Road Segmentation using Neural Networks

CS-403 - Machine Learning - 7 credits

Clélie de Witasse, Léopold Bouraux, David Sollander
Department of Computer Science, EPFL, Switzerland

***Abstract*—This report present the methods that have been implemented to perform road segmentation on Google Maps images. The first method uses a logistic regression model. The results are quite unsatisfying with a maximum F1-score of 79.2. The second method uses a U-Net algorithm and is much more consistent for such a task. The maximum F1-score is 88.4. Even though neural nets are widely used in image processing, it still requires well tuned preprocessing and postprocessing steps to get satisfying results. A cross validation is also useful to reduce the variance of the trained model. The method that gave the best results was to run several time the model with slightly different training sets, get a prediction for each model and compute the mean of each temporary predictions to get the final results.**

I. Introduction

Image segmentation is an image processing task that is very studied in computational sciences, and consists of partitioning an input image in several segments that carries meaningful different information. The use of neural nets has brought a new approach to solve this task very accurately. This project present different ways to segment satellite images of roads, including a classic approach and neural nets approaches. The classic approach uses logistic regression, which gives quite inadequate results, especially because of the fact that roads can be obstructed by streets elements, like trees, vehicles and the shadows of the buildings. Moreover, some structures like, sidewalks, rails roads, roof tops and parking lots looks like road and thus might be misclassified. Neural nets are more efficient to deal with such issues.

II. Problem formulation

Considering a set of satellite images and its associated groundtruth images I and G. We set $G(i, j) = 1$ if in the set I the corresponding pixel at location (i, j) is a road. We set $G(i, j) = 0$ otherwise. The main goal of the project is thus to find $\mathbb{P}[G(i, j)|I]$.

In order to obtain a good prediction measurement, we use the F1-score instead of classic accuracy. It can be computed as follow :

$$2 \cdot \frac{precision \cdot recall}{precision + recall}$$

The F1-score takes a value between $[0, 1]$. The closer the score is to 1 the more accurate the prediction is.

III. Material and Methods

A. Data analysis

The provided dataset is made of 100 RGB satellite image of streets of size 400x400 pixels. For each image, the corresponding ground truth binary image is given, where white pixels represent the streets to be segmented. The Fig 1 shows a satellite image with its associated groundtruth. The

(a) Satellite image (b) Ground truth image

Fig. 1. Road image and its ground truth to be predicted

test dataset consists of 50 satellite images of 608 pixels by 608 pixels. These images are provided without the associated groundtruths, which we have to predict.

B. Data augmentation

One crucial step while trying to segment images is to preprocess them well. This step will influence the computation time and the accuracy of the results. First, we noticed that roads which are skewed in the images were more often badly segmented than horizontal and vertical roads. This can be explained by the fact that in the provided training dataset, the number of skewed roads is much less important than the number of vertical and horizontal roads. In this way, our model trains less on these skewed roads. In order to handle this problem, the dataset has been augmented a first time by rotating each picture by $45°$. Thus all routes that are initially horizontally or vertically oriented are skewed and we get an equal proportion of both types of roads. Then, during the U-Net training, a data generator create even more pictures by rotating , flipping, shifting, zooming and applying a small shear transformation to each image in a non deterministic way. This second data augmentation processing is applied to images as well as their groundtruths. It is this data, twice pre-processed, which we use to create training and validation sets.

On top of that, as mentioned earlier, we have 400x400 px images at the start. We realized that by compressing these images to 256x256 px, the computing time was improved approximately 2.5 times. Although we lost in image resolution, the results were only slightly less accurate. We therefore

preferred to optimize other parameters rather than having a too long computation time. It should be noticed that the number of 256 pixels is not chosen arbitrarily: the width of images must be a large enough multiple of 2 in order to make U-Net possible. [See III-C2 U-Net model section]

C. Learning the models

Two different models have been trained to compare their accuracies. The first one is a logistic regression which is a rather classical model. The second one is a more advanced model called U-Net algorithm.

1) Logistic Regression model: Logistic regression is a quite straightforward model for classifying images. Thus, a good preprocessing of the data is crucial in order to improve the prediction. The first thing was to obtain the 3 most important features for image segmentation : the average color, the standard deviation between pixels and their variation. We've then preprocess those features using the polynomial feature function from the `scikit-learn` library [1]. This function enabled us to generate a new feature matrix consisting of all polynomial combinations of the features with degree less than or equal to the one chosen. We've observed that best results where obtained with a degree of 4. The feature is then used to train a logistic regression, also from the `scikit-learn` library [1]. A cross-validation using a $\mathbf{k_{fold}} = 4$ has been implemented to enhance our overall accuracy.

2) U-Net model: Convolutional networks for classification tasks usually have an output towards a single class label. However in road segmentation, the output wishes to assign to each pixel a class label different from the one of the whole image. To overcome this problem, A.Giusti et al. [2] trained a network in a sliding-window setup to predict the class label of each pixel by providing a patch around each pixel as input. This allows two things: to process the image pixel by pixel and to have much more data to practice on. However, this strategy has two drawbacks. On the one hand, there is a lot of repetition due to the overlap of patches. On the other hand, one can observe that a large patching size increases max-pooling thus reducing localization accuracy, whereas a small patching size implies that the network will consider only little context. Hence, a trade-off between localization accuracy and pixels neighborhood must be chosen.

Later approaches have suggested a classifier output which considers the characteristics of several layers. O.Ronneberge et al. [3] from Freiburg University have designed a Fully Convolutional Network (FCN) for biomedical image segmentation where some improvements were made to better segment images. The key insight of FCN is to create successive layers where pooling operators are replaced by upsampling operators. This way, these layers improve resulting resolution of images. This structure can be trained with a rather small training dataset[3].

The network architecture is illustrated in Figure 2. It is made of a contracting path, that we can observe on the left side

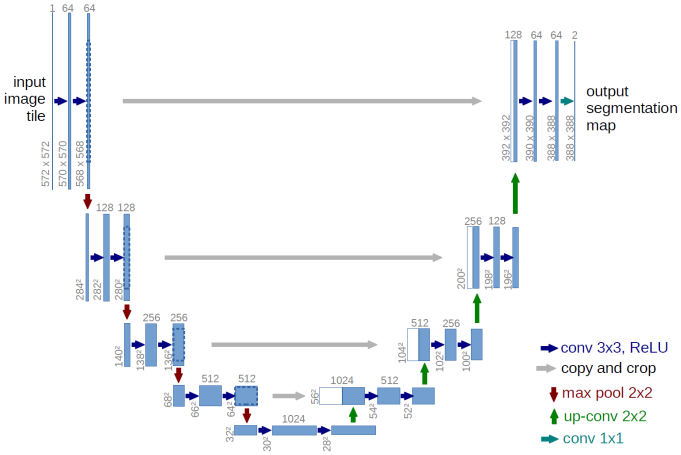


Fig. 2. U-Net architecture[3]. On the top of each layer is given the number of channels. The size of the image is shown on the left of each layer.

and an expansive path on the right side. The contracting path follows the architecture of a classical convolutional network: repeat application of two 3×3 convolutions followed by a ReLU function and a 2×2 max pooling operation with stride 2 for downsampling in which the feature channels number is doubled.

Looking at the expansive path, each step involve an up-sampling of the feature map followed by a equal division of the number of feature channel using a 2x2 convolution. The next step is a concatenation with the correspondingly cropped feature map from the contracting path, and the last step consist of two 3x3 convolutions, followed by a ReLU function. Due to losses of border pixels in each convolution, the data crop is highly important. Finally, to map all the feature vector components to the expected number of classes, a 1×1 convolution is used on the last layer. This brings us to a total of 23 convolutional layers in our network.

Our own U-Net has been built using the `keras` library [4][5] and has been inspired by [3]. It trains approximately 31 million parameters[6]. A 5-fold cross-validation has been implemented in this algorithm to get a smaller variance on the trained model. We have observed that the roads are easily predicted when they are horizontal or vertical, but skewed and curved roads are less accurately predicted. As mentioned earlier, in order to handle this issue, images for training and validation set are predicted two times in different situations during the cross validation:

- In the first situation, images are rotated by a random angle $\alpha_1 < 5°$ before fitting the model, to handle with vertical and horizontal roads.
- In the second situation, images are rotated by a random angle $\alpha_2 < 90°$ to handle with skewed roads.

On top of that,

In both situations, the images are randomly flipped horizontally and several others processing transformations such as shearing, zoom and shift have been applied. Those two

situations are implemented in each iteration of the cross validation. To get our final predictions, many approaches have been tested. First, it has been noticed that the models, once trained, could sometimes give mixed results with either a high or a low accuracy. We've then decided to train a model for each parameter sets of each 5 k_folds, and predict test images for these models. As we used a 5-fold validation each time with 2 different preprocessed data, we obtain $5 \times 2 = 10$ images. From these 10 images, we computed the average image and it gave us much more successful outcomes. We tried as well to compute the median image or a weighted mean over predictions, but the simple average showed the better results.

D. Evaluation metric

To evaluate models for segmentation task, the accuracy might not be the better choice since for some images, the proportion of roads is rather high, thus a model that classify the entire image as a road would get a good accuracy. One robust evaluation metric for such task is F1-score, which is to be maximized when the task is well performed.

E. Submission

In order to submit the results on AICrowd, the predictions must be patched with 16x16 patches. Thus, the resulting average predictions are patched such as each patch is assigned to a binary value depending on its features: if more than 20% of its pixels are labeled as a road, the patch itself is labeled as a road. A csv file is created for the submission. However, the visual result is much more satisfying on unpatched predictions. This patching step is to be considered only for the challenging part but the considered results and discussions will focus on the real predictions.

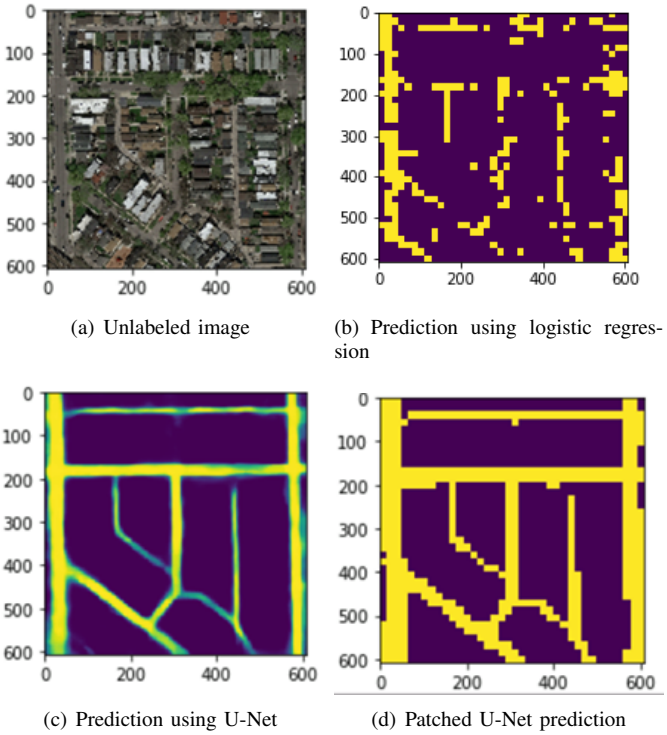
IV. Results

As expected, the logistic regression provides a rather small accuracy. Neural nets methods are much more satisfying. Accuracies, F1-scores and computation times are summed up in Tab I.

Segmentation method	F1-score	Computation time
Logistic regression	79.2	193.08 s
U-Net (weighted avg on predictions)	87.9	7.5h
U-Net (median on predictions)	88.2	7.5h
U-Net (average on predictions)	88.4	7.5h

TABLE I
CHARACTERISTICS OF THE TESTED METHODS

On the Fig 3, one can see a satellite image, its prediction using logistic regression, U-Net and the patched U-Net prediction. As a intermediary result, the F1 score and accuracy of the models with $\alpha = 5°$ are better than $\alpha = 90°$. The evolution of the F1-score and the accuracy has also been plotted as a function of the epoch and the rotation range, the results is visible on Fig 4 and 5. The F1 value tends to stabilize around a final value, thus the number of epochs used to train the model is high enough. The shape of the curves are similar for the two rotation ranges. This result is consistent with the dataset augmentation, since all



(a) Unlabeled image

(b) Prediction using logistic regression

(c) Prediction using U-Net

(d) Patched U-Net prediction

Fig. 3. Predictions of a test image

situations should have a similar amount of horizontal/vertical and skewed roads. Eventually, if a lot of instabilities are visible at the beginning of the training, they tend to disappear as the number of epochs increased. Nevertheless, models with a maximum rotation of 90° vary more, even at the end of the training. This is due to the greater difference in image rotation. The same image rotated randomly from 0 to 5 degrees is more likely to be trained several times compared to an image rotated from 0 to 90 degrees.

Concerning the different U-Net predictions, the best f1-score is obtained by averaging all predictions without weights. Computing the median is almost as good as the average.

V. Discussion

A. Logistic regression

Regarding the logistic regression, one can observe that the prediction, although crude compared to the one obtained with the U-Net model, gives an acceptable general idea of the roads position on the satellite images. The time spent versus quality ratio is fair enough to consider this model as a fast way to segment images. To increase the general accuracy of the model, a further improvement could consist on a more intense preprocessing of the initial data set. The use of a histogram equalization filter would increase the contrast thus improving the intensity distribution. Moreover, noise could be removed using adaptive median filters and variance could be reduced

using average filters. However, one should be careful on the use of those filters which could lead to blurring of an image and may affect features localization.

B. U-Net

1) Resizing the images: Data augmentation is crucial to have a big enough training dataset, but a huge training dataset made of images with low contrast or low quality could lead to unsatisfying results. Resizing the input images was crucial to have a short computation time. Indeed, a U-Net tuned for 400×400 images has been tested but one loop on the cross-validation took around 3 times longer. However, the information due to the resizing was in part compensated by the upsampling operators from the algorithm. One big improvement could be to tune again the algorithm to have the right convolution layers sizes, and take the time to train the model.

2) Preprocess the image using image processing filters: Another way to preprocess the dataset consists on improving the features of the images. Again, methods that could improve the preprocessing for logistic regression can be applied for the U-Net model (average and median filters). Changing the images saturation may also improve the general accuracy of the model by increasing the gap between roads and background features.

3) Windowing the images:

4) Improving the training dataset: Parking lots and small roads leading to them were a source of errors. Those paths were sometimes considered as road in the ground truth images from the training data set creating then some inconsistencies. Managing by hand these inconsistencies in the preprocessing may had improve our accuracy.

5) Patching the predictions: Patched predictions provides a smaller F1-score than unpatched predictions. Reduce the patch size to 4x4 or 8x8 pixels might reduce the difference, but would also require a longer computation time. This wasn't possible for the AI-Crowd challenge, but it could be interesting to go further.

VI. Summary

An image segmentation task has been realized with two different machine learning methods. Logistic regression is easy to implement and doesn't require a big computation time, but results are quite unsatisfying if the images are complex (in our case lots of artifacts such as cars or trees are present). The U-Net algorithm is an efficient neural network to segment complex images, and the results obtained by a 5-fold cross-validation gave a good accuracy. The best F1-score has been obtained by averaging the predictions of 10 times the same model fitted with different training sets.

References

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[2] A. Giusti, D. C. Ciresan, L. M. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," 2012.

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, ser. LNCS, vol. 9351. Springer, 2015, pp. 234–241, (available on arXiv:1505.04597 [cs.CV]). [Online]. Available: http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a

[4] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[5] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[6] "Implementation of deep learning framework – unet, using Keras," 2017, available on. [Online]. Available: https://github.com/zhixuhao/unet

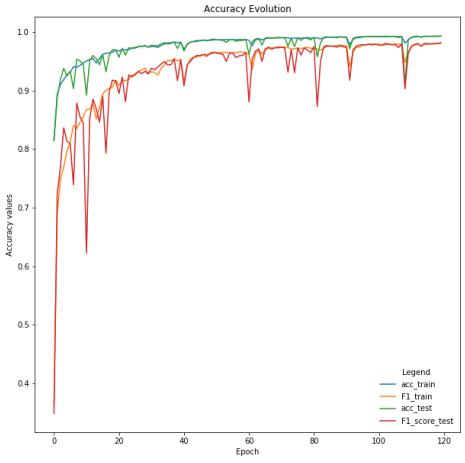


Fig. 4. Accuracy and F1 evolution during training - Rotation max : 5°

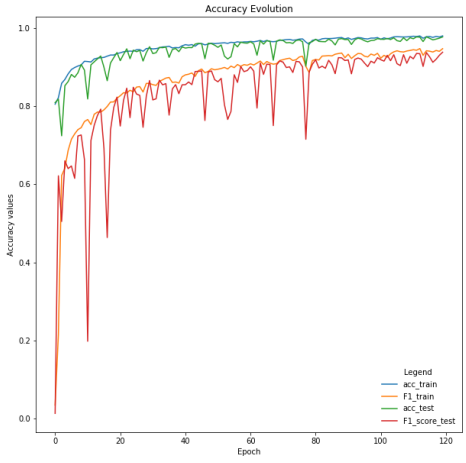


Fig. 5. Accuracy and F1 evolution during training - Rotation max : 90°