Road Segmentation of Aerial Images Using Convolutional Neural Networks

Gianluca Danieletto, Guillaume Dugat, Felix Yang, Daiwei Zhang
Kaggle Team: WeNeedANameHere
Department of Computer Science, ETH Zurich, Switzerland

***Abstract*—We present various methods for the segmentation of roads in aerial images using convolutional neural networks (CNNs). A common theme among our approaches is how to deal with the extremely small size of the provided dataset for this challenging task. Besides exploring different model architectures, we also investigate some unconventional approaches based on style transfer and K-means clustering. In our final approach, we exploit external datasets and perform an enhanced inference procedure that is able to remove false positives from the predictions, which achieves an F1 score of 0.91648 on the Kaggle leaderboard.**

I. Introduction

Image segmentation is an important component of many image processing and computer vision tasks. The purpose of image segmentation is to divide images into regions with specific meanings. Classical approaches involve methods based on thresholding, edge detection, clustering or graph cuts [1]. However, deep neural networks have become increasingly popular for image segmentation in recent years. These powerful models often show very impressive performance and high accuracy rates on common benchmarks [2].

This project is about road segmentation and we are provided with 144 satellite images from Google Maps and their corresponding binary segmentation masks (1: road, 0: background). All images have a size of 400 $\times$ 400 pixels and the task is to assign a binary label to each of the 625 non-overlapping 16 $\times$ 16 patches in a given satellite image. A patch is thereby considered as road (label 1) if the mean of the corresponding labels is at least 25%.

To deal with such little data, we propose various approaches based on data augmentation, style transfer, K-means clustering, transfer learning and exploitation of external data sources. In the following sections, we will describe, evaluate and discuss each approach. We will also compare our methods to two CNN-based baseline models that were presented during the exercise sessions.

II. Models and Methods

A. Baseline Models

Our first baseline model is a **Patch-based CNN**. As presented by [3], this approach consists of using a typical convolutional network for image classification by providing the patches directly as input. The output is then a single number which represents the prediction of the label. Our model is composed of 3 convolutional layers, with ReLU, max-pooling and batch normalization. There is a second part that has two dense linear layers, with a ReLU and two dropouts. The drawback of this approach is that it does not take the information of the neighboring patches into account.

Our second baseline consists of a **U-Net architecture**, as presented by [4]. Unlike the previous approach, the model takes the whole image (possibly resized) as an input, and gives as an output a one-channel image with the same resolution in order to obtain pixel-wise predictions of the labels. It is made up of a repeated block of 3×3 convolution, followed by a ReLU, a batch normalization, another 3×3 convolution and another ReLU. In the left part of the network this block is repeated 5 times with max-pooling down-sampling in between. The bottleneck is a 3×3 convolution which outputs a 1024-channels image of resolution 12×12. It enables to learn rich features. The right part is symmetrical to the left part, with 5 times the repeated block and transposed convolution up-sampling in between. Last but not least, there are skip connections from left to right between images with the same resolution, so that the localization information can propagate to the deepest layers of the architecture.

As a first step, we wanted to see how much the U-Net baseline model can be improved by applying basic methods such as per-image normalization as a preprocessing step. Since there are much more background labels than road labels - approximately 4.6 times more in the training set - we also tried out a weighted loss to account for the imbalance. Finally, we generated an extended dataset using data augmentation methods to increase the number of training samples. In particular, we applied random flips and rotations to each original training sample and filled in the missing parts that arise due to the rotations with reflection padding. Moreover, we added random color jitters to vary the brightness, contrast, saturation and hue of the images. In total, we increased the original dataset size by 10x.

B. Style Transfer

To increase our dataset beyond the data augmentation, we thought about generating new training and groundtruth images ourselves. A procedural automated approach seemed interesting, as it gives full control over the groundtruths and possible modifications for the input. The first idea was to first generate a road segmentation using random lines and Bézier curves and then automatically generate a realistic aerial image from it using style transfer networks [5].

For this, we wrote a simple random road generator using the Processing[1] framework for Python and fed the resulting groundtruth into a 19 layer VGG network for style transfer using a given training image as a style reference. Here it was important to use a lot of noise to even get a remotely realistic looking input. To further investigate how to tweak the input to get realistic outputs, we even created some hand-painted grountruth images.

C. Res-U-Net Models

In our next approach, we experimented with different Res-U-Net models as proposed by [6]. For that, we replaced the down-sampling encoder of the U-Net model with different ResNet [7] models. According to the authors, this approach is able to combine the strenghts of both the U-Net and ResNet architectures while achieving better performance with much fewer model parameters [6]. We first used the ResNet-18 model for the encoder of the U-Net to compare it with the U-Net baseline model and to evaluate the impact of transfer learning by training our model with and without weights that were pre-trained on the ImageNet dataset [8]. Next, we repeated our experiments using the ResNet-50 and ResNet-152 models as U-Net encoder. Finally, we modified the architecture of our models by replacing the last two layers of the ResNet-18 model and the bottleneck of the Res-U-Net-18 by dilated convolutional layers. All models were trained on our augmented dataset.

D. Class-Specific Models

After taking a closer look at the provided dataset, we felt like there were some "implicit classes" of satellite images with very different appearances and types of roads. Based on this observation, we computed image embeddings for each training image and applied K-means clustering to the embeddings. For that, we simply took the ResNet-18 model with pre-trained weights and used the flattened output of the last layer before the fully-connected layer as image embedding. We then tried out different values for K and in our opinion $K = 3$ produced the most convincing results. See figure 1 to see the latent class assignment of some of the training images. The results can be interpreted as follows:

- Class 0: Small and medium-sized buildings and roads, roads are mostly straight and well visible.
- Class 1: Small and curvy roads, many streets are occluded by trees.
- Class 2: Medium and large buildings, many wide roads and highways.

For each identified class, we trained a class-specific Res-U-Net-18 model (ResNet-18 as encoder of the U-Net) on the corresponding subset of the augmented training set. To predict the segmentation map for a test image, we first compute its image embedding and use the fitted K-means model to predict the class label. Then, the corresponding class-specific model is used for the prediction of the road segmentation map.

[1]https://processing.org/

Figure 1. Latent class assignments of some training images. Class order from top to bottom row: class 0, class 1, class 2.

Since we only had very few training samples available, our thesis was that training such class-specific models may be more successful than trying to train a single model that has to generalize to all kinds of road types and satellite image appearances using very little data.

E. Generation of Additional Training Data

In this approach, we used the **Google Maps API** to generate additional training data. In particular, we downloaded satellite images and their corresponding road segmentation maps of size 1200 $\times$ 1200. The road segmentation map was generated with a custom made Google Maps layer, which paints everything except roads black. The zoom was chosen such that it is similar to the training data and all roads are visible. Then, we processed this data by slicing the large images into smaller ones and applying rotations and flips. In total, we generated 12'960 training samples. This additional dataset was then used to train a Res-U-Net-18 model. After that, we used the original dataset to finetune the model.

F. External Training Data and Enhanced Inference

In our final approach, we incorporated two public datasets of manually-labeled satellite images into our training procedure, namely the **Massachusetts Roads Dataset (MA)** [9], and **DeepGlobe 2018 Road Extraction Challenge (DG)** [10]. The MA dataset consists of 1171 aerial images of 1500 $\times$ 1500 pixels in size, with the ground sampling distance (GSD) of 1.2m/pixel; The DG dataset consists of 6226 annotated aerial images of size 1024 $\times$ 1024 with the GSD of 50cm/pixel. We expected the greater variety of geolocations (urban, suburban, and rural regions) and scales

(determined by GSD) to improve the network's accuracy and ability to generalize.

All these images were split into tiles of size 256×256. A total of 84'015 images were obtained, while there were only 1152 images of size 400×400 after performing rotation and flip on our given CIL training set. To address such data imbalance, we designed a pertinent training pipeline: while all of our external data was used for training, the CIL dataset was split into 800 training and 352 validation samples. Two dataloaders were iterated separately for the external and CIL data during training. For each iteration, 3 images with sizes of 256×256 were randomly sampled and cropped from the 800 CIL training images; they were then concatenated with 42 external images from MA and DG to form one batch of the network's input. We expected that such a training batch composition would attain a better balance between the accuracy with respect to our given dataset and the generalization ability to various locations. Each epoch consisted of 2000 iterations.

Moreover, we expected that performing basic data augmentation techniques during the inference step could also enhance the performance: using rotations and flips, 8 images were obtained from each original test image. Sliding window inference was performed on these images with a window size of 256×256 and an overlapping ratio of 0.75 between neighboring windows. The eight binary segmentation results were transformed back to the original state; then we took the average over these results to get our final prediction.

III. Results

A. Baseline Models

The U-Net baseline model, even without any improvements, clearly outperformed the patch CNN baseline model: we obtained a gain of about 0.044 regarding the F1 score, which is about 5.3% better (see table I). These results clearly show the advantages of the more sophisticated U-Net architecture and the access to global information as opposed to local information inside 16×16 patches.

We observed a significant speedup of the training with respect to the loss when using per-image normalization. However, the weighted loss did not provide any clear advantage with respect to the F1 score. Finally, training with our augmented dataset resulted in a much better F1 score than without (see table I). Another interesting thing we observed is that the patch F1 score did not decrease or sometimes even kept increasing as the model started to overfit. In some cases, we got better results by letting the model overfit on purpose.

B. Style Transfer

Unfortunately, we were not able to achieve satisfying and convincing results using the style transfer approach (see figure 2). Moreover, roads were never really obstructed by shadow or trees, too large content-loss made the groundtruths inaccurate and the background parts were not always free of roads. Thus, we quickly abandoned this approach as it proved to be too ambitious and difficult. Another interesting idea would be to use procedural 3D models and ray tracing to generate satellite images as realistically as possible, but this would still incur the difficulty of creating a realistic placement of roads and buildings, which was not in the scope of this project.

Model	F1 Score
Patch CNN baseline	0.82820
U-Net baseline	0.87240
U-Net baseline (with improvements)	0.89742
Res-U-Net-18	0.88940
Res-U-Net-18 (with ImageNet pertained weights)	0.90528
Class-specific models	0.91037
Res-U-Net-18 (with additional Google Maps data)	0.91504
Res-U-Net-34 (with external data and enhanced inference)	0.91648

Table I
COMPARISON OF MOST IMPORTANT MODELS.

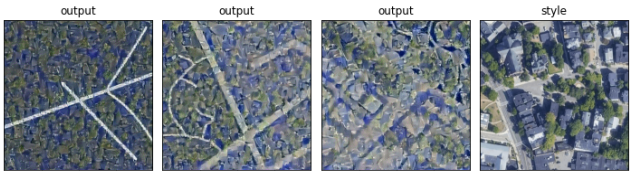

Figure 2. Some intermediate results we obtained using the image on the right as style reference.

C. Res-U-Net Models

Using the Res-U-Net-18 model, we were only able to consistently outperform our improved version of the U-Net baseline model when using the pre-trained ResNet-18 weights. Transfer learning was clearly beneficial in our case, especially because our augmented dataset was still rather small. We also observed that the training time much shorter since the Res-U-Net-18 model has much fewer parameters than the U-Net baseline model. When we repeated our experiments with the Res-U-Net-50 and Res-U-Net-152 models (again with pre-trained weights), we did not observe a clear improvement with respect to the F1 score. We were also not able to achieve any improvements with our modifications to the model architecture where we applied dilated convolutions.

D. Class-Specific Models

The class-specific Res-U-Net-18 models were able to slightly outperform a single Res-U-Net-18 model. As expected, the model for class 0 works best and has the highest validation F1 score since almost all roads are very well visible and rather straightforward to separate from the background. We assumed that the model for class 1 would

perform the worst due to all the tree occlusions. Contrary to our expectations, the model for class 2 actually has the worst performance. A reason for that might be that the distinction between the roads and the background is often quite ambiguous because of the numerous alleyways between the large buildings that look like roads but are not labeled as such.

Finally, we would like to note that this approach would not be suitable in a more general setting. This method assumes that the shown images are drawn from the exact same distribution (locations) as our training set, which is only the case in this project. If we would apply our trained class-specific models to arbitrary satellite images, this approach would not perform as well since there are obviously many locations on Earth that look completely different than the ones in the provided dataset and thus not properly covered by the K-means model. However, we believe that this was a creative approach worth exploring.

E. Generation of Additional Training Data

By fine-tuning the Res-U-Net-18 model with weights that were trained on our additionally generated Google Maps dataset, we were once again able to improve our F1 score a bit. As expected, we achieved a better model performance than by simply using pre-trained ImageNet weights for the ResNet-18 encoder of the Res-U-Net-18 model.

F. External Training Data and Enhanced Inference

For this approach, we used all the available manually-labeled images, namely the MA and DG datasets, to train an efficient U-Net built with pre-trained ResNet backbones. Based on our preliminary experiments on smaller datasets, we chose ResNet-34, implemented in Segmentation Models Pytorch library [11], as our encoder structure to balance between the expressive power (depth) of the network and the training efficiency on a single-GPU desktop. In the contracting path of the network, the number of filters in each block increase by a factor of 2: starting from 64 filters to 512 filters in the fourth and the fifth block. The whole model had 35,041,409 parameters in total.

Training was run with decaying learning rate until the average dice score metric on the validation set stopped improving, which took over 60 epochs. Each epoch took around 30 minutes on a Nvidia RTX 3080 GPU.

The basic sliding window inference with this model improved the F_1 score significantly from the same Res-U-Net-34 network structure but trained with only the data; while it achieves slightly better F_1 score (≈ 0.001) than the Res-U-Net-18 structure trained with Google Maps data (see table I). By adopting the enhanced inference techniques, our F_1 score achieves 0.91648. Some sample predictions are shown in figure 3 to demonstrate the model's improved capabilities regarding generalization (c) and the removal of false positives by the enhanced inference (d).

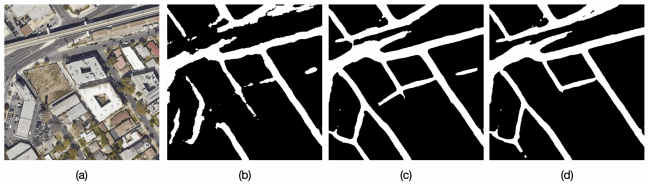

Figure 3. Sample test image (a) and its predictions by Res-U-Net-34 trained with (b) only the given training set (converged), (c) with additional MA and DG datasets, and (d) with enhanced inference.

IV. Conclusion

We presented a variety of approaches to predict road segmentation maps from aerial images using convolutional neural networks. On one hand, our work shows the advantages of (1) incorporating additional training data, either the manually-labeled public dataset or the data generated automatically from the Google Maps API, and (2) training with custom U-Net models built with pre-trained ResNet backbones, when our given training data is limited. We reported consistent performance increases and showed that our model generalizes well to the test set, which conforms to previous research on other image segmentation tasks. However, the optimized network structure, namely the depth and the number of channels for each layer, requires further experiments.

On the other hand, we explored several creative and rather unconventional approaches: A style transfer-based approach to generate additional training samples, which turned out to be too ambitious, and an approach based on an unsupervised partitioning of the data where we use a specialized model for each implicit class.

References

[1] S. Abdulateef and M. Salman, "A comprehensive review of image segmentation techniques," *Iraqi Journal for Electrical and Electronic Engineering*, vol. 17, pp. 166–175, 12 2021.

[2] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *CoRR*, vol. abs/2001.05566, 2020. [Online]. Available: https://arxiv.org/abs/2001.05566

[3] Z. Cui, J. Yang, and Y. Qiao, "Brain mri segmentation with patch-based cnn approach," in *2016 35th Chinese Control Conference (CCC)*, 2016, pp. 7026–7031.

[4] 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*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.

[5] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," *CoRR*, vol. abs/1508.06576, 2015. [Online]. Available: http://arxiv.org/abs/1508.06576

[6] Z. Zhang, Q. Liu, and Y. Wang, "Road extraction by deep residual u-net," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 5, pp. 749–753, 2018.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li, "Imagenet: a large-scale hierarchical image database," 06 2009, pp. 248–255.

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

[10] 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.

[11] P. Iakubovskii, "Segmentation models pytorch," https://github.com/qubvel/segmentation_models.pytorch, 2019.