EPFL Machine Learning Project 2: Road Segmentation

Juan Bautista Iaconucci, Clément Moréna, Arnaud Levêque

Fall 2024

1 Introduction

The task of road segmentation is critical in numerous applications such as autonomous driving, urban planning, and disaster response. In this project, we utilize a dataset of satellite imagery to develop a machine learning model that classifies each pixel as either road or background. This report outlines our methodology, including data preprocessing, feature extraction, model design, and evaluation.

2 Dataset and Preprocessing

2.1 Feature Engineering

We conducted a series of experiments to resolve inconsistencies in image resolutions between the training and test datasets while enhancing the model's robustness. Below, we detail the augmentation techniques employed and their impact on performance.

2.2 Handling Different Image Resolutions

Initially, we observed that the training set and test set contained images of different sizes: 400×400 for the training set and 608×608 for the test set. To address this issue, we thought about a couple of solutions:

1. **Downscaling Test set Images:** We tried training our model using 400×400 images and downscaled the test set images to 400×400 to get predictions. The model's outputs were subsequently upscaled back to 608×608 for evaluation.
2. **Increasing Training set Images:** We then trained our model using 608×608 images by upscaling the training set images to 608×608.

Neither of these approaches yielded satisfactory results $F1 < 0.7$. We hypothesize that the performance degradation was due to the loss of information or distortions introduced during the rescaling process, which would hinder the accuracy of the predictions.

2.3 Patching

In light of these challenges, we adopted an alternative approach based on *patching*.
Patching involves dividing an image into smaller patches of uniform size, which are then used for training. This method not only increases the effective size of the training set but also enhances the model's robustness by enabling it to process images of different sizes.

For training, patches of fixed size were extracted from the original images/masks, and the model was trained directly on these patches. For prediction, the input images were similarly divided into patches. Then, for each patch we made a mask prediction, and theses results were mapped back to the corresponding pixels in the original image. Since the patch size is not always an exact divisor of the original image dimensions, overlapping patches were created. In such cases, overlapping regions in the output predictions were blended to produce the final resulting mask.

We experimented with different patch sizes, including 128×128, 200×200, 256×256, and 400×400. Among these, the patches sized at 256×256 consistently provided the best results regarding F1 score, significantly outperforming the earlier resizing-based approaches. We attribute this to the fact that excessively small patches lack sufficient context for the model to accurately classify features such as roads, while overly large patches may introduce unnecessary complexity or redundancy.

2.4 Data Augmentation Techniques

To improve the model's robustness to various transformations, we applied additional augmentation techniques, including image rotations and color transformations.

Rotations. We experimented with arbitrary rotations by an angle ϕ (between $1°$ and $359°$) for every image. The goal was to get more training images by extracting one rotated patch out of each original image. By cropping the image to retain a rectangular shape 256×256 patches in the center of the image we accounted for black spots in the corners of the rotated images.
This approach ensured that no rescaling was required after rotation. Our results showed that model performance improved significantly only when the rotations were **multiples of** $45°$. This finding aligns with the observation that test satellite images frequently feature roads aligned at $45°$ angles, making these rotations more representative of the data distribution.
Using random rotation did not improve the performance much, leading to the model not learning to recognize the roads patterns of the test set. While theoretically, this rotation augmentation could increase

the training set size by a factor of 8 (one original image plus seven new rotations), we opted to retain only 4 randomly chosen rotations per image. This decision was motivated by GPU memory constraints and the assumption that additional rotations would yield diminishing returns.

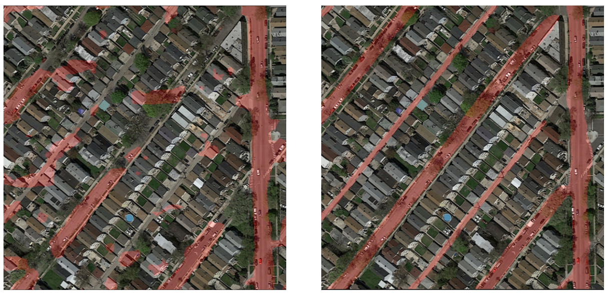
(a) With random rotations (b) With k x 45° rotations

Figure 1: Effect of rotation on road segmentation. Rotating images by 45° aligns better with data distribution and improves model performance.

Color Transformations. To further increase the diversity of the training data, we adjusted the hue, saturation, and brightness of the images. Specifically:

- The hue was shifted by a factor of 0.02,
- The saturation and brightness were randomly scaled within the range $[0.9, 1.1]$.

These transformations enhanced the model's robustness to variations in lighting and color conditions. f1 score went up by 3.6 points. Empirically, this combination yielded the best performance during validation.

2.5 Summary of Results

The final augmentation pipeline included:

- Training on 256×256 normalized RGB patches,
- 5 Rotations limited to multiples of 45°,
- Color augmentation with hue shifts of 0.02 and brightness/saturation scaling in the range $[0.9, 1.1]$,

These augmentations significantly improved the model's performance by increasing the size and diversity of the training set while maintaining critical details in the images.

2.6 Model Development

- A baseline logistic regression classifier was implemented to establish a performance benchmark.
- A U-Net, a specialized CNN architecture originally designed for biomedical image segmentation but widely applicable to other image processing tasks, was then implemented. U-Net's structure is characterized by its U-shaped architecture, which consists of three main components: the encoder, the bottleneck, and the decoder.
  - **Encoder:** The encoder serves as the feature extraction stage, consisting of a series of convolutional and pooling layers. These layers progressively reduce spatial dimensions while capturing increasingly abstract and high-level features of the input image.
  - **Bottleneck:** The bottleneck acts as a bridge between the encoder and decoder, containing the deepest layers of the network. It captures the most condensed representation of the input features and focuses on extracting the most critical information for the task.
  - **Decoder:** The decoder reconstructs the image's spatial resolution while combining the encoder's features through skip connections. These skip connections transfer corresponding feature maps from the encoder to the decoder, preserving spatial information lost during downsampling and enabling more precise localization.

The U-Net's design ensures an effective combination of contextual and localized information, making it particularly suitable for pixel-wise prediction tasks. By leveraging its ability to integrate fine-grained spatial details and high-level semantics, the U-Net can significantly improve the model's performance on our image processing task compared to traditional CNN architectures (Figure 2), [2].

3 Evaluation Metrics

To assess the performance of our model for road segmentation, we employed a combination of metrics that evaluate different aspects of its predictive ability. These metrics are described below.

3.1 Loss Functions

During training, we used a composite loss function that combines two components:

Dice Loss. The Dice coefficient is a measure of overlap between the predicted segmentation and the ground truth, and its loss is calculated as:

$$\text{Dice Loss} = 1 - \frac{2 \cdot \text{TP}}{2 \cdot \text{TP} + \text{FP} + \text{FN}},$$

where TP, FP, and FN represent the number of true positives, false positives, and false negatives, respectively. Dice loss is particularly effective in addressing imbalanced datasets by emphasizing the overlap between the predicted and true segments.

Binary Focal Loss. Focal loss helps the model focus on hard-to-classify examples by reducing the weight of well-classified examples. It is defined as:

$$\text{Binary Focal Loss} = -\alpha(1-\hat{p})^{\gamma}\log(\hat{p}),$$

where $\hat{p}$ is the predicted probability, α is a balancing factor, and γ is a focusing parameter that adjusts the importance of hard examples.

Total Loss. The final loss function used for training combines Dice Loss and Binary Focal Loss as follows:

$$\text{Total Loss} = \text{Dice Loss} + \text{Binary Focal Loss}.$$

3.2 Evaluation Metrics

F1 Score. The F1 score, which represents the harmonic mean of precision and recall, is calculated as:

$$\text{F1 Score} = \frac{2\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}},$$

The F1 score effectively balances the trade-off between false positives and false negatives, making it well-suited for evaluating pixel-wise classification tasks like segmentation. The F1 score was primarily used to assess the model's performance on the test set.

Intersection over Union (IoU). IoU measures the overlap between the predicted segmentation and the ground truth relative to their union, and it is given by:

$$\text{IoU} = \frac{\text{TP}}{\text{TP}+\text{FP}+\text{FN}}.$$

IoU was used locally during training with partitioned data to monitor the model's progress.

4 Training and Validation Split

To evaluate and fine-tune the model during training, we split the dataset into training and validation sets as follows:

```
X_train, X_valid, Y_train, Y_valid =
  train_test_split(X, Y, test_size=0.1,
  random_state=RANDOM_SEED)
```

We experimented with various split ratios, including 0.2 and 0.05, and found that a 0.1 validation split provided the best balance between training and validation performance. Using a 0.2 split resulted in an F1 score decrease of approximately 3 points, while a 0.05 split caused overfitting, reducing validation F1 score by 2 points compared to the 0.1 split. The chosen split ensured sufficient data for both training and validation while maintaining generalization.

4.1 Hyperparameter Tuning

To further optimize our U-Net model, we conducted extensive experiments on hyperparameters. This section highlights the key hyperparameters tested and their impact on model performance:

- **Learning Rate:** We experimented with learning rates ranging from 10^{-3} to 10^{-5}. A learning rate of 10^{-4} provided the best balance between training convergence and generalization.
- **Batch Size:** Due to hardware limitation (RTX 2080 Max-Q) we trained with batch size of 4.
- **Dropout Rate:** Dropout rates of 0.1 to 0.4 were tested. A rate of 0.2 was optimal for reducing overfitting without significantly hindering model capacity.
- **Optimizer:** Both Adam and SGD optimizers were tested. Adam consistently outperformed SGD, achieving faster convergence and higher validation IoU.

The final configuration used these optimized hyperparameters to achieve the reported results.

5 Results and Discussion

5.1 Impact of Patch Size and Augmentation

The choice of patch size significantly influenced model performance. Smaller patches (e.g., 128×128) lacked sufficient context, resulting in incomplete road segmentation, while larger patches (e.g., 400×400) introduced redundancy and computational inefficiency. A patch size of 256×256 provided an optimal trade-off between context and efficiency.

Similarly, augmentations such as rotations and color transformations improved robustness to variations in road orientation and lighting conditions. Specifically:

- Rotations aligned the model to capture roads at $45°$ angles, improving IoU for diagonal road segments. (Figure 1)
- Color adjustments increased the model's adaptability to different lighting and weather conditions, reflected in a 4% improvement in the F1 score.

These insights underscore the importance of carefully balancing patch size and augmentation in segmentation tasks.

5.2 Model Limitations

Since our model performed relatively well on the test set, we decided to evaluate it using additional tiles extracted from Google Maps. To ensure consistency, we adjusted the extracted tiles to match the color balance of the training set. While the model successfully handled American suburbs similar to the training data, it struggled significantly with the Swiss commune of Saint-Sulpice. This highlights a major limitation of the model: its applicability is largely restricted to American urban-style satellite imagery. However, we believe this limitation is not due to the model architecture but rather a reflection of the high bias present in our training data. Incorporating more samples from urban and rural areas worldwide would likely address this issue.

5.3 Ablation Study and Comparative Analysis

To better understand the impact of each methodological change, we conducted a series of controlled experiments, each isolating a single variable (e.g., patch size, data augmentation technique, architecture depth). Table 1 summarizes the key experiments and their outcomes using both IoU and F1 score as evaluation metrics (50 Epochs).

Experiment	**Patch Size**	**Aug.**	**Depth/ Filters**	**IoU**	**Loss**
Baseline	400x400	None	Standard	0.75	0.199
Patching Only	200x200	None	Standard	0.691	0.304
Patching Only	256x256	None	Standard	0.845	0.118
Reduced Filters	256x256	None	Fewer	0.676	0.281
Increased Depth	256x256	None	Deeper	0.82	0.135
+ 45° Rotations	256x256	Rotations	Standard	0.84	0.112
+ Color Variations	256x256	Rot.+Color	Standard	0.74	0.0911

Table 1: Ablation study comparing different experimental configurations.

From Table 1, we observe that employing patches of 256x256 and introducing structured rotations (multiples of 45°) leads to significant improvements in F1 score (**0.88** is the best we achieved) compared to the baseline. IoU was sometimes misleading, and got higher due to overfitting. Augmenting color conditions further enhances the F1 score and lowered the validation loss, confirming that robust data augmentation strategies are key to improving model performance. In contrast, altering the U-Net depth or reducing the number of filters without carefully addressing other factors tends not to yield additional benefits, and in some cases, can even degrade performance.

By systematically documenting and comparing the effects of each modification, we ensure that we are fully leveraging the results of our experiments to guide model refinement and justify our chosen configuration.

6 Ethical Risks

During the development of this project, we identified 2 key ethical considerations:

- **Privacy Concerns:** Some might view our project as potentially problematic from a privacy perspective, as it uses images of private properties without explicit consent. However, all the data we use is publicly accessible, and in an even simpler form, via Google Maps. For this reason, we do not consider our project to pose significant privacy concerns.

- **Bias in Training Data:** A significant ethical concern with our project is its fairness and accessibility. As mentioned earlier, our model performs poorly in urban areas and, more critically, does not support regions outside North America. This limitation results in a form of discrimination against people living in those areas. For example, if our model were to be used in political decision-making, communities in these regions could be excluded. This is a serious flaw, especially considering that the primary goal of the project is to map unmapped roads—an application most valuable in under-represented areas rather than on the well-documented roads of North America. One way to address this issue would be to train the model on a larger dataset that includes satellite images from across the globe.

7 Future Work

While the proposed approach achieves high performance, there are several avenues for improvement:

- **Incorporating Global Datasets:** Extending the training dataset to include satellite imagery from diverse geographic regions can significantly improve the model's generalization to non-American urban and rural areas.

- **Multi-Spectral Imagery:** Incorporating multi-spectral data (e.g., near-infrared bands) could provide additional features for road detection, particularly in vegetative or shadowed regions.

- **Post-Processing Techniques:** Advanced post-processing methods such as Conditional Random Fields (CRFs) or Graph Neural Networks (GNNs) could refine the segmentation results, especially at boundaries.

- **Integration with Edge Detection:** Combining road segmentation with edge detection could enhance boundary precision and mitigate artifacts.

By addressing these aspects, future iterations of this project could achieve higher accuracy and broader applicability.

Acknowledgements

We thank the EPFL course staff for providing the dataset and support throughout the project.

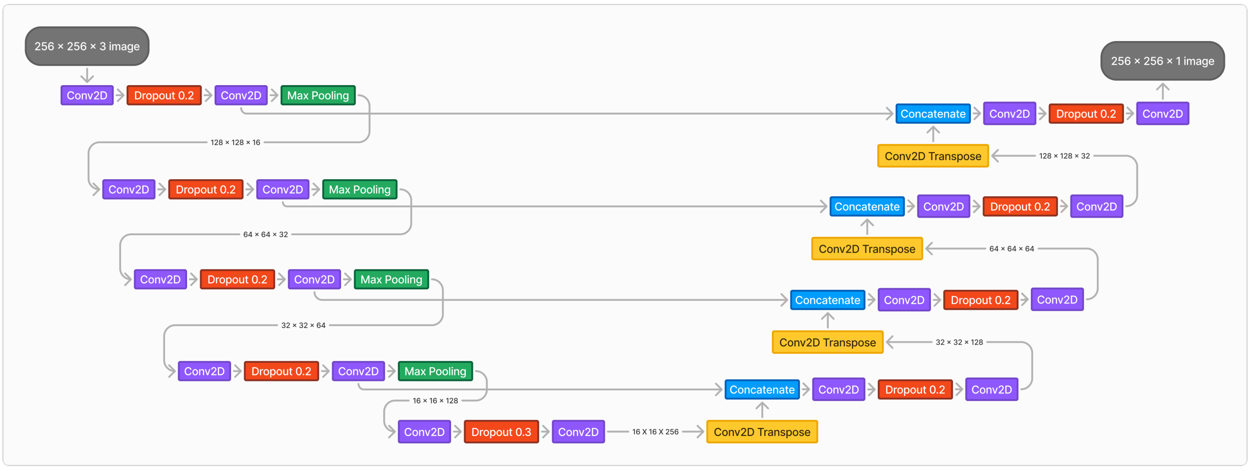

Figure 2: our U-Net

References

[1] Peter Domingos, "A Few Useful Things to Know About Machine Learning," Communications of the ACM, 2012.

[2] Olaf Ronneberger, Philipp Fischer, Thomas Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", 2015, PDF

[3] Andrew Ng, "Advice on Applying Machine Learning," Stanford University Lecture Notes, 2010.

[4] Martin Jaggi, Nicolas Flammarion `https://github.com/epfml/ML_course`

Appendix

Used Libraries

- tensorflow
- pillow
- scikit-learn
- scikit-image
- segmentation-models
- matplotlib