# Preprocessing of High-Resolution Satellite Imagery for subsequent Segmentation using Convolutional Neural Networks

Balz Guenat & Lukas Bischofberger
Department of Computer Science, ETH Zurich, Switzerland

***Abstract*—**

## I. Introduction

Image segmentation is a classic computer vision problem. But the reappearance of neural networks also made this problem interesting in new dicipline, the convolutional neural network (CNN). We tackle the problem of road segmentation in images taken from Google Maps with a CNN. Therefore we developed a new pipeline based on CNN, coupled with pre- and postprocessing to solve the challenging task. Our approach makes it's main contribution with a preprocessing and sample generation method.

## II. [Models and Methods

First we preprocess the images to obtain more additional information which is not captured in RGB image data. Section .. explains the different methods we tried. Then we divided the image in patches of different sizes. We tried different parameters (2,4,8,16,32) for the patch size resulting in different coarseness levels. Section .. will explain how specifically an individual patch has been generated. Those patches are packed into batches and fed into the CNN whose configuration is further detailed in section ... Finally we have developed a postprocessing method to combine predictions from different coarseness levels and remove false positives from the final prediction.

### A. Preprocessing

We use different preprocessing methods to extend the input data with additional channels. In a first step we compute the saturation $S$ and lightness $L$ for each pixel $(R, G, B) \in [0, 1]^3$ according to the HSL[1] system as follows.

$$Cmax := max(R, G, B)$$

$$Cmin := min(R, G, B)$$

$$d := Cmax - Cmin$$

$$L := \frac{Cmax + Cmin}{2}$$

$$S := \begin{cases} 0, & \text{if } d = 0 \\ \frac{d}{1-|2L-1|}, & \text{otherwise} \end{cases}$$

[1]Hue, Saturation, Lightness coordinate representation

Figure 1. Original image and manually set labels

We then extend the images by these newly computed channels, such that every pixel is now comprised of 5 values $(R, G, B, S, L) \in [0, 1]^5$. Figure 2 shows the resulting channels for the image shown in figure 1.

In a second step we compute texture features. We present two different methods which we shall call the *GLCM method* and the *MaxDiff method*, only one of which is used in each model. The *GLCM method* utilizes the Gray-Level Co-occurrence Matrix (GLCM) [1] as computed on the lightness channel. Kirthika et al. [2] achieved good results using a very similar method. For each pixel we compute the GLCM for an offset of 1, angles of 0 and 90 and 8 levels. We then use this GLCM to compute the contrast, correlation, energy and homogeneity features of the pixel, yielding 4 additional channels which we append to the pixel. Using this method, the preprocessed images are comprised of 9 channels $(R, G, B, S, L, T1, T2, T3, T4)$. Figure 3 shows the resulting channels. The *MaxDiff method* is a more lightweight alternative to the above. It is also computed on the lightness channel. For each pixel $p$ and its right and lower neighbors $p_r$ and $p_d$, we compute the following contrast channel and append it to the pixel.

$$T := max(|p - p_r|, |p - p_d|) \in [0, 1]$$

Using this method, the preprocessed images are comprised of 6 channels $(R, G, B, S, L, T)$. Figure 4 shows the resulting channel.

### B. Sample generation

1 random generation 2 overlapping patches 3 flip and rotate

Figure 2. Saturation and lightness channels, computed in preprocessing step 1.

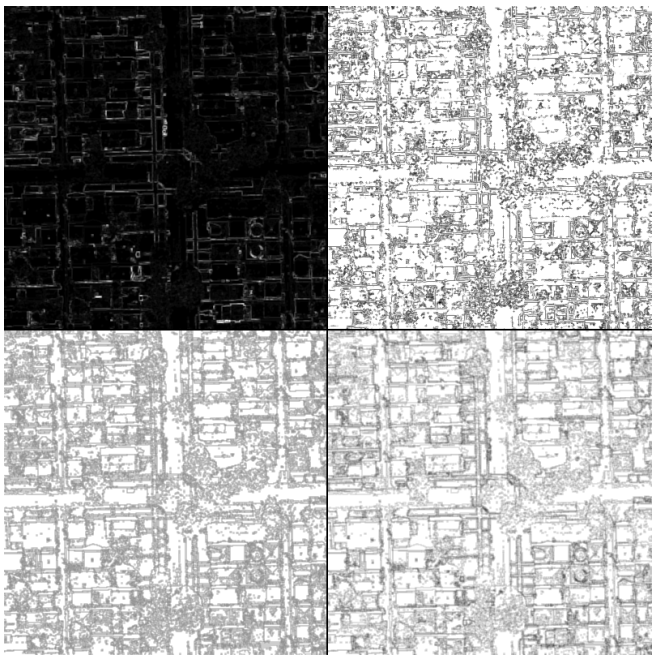

Figure 3. Contrast, correlation, energy and homogeneity channels, computed in preprocessing step 2A.

Figure 4. Contrast channel, computed in preprocessing step 2B.

### C. CNN model

We used a simple convolutional network with three convolutional layers each followed by a RELU and maxpooling. The conv layers are followed by two fully connected layers with depth...

### D. Postprocessing

As stated above we developed different models with changing patch sizes. These make all predictions which have their advantages. For example a CNN with patch sizes 2 can accurately predict small areas, this is specifically useful for removing false positives as e.g. green areas of tress should never be predicted as road. On the other hand with small patch sizes our model is not able to distinguish between a rooftop and a road which are both gray. Therefore we also need the bigger context to avoid these kind of false positives. In the postprecessing we are combining the predictions of models with patch sizes of [2,4,8,16,32] pixels. A final prediction of a pixel is basically classified as road if we obtain a majority vote looking at a pixel in all predictions of different patch sizes.

$p(x,y) = 1if(p2(x,y) + p4(x,y) + p8(x,y)..))/n > thp(x,y) = 0otherwise$

## III. Evaluation

We compare our pipeline to the baseline algorithm. Further we evaluate our preprocessing methods quantitatively and the postprocessing method visually.

### A. Preprocessing

Three cases: 1 w/o data augmentation. 2 with saturation and lumination 3 with edge detection 4 with texture from literature

### B. Baseline comparision

### C. Postprocessing

e.g. image comparision

## IV. Discussion

## V. Summary

## References


[1] M. Hall-Beyer. (2007) The glcm tutorial home page v2.10. [Online]. Available: http://www.fp.ucalgary.ca/mhallbey/tutorial.htm

[2] A. Kirthika and A. Mookambiga, "Automated road network extraction using artificial neural network," in *Recent Trends in Information Technology (ICRTIT), 2011 International Conference on*. IEEE, 2011, pp. 1061–1065.