Edge-related Deep Learning for Semantic Segmentation

MONA GOEBEL[1], LINA E. BUDDE[1] & DOROTA IWASZCZUK[1]

Abstract: The identification of Land Use and Land Cover is an important task for creating maps or monitoring surface changes. The results in this work were obtained using a deep neural network called VGG19-Unet. Influenced by the increasing success of separately implementing edge-extracted inputs in a model, this work used the Canny method to create such edge extracted images. In the final model, one band was discarded from each of the original RGB and NIRRG images and replaced with an edge-extracted band, generating a closer connection between colour and edge bands. This model simultaneously learned eight classes. Segmentation results were found to be predominantly sharper when edge-extracted bands were included, and the model was more confident in its choice of classes. Lastly, improvements were achieved for objects with clear edges, such as buildings, as well as for objects with unclear edges, such as vegetation.

1 Introduction

The identification of Land Use and Land Cover (LULC) is an important task for creating maps or monitoring surface changes. Applications for such maps include environmental monitoring, agriculture, forestry and urban planning. These classification tasks combine remote sensing and computer vision (KAMPFFMEYER et al. 2016). To do this, an accurate raster map is needed in which each pixel is assigned to a specific class. This is called semantic image segmentation or pixel-level prediction and is a complex task in scene understanding (ZOU et al. 2019; LIU et al. 2019).With the introduction of Fully Convolutional Networks (FCNs) by LONG et al. (2015), this area became known and computationally possible (ZOU et al. 2019). Since then, models can also generate spatial maps and not only class labels per image (MARMANIS et al. 2017). Moreover, research advance in this field is not limited to Earth observation, but is also relevant for automatic transportation, robotics and overall automatic image analysis and sorting.

The success of model prediction is highly dependent on image pre-processing and the chosen methods and model architecture. Overall, the Deep Learning approach is more successful than methods such as support vector machines or decision trees (MA et al. 2019). Recently, researchers integrated edge detection into models to achieve sharper and cleaner class boundaries. Examples are EdgeNet introduced by CHEN & BARRON et al. (2016) including an edge-preserving mechanism, as well as models by MARMANIS et al. (2017) or ZOU et al. (2019) which use Holistically-Nested Edge Detection (HED) to improve their models.

This work investigates whether adding separately extracted edges to a deep neural network returns better results than without, even though convolutional filters within an FCN already extract edges. Furthermore, an investigation is made whether objects with clear edges, such as buildings, as well

[1] Technical University Darmstadt, Department of Civil and Environmental Engineering Sciences, Remote Sensing and Image analysis, Franziska-Braun-Str. 7, D-64287 Darmstadt,
E-Mail: [mona.goebel, lina.budde, dorota.iwaszczuk]@tu-darmstadt.de

 DOI: 10.24407/KXP:1796047864

as objects with unclear edges, such as vegetation, can be predicted better with the integrated edge extraction. For this, the focus will be on pixel-wise semantic segmentation with satellite images of urban areas using supervised Deep Learning. Training and testing of the model will be performed using the SemCity Toulouse benchmark by ROSCHER et al. (2020). The authors discuss in the published paper (ROSCHER et al. 2020) only instance segmentation and solely for buildings. In this work, all seven classes plus unclassified pixels (equals to eight classes) will be addressed simultaneously.

2 Background and Related work

2.1 Edge Detection

Detecting edges in images is essential in order to simplify images, yet preserving important features. Therefore, an errorless edge detection is crucial for the model prediction. If edges are missed, the model will not notice the importance of a specific pixel and on the contrary, if too many unimportant edges are present, then the model will not recognise the meaningful ones. Summarised, three key points of a successful edge detection can be listed: First, the result should have a low chance of missing the real edge point, while also not misclassifying non-edge points. Secondly the marked edge point should be near the centre of the real edge point. Lastly, the possibility of showing several maxima for one edge should be low, hence resulting only to one signal for one edge. (CANNY, 1986)

The edge detection function implementation in the python package scikit-image was applied in this work and is based on the paper by CANNY (1986). Some adjustments were made to the original idea, which led to the following process: At the beginning the image is smoothed with derivative of the Gaussian operator, just as in the base algorithm by CANNY (1986). Thereafter the Sobel operator in the horizontal and vertical direction is calculated, detecting high pixel intensities. The wide intensities are narrowed to a 1-pixel wide curve, by analysing the gradient directions. Possible orientations can be horizontal, vertical, diagonal and anti-diagonal. A check is then made on the focused pixel to see if it is a maximum, a minimum, or neither. By calculating the interpolation of the four directions, the result has a higher probability of being accurate than choosing the direction that is the closest. Finally, the thresholds define all pixels above the upper threshold to be edges, while the pixels below the lower threshold are background. The pixels in between are edges if they are maxima of the surrounding neighbouring pixels. (SCIKIT-IMAGE n.d.)

Just as the Laplacian of Gaussian, the Canny edge detector is categorised as Gaussian-based. It is known to be robust to noise and to be the best method, by reason of detecting edges without modifying or influencing the image features (KRISHNAN et al. 2017). The idea of including different resolutions is also covered in the Canny operator (HAFFNER & RAVAS 2014). Additionally, the Canny operator's success is extremely dependent on the choice of thresholds and width of the Gaussian σ. The latter sets the frame of neighbouring pixels that are included in the decision whether a point is an edge or not. Setting an unsuitable threshold can suppress true edges while reducing noise (HAFFNER & RAVAS 2014).

2.2 VGG and U-Net concept

The model used in this work is based on the VGG convolutional neural networks and the U-Net concept. ConvNet was originally presented by SIMONYAN & ZISSERMAN (2014) and is commonly called VGG. It won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2014. Different VGG networks were adapted in many works, such as ResNet (HE et al. 2015) and DeepLab (CHEN et al. 2017).

U-Net was first presented by RONNEBERGER et al. (2015) and represents an encoder-decoder. The paper mentions the utility for pixel-wise classification for biomedical images, as well as the difference between a large training set like ImageNet and the typical small dataset in biomedical cases. The authors also aimed to reduce redundancy in training and improve the processing time of the approaches commonly used at that time. Their FCN idea was based on a balance between good localisation and using the context around a pixel (RONNEBERGER et al. 2015). Original about the U-Net architecture was the symmetry between the encoding and decoding parts. During compression within increasing depth of the model, fine details are lost. Skip connections merge outputs from encoding and decoding, thus reintroduce details and improve prediction results.

2.3 Benchmark

SemCity Toulouse was published in 2020 by a cooperation of research establishments in Germany, France and Switzerland (ROSCHER et al. 2020). The data focuses on the densely populated city of Toulouse in France, where each building is labelled separately. Taken from the Worldview-2 satellite in April 2011, the 8-band image was tiled into 16 smaller images, each covering an area of about 3 km². As the paper states, tiles number 3 and 7 are for training, while 4 and 8 for testing. The benchmark includes multispectral imagery, panchromatic imagery as well as corresponding ground truths in instance and semantic segmentation (ROSCHER et al. 2020).

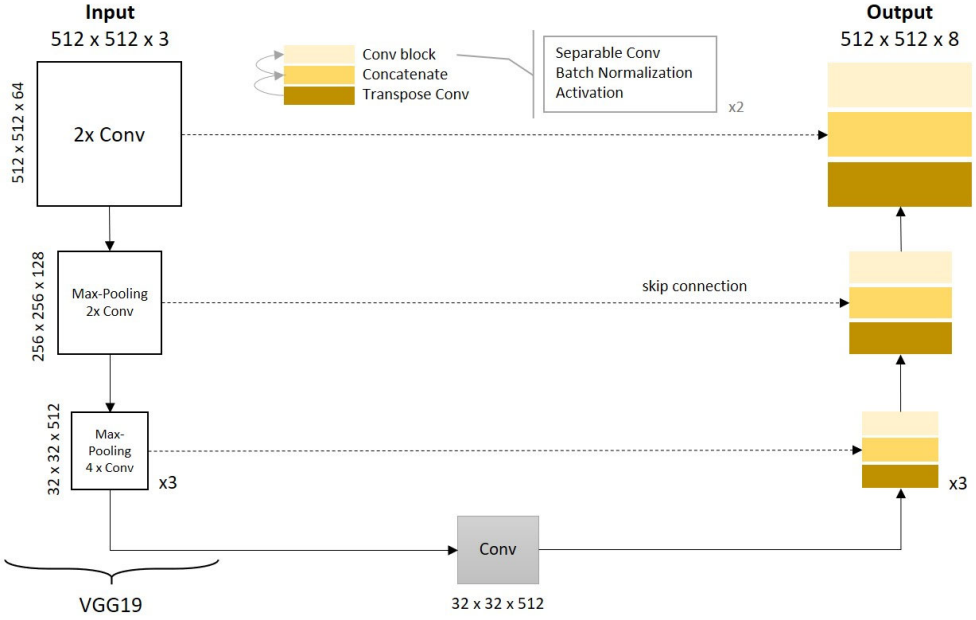

Fig. 1: VGG19-Unet architecture, shown without dropout layers. Conv: convolution, AAA x BBB x C: image size of the layer output, “x3”: sequence is run three times

3 Method

The following section describes the model architecture, model uncertainty, edge extraction, input and finally metrics. The experimental process included in the first phase to investigate the best combinations of layers and channels as well as settings of parameters. Only one comparison will be shown in this work, namely the different combinations of image channels given to the model. The best results were fused into a "final model". Lastly, to interpret the results of this model, a "reference model" was additionally run. The final and the reference model have the same architecture and settings. However, they differ in the input images, as the reference model only receives RGB and NIRRG images and none edge extracted images. The models were run on a computer with 128 GB RAM, an AMD Ryzen Threadripper 2970WX processor with 24 cores, as well as an NVIDIA TITAN RTX graphic card with 24GB.

3.1 Model architecture

The model used in this work is called VGG19-Unet, a VGG19-based encoder-decoder model extended into the U-Net form (Fig. 1). The model code was adapted from VGG19 Unet by Nikhil Tomar (NIKHILROXTOMAR 2021). VGG19 is implemented with Keras (keras.io) which comes with pre-trained weights on the ImageNet benchmark. VGG19 was chosen as the backbone because it is easy and not as complex as the state of the art (Xception, DeepLabV3+), as well as being widely used as a backbone. As shown in Fig. 1, the encoding part of the U-Net architecture consists of an adapted version of VGG19 (on the left side of the figure). Dimensions such as 512 × 512 × 64 represent the output dimensions of the images after each block. The decoding part was added on top of the VGG19 model. The significant difference in the decoding part is the use of depth-wise separable convolution instead of normal convolutional layer. Skip connections are implemented before each max-pooling layer, which merges the layers from the encoder and the previous layer via concatenation. When merging the layers from encoding and decoding, the images must have the same dimensions. Lastly it has an initial input size of 512 × 512 × 3 pixels. The output has the same image size but now with 8 channels instead of 3. For the reason of each channel storing the probabilistic distribution for each pixel and class (Softmax output). Dropout layers were included in the final and reference model after each convolution block with 20 % dropout rate and one dropout layer prior the output layer with a 50 % dropout rate. For the visual interpretation of the result and discussion, the highest probability for each pixel is taken and assigned to that class, resulting in a size of 512 × 512 × 1 pixels. As optimiser, the algorithm Adam by KINGMA et al. (2014) and categorical cross entropy as loss function were chosen.

3.2 Model Uncertainty

To better understand the confidence of the final model, the dropout layers were active during training and testing in order to use the Monte Carlo dropout (GAL & GHAHRAMANI 2016). Since the results differ with this method with each prediction run, it was carried out ten times. Then the median was determined, creating a more stable result. Thereafter, for each class the corresponding pixel indices were extracted from the target labels. With such a class selection the prediction includes, for instance, only the building pixels. Finally, the median and standard deviation for each class channel were calculated, based on the Softmax output.

3.3 Input and edge extraction

As input data, the images from the benchmark SemCity Toulouse were taken, showing the French city Toulouse. The utilised panfused images have a spatial resolution of 0.5 m and are either RGB (red, green, blue) or NIRRG (near-infrared, red, green) images. All images that are given annotated were used, which are images 3, 4, 7 and 8. The original size of 3504 × 3452 × 3 pixels was tiled into smaller squares in the form of 512 × 512 × 3 pixels, with an overlap of 50 %, which results in 169 tiles. In addition, the images were normalised to range from 0 to 1 instead of reaching between 0 and 255. NIRRG and RGB images were treated in the same way. Edge extraction was carried out based on either of the three RGB or NIRRG channels. The extraction was implemented via the Canny function of the feature package of scikit-image, based on CANNY (1986). The width of the Gaussian (σ) turned out to be best fitting with σ = 3.6 for most of the channels. Each tile median is used as a guide for setting the lower and upper thresholds in the Canny function.
Three edge-colour combinations were tested, see Fig. 2 for an overview. The best method was chosen for the "final model". The final model was trained with 906 randomly selected tiles from 1352 tiles, while the remaining 446 tiles were kept for testing.

3.4 Metrics

Comparing the results in numbers, metrics F-score and overall accuracy were selected. The F-score is based on precision and recall, where the first indicates if the found objects are true or false, while recall indicates whether all relevant objects were found from all the objects that should have been found. With β defining the balance between precision (p) and recall (r), the equation is as follows (CHINCHOR 1992):

$$F_\beta = \frac{(1+\beta^2)\times p \times r}{\beta^2 \times p + r}.$$

Normally β is set to 1, just as in the SemCity Toulouse paper (ROSCHER et al. 2020), further called F1. The weighted F1 value takes into account the number of pixels per class, to compensate for imbalance of classes.

4 Results and Discussion

First, models called RGB edges, NIRRG edges and mixed edges are compared and discussed. In the second part, the favourable input method is chosen for the "final model". Overview of the model inputs is given in Fig. 2.

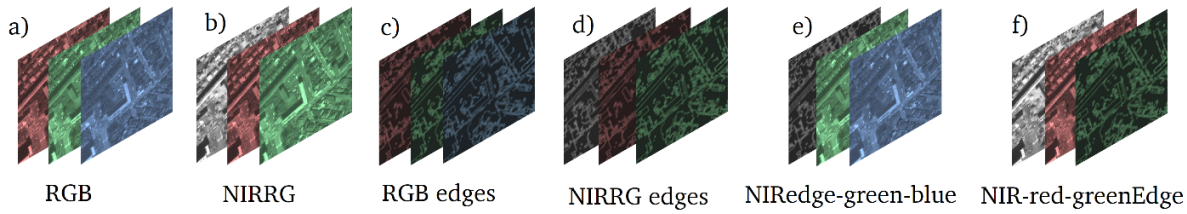

Fig. 2: Overview of model inputs. Model "RGB edges" includes input a), b) and c); "NIRRG edges" a), b) and d); model "mixed edges" and the final model include e) and f); the reference model only a) and b)

4.1 Edge input comparison

The model with RGB edges returns good results, for example in detached housing groups located in the upper left corner in each tile of Fig. 3, however the model has difficulties to distinguish between impervious surfaces and sport venues. In comparison, the model with NIRRG edges as input is challenged between pervious and impervious surfaces as well as confusing impervious surfaces and water pixel. With mixed edges the model returns overall good results. Both NIRRG and mixed edges performing less accurate in the detached housing area. In terms of numbers, the model with RGB edges reached an overall accuracy of 0.56 and a weighted F1 score of 0.55 (Tab. 1). As for the model with NIRRG edges the values are both 0.54 respectively. Lastly, the model with mixed edges achieved an overall accuracy and weighted F1 score of 0.70. This underlines that a closer connection between colour bands and edge bands improves model performance. Consequently, mixed edges were chosen as input for the final model.

4.2 Final model results

The final model needed 1 day and 4 hours and ended training after 27 epochs, whereas the reference model needed 29 epochs and 2 hours more. Fig. 4 shows the visual results. Edges in the final model look overall sharper than from the reference model. Both models seem to have similar problems, for example the objects in white were not detected (Fig. 4, bottom row). Nonetheless, the final model has troubles in the detached housing area (Fig. 4, top row), unlike the reference model. An explanation for this are the extracted edges of this particular area, illustrated in Fig. 5. Thus, if the extracted edges are in a bad quality, the prediction accuracy will decline. This also shows the influence of given edges on the model's success.

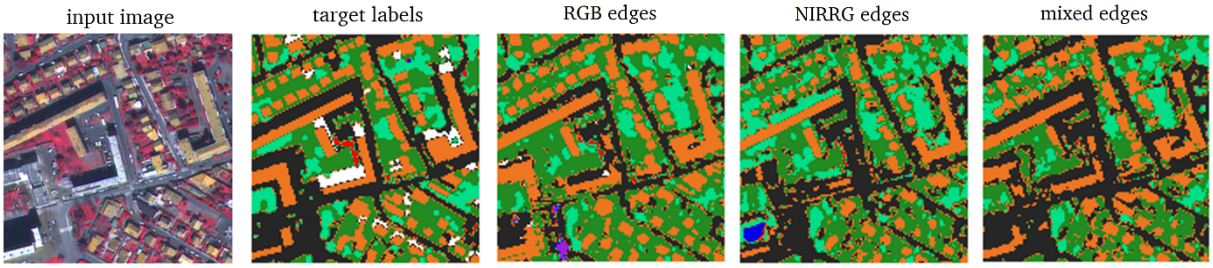

Fig. 3: Results of models with RGB edges, NIRRG edges or mixed edges. The colours represent buildings in orange, impervious surfaces in black, pervious surfaces in dark green, high vegetation in light green, cars in red, water in blue, sport venues in purple and void in white

Tab. 1: Numerical results of models with RGB, NIRRG and mixed edges.
Precision, recall and F1 are given as weighted average, accuracy as overall average

Input	Precision	Recall	F1	accuracy
RGB edges	0.60	0.56	0.55	0.56
NIRRG edges	0.60	0.54	0.54	0.54
Mixed edges	0.72	0.70	0.70	0.70

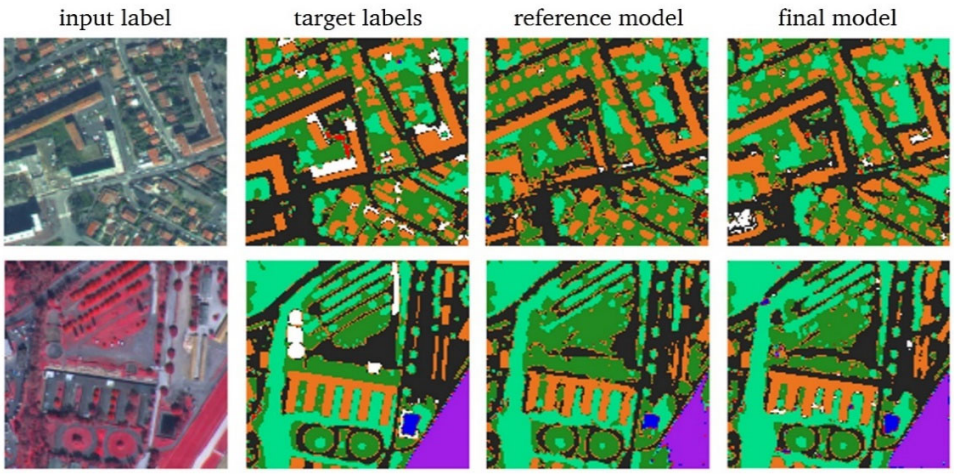

Fig. 4: Visual results of the reference and final model. The colours represent buildings in orange, impervious surfaces in black, pervious surfaces in dark green, high vegetation in light green, cars in red, water in blue, sport venues in purple and void in white

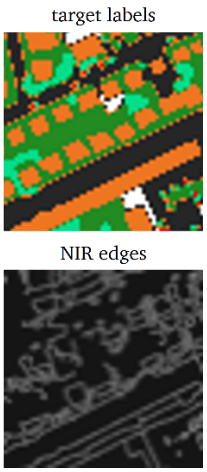

Fig. 5: Example of poorly extracted NIR edges

4.2.1 Numerical results

For the final model the highest F1 score was achieved by the building class (0.90), while the reference model reached 0.91. One class which was only rarely detected in all prior models, namely the sport venues, was now found 83 % of times (recall) by the final model, which is 12 % more than the reference model. Leading to a sport venues F1 value of 0.83 for the final model and 0.81 for the reference model. The least success was found in cars with an F1 score of 0.69, while the reference model reached 0.73. Lower results can derive from incorrect edges that guide the model into false directions. The overall F1 and accuracy is 0.80 for the final model, and 0.82 for the reference model.

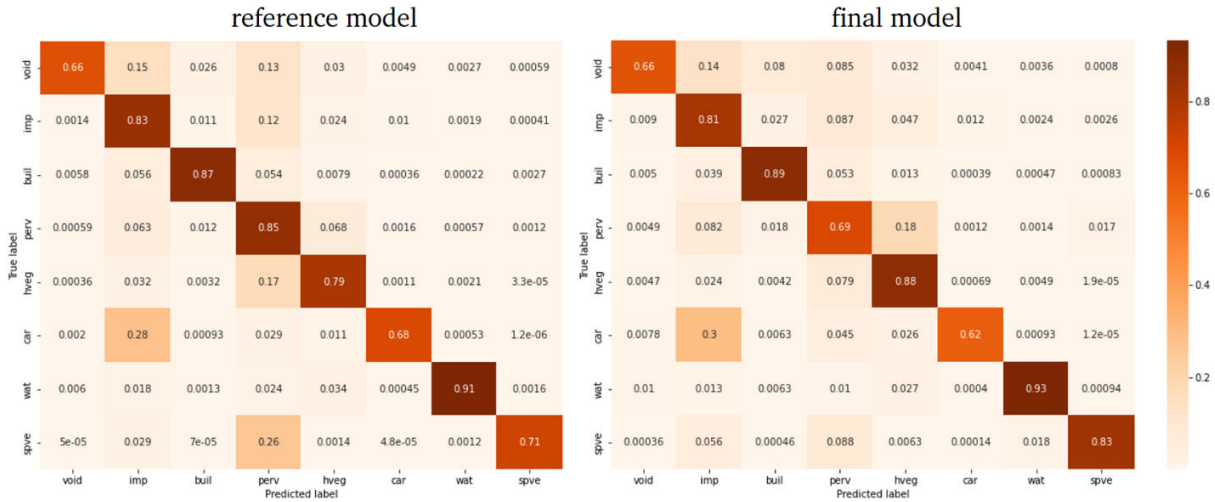

Fig. 6: Normalised confusion matrices of the reference and final model, with the latter including mixed edges. Given numbers represent consensus between predicted and true labels, with 1.00 as complete match. The abbreviations stand for: impervious surface (imp), building (buil), pervious surface (perv), high vegetation (hveg), water (wat) and sport venues (spve)

4.2.2 Model uncertainty

The class-wise recall for the reference and final model are portrayed in the confusion matrices in Fig. 6. Uncertainties of the reference and final model are shown in Fig. 7 for three classes, calculated as explained in the method (section 3). Selected exemplary classes were chosen because they have clean edges (building), unclear edges (high vegetation) or are a challenging class for the models (sport venues). The two confusion matrices show a good diagonal line which implicates that the models perform good and are rather confident.

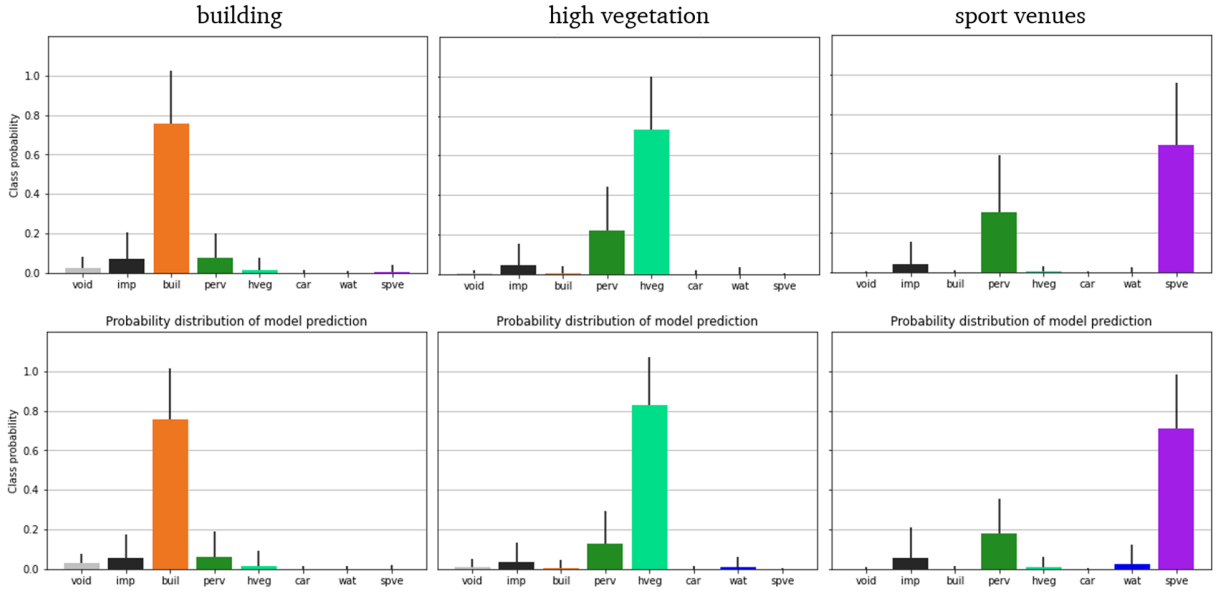

Fig. 7: Graphs show the probabilistic distribution per class of the reference model (top) and final model (bottom), with the latter including mixed edges. Median and standard deviation were calculated only on the pixels that belong to that specific class. For example, for building the pixel that are classified as buildings in the true labels were extracted. The black vertical line represents the standard deviation. See Fig. 6 for abbreviation explanations

The highest recall was reached in the water class for both models, with 0.91 in the reference model and 0.93 by the final model. The lowest percentage of consensus is the car class for both models. Cars were often classified as impervious surface, which is reasonable, as they are mostly surrounded by roads and parking lots. Other than that, they are small objects and only occupy 2 % of the total annotated pixels. Unfortunately, cars were often annotated in the benchmark in a poor quality, thus the true labels could in fact not correspond to the truth.

For the building class, the final model achieved an average confidence of approx. 0.76 ±0.26, very similar to the reference model (Fig. 7). Building pixels matched by 0.89 in the confusion matrix for the final model. Concerning high vegetation with a confidence of 0.83 ±0.24, the final model confused it at times with pervious surface. This is not surprising, as they can be very similar in the green and NIR channel. Including edges in the input dataset, improved the confidence in this class by 0.10 ±0.02, and recall from 0.79 to 0.88. As discussed before, the reference model had more difficulties with the sport venues class and misclassified 26 % of pixels as pervious surface. Naturally, these sites can include pervious surface (e.g. football fields), impervious surface (e.g. Tartan track), water (e.g. swimming pools) and buildings (e.g. stadium). For these reasons it is the

most diverse class and difficult to classify. Thus, the finale model reaches an average confidence of 0.71 ±0.28, significantly better than the reference with 0.64 ±0.32. Interestingly, the other way around, the models only rarely misclassify pervious surface as sport venues.

4.2.3 Comparison with other work

For comparison, MARMANIS et al. (2017) achieved with their most comparable network HED+FCN-H-V which is based on an FCN plus VGG weights, an overall accuracy of 0.85. It includes HED method and the ISPRS Vaihingen benchmark and Potsdam benchmark as input (include ground elevation data). Their overall accuracy is 0.05 better than the final model in this work. However, the model in this work only involves 23×10^6 trainable parameters, while their model has 300×10^6 (MARMANIS et al. 2017). On the Virtual KITTI dataset, semantic segmentation was performed with SSDNet-Sem by ZOU et al. (2020). These images include 13 classes plus void. Their complex model includes boundary-detection without depth maps. SSDNet-Sem needed 25×10^6 parameters and achieved an accuracy of 0.76 (ZOU et al. 2020). Therefore, it needed a few more parameters than the final model discussed in this work, as well as reaching 0.04 less in accuracy. In conclusion, the number of classes to predict is next to the architecture and input another crucial part for the success of a model, as the Vaihingen and Potsdam benchmark include 6 classes, this work 8 and the KITTY dataset 14 classes.

To summarise, the visual results seemed to be generally better in the final model. Contrarily, overall accuracy and (weighted) F1 value was 0.02 higher in the reference than the final model. However, recall was better in the final model for the classes building, high vegetation, water and sport venues. These classes involve clear and unclear edges as well as poorly distributed and challenging classes. Furthermore, the final model showed are clearer diagonal in the confusion matrix, thus delivers a higher confidence, which is also reflected in the class probability outputs (Fig 7). This indicates that the additional weights on the edge pixels guide the model's decisions. Unfortunately, if the edges are extracted in a poor quality, the model will be led into wrong directions during its training. In consequence, errors in the edge extracted images are an explanation for overall better accuracy of the reference model compared to the final model.

5 Conclusion and future work

In this work, experiments of edge extractions based on different image channels and combinations of image bands were analysed. Better results were achieved when constructing a closer connection between colour bands such as red, green, blue and NIR, and edge extracted bands within one image. The final model included these mixed bands, as well as dropout layers and depth-wise separable convolution.

To improve the model, more fine tuning and further variation of the architecture could be tested. In this work edges were extracted via the Canny operator. It is a simple and fast method, however with a more intricate approach the result could be improved. MARMANIS et al. (2017) mentioned to include information of class specific boundaries, so that the model not only classifies pixel by pixel but is also aware which classes are separated. This idea would immensely reduce misclassification of single pixels and reduce processing (MARMANIS et al. 2017).

SemCity Toulouse is a good benchmark for pixel-wise semantic segmentation, especially for remote sensing tasks. However, just as the ISPRS Vaihingen and Potsdam benchmark, it needs a higher annotation quality. Nonetheless the final model also showed good results in under-represented classes such as sport venues and water.
Final model results with edge extracted bands reach an accuracy of 0.80 and the reference model without edges 0.82. However, mean values cannot fully show the challenging semantic segmentation task for urban areas. The confusion matrices showed that including edge extraction improved classes with clear edges such as buildings as well as unclear edges such as high vegetation. Additionally, the confidence of the model and the visual results increased with edges. The final model with edges obtained predominantly clearer edge results than the reference. This work showed the impact of edge extracted bands on the model's performance. Finally, investing in high quality edge extracted images as well as combining edges and colour bands within single images will improve semantic segmentation results and is advised to include in future models.

6 References

CANNY, J., 1986: A Computational Approach to Edge Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-8.6, 679-698. https://doi.org/10.1109/TPAMI.1986.4767851.

CHEN, L., BARRON, J. T., PAPANDREOU, G., MURPHY, K. & YUILLE, A. L., 2016: Semantic Image Segmentation with Task-Specific Edge Detection Using CNNs and a Discriminatively Trained Domain Transform. IEEE Conference on Computer Vision and Pattern Recognition, 4545-4554.

CHEN, L., PAPANDREOU, G., KOKKINOS, I., MURPHY, K. & YUILLE, A.L., 2017: DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. IEEE Transactions on Pattern Analysis and Machine Intelligence, **40**(4), 834-848, https://doi.org/10.1109/TPAMI.2017.2699184.

CHINCHOR, N., 1992: MUC-4 evaluation metrics. 4th Conference on Message understanding - MUC4 '92, Association for Computational Linguistics, 22, https://doi.org/10.3115/1072064.1072067.8.

GAL, Y. & GHAHRAMANI, Z., 2016: Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. https://arxiv.org/abs/1506.02142.

HAFFNER, O. & RAVAS, R., 2014: Edge detection in image. https://www.researchgate.net/publication/269108799_EDGE_DETECTION_IN_IMAGE.

HE, K., ZHANG, X., REN, S. & SUN, J., 2015: Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition, 770-778, https://arxiv.org/abs/1512.03385.

KAMPFFMEYER, M., SALBERG, A. & JENSSEN, R., 2016: Semantic Segmentation of Small Objects and Modeling of Uncertainty in Urban Remote Sensing Images Using Deep Convolutional Neural Networks. IEEE Conference on Computer Vision and Pattern Recognition, 1-9, https://doi.org/10.1109/CVPRW.2016.90.

KINGMA, D. P. & BA, J., 2014: Adam: A Method for Stochastic Optimization. 3rd International Conference for Learning Representations, http://arxiv.org/pdf/1412.6980v9.

KRISHNAN, K. B., RANGA, S.P. & GUPTHA, N., 2017: A Survey on Different Edge Detection Techniques for Image Segmentation. Indian Journal of Science and Technology, **10**(4), 1-8, https://doi.org/10.17485/ijst/2017/v10i4/108963.

LIU, X., DENG, Z. & YANG, Y., 2019: Recent progress in semantic image segmentation. Artificial Intelligence Review, 52(2), 1089-1106, https://doi.org/10.1007/s10462-018-9641-3.

LONG, J., SHELHAMER, E. & DARRELL, T., 2015: Fully convolutional networks for semantic segmentation. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 3431-3440, https://doi.org/10.1109/CVPR.2015.7298965.

MA, L., Liu, Y., Zhang, X., Ye, Y., Yin, G. & Johnson, B.A., 2019: Deep learning in remote sensing applications: A meta-analysis and review. ISPRS Journal of Photogrammetry and Remote Sensing, **152**, 166-177, https://doi.org/10.1016/j.isprsjprs.2019.04.015.

MARMANIS, D., SCHINDLER, K., WEGNER, J.D., GALLIANI, S., DATCU, M. & STILLA., U., 2017: Classification With an Edge: Improving Semantic Image Segmentation with Boundary Detection. ISPRS Journal of Photogrammetry and Remote Sensing, **184**, https://doi.org/10.1016/j.isprsjprs.2017.11.009.

NIKHILROXTOMAR, 2021: Semantic-Segmentation-Architecture: vgg19_unet.py. https://github.com/nikhilroxtomar/Semantic-Segmentation-Architecture/blob/main/TensorFlow/vgg19_unet.py, last access 21.08.2021.

RONNEBERGER, O., FISCHER, P. & BROX, T., 2015: U-Net: Convolutional Networks for Biomedical Image Segmentation. International Conference on Medical image computing and computer-assisted intervention, 234-241, https://doi.org/10.1007/978-3-319-24574-4_28.

ROSCHER, R., VOLPI, M., MALLET, C., DREES, L., & WEGNER, J. D., 2020: SemCity Toulouse: A benchmark for building instance segmentation in satellite images. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, **V-5**-2020, 109-116, https://doi.org/10.5194/isprs-annals-V-5-2020-109-2020.

SCIKIT-IMAGE, n.d.: Module: feature — skimage v0.19.0.dev0 docs: canny. https://scikit-image.org/docs/dev/api/skimage.feature.html#skimage.feature.canny, last access 03.09.2021.

SIMONYAN, K. & ZISSERMAN, A., 2014: Very Deep Convolutional Networks for Large-Scale Image Recognition. https://arxiv.org/pdf/1409.1556.

ZOU, N., XIANG, Z., CHEN, Y., CHEN, S. & QIAO C., 2019: Boundary-Aware CNN for Semantic Segmentation. IEEE Access, 7, 114520-114528, https://doi.org/10.1109/ACCESS.2019.2935816.

ZOU, N., XIANG, Z., CHEN, Y., CHEN, S. & QIAO C., 2020: Simultaneous Semantic Segmentation and Depth Completion with Constraint of Boundary. Sensors, **20**(3), 635, https://doi.org/10.3390/s20030635.