.65% and 1.84 respectively, while test accuracy and test loss for FCN-32 were 37.86% and 1.71 respectively.

These performance results are in line with our expectations, since U-Net is extracting features more efficiently due to its large encoder module and upsampling them effectively by retrieving the lost features using the skip connection from the corresponding encoder convolutional layers. We also expected FCN-8 to perform better than FCN-32 due to more precise convolutional filters/feature maps and two skip connections that retrieve more features. FCN-32, on the other hand, does not get any additional information from the skip connections, and therefore relies only on its large 32x32 kernels to retain most of the features. We speculate that these results show that kernel size is not as good at retaining initial image features compared to the skip connections in the decoder. The previous literature also suggests that FCN-like models perform better on the high-resolution images, but since we resized our inputs from 512x512 to 128x128 for the sake of fair comparison between the models, we might have not utilized the full architectural potential of FCN. It is also worth mentioning that the inference on the test set for 500 images for U-Net took around 30 seconds and both FCNs took a little less than 6 seconds on the same machine. This indicates that the FCN architecture might be more efficient for inference, but more tests on different dataset sizes, machines, and frameworks are needed.

B. Performance Analyses

1) Quantitative Analyses: There is an obvious pattern in the confusion matrix shown in Figure 12 for U-Net, that is, most of the predictions made by U-Net are correct. This corresponds to the Precision, Recall, F1 table labeled as Table 1, where U-Net has a very high recall and F-1 score for all of the categories. From the confusion matrix, it is interesting to observe that the water, grassland, forest, and hills are often misclassified. This could be attributed to the presence of small water bodies (such as lakes) within grasslands, forests, or hills on map representations. Consequently, it is challenging to predict these small chunks accurately. Additionally, the category "Hills" has the lowest precision and F-1 score among all the categories, meaning when the model predicts hills, it is less likely the prediction is correct. The two 0's in the confusion matrix suggest that mountain and grassland are never confused by each other by the model.

For FCN-8, based on the confusion matrix shown in Figure 12 and Table 1, it becomes evident that water is consistently the only category correctly classified most of the time. All other categories have low precision, recall, and F1 scores. Notably, grassland often gets misclassified as forest or hills, while forest is frequently mistaken for water or grassland. Additionally, hills and

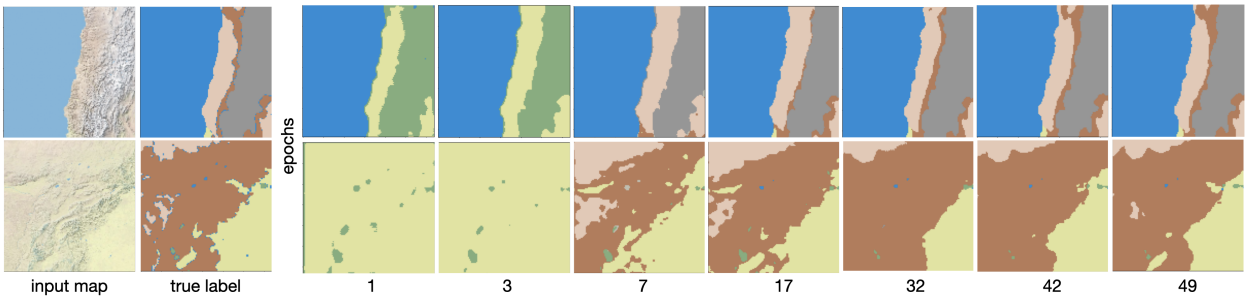

Fig. 9: U-Net Image Outputs across Epochs

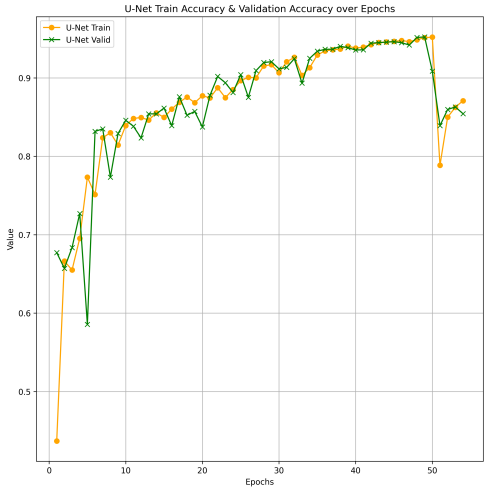

Fig. 10: U-Net Training & Validation

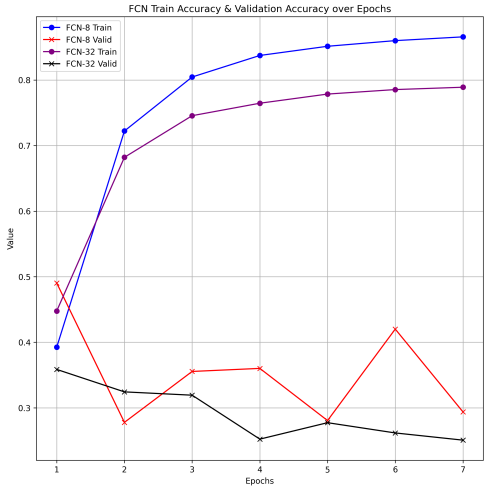

Fig. 11: FCN Training & Validation

desert tend to be misclassified as grassland, whereas mountain and tundra are commonly misclassified as water or forest. This is likely due to water being over-represented in this dataset. To improve recall and precision for other classes, we need either higher-resolution images or a larger and more balanced dataset.

The confusion matrix for FCN-32 suggests that water and desert are the only two categories that have high recalls. However, the precison for desert is very low, indicating that the model tends to predict a pixel as the desert most of the time. The 0's in the confusion matrix means desert is never misclassified as forest or grassland.

TABLE I: Precision, Recall, F1 for Three Models

Class	Precision			Recall			F1		
	U-Net	FCN-8	FCN-32	U-Net	FCN-8	FCN-32	U-Net	FCN-8	FCN-32
Water	0.99	0.75	0.77	0.95	0.78	0.85	0.97	0.77	0.81
Grassland	0.94	0.37	0.50	0.95	0.41	0.02	0.95	0.40	0.05
Forest	0.95	0.45	0.56	0.95	0.56	0.00	0.95	0.50	0.00
Hills	0.88	0.23	0.14	0.94	0.20	0.29	0.91	0.21	0.19
Desert	0.94	0.15	0.17	0.96	0.06	0.96	0.95	0.08	0.28
Mountain	0.95	0.11	0.02	0.94	0.06	0.02	0.94	0.08	0.02
Tundra	0.92	0.11	0.10	0.96	0.05	0.01	0.94	0.06	0.02
Total	0.98	0.89	0.92	0.99	0.87	0.87	0.99	0.88	0.90

2) Qualitative Analyses: The Figure 13 shows the predicted output from each model along with the true segmented image. From first glance, it is very clear that U-Net had produced an output that most resembled the true segmented images and outperformed the other two models in its prediction. The U-Net model was able to produce almost all the details within the true label. Compared to U-Net, FCN-8 and FCN-32 were not able to accurately produce a prediction that resembled the true segmentation. Unlike U-Net, the kernel size of FCN-8 is 8x8 which we can visibly observe the filter being applied in each pixel as there is a clear distinction between each 8x8 pixel. In test_img3, although it seems like a completely incorrect prediction, it still retains the general features of the label like the outline of the forest.

Similar to FCN8, FCN32 has a kernel size of 32x32. We can see the traces of the convolutional operations on the predicted image with the noise repeating in

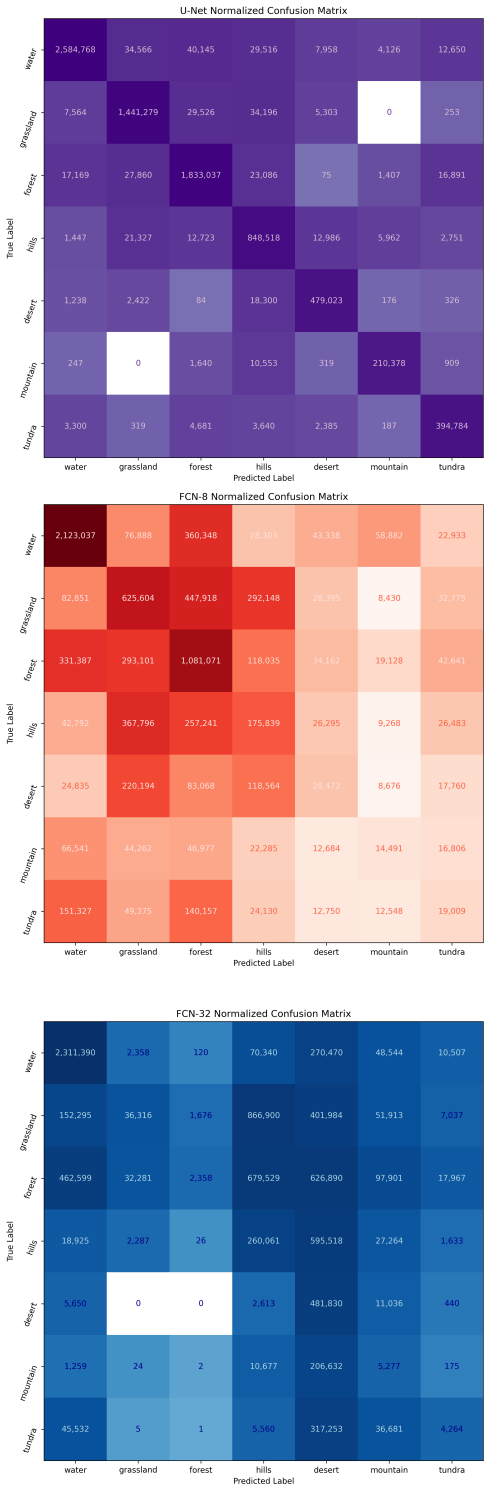

Fig. 12: Confusion Matrices

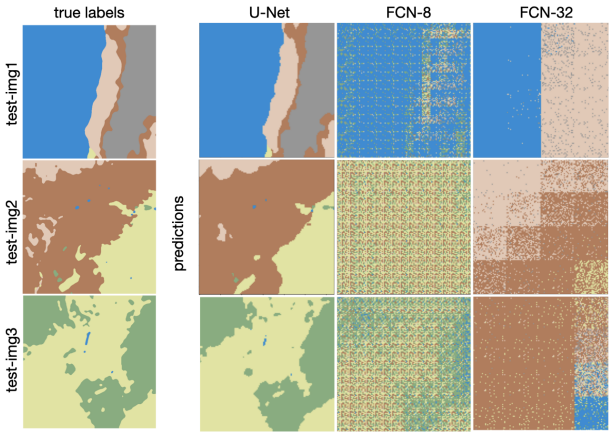

Fig. 13: Comparison of picture quality from the three models

the squares of 32x32 pixels. When observing the first prediction image of FCN-32, the main feature of the image was captured in the output as there is a clear division between water and desert; however, as for the mountain terrain, it is scattered throughout each 32x32 pixels. However, FCN-32 was not able to predict well in test_img3 as it predicted hill for most terrain and a patch of water in the bottom right of the image, even though there is little to none presence of hill and water on the true map. Overall, U-Net outperforms FCN-8 and FCN-32 in image segmentation, demonstrating superior accuracy and detail retention, while the latter models exhibit limitations due to kernel size and precision issues which are evident in Figure 13.

Convolutional Filters

To analyze the convolutional filters of U-Net and look into the patterns they produce when applied to the real input, we extracted the first two parts of the U-Net encoder [Figure 14]. The input layer was always present in the chunk we extracted, then all other model chunks included all previous encoder components and the next one included the max pooling layer. The resulting model chunks were saved and the input image was run through all three of them. We plotted the weight intensities for the first 32 filters of each model chunk to see if any patterns emerged.

The first few layers of a convolutional network usually represent the most basic features extracted from an image: edges, lighting, and basic shapes. This is what we notice at the top layer of U-Net. We see that some filters highlight the boundary between the island and the water and others separate regions that have a different color intensity in the raw input image. As the encoder further down-samples the image, the feature

maps become more obscure and specific, and their number becomes much larger. In the second encoder layer of U-Net, we see more specific features related to the image: the distinction between mountains and desert, hills and desert, etc. If we go further down, the applied filters become somewhat hard to read.

Through this analysis, we saw approximately what features the model considers important on one sample input image.

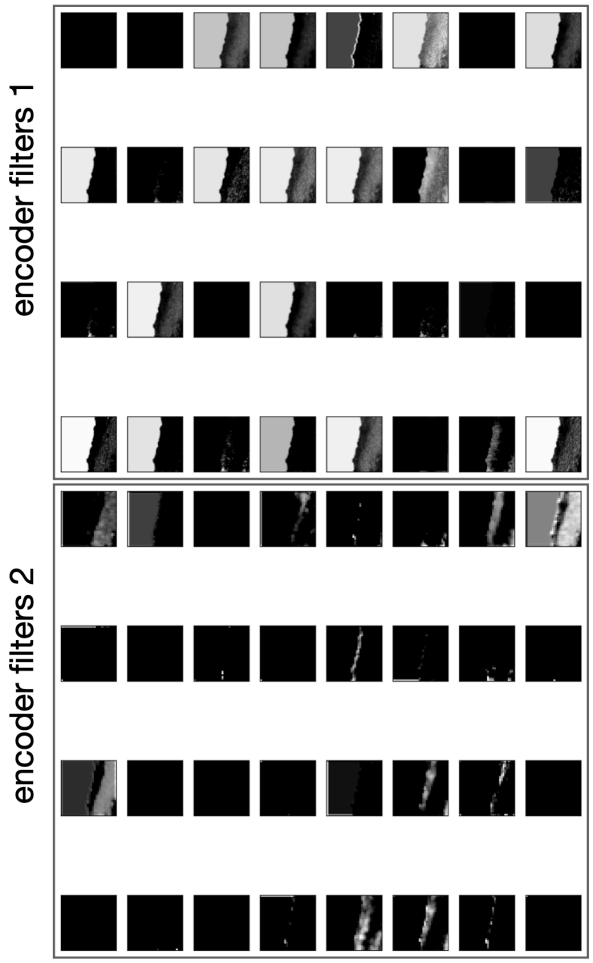

Fig. 14: U-Net Filters from the First Two Encoder Layers

IX. Conclusion

In the current project, we implemented and trained three map segmentation models: U-Net, FCN-8, and FCN-32. We conducted quantitative and qualitative analyses to examine the model performance. The results showed that U-Net performs the best with the test accuracy of 95.11%. The FCN-8 and FCN-32 had test accuracies of 49.65% and 37.86% respectively. We attribute the discrepancies between the model performance to different model architectures, dataset size, and picture size and resolution.

One of the weaknesses of our approach is that we did not use the 512x512 images for the FCN model training. According to previous studies, using higher resolution images might be helpful for the FCN performance prediction, since the convolutional layers would retain more spatial information. Another weakness is the dataset size. Though 5,000 pictures is enough to train the U-Net model, it is likely not enough for FCN, so getting a larger dataset size is necessary for future analyses. Lastly, our models are overfit to the dataset and are unlikely to perform well on the real-world data with different raw image maps.

For future work, we aim to expand the scope of our research by incorporating 512x512 high-resolution images for FCN models to enhance model accuracy and detail resolution in segmentation tasks. Additionally, we would like to find a larger dataset which would allow us to utilize 512x512 resolution input images in U-Net models, thereby avoiding the need for image size reduction. Furthermore, we would like to explore the capabilities of Generative Adversarial Networks (GANs) on this task. Lastly, to improve the applicability of our models, we intend to introduce more terrain categories. This diversification will enable our models to handle a wider range of geographic features and environmental conditions.

X. Appendix: Role of Each Member

All members of the group contribute to model development. Win and Ginny worked on the data processing, and Tolya and Sophie worked on the analyses. All members contributed to the report equally.

XI. Works Cited

[1] "First Digital Image," NIST, Mar. 2022. Available: https://www.nist.gov/mathematics-statistics/first-digital-image#: :text=Date3A201957.

[2] S. Du, S. Du, B. Liu, and X. Zhang, "Mapping Largescale and Finegrained Urban Functional Zones from VHR Images Using a Multiscale Semantic Segmentation Network and Object Based Approach," Remote Sensing of Environment, vol. 261, p. 112480, 2021. doi: https://doi.org/10.1016/j.rse.2021.112480.

[3] D. Gupta, "image-segmentation-keras" at masterdivamgupta/image-segmentation-keras," GitHub, Mar. 03, 2021. https://github.com/divamgupta/image-segmentation-keras (accessed May 03, 2024).

[4] O. Ozturk, B. Sariturk, and D. Z. Seker, "Comparison of Fully Convolutional Networks (FCN) and U-Net for Road Segmentation from High Resolution Imageries," International Journal of Environment and Geoinformatics, vol. 7, no. 3, pp. 272–279, Sep. 2020. doi: https://doi.org/10.30897/ijegeo.737993.

[5] T. Pappas, "Earth Terrain, Height, and Segmentation Map Images," www.kaggle.com, 2020. https://www.kaggle.com/datasets/tpapp157/earth-terrain-height-and-segmentation-map-images/data (accessed May 03, 2024).

[6] O. Ronneberger, P. Fischer, and T. Brox, "UNet: Convolutional Networks for Biomedical Image Segmentation," in Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds., Cham: Springer International Publishing, 2015, pp. 234–241.

[7] N. Siddique, S. Paheding, C. P. Elkin, and V. Devabhaktuni, "UNet and Its Variants for Medical Image Segmentation: A Review of Theory and Applications," IEEE Access, vol. 9, pp. 82031–82057, 2021. doi: https://doi.org/10.1109/ACCESS.2021.3086020.

[8] N. J. Singh and K. Nongmeikapam, "Semantic Segmentation of Satellite Images Using DeepUnet," Arabian Journal for Science and Engineering, vol. 48, no. 2, pp. 1193–1205, 2023. doi: https://doi.org/10.1007/s13369022067344.

[9] W. Wang, Y. Xing, L. Zhong, and X. Zhong, "An Encoder-Decoder Network Based FCN Architecture for Semantic Segmentation," Wireless Communications and Mobile Computing, vol. 2020, p. 8861886, 2020. doi: https://doi.org/10.1155/2020/8861886.

[10] J. Zhang et al., "LCUNet: A novel low-cost UNet for environmental microorganism image segmentation," Pattern Recognition, vol. 115, p. 107885, 2021. doi: https://doi.org/10.1016/j.patcog.2021.107885

[11] R. Zhao, W. Chen, and G. Cao, "Edge-Boosted UNet for 2D Medical Image Segmentation," IEEE Access, vol. 7, pp. 171214–171222, 2019. doi: https://doi.org/10.1109/ACCESS.2019.2953727.