U-Net like architecture for Road Extraction

Anca Badiu, Paul Guillon, Endrit Vorfaj
CS-433: Machine Learning, EPFL, Switzerland

***Abstract*—Road extraction from satellite imagery is essential for applications such as urban planning and infrastructure development. In this project, we propose a machine learning-based approach to extract road networks from high-resolution satellite images, producing accurate representations of road structures.**

The process begins with pre-processing high-resolution satellite images to reduce noise and enhance clarity through normalization and contrast adjustment. We use a lightweight U-Net, a convolutional neural network (CNN) architecture, to segment road networks. The lightweight U-Net's encoder-decoder structure with skip connections enables precise segmentation by learning from ground truth masks while maintaining computational efficiency.

The proposed method is evaluated using multiple satellite imagery datasets in various regions and conditions. Performance is assessed using precision and F1-score, which demonstrate the accuracy and reliability of the approach.

Overall, our methodology provides an efficient solution for road extraction, offering valuable insights for urban planning, transportation management, and emergency response. By providing accurate road maps, this approach supports improved spatial data analysis and decision-making processes.

I. Introduction

High-resolution satellite imagery has revolutionized fields such as urban planning, transportation management, and disaster response. Road extraction plays a critical role in these areas by enabling the creation of accurate and up-to-date road maps. However, it remains a challenging task due to varying land cover types, environmental conditions, and imaging artifacts.

Traditional road extraction methods are often manual or rely on basic image processing, making them time-consuming and error-prone. With advancements in machine learning, particularly deep learning, automated approaches have become more accurate and scalable. In this project, we develop a lightweight U-Net-based method for road extraction. U-Net's encoder-decoder structure with skip connections enables precise segmentation of road networks while maintaining computational efficiency by learning from ground truth masks.

As part of this student project, we also use the 100 images provided in the dataset to evaluate our approach. While the outcomes are primarily academic and do not have direct real-world applications, they demonstrate the potential of automated methods for road extraction. We assess our model's performance using metrics such as precision and F1-score to ensure accuracy and reliability.

The results provide insights into the feasibility of machine learning for road extraction, contributing to ongoing research efforts in satellite image analysis and segmentation.

II. Ethical risks

The main benefit offered by our solution is the ability to automatically extract roads from satellite images, which is especially useful for areas that are difficult to access or change often. However, this benefit should not overshadow the ethical risks that such a solution can cause. To address this ethical aspect, we will focus on the sustainability of our solution. It is a well-documented fact that training large neural networks - a computationally heavy task - requires a lot of energy [1]. This energetic aspect is more important than ever, as global warming and its effects are central issues affecting everyone in today's world. In particular, because the resources consumed by our computing technologies are not directly visible when using them, this ecological issue is often underestimated in the field of computer science. Of course, while our model doesn't have the scale of large language models such as GPT-4 and its trillions of parameters, we still believe that smaller models should take this question seriously. To mitigate the impact of our project, we will try to reduce its carbon footprint, as CO2 emissions are one of the main causes of climate change. Most of the ecological impact of our solution will happen during development, in the training phase of the model. To monitor the carbon footprint of this step, we use the *carbontracker* [2] library. This tool gives an estimate of the energy consumption and CO2 emissions of the training process, as well as what distance driven by car it corresponds to. Furthermore, to mitigate the carbon footprint of our model, we chose to use a lightweight version of the selected U-Net architecture, reducing the number of parameters of the model. The goal is to reduce training time and energy consumption. In the end, our training phase performed on Google Colab took around 20 minutes, and *carbontracker* reported an energetic cost of 0.033 kWh, equivalent to 15.8 g of CO2. This corresponds to around 150 meters traveled by car which seems totally acceptable.

III. Dataset Description

For this project, we were provided with a dataset of 100 RGB training images each of size 400x400 pixels. Initially, we considered using the DeepGlobe 2018 Road Extraction dataset [3], which is designed for semantic segmentation

tasks and has applications in urban planning, geospatial analysis, and the environmental industry. The dataset consists of 8570 images with pixel-level annotations for a single class: roads. Of these, 6226 images are labeled, while 2344 images remain unlabeled. The dataset is split into three subsets: train (6226 images), validation (1243 images), and test (1101 images). It was released in 2018 by Facebook, DigitalGlobe, CosmiQ Works, Wageningen University, and The MIT Media Lab. For this project, we only considered the 6226 labeled images and their adequate masks. After running the lightweight U-net model on the augmented dataset and on the original one, we noticed the F1 score didn't improve and we suspect the reason for it is that the images from DeepGlobe originally were of size 1024x1024 and had to be resized to match the size of the given dataset. Furthermore, since we had to use Colab and we wanted to minimize the carbon print of our model, we decided to only train on the provided 100 images and not use DeepGlobe at all. We mention it here and give a reference for those people who want to train our model on a different and maybe larger dataset.

IV. Model description

A. Baseline model

As a baseline model, we decided to start with a simple logistic regression model, trained on the 100 provided images. The images are split into patches of 16x16 pixels each. We then extract 3 features for each patch: median, mean, and variance of the pixels. Since the images are RGB, there are 3 channels per patch, which gives a total of 9 features per patch. To enrich our set of features, we use polynomial expansion of degree up to 2 and end up with a total of 55 features. Finally, we train a logistic regression to predict the class of each patch (road or background) and recombine the predicted patches into our final mask. With an 80-20 train-validation split, this model achieves an F1-score of 0.55 on the validation set, with 0.77 recall and 0.67 accuracy. These results are already pretty good for such a simple model, and our aim is to beat them with a more complex approach. See Figure 1 for an example output of the baseline model. The red parts represent where the model detects a road.

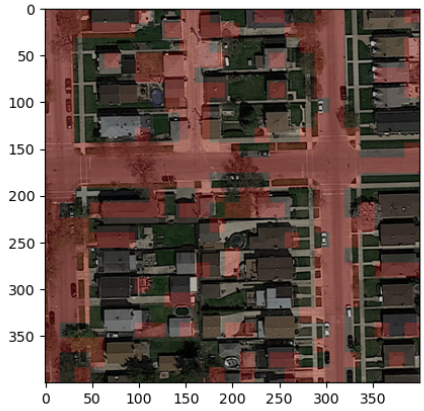

Figure 1. Example output from the baseline model

B. Lightweight U-Net

To achieve better results, we decided to go with a U-Net architecture. U-Net models are convolution neural networks specialized in image segmentation. They also have the advantage of achieving good results with only a few training samples [4]. Our implementation is a lightweight version of this architecture which is computationally efficient as its architecture combines the principles of the U-Net design with depthwise separable convolutions [5] to reduce the model's parameter count and computational overhead, making it suitable for use in resource-constrained environments such as our case with only 100 training images. We will now briefly describe the architecture of the proposed model: each convolutional layer is replaced with a depthwise separable convolution, which splits the convolution process into two stages:

- *Depthwise convolution*: Performs spatial filtering independently on each input channel.
- *Pointwise convolution*: Combines channel-wise information using 1×1 convolutions.

This significantly reduces the number of parameters and computations while maintaining performance. The encoder extracts hierarchical features from the input image. It consists of four *Encoder Blocks*, each comprising of two depthwise separable convolutional layers with batch normalization and ReLU activation as well as a max-pooling layer for spatial downsampling. The decoder reconstructs the output resolution using *Decoder Blocks*. Each block consists of:

- An upsampling operation via a transposed convolution to double the spatial resolution.
- A concatenation of the upsampled feature map with the corresponding skip connection from the encoder.
- Two depthwise separable convolutional layers with batch normalization and ReLU activation.

Output Layer: The final layer is a 1×1 convolutional layer that maps the decoder's output to the desired number of classes for pixel-level prediction.

V. Parameter Selection

For the lightweight model, the main challenge was to extract the optimal values for epochs, learning rate, and threshold values for the predicted masks. We started with an analysis of the effect of the learning rate on the F1 score and as we can see in Figure 2, the best learning rate is 0.001. Upon establishing this first parameter, we then focused on trying to find the optimal number of epochs so that we minimize the carbon print of the training process. As we can

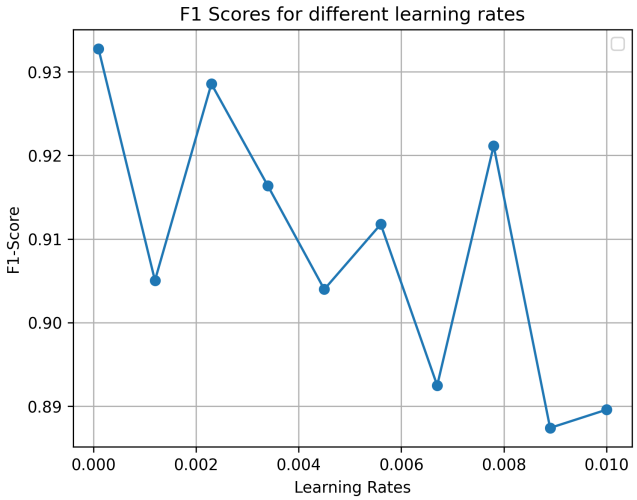

Figure 2. F1 Score for different learning rates with constant epochs number

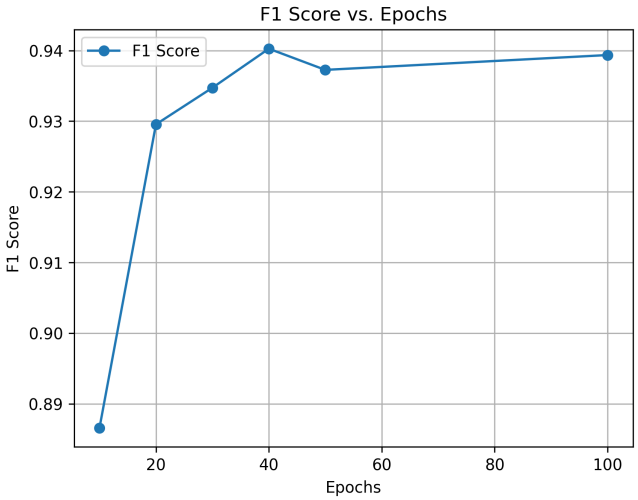

Figure 4. F1 Score for different epochs

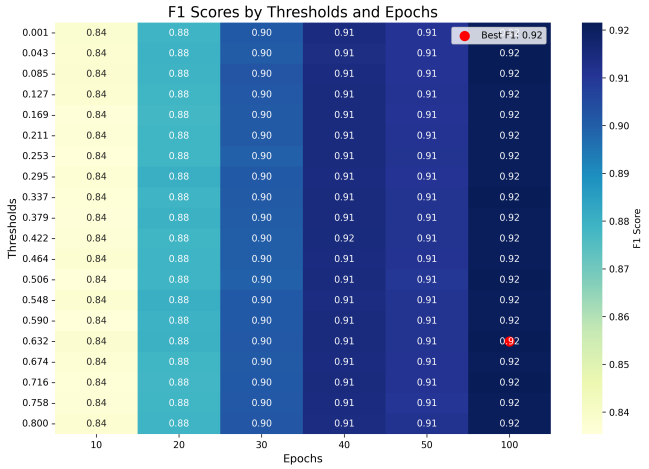

Figure 3. Best parameters

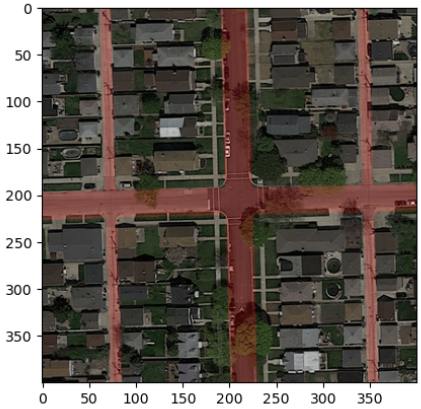

Figure 5. Example output from the final model

see in Figure 4, running the model for 40 epochs produces a higher F1 score than running it for 50. However, we noticed that running the model for 100 epochs results in a better F1 score on the AiCrowd platform even though on the validation set, the scores are close. We thus decided to go with 100 epochs to provide a maximum F1 score. Finally, we tried to establish the best threshold for a higher F1 score, and in Figure 3 we can see that the optimal threshold is 0.632. In the figure, we don't see the differences between the F1 scores due to the number of decimal places but the one highlighted in red is the maximum F1 score. So in the final version, we have a model that runs for 100 epochs with a learning rate of 0.001 and the threshold for the predictions is 0.632.

VI. Results

The model takes about 20 minutes to run on Google Colab and produces 50 predictions for the provided 50 test images. The F1 score in AiCrowd is 0.845 while the local one on the validation set is 0.92. Each prediction is a black-and-white mask where the white pixels represent the roads and the black represents anything else in the test image. See Figure 5 for an example output of the final model. The red parts correspond to the pixels labeled as roads by the model.

VII. Conclusions

In this report, we analyzed the dataset from DeepGlobe as well as our provided dataset of 100 images and ground truth. We explained the training process and how we came to select the best parameters. We also introduced and talked about the ethical risks of this project and ML projects in general. We provided some details on the baseline compared to lightweight U-Net approach that we used. Based on the current feedback, we believe that expanding the original dataset with the DeepGlobe datasets will yield more accurate

results that can be efficiently applied in different research areas, leading to new exciting developments at the intersection of computer vision, machine learning, remote sensing, and geosciences.

REFERENCES

[1] D. Patterson, J. Gonzalez, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. So, M. Texier, and J. Dean, "Carbon emissions and large neural network training," 2021. [Online]. Available: https://arxiv.org/abs/2104.10350

[2] L. F. W. Anthony, B. Kanding, and R. Selvan, "Carbontracker: Tracking and predicting the carbon footprint of training deep learning models," ICML Workshop on Challenges in Deploying and monitoring Machine Learning Systems, July 2020, arXiv:2007.03051.

[3] I. Demir, K. Koperski, D. Lindenbaum, G. Pang, J. Huang, S. Basu, F. Hughes, D. Tuia, and R. Raskar, "Deepglobe 2018: A challenge to parse the earth through satellite images," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.

[4] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1505.04597

[5] A. G. Howard, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.