A Modern Approach to Road Segmentation

Julian Minder, Andrew Dobis, Daniel Frey, and Marie Jaillot; Group: CIL Team Six
Department of Computer Science, ETH Zurich, Switzerland

***Abstract*—Modern-day artificial intelligence applications find themselves relying ever-more heavily on image segmentation in order to accomplish tasks which require an interaction with their surrounding environment. In this work, we focus on road segmentation for satellite imagery using only a small amount of labeled training data. For this, we propose two approaches. First, we leverage the existing large pretrained image backbones available with the UPerNet architecture and fine-tune it on our small training dataset to achieve highly accurate results. Additionally, we alleviate the shortcomings of having a small labeled training dataset by using a Denoising Diffusion Probabilistic Model (DDPM) to utilize unlabeled satellite imagery for the segmentation task. We obtain a 16×16 patch accuracy of 93.13% using the UPerNet model with a pretrained ConvNeXt backbone, and 87% using a DDPM, both trained on a dataset containing 115 labeled images.**

I. Introduction

Many contemporary applications are highly dependent on a program's capability of understanding the contents of an image at a high level in order to function. Out of these contents, the localization of roads has been shown to be very useful for critical applications such as map generation or in autonomous vehicles. This task is called road segmentation and its goal is to generate a mask, in the form of an alpha image, that highlights the roads which can be found on the input image. Many images may be used for this task, however, we will be focusing on satellite imagery. A common approach, often used when solving this problem, is to train an encoder-decoder style model, known as a UNet, on a very large dataset containing satellite imagery labeled with the corresponding road masks [1]. However, solving this task using only a small training dataset is a much more difficult problem, and requires a different approach. To tackle this problem, we leverage the existing pretrained large image backbones available with the UPerNet architecture [2], and use our small training dataset, with a few augmentations, to fine-tune it on our task, thus extracting maximum accuracy out of a small amount of labeled data.

This solution yields good results, however it does not focus on the core problem, which is the lack of a sufficient amount of labeled training data. To solve this, we take an even more drastic approach aimed at truly alleviating these shortcomings by using a Denoising Diffusion Probabilistic Model (DDPM) to generate new satellite images based on the existing dataset. DDPMs [3], [4] have recently become very popular and have outperformed the state-of-the-art hold by GANs in generative modeling [5]. It has been shown that one can leverage the latent space produced by GANs to produce label efficient semantic segmentation models [6], [7], [8]. Following up on their work, Baranchuk et al. [9] propose a DDPM based approach for creating segmentation maps, and we adapt this solution for our task of generating road segmentation masks.

In this work, we propose the following contributions:

- A novel approach to road segmentation using the UPerNet architecture with the recently proposed ConvNeXt backbone.
- A novel approach, as well as an exploratory study, for adapting a DDPM for road segmentation using only a small training dataset.

II. Models and Methods

In this section we describe all the approaches used when solving this problem. We start with a basic UNet, followed by more novel architectures and finally expand them with various pre- and post-processing mechanisms.

A. UNet

A UNet is a convolutional network architecture for small training sets first proposed for biological image segmentation [1]. We only modified the network by adding batch normalizing layers.

B. UPerNet

The best performing model in our solution is called a Unified Perceptual Network (UPerNet) [2]. It was originally designed for multi-task image segmentation, a.k.a. scene parsing, and does so by adding many output heads which use intermediary features from the model's core to obtain different semantic information about an image. However, in our version, we do not require the multi-task capabilities of the model, and thus only use the result from the object head to obtain our road mask. Figure 1 illustrates and describes the architecture of the model. We adapt the model in order to use either a `ResNet50` [10] or ConvNeXt [11] backbone. The model uses a total of 108M parameters when used with the ConvNeXt *base* backbone, which is the one that yields the highest F1-score [12].

C. DDPM

DDPMs function by gradually transforming noise $x_T \sim N(0, I)$ into less noisy samples x_t until we reach the final denoised image x_0. This is done by repeatedly applying a

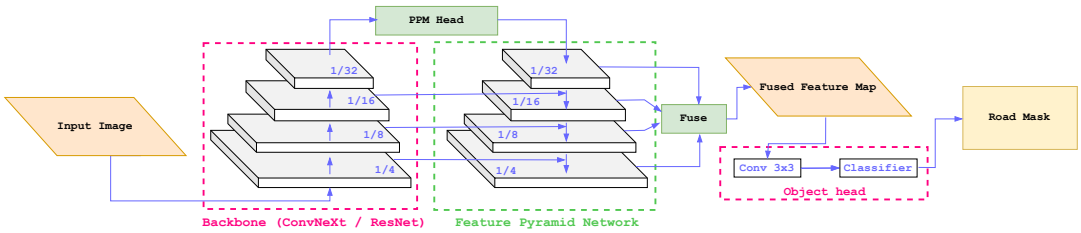


Figure 1: Architecture of our adapted UPerNet model. Here, the image is first passed through a feature extracting backbone, for which we use either ConvNeXt or ResNet models, after which it goes through a Pyramid Pooling Module (PPM) before being passed to the Feature Pyramid Network (FPN). Finally, the intermediary results from the FPN are fused into a single feature map and passed through the object head to achieve the final road mask of our input image.

noise predictor network $\epsilon_\theta(x_t, t)$, which is typically a variation of the UNet [1]. We use the state-of-the-art variation proposed by [5]. The network is trained by creating noisy samples x_t from training images x_0 and then predicting the original sample $\hat{x_0}$ from x_t.

To obtain segmentation maps, we pretrain a DDPM on a large corpus of unlabeled satellite images. Once this is done, the masks can be extracted by following [9] by adding noise to training images. This is done for a set $\mathcal{T}$ of different time steps t. The noisy images x_t are then passed through the pretrained $\epsilon_\theta(x_t, t)$. During the denoising step, we extract features f_t from various points in the Network, as illustrated in Figure 2. The assumption here is that the noise predictor $\epsilon_\theta(x_t, t)$ has learned a semantic representation of the satellite images in a latent space $\mathcal{L}$, including what roads look like, and leverages this to denoise the image. We upscale the features back to the original image size and stack them across all time steps. Finally we train a small MLP pixel classifier that predicts whether or not each pixel of the stacked features whether is a road.

To our knowledge, there is no open-source pretrained DDPM for satellite images. We thus pretrain it from scratch using hyperparameters adapted from a previous DDPM work [5]. Note that this is extremely compute heavy task, leading to us having to reduce the number of channels in the noise predictor from 128 to 96, and to use microbatches of size 8 due to memory limitations. The DDPM was trained for $44k$ steps, which is also much lower than the $500k$ steps proposed by the authors. This is, again, simply due to the limitations in available computational resources.

The pixel predictor is a simple MPL with three hidden linear layers of sizes $[128, 128, 32]$. For the feature extraction, we use $\mathcal{T} = \{50, 150, 200, 250\}$ which results in 5760 features per pixel. The pixel predictor is trained on those individual pixel features. Due to the highly unbalanced nature of the data, i.e. much more background than road pixels, we sample the image pixels and features such that the number of road and non-road pixels are equal. Augmentations are not applied during the pixel classifier's training, as this would require extracting features from every randomly augmented training image, yielding an excessive computational complexity.

D. Augmentations

During training, we apply heavy augmentations to expand the size of the available training set. Up to three transformations are randomly applied to each input image. We applied the methods *Flip*, *ShiftScaleRotate* and *RandomBrightnessContrast*, from the Albumentations library [13]. Details can be found in Table II in the appendix.

1) Test-Time Augmentation (TTA): Augmentations were also applied throughout the prediction pipeline. With rotation and flipping, we obtain 8 different versions of a test image, all of which are passed trough the network. The 8 predictions are combined into the final segmentation map by either *thresholding* the sum at value 4 or by taking the *mean*.

2) Post-Processing: Additionally to the aforementioned augmentations, we tested post-processing methods adapted from an existing road segmentation work [14]. Namely *median blurring*, *removal of low noise*: set all values with a value smaller than 0.1 to 0, *gaussian adaptive thresholding* and *removal of small connected objects*. Details can be found in Table IV in the appendix.

These post-processing steps, however, did not improve the results. The test-time augmentation step already has a significant noise cleaning effect as the resulting segmentation images show. Additionally, since the final score is calculated patch-wise, small amounts of noise in a prediction image have only a minor influence on the final score.

3) Stochastic Weight Averaging: Finally, stochastic weight averaging was done as an attempt to improve the results achieved with the UPerNet model. However, this had very little effect and thus is left as a side-note.

III. Results

Now that our models and data augmentation methods have been described, we can move on to presenting the results obtained throughout our experimentation. In this section, we start by describing our evaluation methodology, followed by an ablation study on our two different models, i.e. UNet

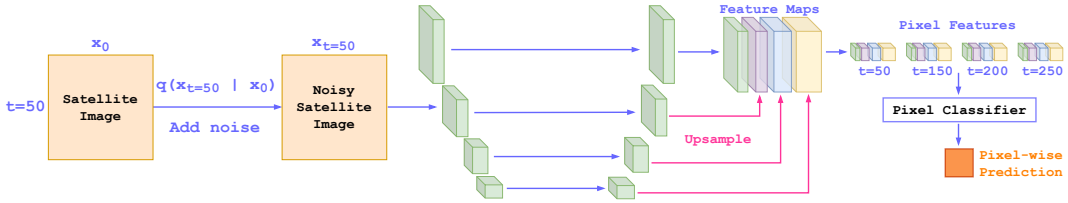


Figure 2: Architecture of the DDPM. x_t is obtained by adding noise according to $q(x_t|x)$. Feature maps are then extracted from the pretrained noise predictor $\epsilon_\theta(x_t, t)$, and upsampled and concatenated for every time step t to obtain pixel-wise feature maps. A pixel classifier Multi-layer Perceptron (MLP) is then trained to predict the labels based on the pixel-wise features of all timesteps.

Model	# Parameters	Pretrained	Patch Acc [%]
`All Background`	0	-	75.55
`Patch CNN`	26.4K	-	82.77
`UNet`	31M	-	91.74
`UNet`	31M	MS	91.87
`UNet`*	31M	MS	92.18
`UNet`†	31M	MS	92.32
`UNet`†	31M	AIRS	92.20
`DDPM`†	64M / 0.7M	MS (unlabeled)	87.00
`UPerNet`†	108M	-	90.80
`UPerNet(ResNet)`†	108M	IN	91.66
`UPerNet`†	108M	IN	92.32
`UPerNet`†	108M	IN + MS	**93.13**
`UPerNet`$^{\dagger,\diamond}$	108M	IN + MS	93.00
`UPerNet`$^{\dagger,\diamond}$	108M	IN + AIRS	92.86

Table I: Patched accuracy on the CIL test dataset. The symbol $*$ refers to *mean TTA*, † to *thresholded TTA* and $\diamond$ to training with *Stochastic Weight Averaging*. If not specified, UPerNet is trained with the ConvNeXt backbone.

and UPerNet. Finally, we present our exploratory results obtained using the DDPM architectures.

A. Evaluation Methodology

Our models were trained on the following image datasets.

- *AIRS:* Aerial Imagery for Roof Segmentation [15], a roof segmentation dataset for aerial images.
- *MS:* Massachusetts Roads Dataset [16], a road segmentation dataset for aerial images.
- *CIL:* Dataset from the ETHZ CIL Road Segmentation 2022 Project [17].

The AIRS and the MS datasets are used for pretraining and the labeled part of the CIL dataset is used for subsequent fine-tuning while keeping a fixed image subset of 20% for the validation process. Since the AIRS dataset is comprised of very high resolution images, we resize them to 2000×2000 so that the resulting resolution is similar to the ones found in the CIL dataset. During training, each image is divided into multiple small patches, which can overlap depending on a specified stride value. Details on the patching can be found in Table III in the appendix. Patches that contain a mask percentage below 0.1% are removed from the training set.

We pretrained the different models for 24 hours on an Nvidia RTX 2080Ti GPU. This is done on the AIRS and MS datasets for 73 and 24 epochs, respectively. The models are then fine-tuned on the CIL dataset for up to 600 epochs. All computations were run on the ETH Euler cluster [18] with a batch size of 32.

We combined dice and binary cross entropy loss to obtain the loss function $\ell = \ell_{DICE} + \ell_{BCE}$ to train our models.

$$\ell_{DICE}(x, y) = 1 - \frac{1}{N} \frac{2 \sum_{i=1}^{N} x_i y_i}{\sum_{i=1}^{N} x_i + \sum_{i=1}^{N} y_i} \tag{1}$$

$$\ell_{BCE}(x, y) = -\frac{1}{N} \sum_{i=1}^{N} (y_i \ln x_i + (1 - y_i) \ln(1 - x_i)) \tag{2}$$

B. Analysis of UNet and UPerNet results

Now that our evaluation methodology has been described, we can move on to analysing the experimental results. Table I shows the results from the experiments run on our different models. As a set of baselines for our comparison, we define a trivial model which always predicts a value of zero, as well as the patch Convolutional Neural Network (CNN) provided to us by the project handout [17]. This baseline will be compared to the various versions of our subsequent models. In this paragraph, we focus mainly on our UNet, and UPerNet models, since these yielded the most promising results. We start by taking a look at our two models without pretraining. In this scenario, UNet outperforms UPerNet by 1.06%. Using pretrained weights on `imagenet1k` (IN) [19] for the UPerNet's backbone yields a slight improvement of 0.8% with the ResNet backbone, and 1.5% for the ConvNeXt backbone. This shows that using the `ResNet50` backbone does not yield a satisfactory result, since it performs poorer than UNet when using the IN configuration. In order to improve the UNet's results, we pretrain it on-domain with the Massachusetts Roads (MS) dataset, and augment our training dataset using the aforementioned *mean TTA* and *thresholded TTA* methods, yielding

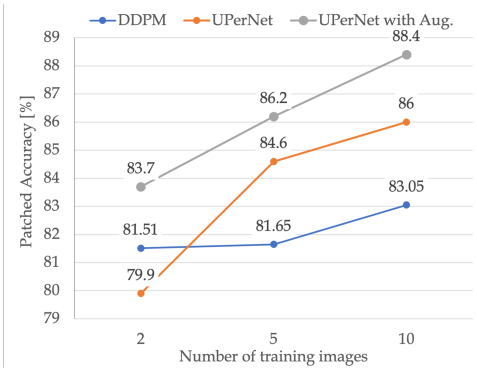


Figure 3: Comparison of the models performance, trained on only 2, 5 and 10 labeled images. As the DDPM performance is measured without Augmentations due to computational limits, the figure shows a UPerNet with and without Augmentations.

an improvement over our base UNet of 0.4% and 0.5% respectively. The ladder configuration yielded the best results using the UNet model with a patch accuracy of 92.32%. An equivalent result is obtained using a UPerNet architecture with a ConvNeXt backbone in the IN configuration using the *thresholded TTA* method. Note that we also tested a version of the UNet pretrained off-domain using the AIRS dataset, however, unsurprisingly, this under-performed with regards to our previous configuration. A similarly under-performing result is obtained when pertraining the UPerNet off-domain. Nonetheless it is shown that leveraging a related domain for pretraining results in increased performance. The best results were obtained using the UPerNet architecture by pretraining on both the IN and MS datasets. Doing so, all while using the *thresholded TTA* method, yielded a final patch accuracy of 93.13%, which is 0.8% better than the best result obtained using a UNet. Anecdotally, we tried to improve the result using stochastic weight averaging on the last 25% of the training epochs, however this yielded unsatisfactory results. Our best configuration for this task is thus the UPerNet model with the ConvNeXt backbone pretrained on the IN and MS datasets and using the *thresholded TTA* method.

C. Analysis of DDPM performance

The DDPM patched accuracy lies quite far behind the other approaches. This section shall hypothesise why this is the case and how one could improve it. Figure 4 shows two segmentation predictions using the DDPM approach. Manual inspection shows that shadows, trees, and less common road types are often not well predicted. Note that the pixel classifier works on a pixel level and does not access the pixel's neighborhood like typical convolutional networks. Therefore, information about a hidden road must entirely be stored by the features that the noise predictor produces. Figure 6 in the Appendix shows a random selection of 16 256×256 images sampled from the generative model. The

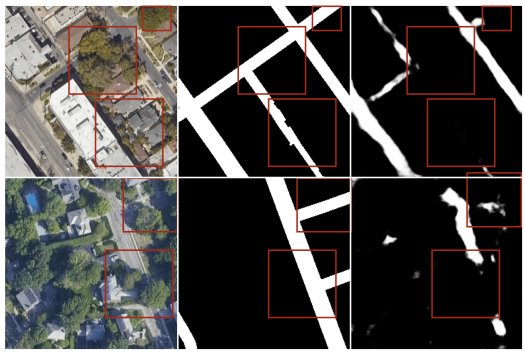


Figure 4: Left: Satellite image; Mid: Ground Truth; Right: DDPM Prediction. Highlighted are problem zones.

samples obtained in this work look quite good, but it is inherent that the model could benefit from more training. This is especially apparent when comparing them to the degree of quality the original authors achieve [5]. Thus, the model would benefit from longer pretraining. To further evaluate the diffusion approach, we look at how it performs in comparison to the UPerNet (ConvNeXt backbone) on even less data. Figure 3 shows the patched accuracy of the two approaches on 2, 5 and 10 training images. The displayed numbers are the highest patched accuracy scores that the models reached on the validation set during the training process. The slopes show that the UPerNet benefits much more from having more available training data. This further reinforces our hypothesis that the information stored in the features is limited, and more labeled training data for the pixel classifier only partially helps. For only 2 training images, the DDPM can, despite its limits, outperform the otherwise much more capable UPerNet. This highlights its capability to deal with minimal training data. Also noteworthy is the remarkable performance of 88% (87.8% on the CIL test set) of the UPerNet on only 10 images. The Appendix contains the figures for pixel-wise accuracy and pixel-wise F1-score.

IV. Conclusions

Throughout this work, we explored the use of modern architecture for road segmentation in the presence of limited training data. We found that the UPerNet model with a ConvNeXt backbone pretrained on-domain using the IN and MS datasets outperformed our best UNet model by 0.8%, yielding a best patch accuracy of 93.13%. We also explored the use of a DDPM to alleviate the small training set size, however this under-performed in comparison to the UPerNet model. We find, however, that such an approach can be especially valuable for dynamic tasks with regularly updated segmentation targets, because the DDPM must only be trained once and can be used for many tasks. DDPM generated images, as well as a comparison of the different road masks generated by our models, can be found in the appendix.

References

[1] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Springer. Springer International Publishing, 2015, pp. 234–241.

[2] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, "Unified perceptual parsing for scene understanding," in *European Conference on Computer Vision*. Springer, 2018.

[3] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *International Conference on Machine Learning*. PMLR, 2015, pp. 2256–2265.

[4] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851, 2020.

[5] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8162–8171.

[6] Y. Zhang, H. Ling, J. Gao, K. Yin, J.-F. Lafleche, A. Barriuso, A. Torralba, and S. Fidler, "Datasetgan: Efficient labeled data factory with minimal human effort," in *CVPR*, 2021.

[7] N. Tritrong, P. Rewatbowornwong, and S. Suwajanakorn, "Repurposing gans for one-shot semantic part segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 4475–4485.

[8] D. Galeev, K. Sofiiuk, D. Rukhovich, M. Romanov, O. Barinova, and A. Konushin, "Learning high-resolution domain-specific representations with a gan generator," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 2021, pp. 108–118.

[9] D. Baranchuk, A. Voynov, I. Rubachev, V. Khrulkov, and A. Babenko, "Label-efficient semantic segmentation with diffusion models," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=SlxSY2UZQT

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[11] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," *CoRR*, vol. abs/2201.03545, 2022. [Online]. Available: https://arxiv.org/abs/2201.03545

[12] M. Sokolova, N. Japkowicz, and S. Szpakowicz, "Beyond accuracy, f-score and roc: A family of discriminant measures for performance evaluation," in *AI 2006: Advances in Artificial Intelligence*, A. Sattar and B.-h. Kang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1015–1021.

[13] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, 2020. [Online]. Available: https://www.mdpi.com/2078-2489/11/2/125

[14] V. Yerram, H. Takeshita, Y. Iwahori, Y. Hayashi, M. K. Bhuyan, S. Fukui, B. Kijsirikul, and A. Wang, "Extraction and calculation of roadway area from satellite images using improved deep learning model and post-processing," *Journal of Imaging*, vol. 8, no. 5, 2022. [Online]. Available: https://www.mdpi.com/2313-433X/8/5/124

[15] Q. Chen, L. Wang, Y. Wu, G. Wu, Z. Guo, and S. L. Waslander, "Aerial imagery for roof segmentation: A large-scale dataset towards automatic mapping of buildings," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 147, pp. 42–55, 2019.

[16] V. Mnih, "Machine learning for aerial image labeling," Ph.D. dissertation, University of Toronto, 2013.

[17] (2022) Ethz computational intelligence lab, road segmentation project 2022. [Online]. Available: https://www.kaggle.com/competitions/cil-road-segmentation-2022

[18] (2022) Eth euler cluster. [Online]. Available: https://scicomp.ethz.ch/wiki/Euler

[19] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

Appendix

	probability	non-default parameters
`Flip`	0.5	
`ShiftScaleRotate`	0.9	`rotate_limit=170`
`RandomBrightnessContrast`	0.75	

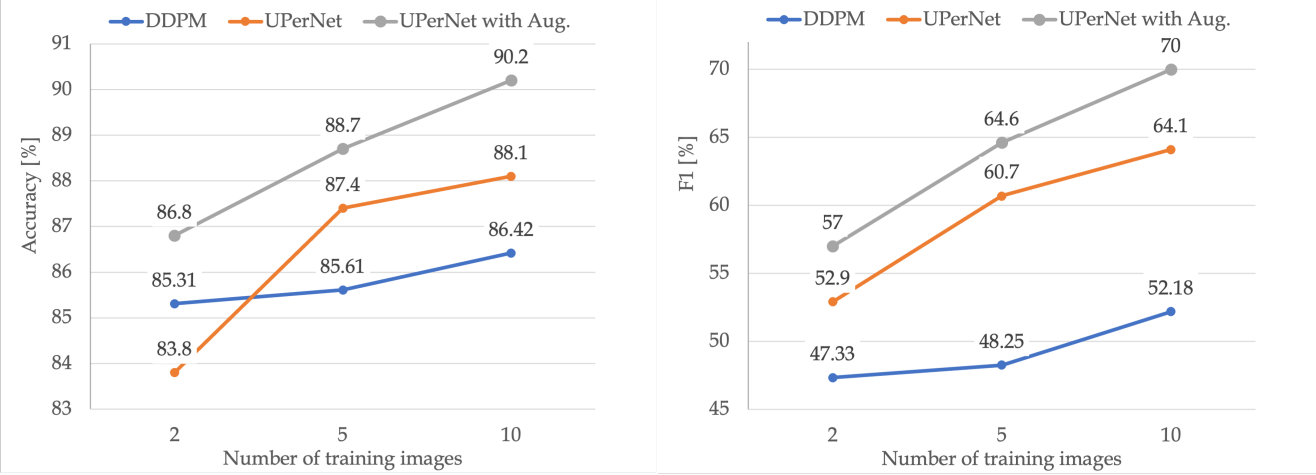
Table II: Transformation methods used in the augmentation step of the training process.

	patch size	stride	patching used in validation
`CIL`	208	96	No
`MS`	224	112	Yes
`AIRS`	224	112	Yes

Table III: Configurations used for image patching during training.

`Median Blurring`	kernel 15×15
`Gaussian Thresholding.`	kernel 85×85
`Removal of small connected objects`	size limit 200

Table IV: Configurations used for mask postprocessing.



(a) Accuracy

(b) F1-Score

Figure 5: Comparison of the models performance, trained on only 2, 5 and 10 labeled images. As the DDPM performance is measured without Augmentations due to computational limits, the figure shows both UPerNet with and UPerNet without Augmentations.

Figure 6: 16 256×256 images sampled from the DDPM model.

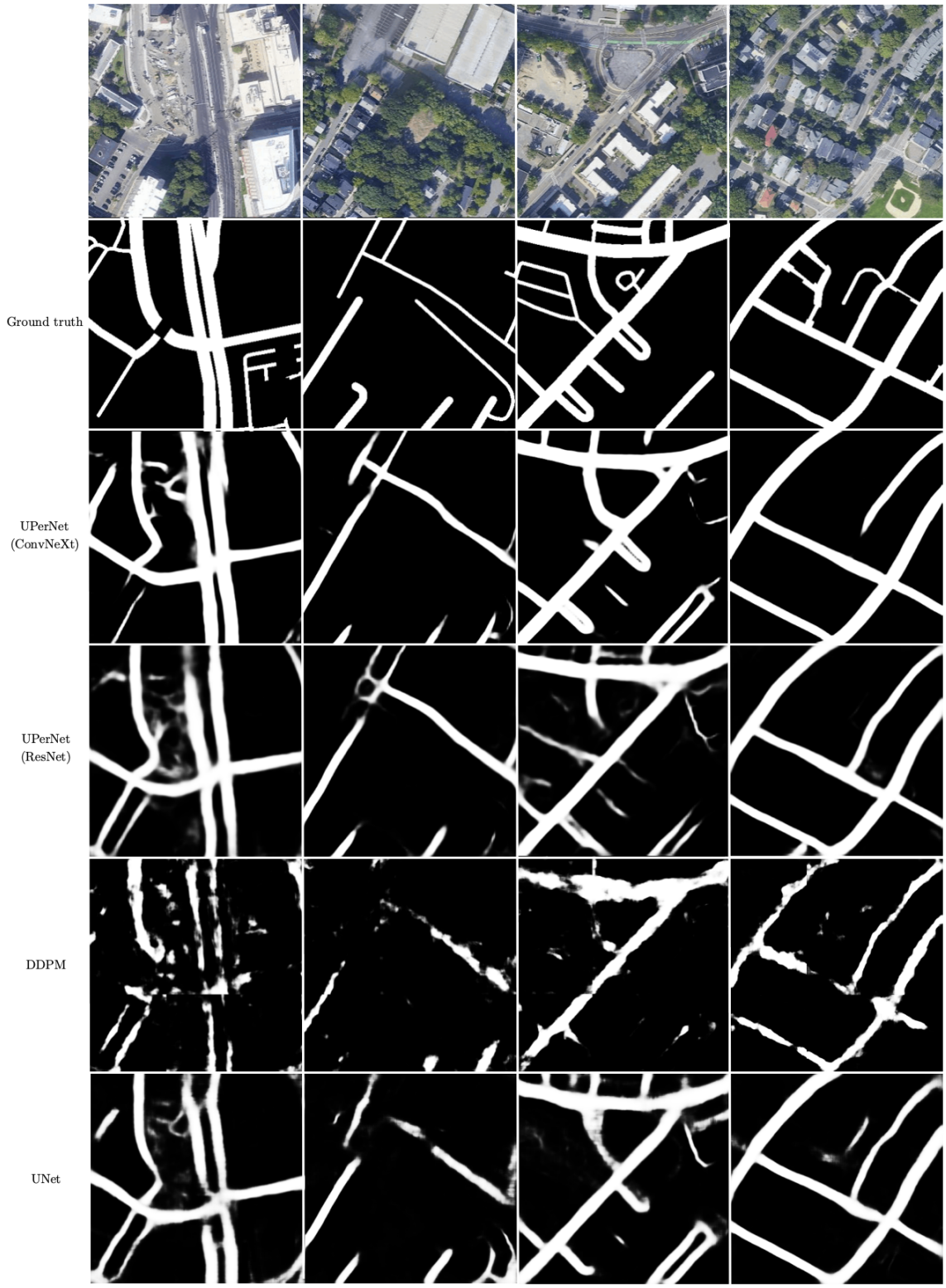


Figure 7: Prediction examples from different architectures. No postprocessing is applied to the outputs. The truncated artifacts on the leftmost prediction of the DDPM is due to how the four 256×256 patches are combined to one 400×400 image.