Road Segmentation challenge

Niccolò Sacchi & Valentin Nigolian

Group: RoadSegmentationFault
École polytechnique fédérale de Lausanne

December 18, 2017

***Abstract*—The goal of this project is to segment satellite images in order to discriminate the roads from the background. Several approaches have been tried in the past, most of them required a lot of preprocessing and did not gained great performances. Invented in the late 20th century, the convolutional neural networks (CNNs) only recently were successfully applied to images classification and segmentation problems and performed considerably better than other techniques. The model we used in this project consists of a CNN whose input can be an image of any size and the output is the respective mask of roads (ground-truth). Our model makes use of on-the-fly data augmentation, Dropout layers, LeakyReLU activation functions to reduce overfitting and improve the training. Moreover, we adopted two simple but effective techniques to improve the predictions. We will also show that, according to our tests, our model outperforms the considered baselines.**

I. Introduction

The segmentation problem can be defined as the problem of assigning a class label to a set of pixels so to classify part(s) of an image. The goal of this project is to segment the roads in satellite images so to extract the mask of the roads. The metric used to evaluate the prediction of the roads is the F1 score that combines the precision, i.e. the fraction of correctly predicted roads among all the predicted roads, and the recall, i.e. the fraction of correctly predicted roads among all the true roads. The formula of the F1 score is the following:

$$F1 = 2\frac{pr}{p+r}$$

$$\text{where} \quad p = \frac{TP}{TP+FP}, \quad r = \frac{TP}{TP+FN}$$

Only in recent years, thanks the increase in computing performance and the possibility to exploit GPUs to parallelize computation, CNNs were able outperform most of the techniques in solving different tasks, e.g. image classification, image segmentation, natural language processing, handwriting generation, automatic game playing and many others. The strengths of the CNNs derives from a hierarchy of filters which are learnt during the training phase. Thanks to these filters, the CNNs are capable of producing more and more complex features, i.e. high level representation of the input, as the image goes deeper in the network. However, a drawback of CNNs is the need of a large, fully-labeled dataset. This drawback can be compensated with data augmentation which has proven to be very effective on images. In the following sections we will give an overview of the techniques we tried focusing in particular to the CNN we implemented and tuned. Section II explains ..., Section III ..., Section IV ..., Section V ..., Section VI ...

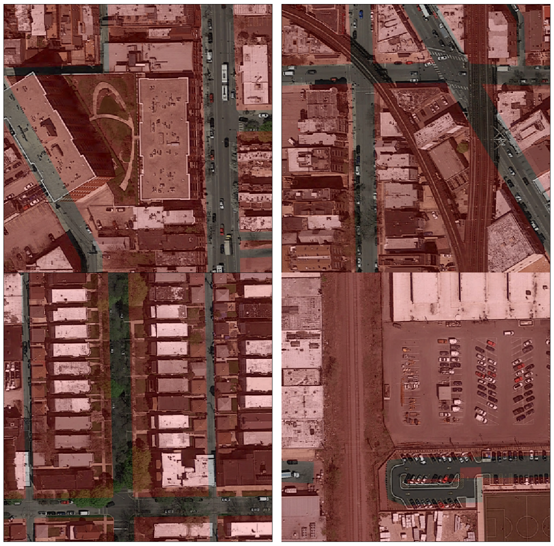

Figure 1. Corner cases in which roads are covered by (a) buildings, (b) railroads, (d) trees and also show the ambiguity of parking lots in (e)

II. Exploratory data analysis

We are given two sets of images acquired from GoogleMaps: (1) a train set composed by RGB images of size $400*400$ pixels and the respective ground-truth images, i.e. images of the same size where white pixels represent roads and black pixels represent the background, (2) a test set composed by RGB images $608 * 608$ pixels whose ground-truth has to be produced. The aim of the project is to label patches of $16*16$ pixels, therefore, depending on the methodology used, pixels of the provided ground-truth may have to be grouped and averaged so to provide the class label for that patch. We adopted a threshold of 0.25, if the average of the pixels in a patch was above the threshold then that patch was labeled as road, otherwise it was labeled as background. By taking a look at the train set and its masks we could notice the problem had its non-trivialities, e.g. ambiguity of parking lots, roofs of the same color of the roads and trees, shadows, buildings, railroads and cars covering the surface of the roads, Figure 1. Those problems can be solved only if we consider also the neighborhood of each patch to correctly classify it. That task can be achieved with feature extraction, either manually, with complex and advanced techniques, or automatically by a CNN.

III. Models and Methods

The first techniques we studied involved feature extraction using interesting, still complex and not very efficient techniques that fails in more extreme cases, e.g. when there are object obstructing the view of the road or there is a change of brightness through different images. Such techniques involved for example computation of the gradient at each pixel (so to exploit the fact that roads have continuity in one direction, i.e. a gradient ~0), the use of an edge detection algorithm to extract the boundaries of the objects and the use of vector graphics to represent images with geometrical primitives such as points, lines, curves, and shapes or polygons [1]. Given the complexity of those solutions and poorness of the results when compared to the CNNs, we opted for the latter which are well know for the little preprocessing needed and great efficiency in solving these kind of tasks. We started from three very simple baselines: (1) a model that predicts all as road (notice that, since the metric chosen is the F1 score computed on the road predictions, a model that predicts all as background would have higher accuracy but an F1 score of zero), (2-3) we considered each $16 * 16$ patch as an input and for each one of them we produced 6 simple features extracted from the mean and the standard deviation of each RGB channel in the patch. The obtained dataset has been used to train a logistic regression and a random forest. The F1 scores of these three baselines is shown in Figure 2. As expected, this analysis shows that those methods fails in this complex task if more advanced features extraction is not in place. Therefore, we started searching and studying solutions based on CNNs. We found two main approaches: (1) design a CNN which segment the input image and outputs directly the mask for the roads [2], (2) reduce the segmentation problem to a classification problem, i.e. build a CNN that given as input a single patch, possibly with a bigger context (neighborhood of that patch), labels that patch as either road or background [3]. We tried both the techniques but then opted and focused on (1) since it required almost no preprocessing and showed better results.

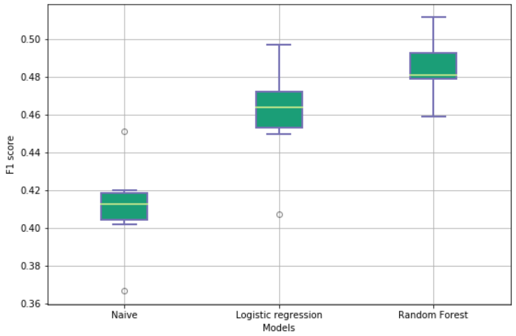

Figure 2. Distribution of the F1 scores computed on a 6-fold cross validation on the three considered baselines.

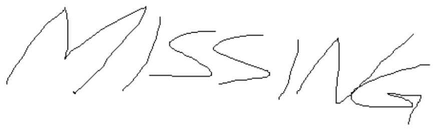

Figure 3. Scheme of the final model used to obtain an F1 score of ...

A. CNN design

Given the differences in size between train images and test images, we decided to build a CNN which could take as input an image of any size and output the estimated probability of being a road for each patch of that image. To achieve such a CNN we did not used any dense layer, since it requires an input of fixed size, and designed it such that it reduced each side of the input image by a factor equal to the width of the patch we wanted to predict so to obtain an output mask of size $H/n * W/n$ where n is the width of the patch. We tried both predicting $8 * 8$ patch-wise and $16 * 16$ patch-wise and observed better results when predicting $8 * 8$ patch-wise. The structure of our CNN is depicted in Figure 3. Since we experienced problems of GPU memory filling, we could only adopt convolutional layers of higher number

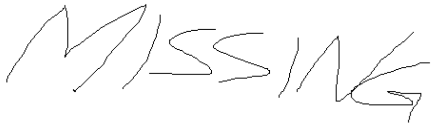

Figure 4. Activations of the first convolutional layer: with ReLU on the left and with Leaky ReLU on the right. As can be noticed the number of dead filter decreased sensibly meaning the model was learning more reasonable features.

of filters after reducing the size of the input image, i.e. the first layers have bigger inputs and less filters while the last layers have smaller inputs but can have more filters [1].

The most common activation function used after convolutional layer is the ReLU. However, since we have experienced the problem of dead filters [2], we have switched to the Leaky ReLU, i.e. a variant of the ReLU which has a small positive gradient for negative inputs:

$$f(x) = max(\alpha x, x)$$

Where α is the slope for negative inputs and is usually a small value ~0.01. This choice solved the problem of dead filters as can be seen in Figure 4.

We used k max pooling layers with a pool size of $2 * 2$ where 2^k is the size of the patch we wanted to predict, e.g. to predict $8*8$ patch-wise we adopted 3 max pooling layers so to reduce the size of the input image by a factor of 8 on each side. Note that increasing the strides of the convolutional layer would lead to the same size reduction but we opted for max pooling so to automatically select the higher activations.

Moreover, after each max pooling layer we added a dropout layer, i.e. a regularization technique used to prevent overfitting by randomly setting to 0 a fraction of the input during the training phase. We have set the dropping rate for all the dropout layers to 0.25.

B. Data augmentation

CNNs need a big training set in order to recognize the patterns proper of the roads otherwise it could overfit and just memorize the training images. Increasing the training set is a must and could greatly increase the performance of the model while reducing overfitting. Our model builds the batches of images on-the-fly applying rotation and/or mirroring to the original image. While each image of the batch is mirrored with a probability of 0.5 we decided not to rotate each image by a random degree since most of the images, both in the test set and train set, have vertical and horizontal roads and we wanted our model to master the recognition of those roads. Rotation is done in the following way: an image of the batch is rotated by any degree with a probability of 0.2 while is rotated by $90° * k$ where $k = 1, 2, 3, 4$ with a probability of 0.8. Figure 5 shows an examples of data augmentation applied on an image. This data augmentation proved to be effective and our model learnt not only to recognize horizontal and vertical roads but also diagonal roads.

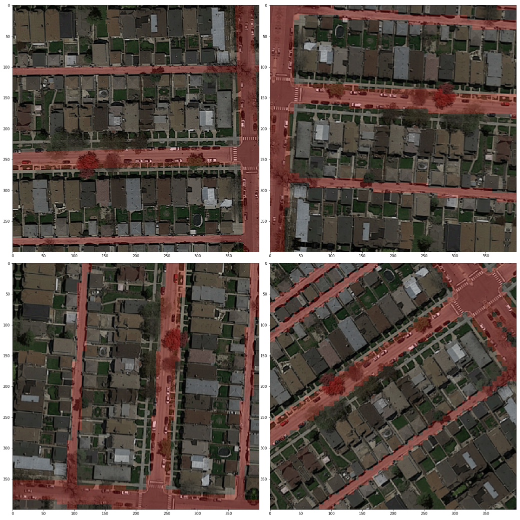

Figure 5. At the top-left the original image, the other are example of augmentation of that image. After rotating the image, the points outside the boundaries are filled by reflecting the image so to always obtain an image of the same size ($400 * 400$).

C. Cross validation

In order to compare different CNNs models we used 4-fold cross validation. We couldn't use a greater number of folds as the training may already take a lot of time.

D. Post-processing

Once the model have been trained, its the predictions can be further improved in order to increase the F1 score. However, even if roads have some intuitive properties, e.g. starting from a road you can always reach the border the image, it is not trivial to improve a predicted mask of roads such that those properties are fulfilled. Therefore, we opted for two easier solutions that proved to increase the F1 score by a couple of percentage points.

- We compute on the on the train images and its predictions the optimal threshold so that each patch whose probability is beyond that threshold is labeled as road. This step is simply done with a grid search and retrieving the threshold which corresponds to the maximum F1 score. This estimate optima threshold will be used on the predictions of the test set to label each patch.
- To obtain the predictions of a test image, we mirror it and/or rotate it by $90° * k$ where $k = 1, 2, 3, 4$ so to obtain 8 different predictions. Those predictions are then averaged to obtain the final prediction.

IV. Results

V. Discussion

VI. Summary

References

[1] J. Hormese and C. Saravanan, "Automated road extraction from high resolution satellite images," *Procedia Technology*,

[1] The output of a convolutional layer, when strides is $1 * 1$ and padding is used, is $H * W * \#filters$ where $H * W$ is the size of the input.

[2] Dead filters are filters that always output zero for any input. This may happen when, during training, the weights are updated in such a way that the neuron will never activate on any input again which lead to a gradient of zero and no update of the weights.

vol. 24, no. Supplement C, pp. 1460 – 1467, 2016, international Conference on Emerging Trends in Engineering, Science and Technology (ICETEST - 2015). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2212017316302705

[2] K. Lis and I. Roesch, “Road extraction from aerial images,” 2016.

[3] D. Pavllo, M. Martinelliy, and C. Hwang, “Epfl machine learning project 2 report road segmentation,” 2016, https://github.com/dariopavllo/road-segmentationl.