Architectural Enhancements to U-Net for Road Segmentation

Yoann Lafore, Sacha Rolle and Nicholas Thole
EPFL, Lausanne, Switzerland
{yoann.lafore, sacha.rolle, nicholas.thole}@epfl.ch

***Abstract*—We propose two approaches to enhance U-Net for road segmentation in satellite imagery: a cascaded U-Net used as a refinement module and the integration of ResNet encoders. Cross-validated experiments show that both approaches outperform the baseline when trained for 300 epochs. Among them, a pretrained ResNet-34 backbone achieves the highest F1 score (0.8904) on validation data and reaches 0.915 on the competitive test set.**

I. Introduction

Image segmentation is a fundamental task in computer vision aiming at partitioning an image into meaningful regions or segments. In the context of autonomous driving, urban planning, and geographic information systems, accurate road segmentation from satellite imagery plays a critical role for accurate map generation and infrastructure analysis. This project addresses the challenge of road segmentation by exploring architectural improvements to the U-Net model [1], a widely adopted encoder-decoder architecture known for its effectiveness in biomedical and satellite image segmentation tasks. We explore several strategies to enhance the baseline model, including:

1) Adding a refinement model: a second U-Net trying to correct the first predicted mask.
2) Using ResNet backbones instead of the encoding part of the baseline U-Net.

Each proposed improvement is evaluated by comparing its F1 score performance to the baseline U-Net model.

Beyond technical performance, automated road segmentation systems raise important ethical considerations. A dedicated section addressing these ethical aspects can be found at the end of the paper.

II. Exploratory Data Analysis

The dataset used in this project consists of 100 high-resolution satellite images from the Massachusetts Road Dataset. Each image has dimensions of 400×400 pixels and is paired with a corresponding binary ground truth mask of the same size, where road pixels are labeled distinctly from the background. These satellite images capture diverse urban and suburban scenes containing a variety of features, including roads, buildings, railway tracks, parking lots, and other miscellaneous structures. Notably, the testing images provided for model evaluation have dimensions of 608×608 pixels, which deviates from that of the training dataset.

III. Data Augmentation

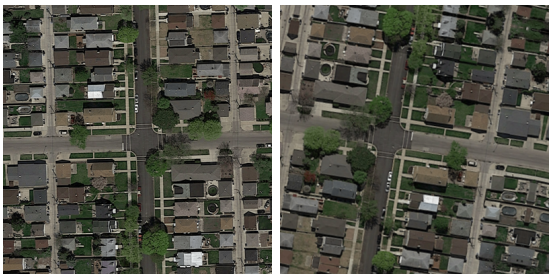
Due to the limited size of the training dataset, data augmentation is employed to mitigate overfitting and improve model robustness. The same augmentation pipeline is applied across all evaluated models and consists of geometric and colorimetric transformations.

Geometric augmentations are used to increase invariance to scale, orientation, and minor spatial variations commonly observed in satellite imagery. Specifically, images undergo random resizing and cropping, followed by random horizontal and vertical flipping and 90° rotations. Additionally, small random affine transformations are applied to account for local distortions. Each transformation is applied independently with a probability of 0.5, ensuring a diverse and uniform exploration of geometric variations.

To enhance robustness to varying imaging conditions, colorimetric augmentations are also applied. These include random adjustments of brightness and contrast, as well as variations in hue, saturation, and value.

After augmentation, images are converted to PyTorch tensors for model processing. Augmentations are applied exclusively to the training set; validation images are only converted to tensors to ensure an unbiased and representative evaluation.

To guarantee consistency between images and their corresponding segmentation masks, augmentations are implemented using the `albumentations` library, which natively supports synchronized transformations. The full augmentation pipeline is detailed in the appendix A.



(a) Before augmentation (b) After augmentation

Fig. 1: Sample image before and after augmentation.

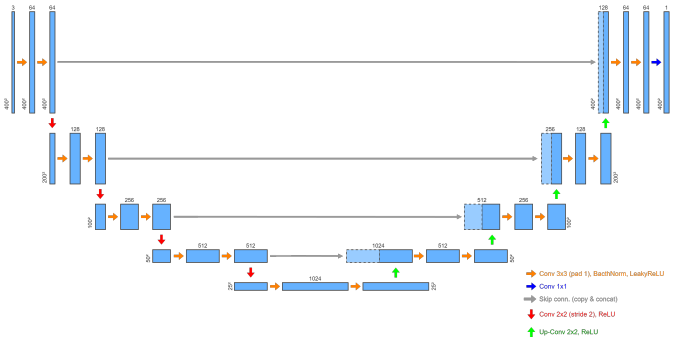


Fig. 2: U-Net architecture used in experiments.

IV. Models

A. U-Net

As our objective is to study architectural modifications tailored to U-Net–based models, we adopt the standard U-Net architecture as our primary baseline. U-Net is a widely used and well-established architecture for dense, pixel-wise segmentation tasks, making it a natural reference point for evaluating the impact of architectural changes in the context of road segmentation.

Originally introduced for biomedical image segmentation, U-Net is specifically designed for dense prediction tasks [1]. It follows an encoder–decoder structure composed of three main components. The encoder progressively downsamples the input using convolutional layers, while preserving intermediate feature maps for skip connections. The bottleneck captures high-level semantic representations. The decoder then progressively restores the spatial resolution through upsampling operations, combining high-level features with corresponding encoder features via skip connections. This design enables precise localization while retaining global contextual information.

U-Net has demonstrated strong performance in segmentation tasks involving thin, connected structures and limited annotated data, which closely resemble the challenges of road segmentation in satellite imagery. Furthermore, as the architecture is fully convolutional, it naturally supports inputs of arbitrary spatial dimensions.

In our implementation, we introduce minor modifications to the original architecture. Specifically, strided convolutions are used for downsampling instead of max pooling to allow learnable spatial reduction and improved feature preservation [2]. Convolutions are applied with a padding of 1 to preserve spatial dimensions throughout the network. Additionally, LeakyReLU activations (slope 0.01) are employed to facilitate gradient flow and improve convergence, and a dropout layer with a probability of 0.5 is applied at the bottleneck to improve robustness during training. Finally, no explicit input normalization is applied, as normalization is handled internally by Batch Normalization layers. The final architecture is illustrated in Fig. 2.

B. ResNet-backboned U-Net

The previously implemented U-Net already provides a strong baseline, but still fails in specific cases such as scenarios requiring long-range spatial relationships, ambiguous regions (e.g., parking areas), or the segmentation of small roads.

To address these limitations, we explore architectural modifications of the base U-Net. Our first approach consists in replacing the encoding part of the U-Net with the encoding layers of a ResNet [3]. To accommodate dimensional mismatches between encoder and decoder feature maps, bilinear interpolation is used to align spatial resolutions.

A ResNet is composed of residual blocks, where each block learns a transformation that is added to its input through an identity skip connection. This key characteristic may help preserve information across layers and improve road connectivity in the segmentation output.

Several ResNet variants exist, mainly differing in depth and number of parameters. In this work, we evaluate ResNet-18 (11.7M parameters), ResNet-34 (21.8M parameters), ResNet-50 (25.6M parameters), and ResNet-101 (44.5M parameters). While these architectures are similar overall, they differ in the number and type of residual blocks, as illustrated in Fig. 3.

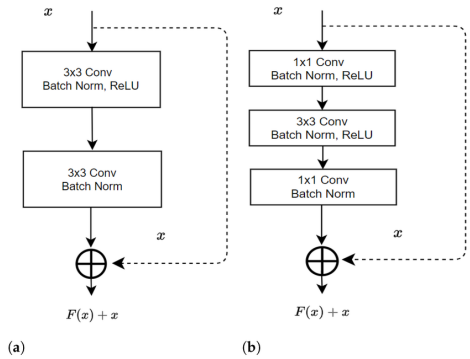


Fig. 3: Basic blocks for (a): ResNet-18,34 and (b): ResNet-50,101 [4]

C. Refinement Module

The main idea of this approach is to introduce a second U-Net placed after the first one (referred to as the *base* model), whose role is to refine the predictions produced by the base network.

The refinement module does not have access to the original input image. Instead, it operates only on a limited set of features derived from the base U-Net. As a result, it is encouraged to learn a more global and structural representation of roads, such as connectivity and continuity, rather than relying on low-level visual cues.

We evaluate two variants of this refinement strategy. In the first variant, the refinement module takes as input only the prediction map generated by the base model, resulting in a single-channel input. In this setting, the refinement module must rely on a very high-level representation. In the second variant, the refinement module receives both the prediction map and the final-stage feature maps of the base U-Net, leading to a total of

65 input channels (64 feature channels + 1 prediction channel). This allows us to assess whether providing additional feature information improves the refinement process.

V. Experiments

A. Experimental setup

We evaluate all models using a 5-fold cross-validation protocol and report results averaged across folds to ensure robust evaluation.

For each fold, models are trained for 300 epochs, which was empirically determined to be sufficient to observe stable and meaningful performance differences across architectures. To reduce variability in the final performance, we employ a decaying learning rate implemented via a cosine annealing schedule [5], decreasing from 5×10^{-4} to 5×10^{-6}. These values were selected empirically through a manual grid search.

A batch size of 8 is used, corresponding to the maximum size that allows all model variants to fit within GPU memory constraints. As described in the data augmentation section, each input image undergoes stochastic data augmentation prior to being forwarded to the model. To ensure fair comparison across architectures, we enforce identical augmentation sequences for all models by fixing the random seed of the transform for each fold.

All models are optimized using the Adam optimizer [6] and trained on a single NVIDIA RTX 3090 GPU with 24 GB of memory.

Overall, this experimental setup is designed to ensure fairness, reproducibility, and a reliable assessment of performance differences across architectures.

B. Loss function

We use the binary cross-entropy (BCE) loss for binary segmentation. Given predictions p and binary labels $y \in \{0, 1\}$, the loss is defined as

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N} \sum_{i=1}^{N} \left[y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \right].$$

In practice, the loss is implemented using the `BCEWithLogits` formulation, which combines the sigmoid activation with the loss function, for increased stability. Other loss functions, mainly clDice, were also tested but did not yield conclusive improvements for our use case (see Appendix B).

C. Baselines

Using the experimental setup described above, we evaluate the standard U-Net architecture described in Section IV-A, which serves as a strong and widely adopted baseline against which subsequent architectural modifications are compared.

D. ResNets

We replace the U-Net encoder with the encoding layers of a ResNet architecture, as described in Section IV-B. We first evaluate U-Net models equipped with randomly initialized (non-pretrained) ResNet backbones, considering ResNet-18, ResNet-34, ResNet-50, and ResNet-101, in order to assess the purely architectural benefits brought by increasing model depth and capacity.

As ResNet architectures are widely adopted, pretrained weights are available. We therefore further investigate the impact of encoder pretraining by initializing the ResNet backbones with weights pretrained on the ImageNet-1K dataset [3], [7]. This allows us to evaluate the contribution of large-scale pretraining when transferred to the road segmentation task.

Beyond architectural comparisons, these experiments aim to assess whether features learned from large-scale image classification can be effectively transferred to a domain-specific dense prediction task such as road segmentation.

E. Refinement module

We evaluate the addition of a refinement U-Net appended to the base U-Net architecture (Section IV-C). Two variants are considered: (i) forwarding only the base prediction, and (ii) forwarding both the final-stage feature maps and the base prediction, in order to assess the impact of additional information on refinement performance. We moreover use two training strategies.

a) Separate training: In this strategy, the base U-Net and the refinement module are trained independently using separate losses and optimizers with identical hyperparameters. The input image is first processed by the base U-Net to produce an initial prediction (and feature maps when applicable), which are detached from the computation graph before being passed to the refinement module. Gradients are backpropagated independently to update each module.

b) Joint training: In this strategy, the base U-Net and the refinement module are trained end-to-end as a single model using a shared loss and optimizer. The input image is passed through both modules without detaching intermediate outputs, and gradients from the refined prediction are jointly backpropagated to update the parameters of both networks.

VI. Results

Table I reports the F1 scores and accuracies obtained for each configuration described in Section V.

In addition, to better understand the effect of the base and refinement module configurations, Figure 4 presents a qualitative comparison for a representative input sample obtained at the end of the first fold. This visualization highlights the qualitative differences induced by the various training and feature-forwarding strategies.

VII. Discussions

A. Refinement module

The refinement module improves performance over the baseline U-Net in most settings, with the best configuration (separate training with mask and feature forwarding) increasing the F1 score from 0.8666 to 0.8769 (Table I).

With separate training, both variants exhibit the expected refinement behavior, namely improved connectivity and blob removal (Figure 4). Forwarding additional features further

Model	F1 Score	Accuracy
Baseline UNet		
Baseline UNet	0.8666	0.9468
UNet + Refinement Module		
Refine mask (joint)	0.8724	0.9490
Refine mask + features (joint)	0.8634	0.9448
Refine mask (separate)	0.8693	0.9476
Refine mask + features (separate)	0.8769	0.9507
ResNet Backbone – Not Pretrained		
ResNet18 (NP)	0.8686	0.9476
ResNet34 (NP)	0.8673	0.9471
ResNet50 (NP)	0.8741	0.9497
ResNet101 (NP)	0.8718	0.9490
ResNet Backbone – Pretrained		
ResNet18 (PT)	0.8834	0.9537
ResNet34 (PT)	**0.8904**	**0.9564**
ResNet50 (PT)	0.8897	0.9561
ResNet101 (PT)	0.8893	0.9557

TABLE I: Comparison of F1 Score and Accuracy across models (5-fold cross-validation). Underlined values indicate the best result within each category; bold values indicate the best overall performance.

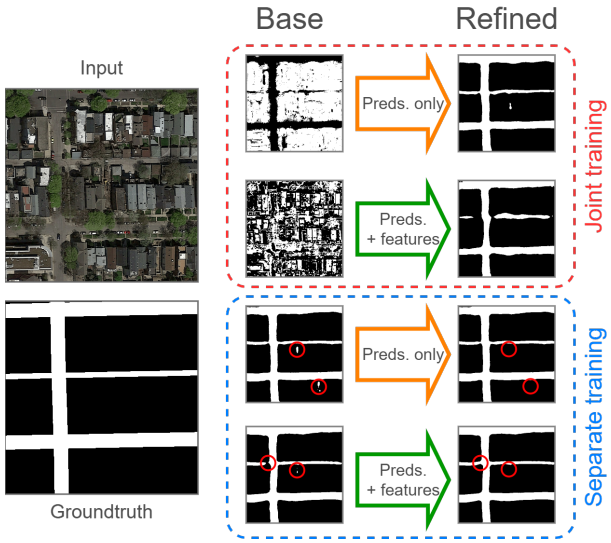


Fig. 4: Qualitative impact of training strategy and feature forwarding in the refinement module.

improves performance (F1 score 0.8769 vs. 0.8693), indicating that richer information benefits refinement when the base model is explicitly constrained.

In joint training, the base prediction is not directly constrained and is therefore free to optimize only for the final refined output. When only the predicted mask is forwarded, this mask still acts as a bottleneck, preserving minimal structure and enabling the refinement module to improve performance (F1 score 0.8724). In contrast, forwarding both features and predictions relaxes this constraint, allowing the base network to rely on high-dimensional features. As a result, the intermediate mask loses meaningful structure (which can be seen in Figure 4), which hinders refinement and leads to performance below the baseline (F1 score 0.8634).

Overall, refinement is effective, but its success depends on how constraints are enforced during training.

B. Architectural benefits of ResNet

Replacing the U-Net encoder with a ResNet backbone leads to consistent performance improvements. Small backbones such as ResNet-18 and ResNet-34 provide only limited gains, while larger architectures yield more noticeable improvements. In particular, ResNet-50 and ResNet-101 achieve higher F1 scores, although ResNet-50 slightly outperforms ResNet-101, suggesting diminishing returns when increasing model depth for this task.

Overall, the ResNet architecture provides clear architectural benefits and broadly follows a scaling trend, with performance gains saturating for deeper models.

C. Impact of pre-training

Pretraining has a strong positive impact on performance, with all pretrained models outperforming the baseline U-Net by approximately two F1 points. The best overall performance is achieved by the pretrained ResNet-34 (F1 score 0.8904), while deeper models obtain slightly lower scores, remaining within 0.1 percentage point.

In addition, all pretrained variants significantly outperform their non-pretrained counterparts by more than one F1 point, highlighting the benefit of ImageNet initialization. This indicates that features learned on a different task and dataset distribution—such as ImageNet, which predominantly contains object-centric images (e.g., animals, everyday objects)—transfer well to road segmentation. This is likely due to pretrained models learning generic low-level and mid-level patterns such as edges, textures, and connectivity cues.

Overall, pretraining provides the best performance across all experiments. The limited gains observed for deeper pretrained models suggest that the benefits of pretraining are already largely exploited by ResNet-34 for this task.

VIII. Conclusion

Adding a refinement model at the output of the U-Net was shown to improve the performance, especially by training the refinement model separately and feeding the last layer's features along with the predicted mask. Using ResNets as the encoding architecture of the U-Net yielded similar results. The best results we obtained on cross-validation was by using pretrained ResNets as backbone, but we note that the edge gets smaller with longer training, showing the efficiency of transfer learning.

We reused the ResNet34 (PT) weights from the the first fold of the experiment, since it yielded the best results. We also predict the mask using rotated and flipped versions of the test images, and then average the predictions to avoid bias and used a slighly lower threshold to improve recall (chosen empirically to 0.4). This resulted in our best competitive F1 score of 0.915 with accuracy of 0.954 (submission #305318).

ETHICAL RISK

A key ethical risk identified in this project concerns the lack of geographic and socio-economic diversity in the training and test datasets. The images used are a subset of the Massachusetts Roads Dataset and consist almost exclusively of urban and suburban environments in developed, first-world regions.

A. Stakeholders impacted

The primary stakeholders affected are end users and communities where such a road-segmentation system could be deployed, particularly those in rural areas or less developed countries. Secondary stakeholders include public authorities which might rely on such models for infrastructure planning, navigation, or safety-critical applications.

B. Negative impact

Our model may fail to generalize to regions with different road characteristics, such as unpaved roads, irregular layouts, sparse markings, or lower-quality imagery. This can lead to systematic performance degradation in under-represented regions, reinforcing existing technological inequalities and potentially excluding certain populations from the benefits of such systems.

C. Risk significance

The likelihood of occurrence is high, as dataset bias is intrinsic to the data provided [8]. However, the severity of this risk is dependent on the deployment context. In this academic project, the real-world impact is near zero as no deployment is performed.

D. Risk evaluation

This risk was evaluated through dataset inspection and literature review on dataset bias and domain shift in semantic segmentation. Prior work in computer vision has shown that models trained predominantly on data from high-income regions experience significant drops in standard segmentation metrics (e.g., IoU or F1-score) when evaluated on underrepresented regions, due to domain shift in visual characteristics [9]. Although these results are reported for general vision tasks, the same mechanisms apply to road segmentation. Relevant indicators include the absence of rural road patterns, low variability in road surface types, and consistent urban road density.

E. Mitigation and project constraints

Due to the fixed nature of the provided test set, this risk could not be directly mitigated within the project. Training on a more diverse dataset including rural roads or images from less developed regions would have likely not improved the measured performance, even if it improved real-world robustness.

REFERENCES

[1] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," May 2015. arXiv:1505.04597 [cs].

[2] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net."

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[4] "Resnet block diagram: Basic block for resnet-18/34 and bottleneck block for resnet-50/101." https://www.researchgate.net/figure/ResNet-block-diagram-a-Basic-block-for-ResNet-18-ResNet-34-b-Bottleneck-block-for_fig3_385334673. Accessed: 2025-03.

[5] P. Janson, V. Singh, P. Mehrbod, E. Belilovsky, *et al.*, "Beyond cosine decay: On the effectiveness of infinite learning rate schedule for continual pre-training," *arXiv preprint arXiv:2503.02844v1*, 2025.

[6] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2015.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[8] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (Colorado Springs, CO, USA), pp. 1521–1528, 2011.

[9] T. D. Vries, I. Misra, C. Wang, and L. van der Maaten, "Does object recognition work for everyone?," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (New Orleans, LA, USA), pp. 3035–3044, 2022.

[10] S. Shit, J. C. Paetzold, A. Sekuboyina, I. Ezhov, A. Unger, A. Zhylka, J. P. W. Pluim, U. Bauer, and B. H. Menze, "clDice – A Novel Topology-Preserving Loss Function for Tubular Structure Segmentation," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16555–16564, June 2021. arXiv:2003.07311 [cs].

APPENDIX A
DATA AUGMENTATION IMPLEMENTATION

```
A.Compose(
    [
        A.RandomResizedCrop(
            size=(height, width),
            scale=(0.8, 1.0),
            ratio=(0.9, 1.1),
            p=0.5,
        ),
        A.HorizontalFlip(p=0.5),
        A.VerticalFlip(p=0.5),
        A.RandomRotate90(p=0.5),
        A.Affine(
            translate_percent=(-0.03, 0.03),
            scale=(0.95, 1.05),
            rotate=(-15, 15),
            border_mode=cv2.
                BORDER_REFLECT_101,
            p=0.5,
        ),
        A.RandomBrightnessContrast(
            brightness_limit=0.1,
            contrast_limit=0.1,
            p=0.4,
        ),
        A.HueSaturationValue(
            hue_shift_limit=5,
            sat_shift_limit=15,
            val_shift_limit=10,
            p=0.25,
        ),
        A.ToFloat(max_value=255.0),
        A.ToTensorV2(),
    ],
)
```

APPENDIX B
LOSS FUNCTION EXPLORATION

Another loss function, clDice [10], was previously proposed in order to improve the topology conservation of segmentation models. The loss function is defined as an interpolation between a standard volumetric loss function, and clDice, a topological loss function. The baseline U-Net model was trained for 300 epochs using this loss function.

Loss function	F1 Score	Accuracy
BCE	**0.8666**	**0.9468**
clDice ($\alpha = 0.1$, iter $= 25$)	0.7979	0.9087

TABLE II: Comparison of F1 Score and Accuracy between BCE and clDice loss functions.

The results are reported in Table II. Both accuracy and F1 score drop, supposedly traded for better topology conservation.