Machine learning project 2:
Using neural networks for road segmentation of satellite images

Luca Delabarre, Thibaud Banderet, Mateo Echeverry Hoyos

***Abstract*—In this lab report, we propose the use of an U-Net model to segment roads in satellite images. The U-Net model is a convolutional neural network (CNN) that has been shown to be effective for image segmentation tasks. We preprocess the dataset by applying data augmentation techniques, which help to improve the performance of the model. The augmented dataset is then used to train the U-Net model. We evaluate the performance of the model using several metrics, including accuracy and F1 score. The results show that the U-Net model achieves an accuracy of 92.7 on the test set, demonstrating its effectiveness at segmenting roads in satellite images. In addition, we also perform a detailed error analysis to understand where the model is making mistakes and identify potential areas for improvement. Overall, our results suggest that the U-Net model is a promising approach for solving this challenge.**

I. Introduction

Road segmentation is the process of identifying roads in satellite images, which is an important task in many applications, such as autonomous vehicles and mapping services. By accurately segmenting roads in satellite images, we can improve the accuracy of these applications, which in turn can have a significant impact on the world.

For example, in the case of autonomous vehicles, accurate road segmentation is essential for the vehicle to be able to navigate safely. By correctly identifying roads in satellite images, the vehicle can more accurately plan its route and avoid obstacles, which can greatly improve its safety and reliability.

Additionally, accurate road segmentation can also improve the accuracy of mapping services, such as Google Maps. By correctly identifying roads in satellite images, these services can more accurately display road networks and provide more accurate directions to users. This can be especially important in areas where the road network is constantly changing, such as in developing countries.

Overall, road segmentation is an important task that has the potential to greatly impact the world by improving the accuracy of many applications. In this lab, we will explore methods for accurately segmenting roads in satellite images and evaluate their performance.

II. Methods

We use a convolutional network in order to work on this problem since our inputs are images, in fact since our outputs are also images we use the U-Net as a general model as it is made for image segmentation [1].

A. Preprocessing

Before training our model, we preprocess our data in the following way.

1) Data augmentation: We start by augmenting our data as we only have 100 images of 400x400 which is far from enough for a U-Net, we use the following transformations : rotations of multiples of 90 degrees, random rotations, flipping, adding gaussian or salt and pepper noise and we fill the images with black pixels so they stay rectangular. Our best working set of augmentation consists of:

- Rotations of 3 random angles, 90, 180 and 270 degrees
- Flipping horizontally and vertically
- Adding gaussian noise of two different variances (usually 50 and 200)
- Salt and pepper with a low corruption ratio (about 2%)

2) Other training set: Another way to grow our training set is by adding images from other training sets found online, namely from [2]. This set is far from clean (has a lot of images with white parts) and is not quite the same as the base training set we are given. We make it more usable by removing all images with white parts, but end up not using it as it is too different from our images to make our model more accurate.

3) From images to patches: Every image we have is 400x400 which is a big sample, it does not make much sense to give a whole image as a training item, instead we cut it into smaller patches (if the image is not a multiple of the patch size we pad it with black again). This gives us more training data without trading anything in return as long as the patches are large enough for the model to find roads easily.

B. Models

As explained before the U-Net is our general model but we try to train different variations of it we give each of them a name to talk about them easier later on :

- ”First UNET” taken from [3]
- ”Short UNET” taken from [4]
- ”Multi UNET” taken from [5]
- ”Large UNET” taken from [6]

The First Unet model was the first model we created, just like the other models, it is a basic implementation of the Unet architecture, it consists of a contracting path and an expansive path. The contracting path is made up of several convolutional and max pooling layers, which reduce the spatial dimensions of the input image while increasing the number of filters. The expansive path is made up of several convolutional and up-sampling layers, which restore the spatial dimensions of the input image while decreasing the number of filters.

The Multi Unet model uses multiple unet modules to improve the performance of the model. This is achieved by concatenating the outputs of the individual unet modules before passing them through the final layers of the model.

The Short Unet uses fewer layers and filters in the contracting and expansive paths. This makes the model faster to train and less prone to overfitting, at the expense of some loss of performance.

The Fat Unet model uses more layers and filters in the contracting and expansive paths. This makes the model more powerful and capable of achieving higher performance, but also makes it slower to train and more prone to overfitting.

On top of using these models as basis we also attempt to tweak them in multiple ways, for example by adding

dropout layers or changing their rate, adding or removing layers or changing activation functions. Note that we always give the preprocessed images with their values divided by 255 (since RGB values go to 255) to the model, hence each model is trained with the same exact input shape and type.

C. Hyperparameters

Here we elaborate on certain hyperparameters we tried varying and optimizing.

1) Training set size: As explained in II-A, our given training set is far too small to train our models well so we have to add more data to it. An important tradeoff to take care of is to not have too much augmented data as opposed to original data, which is why we tried using another set as explained in II-A2. We end up settling on augmenting our data from 100 to 1200 images without adding new original data, as this gave us the best results without over-fitting on the training set.

2) Patch size: Patch size is a very interesting hyperparameter, intuitively the best patch size would be the smallest one where it is still easy to tell a road apart from a roof of a building for example. We still try even smaller sizes such as 16x16 in order to make sure our intuition is right. The models we use also tend to work better with patch sizes as multiples of 32 so we settle on 96x96 for most of our tests.

3) Submission threshold: After training the model, we still need to go from our patch size to the required one (16x16) and give a single value for each of these patches, which we do by averaging the patch image and thresholding that average to either output 0 or 1. Intuitively a threshold of 0.5 sounds like the best option but because of our lack in training data, a lower threshold such as 0.2 tends to yield better results as our model does not seem so confident in his predictions (road pixels do not have values much higher than non road ones). This hyperparameter is important to maximize as it is crucial in getting a higher F1 score, the right ratio of false positive and false negative needs to be met.

4) Others: There are a lot more hyperparameters in our models, most of the remaining ones were optimized for training speed as they did not impact accuracy (such as learning rate, batch size, which metric to monitor while training).

III. Results

A. Performance

showcase all of the performance comparisons for hyperparameters or models or preprocessing

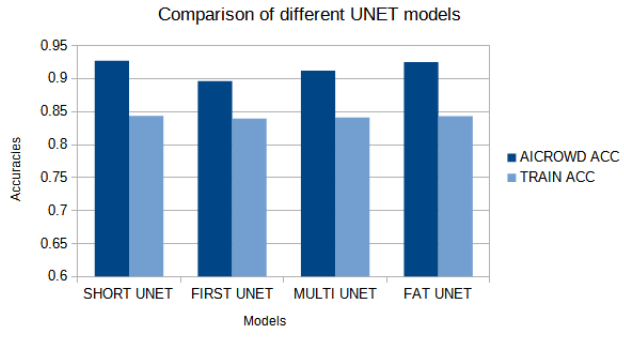

Figure 1: Accuracies of our various models, all using the same hyperparameters : training set augmented to 1200 images, patches of 96x96, submission threshold of 0.2, batch size of 16 and ADAM optimizer

The results show that the Short Unet performed the best, with an F1 score of 0.859 and an accuracy of 0.927. The F1 score is a measure of a model's performance that takes into account both precision and recall, so a high F1 score indicates that the model is able to accurately classify a large number of positive examples (i.e. roads) while also minimizing the number of false positives (i.e. non-road pixels that are classified as roads). The accuracy score measures the overall proportion of correctly classified pixels, so a high accuracy score indicates that the model is able to accurately classify a large number of pixels overall.

The First Unet, Multi Unet, and Fat Unet models all performed relatively similarly, with F1 scores in the range of 0.79 to 0.85 and accuracy scores in the range of 0.896 to 0.912. This indicates that these models are able to accurately classify a significant proportion of the positive examples and overall pixels, but not quite as well as the Short Unet model.

Overall, these results suggest that the Short Unet model is the best-performing model for this task, with a relatively high F1 score and accuracy score. However, it is also worth noting that the performance of all of the models is relatively high, indicating that all of the models are able to effectively classify roads in the satellite images.

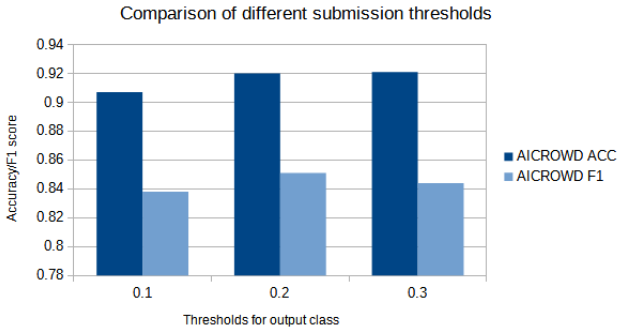

Figure 2: Accuracy and F1 score of short unet with added dropout depending on the submission threshold. The hyperparameters are : training set augmented to 1200 images, patches of 96x96, batch size of 16 and ADAM optimizer

IV. Discussion

V. Summary

References

[1] U-Net, "U-net — Wikipedia, the free encyclopedia," 2022, [Online; accessed 10-December-2022]. [Online]. Available: https://en.wikipedia.org/wiki/U-Net

[2] J. Paul, "Segmentation of roads in aerial images." 2019, [Online; accessed 10-December-2022]. [Online]. Available: https://towardsdatascience.com/road-segmentation-727fb41c51af

[3] zhixuhao, "Implementation of deep learning framework – unet, using keras," 2019, [Online; accessed 10-December-2022]. [Online]. Available: https://github.com/zhixuhao/unet/blob/master/model.py

[4] ashishpatel26, "Semantic-segmentation-keras-tensorflow-example," 2020, [Online; accessed 10-December-2022]. [Online]. Available: https://nbviewer.org/github/ashishpatel26/Semantic-Segmentation-Keras-Tensorflow-Example/blob/main/Areal_Image_segmentation_with_a_U_Net_like_architecture.ipynb

[5] D. S. Bhattiprolu, "Python for microscopists and other image processing enthusiasts," 2022, [Online; accessed 10-December-2022]. [Online]. Available: https://github.com/bnsreenu/python_for_microscopists/blob/master/228_semantic_segmentation_of_aerial_imagery_using_unet/simple_multi_unet_model.py

[6] zhixuhao, "Implementation of deep learning framework – unet, using keras," 2019, [Online; accessed 10-December-2022]. [Online]. Available: https://github.com/zhixuhao/unet/blob/master/model.py