Supe®oad: Road segmentation through multi-objective ensemble and geometric-aware post-processing

Davin Choo, Hsuan-I Ho, Juan-Ting Lin & Vaclav Rozhon*
Group: One Punch LuRouFan
Department of Computer Science, ETH Zurich, Switzerland

***Abstract*—Road segmentation is a popular computer computer vision task. In this project, we modified a state-of-the-art convolutional neural network and developed a geometric-aware post-processing pipeline. As neural networks are data-hungry, we augment the lack of training data from the CIL dataset with external datasets. To further boost performance, we used an ensemble of models. With these techniques, we were able to rank 2^{nd} in the both the public and private leaderboards of the 2019 CIL road segmentation Kaggle contest.**

I. Introduction

The road segmentation problem is determining whether each pixel of a given image is part of a road, or not. Road extraction from aerial images is a popular problem in computer vision with many applications such as automated updating of road maps [1] and detecting road damage [2].

A. Related work

Recently, convolutional neural networks (CNNs) [3], [4] have shown impressive performance in tackling the problem of semantic/instance segmentation [4], [5], [6], [7]. Although the raw output of a CNN performs well in pixel-wise evaluations, it usually lacks visual structure. As such, various techniques have been proposed to post-process the output of CNNs for road extraction. These include probabilistic models [8], structured SVM [9] and heuristics such as sampling the junctions [10], bridging gaps in roads via A^* search [11], or finding shortest paths [12].

B. Issues with state of the art

While CNNs have demonstrated the ability to extract semantic information in a data-driven manner, large amounts of training data are required [13], [14], [15]. Data augmentation and transfer learning are two main methods to combat the lack of training data. Data augmentation involves generating more training data or using external datasets. For instance, Kaiser et al. [16] showed how one can generate coarse supervisions from Google Maps and Open Street Maps. Meanwhile, transfer learning reduces the computational burden of training a neural network by starting with a pre-trained network. In this project, we used external datasets (DeepGlobe [17] and SpaceNet [18]) and adopted the architecture of Xception [19] as our backbone segmentation encoder.

Compared to semantic segmentation tasks on common objects [13], [15], the problem of road segmentation is a specific instance of semantic segmentation with only two classes and it has strong geometrical priors. This motivates the use of heuristics and post-processing techniques on the CNN output. We developed our own post-processing pipeline of performing heuristic line smoothing (Section II-B1) before applying a graph cut algorithm (Section II-B2).

C. Contributions

We propose a road segmentation method consisting of a semantic-aware soft label predicting model and a geometric-aware post-processing pipeline. External datasets were used in the training process to augment the training images provided and we evaluated our approach using the data from the 2019 CIL road segmentation contest[1].

II. Models and Methods

Given a dataset $\mathcal{D}$ consisting of RGB satellite images and road segmentation masks pairs $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$, we aim to design an algorithm that extracts road mask z for an unseen input RGB image x. In this section, we describe our neural network architecture for road segmentation and how we post-process its output using heuristic line smoothing and graph cut. Fig. 1 shows a training pair while Fig. 2 illustrates our entire pipeline: Given an input image x, our network returns a softmax output $\widetilde{y}$. After post-processing it via our heuristic line smoothing to obtain $\widetilde{y}^s$, we run a graph cut algorithm to compute a binary label assignment z. Note that our post-processing steps (line smoothing and graph cut) are applied after the model prediction without any supervised data, thus they do not incur any training computation cost.

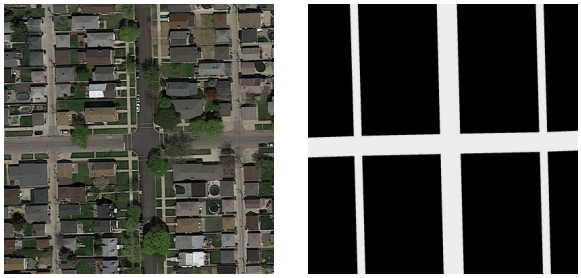
A. Neural network architecture

As suggested in [20], we use a network structure that exploits dilated/atrous convolution and pyramid pooling[2] in cascaded Xception [19] to perform semantic segmentation. However, Yu et al. [21] pointed out that simply applying skip connections to different level layers in the network may

*Names are in alphabetical order of first names.

[1] https://inclass.kaggle.com/c/cil-road-segmentation-2019

[2] This is also known as Atrous Spatial Pyramid Pooling (ASPP).



(a) RGB input x (b) Segmentation mask y

Figure 1: RGB input and its road segmentation mask

not effectively fuse the semantic information in the feature maps. Taking inspiration from their work, we extend our network using interactive deep aggregation. To be precise, the fused feature maps are first generated by applying convolution to the aggregation of low-level feature maps f_1 and f_2, then the resulting fused feature maps are once again concatenated with the semantic feature maps f_s to obtain the final predictions $\widetilde{y}$. See Fig. 2 for an illustration.

In the Kaggle contest, we only care about performance and not inference time. Hence, we consider using an ensemble of neural networks. To encourage diversity within the ensemble, we employed different loss functions to focus on different aspects of the problem. We first describe the loss functions ($L_{CE}, L_{WCE}, L_J, L_F, L_{TV}$) used before explaining how we combined them in the ensemble.

1) Loss functions: Since road segmentation task is a pixel-level binary classification task, a typical loss function to use would be the binary cross-entropy loss function $L_{CE}(\widetilde{Y}, Y)$ where $Y = \{y^{(1)}, \ldots, y^{(N)}\}$ are the binary ground-truth masks with corresponding predicted probabilistic masks $\widetilde{Y}$. To handle the imbalance of classes in the road segmentation dataset, we also considered the weighted cross-entropy loss $L_{WCE}(\widetilde{Y}, Y, \alpha)$ where we weight the loss term from class "roads" by an α factor.

Another way to combat class imbalance is to consider the Jaccard index as suggested by [22]. Given softmax prediction masks $\widetilde{Y}$ and ground-truth masks Y, the Jaccard index is defined as $J(\widetilde{Y}, Y) = \frac{|\widetilde{Y} \cap Y|}{|\widetilde{Y} \cup Y|} = \frac{|\widetilde{Y} \cap Y|}{|\widetilde{Y}| + |Y| - |\widetilde{Y} \cap Y|}$. Suppose there are n pixels across all images, then we can formulate a pixel-wise Jaccard index as follows:

$$J(\widetilde{Y}, Y) = \frac{1}{n} \cdot \frac{\left(\sum_{i=1}^{n} \widetilde{y}_i y_i\right) + \epsilon}{\left(\sum_{i=1}^{n} \widetilde{y}_i + y_i - \widetilde{y}_i y_i\right) + \epsilon}$$

where $\widetilde{y}_i$ is the predicted probability for the i^{th} pixel, y_i is the binary ground truth for the i^{th} pixel, and ϵ is a small smoothing term in case the denominator is zero. We then define the Jaccard loss as

$$L_J(\widetilde{Y}, Y) = 1 - J(\widetilde{Y}, Y)$$

We also considered two other loss functions — focal loss [23] and total variation loss [24]. Focal loss L_F can be viewed as a re-parameterization of the cross-entropy loss by weighting higher training loss for pixels with higher uncertainty. On the other hand, total variation loss L_{TV} penalizes large changes between adjacent pixel predictions. Let $\widetilde{y}_{i,j}$ be the prediction probability for the pixel with coordinate (i, j), and γ be a hyperparameter. Then,

$$L_F(\widetilde{Y}, Y) = \sum_{\widetilde{y} \in \widetilde{Y}} \sum_{i,j} -(1 - \widetilde{y}_{i,j})^{\gamma} \log(\widetilde{y}_{i,j})$$

and

$$L_{TV}(\widetilde{Y}, Y) = \sum_{\widetilde{y} \in \widetilde{Y}} \sum_{i,j} |\widetilde{y}_{i+1,j} - \widetilde{y}_{i,j}| + |\widetilde{y}_{i,j+1} - \widetilde{y}_{i,j}|$$

2) Ensemble: We use the following three combinations of loss functions and output their mean prediction as $\widetilde{y}$.

Model 1 :$\mathcal{L}_{train} = 0.2 \cdot L_J + L_F$
Model 2 :$\mathcal{L}_{train} = 0.2 \cdot L_J + L_{CE}$
Model 3 :$\mathcal{L}_{train} = 0.2 \cdot L_J + L_{WCE} + 10^{-7} \cdot L_{TV}$

Observe that the total variation is very sensitive to relative weighting of the other loss functions. In actuality, we used more than three models because we varied other hyperparameters such as training rate.

B. Post-processing

The segmentation output from a neural network is typically fragmented and lacks structure. We apply two post-processing techniques to the softmax prediction $\widetilde{y}$ to smoothen the prediction mask and connect disjoint roads.

1) Heuristic line smoothing: The first step is to fill in gaps created by objects on the roads such as cars or trees. To do so, we use the probabilistic output mask $\widetilde{Y}$ as a guideline to increase the probability of each pixel being a road by performing line smoothing under rotations.

We first describe vertical line smoothing, then explain how this generalizes to line smoothing under arbitrary rotations. Let r, R, and τ be parameters such that $0 < r < R$. For each pixel (i, j) with soft prediction by the neural network $p_{i,j}$, we consider a strip of $2R + 1$ pixels with (i, j) in the middle. We define 4 sets of pixel probabilities with respect to pixel (i, j):

$$\begin{aligned} S_{down} &= \{p_{i,j}, p_{i,j-1}, \ldots, p_{i,j-R}\}, \\ S_{up} &= \{p_{i,j}, p_{i,j+1}, \ldots, p_{i,j+R}\}, \\ T_{down} &= \{p_{i,j}, p_{i,j-1}, \ldots, p_{i,j-r}\}, \\ T_{up} &= \{p_{i,j}, p_{i,j+1}, \ldots, p_{i,j+r}\}. \end{aligned}$$

The sets S_{down} and S_{up} are our "smoothing windows" while T_{down} and T_{up} are "threshold lookaheads". To fill small gaps in vertical roads, we set the probability of pixel (i, j) being a road to the largest probability in S_{down} and S_{up}. However, this process alone may widen roads unnecessarily. As a remedy, we consider the sets T_{down} and T_{up}, and only

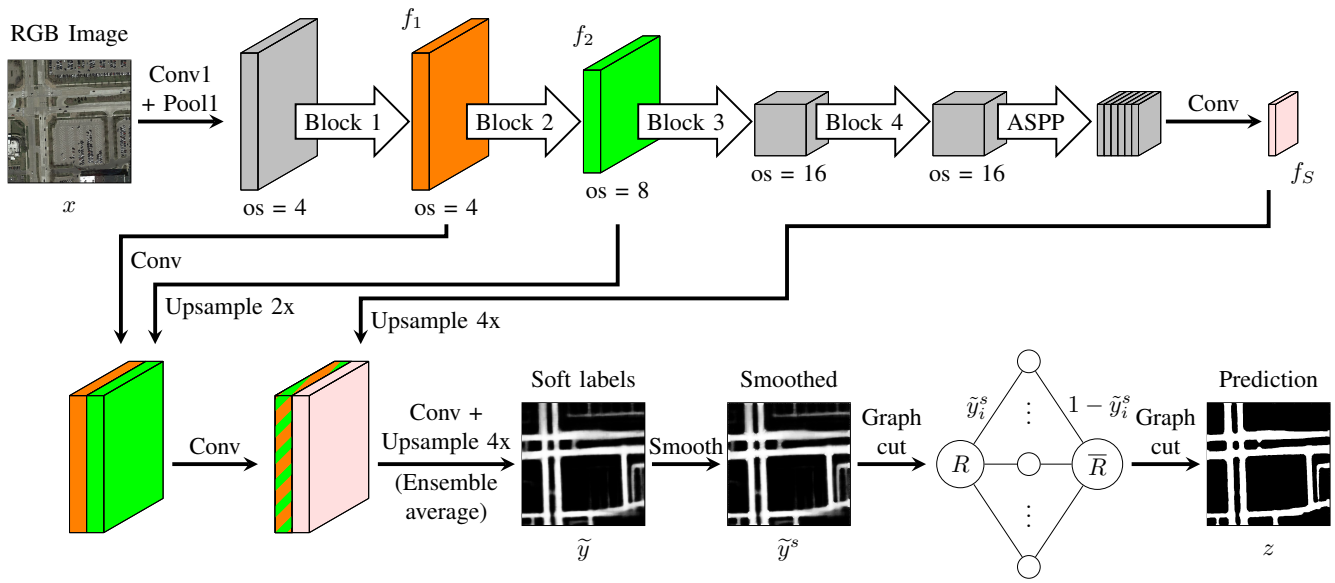


Figure 2: Road segmentation pipeline on `test_25.png`

apply smoothing if its maximum value exceeds τ. To be precise, we define s^*_{down}, s^*_{up}, and s^* as follows:

$$s^*_{down} = \begin{cases} \max_{p \in S_{down}} p & \text{if } \max_{p \in T_{down}} p \geq \tau \\ 0 & \text{otherwise} \end{cases}$$

$$s^*_{up} = \begin{cases} \max_{p \in S_{up}} p & \text{if } \max_{p \in T_{up}} p \geq \tau \\ 0 & \text{otherwise} \end{cases}$$

$$s^* = \max\left\{p_{i,j}, \min\{s^*_{down}, s^*_{up}\}\right\}$$

We generalize the above smoothing method under rotations by computing s^*_θ after rotating the mask $\widetilde{y}$ by $\theta \in \{0°, 45°, 90°, 135°\}$. Then, we set the probability $\widetilde{y}^s_{i,j}$ for pixel (i, j) to the maximum of s^*_0, s^*_{45}, s^*_{90}, and s^*_{135}. Fig. 3 illustrates how s^*_{45} is computed.

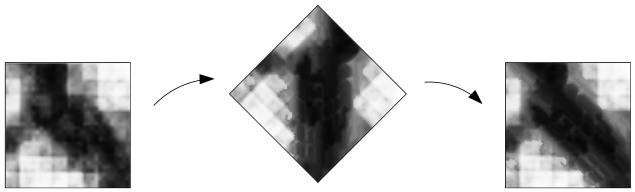


Figure 3: Example of s^*_{45} being computed on a small patch

2) Graph cut: Inspired by [25] and [26], we adopt a graph cut algorithm as our second stage prediction refinement. Consider the following weighted graph $G = (V, E)$:

- Set V contains all n pixels and two sentinel nodes R and $\overline{R}$, indicating "road" and "not road" respectively.
- Each pixel is connected to R with an edge weight of $\widetilde{y}^s$, and to $\overline{R}$ with an edge weight of $1 - \widetilde{y}^s$.
- Adjacent pixels are connected with an edge weight proportional to their similarity in the LAB color space.

We derive the binary mask prediction z by finding a minimum cut separating R and $\overline{R}$, then assigning pixels according to their partition. One can define graph cut as minimizing the following energy function:

$$E(Z) = \sum_{i=1}^{n} \psi_i(z_i) + \lambda \sum_{i=1}^{n} \sum_{j \in N(i)} \phi_{i,j}(z_i, z_j)$$

where λ is the hyperparameter, $z_i \in \{0, 1\}$ is the binary label of the i^{th} pixel, $\psi_i(0) = 1 - \widetilde{y}^s_i$, $\psi_i(1) = \widetilde{y}^s_i$, and

$$\phi_{i,j}(z_i, z_j) = \begin{cases} \exp\left(-\frac{(I_i - I_j)^2}{2\sigma^2}\right) & \text{if } z_i \neq z_j, \\ 0 & \text{otherwise.} \end{cases}$$

I_i and I_j are 3-dimensional LAB color values for the i^{th} and j^{th} pixels. The function $\phi_{i,j}$ penalizes heavily for similar pixel colors when $|I_i - I_j| < \sigma$, while being close to zero when adjacent pixels have very different colors.

III. Experiment

A. Dataset and Evaluation Metrics

We trained our network on the dataset provided by the 2019 CIL road segmentation contest. The training set consists of 100 RGB aerial images of size 400×400 as well as their corresponding binary road masks. Our classification is evaluated on Kaggle leaderboard via the accuracy of the 94 608×608 test images. Due to the lack of validation data, we further perform a $90 - 10$ training-validation split of the given training dataset. In our experiments, we also include the pixel-wise mean Intersection over Union (mIoU) score

of the validation set. mIoU captures the Jaccard index and is a common metric used in object detection tasks.

To prevent over-fitting on the small number of training data, we collected other two road segmentation datasets — DeepGlobe [17] and SpaceNet [18] — to pretrain our model. DeepGlobe contains 6226 RGB satellite images of size 1024×1024 alongside their road segmentation masks, while SpaceNet consists of 1774 16-bit satellite imagery of size 1300×1300 and their road information in GeoJSON format. To use SpaceNet, we converted the 16-bit imagery into RGB format and encoded the GeoJSON data into corresponding image masks before training. Both datasets serve as good sources of data augmentation to pretrain our network.

B. Implementation Details

As our backbone segmentation encoder, we adopted the architecture of Xception [19] with its weights being pre-trained on the ILSVRC-2012-CLS [27] image classification dataset before fine-tuning with the CIL dataset. During the training process, we apply batch normalization to convolution layers and perform simple image manipulation of the training images such as random flips, random rotations and random crops. We train our network using the Adam optimizer with a batch size of 4, first-momentum of 0.9, and second-momentum of 0.99. The learning rate for pre-training the model on DeepGlobeand and SpaceNet is set to 5×10^{-6}, and to 5×10^{-7} when we fine-tune using the CIL dataset. In the network, we use the hyperparameters $\alpha = 1.2$, $\epsilon = 10^{-9}$ and $\gamma = 0.5$. For post-processing, we set $r = 3$, $R = 16$, $\tau = 0.25$, $\sigma = 10$, and $\lambda = 5$.

C. Comparison

In Table I, we compare our work with four baselines (**KNN**, **SVM**, color-based **GraphCut**, **ShallowCNN**) and three state-of-the-art segmentation models (**FCN** [4], **DeepLabV3** [20], **DeepLabV3+** [28]. We first describe how each baseline is compute.

KNN We compute the ratio of roads versus non-roads for each RGB pixel value. That is, we obtain a mapping $f : [256]^3 \rightarrow [0, 1]$. Then, for each pixel x in a test image, let x_i be the i^{th} closest point to x in the RGB space. The soft prediction output for pixel x is then $\frac{1}{\sum_{i=1}^{k} d(x,x_i)} \sum_{i=1}^{k} d(x, x_i) \cdot f(x_i)$, for $k = 5$.

SVM We sampled 5626 RGB pixel values and labels from the training set, and trained a SVM model with Gaussian kernel of bandwidth 0.1. A 16×16 patch is labelled as a road if at least 30% of the pixels are labelled as a road by the SVM. The 30% threshold and the SVM bandwidth were optimized via a 90-10 split.

GraphCut We derive a binary prediction mask by minimizing the energy function similar to the graph cut algorithm in Section II-B1 where ψ is computed by estimating the likelihood of a pixel in the "road" and "non-road" color distribution instead.

	Methods	Public	Private	mIoU
Baseline	KNN	0.83563	0.81630	0.48670
	SVM	0.84752	0.83051	0.21423
	GraphCut	0.57459	0.55377	0.40879
	ShallowCNN	0.44929	0.47613	—
SOTA	FCN	0.87376	0.86878	0.75439
	DeepLabV3	0.87281	0.86406	0.75810
	DeepLabV3+	0.87143	0.86376	0.77778
Ours	W/o pre-training	0.87640	0.86821	0.78766
	W/o post-processing	0.91477	0.91103	0.84934
	Single full model	0.91936	**0.91595**	0.85677
	Model ensemble	**0.92472**	0.91451	**0.87010**

Table I: Performance evaluation and comparison

ShallowCNN In CIL tutorial 11, we were given a 2-layer CNN to classify 16×16 image patches. As it operates on patches, we can only compute patchwise accuracy on Kaggle but not pixelwise mIoU.

D. Discussion

Table I summarizes the quantitative results of our proposed method against baselines and state-of-the-art methods. Our method produced favorable results due to integration of learning segmentation information from external datasets and combination of different training losses.

Approaches simply using color information such as KNN, SVM, color-based graph cut were unable to achieve comparable results due to a lack of ability in modeling complicated aerial images. While different state-of-the-art network architectures have similar performance, we observe that using a pre-trained backbone encoder significantly improves the performance of a neural network, justifying transfer learning.

We performed an ablation study to verify each design choice of our proposed method. The use of external training data provided a significant performance gain, even if the distribution of images is not necessarily the same. While our post-processing pipeline only gave a modest improvement, a further use of an ensemble of networks eventually earned us a second place position in the Kaggle leaderboard.

IV. Summary

We used data augmentation to train an ensemble of neural networks for road segmentation and designed a post-processing pipeline to enforce structure into the output of the ensemble. This resulted in a high accuracy classifier with an accuracy of 0.92472 on the private test set, and mIoU of 0.87010 on the CIL road segmentation dataset.

References

[1] V. Mnih and G. E. Hinton, "Learning to detect roads in high-resolution aerial images," in *European Conference on Computer Vision (ECCV)*, 2010.

[2] H. Ma, N. Lu, L. Ge, Q. Li, X. You, and X. Li, "Automatic road damage detection using high-resolution satellite images and road maps," in *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2013.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[4] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[5] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-assisted Intervention*, 2015.

[6] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017.

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[8] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017.

[9] E. Ricci and R. Perfetti, "Large margin methods for structured output prediction," in *Computational Intelligence Paradigms*, 2008.

[10] D. Chai, W. Forstner, and F. Lafarge, "Recovering line-networks in images by junction-point processes," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[11] G. Máttyus, W. Luo, and R. Urtasun, "Deeproadmapper: Extracting road topology from aerial images," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[12] T. Sun, Z. Chen, W. Yang, and Y. Wang, "Stacked u-nets with multi-output for road extraction," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[13] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision (IJCV)*, 2015.

[14] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[15] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision (ECCV)*, 2014.

[16] P. Kaiser, J. D. Wegner, A. Lucchi, M. Jaggi, T. Hofmann, and K. Schindler, "Learning aerial image segmentation from online maps," *IEEE Transactions on Geoscience and Remote Sensing*, 2017.

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

[18] A. Van Etten, D. Lindenbaum, and T. M. Bacastow, "Spacenet: A remote sensing dataset and challenge series," *arXiv preprint arXiv:1807.01232*, 2018.

[19] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[20] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[21] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep layer aggregation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[22] A. Buslaev, S. S. Seferbekov, V. Iglovikov, and A. Shvets, "Fully convolutional network for automatic road extraction from satellite imagery," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[24] M. Javanmardi, M. Sajjadi, T. Liu, and T. Tasdizen, "Unsupervised total variation loss for semi-supervised deep learning of semantic segmentation," *arXiv preprint arXiv:1605.01368*, 2016.

[25] C. Poullis and S. You, "Delineation and geometric modeling of road networks," *Journal of Photogrammetry and Remote Sensing (ISPRS)*, 2010.

[26] M. Rajeswari, K. Gurumurthy, L. P. Reddy, S. Omkar, and J. Senthilnath, "Automatic road extraction based on normalized cuts and level set methods," *International Journal of Computer Applications (IJCA)*, 2011.

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

[28] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *European Conference on Computer Vision (ECCV)*, 2018.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**

With my signature I confirm that
- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.