CS-433 Machine Learning - Project 2 - Road Segmentation

Justin Deschenaux, Guillaume Follonier, Guillaume Barre
Department of Computer Science, EPFL Lausanne, Switzerland

***Abstract*—For the second project of the Machine Learning class (CS-433), our group decided to undertake the road segmentation challenge. We were provided a set of 100 satellite images of roads with a corresponding mask indicating the ground-truth class of each pixel. The objective was to build a classifier that would segment images correctly. Hence, we investigated the literature in order to discover which architectures were the most widely used, to obtain a powerful model for the task at hand.**

I. Introduction

Because of their essential generality, neural networks are often a choice of preference when it comes to machine learning. In the recent years, research has been flourishing to develop new architectures to solve computer vision problems, and convolutional neural networks (CNN) emerged as the building blocks of many sophisticated solutions. In particular, when it comes to image segmentation, the U-net [1] was a major leap forward. Since then, many improvements were proposed and we were very excited about studying them. Therefore, we used this architecture as a starting point in our journey. We began by reading about skip connections, used in the U-net to connect the descending path and the opposite ascending path. Later, after reading [2] about dense skip connections, we tried to replace some convolutional blocks of the U-net by densely connected blocks. An other general mechanism getting a lot of traction recently is the idea of *attention*. Originally proposed in [3], it fuelled a significant number of breakthroughs such as GPT and BERT for natural language processing or the very recent AlphaFold proposed by DeepMind in 2020 that succeeded to predict the folding of proteins, many years before domain experts were expecting it. Therefore, we tried to use attention mechanisms to improve the performance of the original U-net. We implemented these neural networks using PyTorch [4].

II. Models and Methods

A. The models

Fig. 1 represents the general structure of the U-net. It shows the descending path that tries to extract meaningful features to represent the input and the ascending path, whose objective is to construct a pixel-wise mask (road/background). In a nutshell, the differences between the different models are located in the blue blocks and orange attention mechanisms. The salmon arrows in the middle are called ”skip connections”. They are commonly used in deep architectures as a solution to the vanishing gradient problem. They offer an alternative path to propagate the derivatives from the output to the early layers. In the case of the U-net, they also provide the reconstruction path a direct access to the features extracted in the first half. It is commonly accepted (see [5]) that the filters on the primary layers are tuned to recognize low-level details (spatial location, corner shapes, edges) while the deeper ones focus on more abstract concepts (with less precise position). Therefore, as we want the model to detect accurately the position of the roads, these shortcuts perfectly make sense, as they allow to forward the details of each hierarchical level to the adjunct part. As one could guess from the shape of the blue blocks on Fig. 1, each time we use a max-pooling operation to sub-sample the image by two, we double the number of channels on the subsequent stage. Conversely, at the transpose convolutions (akin to an interpolation with parameters, hence doubles the size of the image), we reduce the number of channels by two. Ultimately, the output of the last (blue) block is combined by a 1x1 convolution into a mask of the same width and height as the input. The CNN expects input images of shape 256x256x3, i.e. squares with 3 color channels and outputs a 256x256x1 gray scale mask. We chose this scale as a good trade-off. Indeed, if the input is too large, it

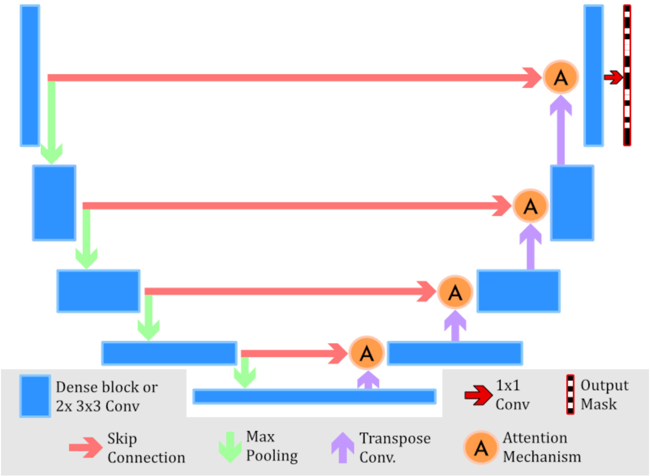


Figure 1: Blue blocks can be dense blocks or two sequential 3x3 convolutions. Orange circles represent the optional attention mechanism whose output is fed to the neared block. Salmon arrows represent the skip connections, forwarding features extracted before.

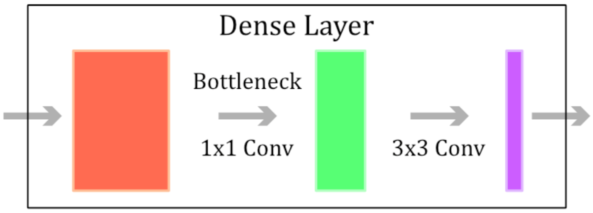


Figure 2: Representation of a dense layer. These are stacked together to form densely connected blocks. For examples, we used 64 such layers in the bottom section of the Dense U-net.

is infeasible to train the model properly and in case the patches are too small, the CNN could be fed with road or background pixels only, which seems a bad idea in order to build a general classifier. In order to segment pictures with a different size, we simply split them into patches of the matching aspect, and combine the individual predictions to form the global mask. Each convolution layer, except the last one and some inside of the attention mechanisms, is followed by a batch normalization (BN, [6]) and a ReLU activation. For the implementation details such as how many channels we used for each stage, please refer to the git repo, where the README file.

U-net: The first model, presented in [1], uses two consecutive convolution layers at the place of the blue blocks. It did not implement an attention mechanism ([3] was proposed two years after the publication of [1]). This architecture was created to solve medical image segmentation problems. Its ingenious structure gave birth to a very rich literature in computer vision. For instance, it was studied for human pose estimation, e.g. for sign language recognition [7].

Dense Blocks: Introduced in [2], densely connected networks (DenseNet) were proposed to improve the performance of deep architecture by shortening the distance between the output and the early sections. Its main component is the dense layer, represented in Fig. 2. Those building parts are stacked together to form a dense block and the output of all previous layers is used as input of the next one. It was shown in the original work, that such construction can improve the capabilities of CNNs by enforcing feature reuse across the network. The ”knowledge” discovered in early layers is not available their direct successor only, but rather to all the followings. However, implementing such an architecture ”as-is” would cause a lot of overhead. Indeed the convolution of the U-net are performed with 3x3 filters. Therefore, for computational efficiency, there is a bottleneck 1x1 convolution that reduces the dimension before feeding the channels to the 3x3 layer. It seemed more meaningful to use dense blocks in the descending path only since this is the section of the CNN that ”analyses” the image, whereas the second half tries to use the results to infer a mask. It is consistent with the fact that [2] uses the construct for image classification tasks rather than ”recovering” task, i.e. to extract discriminating features.

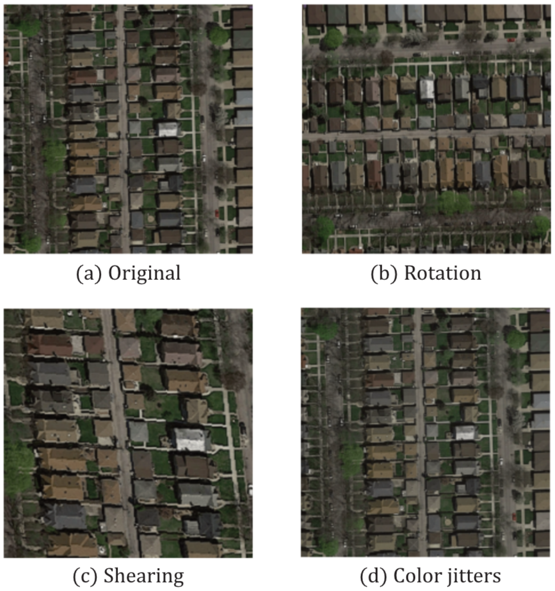

(a) Original
(b) Rotation
(c) Shearing
(d) Color jitters

Figure 3: Data transformations used. For ”color jitters”, we randomly modifies the brightness, contrast, saturation and hue to create a different image. However, we ensured it still looks like a plausible image.

Attention mechanisms: We used two such methods. Namely, the results proposed in [8], referred to as channel attention and [9], referred to as grid attention. To make a long story short [8] uses a three layers fully connected network (FCN) (with a bottleneck in the middle) in order to weight the input sections with respect to their importance. For that purpose, each channels is reduced to a single value using adaptive max pooling. The resulting vector is then fed into the FCN. Finally, the output weights multiply the input channels accordingly. On the other hand, [9] proposes to create a coefficient for each pixel (like a grid) shared between all channels. However, the procedure is a bit technical and harder to explain in a few words. Therefore, we chose not to reproduce the statement of [9] to avoid making this report unnecessarily long.

B. Data processing

The data set: The provided data set contains 100 aerial images and their associated ground truth. Those are square images of 400x400 pixels. The ground truth images have the same shape but are on gray scale. White pixel correspond to roads, and black background. We were also provided 50 unlabelled aerial images (RGB channels). Note that those are larger than the training images (608x608 pixels).

Data augmentation: It is not enough to train the neural network on the provided images directly if we want the model to be robust (e.g. translation, rotation, brightness invariant). For that reason, we preprocessed images to obtain an augmented data set. First, we applied linear transformations with random parameters to the original images, namely shears, rotations, brightness, contrast, saturation and hue change. Note that some modifications would create some blank margins in the output. As the pictures are large enough, we simply cropped them to yield our final training examples. Fig. 3 shows what these modifications look like. Finally, the data preparation step also ensured that all images had a shape matching the expectations of the network.

III. Results

A. Comparing losses

A widely used loss when training CNNs for image segmentation is the (binary) cross-entropy (BCE). We found that a different loss was also in use sometimes. This is the Dice loss, which is based on the Sørensen–Dice coefficient $D(p, g)$ defined as:

$$D(p, g) := \frac{2 \times \sum_i p_i g_i}{\sum_i p_i^2 + \sum_i g_i^2}$$

where p and g are sets to compare. In our case, they represent pixels of the prediction and ground-truth. The Dice loss is then $\ell_{Dice} := 1 - D(p, g)$. Moreover, we found that [10] used it to improve the performance of edge detection in images. Their point was that when the classes (edge/background) are unbalanced, a combination of the two losses can improve performance. Therefore, we tried to use the result for our case. The final loss is defined as $\ell(p, g) := \alpha\ell_{Dice}(p, g) + \beta\ell_{BCE}(p, g)$. For this experiment, the augmented dataset was obtained by first splitting the labelled data into a training set (60%), a validation set (20%) and a test set (20%). Data augmentation was then applied independently over the three sections. The best parameters found in [10] where $\alpha = 1, \beta = 0.001$. Note that β is smaller than α because the two losses have different ranges. The dice loss is defined on $[0, 1]$ while the cross-entropy is in $[0, 100]$ since PyTorch clamps the logs to avoid infinite values during backpropagation. For computational efficiency, we compared the losses on the simplest version of the U-net (no dense block, no attention), although we know that different losses may behave differently depending on the architecture.

F1-score of different losses		
α (Dice)	β (BCE)	F1-score
1	0.0001	0.844
1	0.001	0.835
1	0.01	0.855
1	0.1	0.854
0	**1**	**0.856**
1	0	0.826

At first glance, we were surpised by the results. Intuitively, aerial images *should* form an unbalanced class. However, our training images have on average 45% of road pixels. Therefore, our setup is different from [10]. Given the reasons why the Dice loss was used, the above table makes perfectly sense. Hopefully, we did not blindly used this loss.

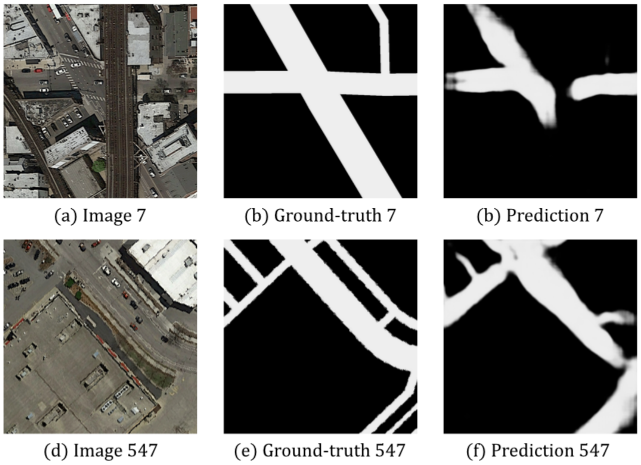

(a) Image 7 (b) Ground-truth 7 (b) Prediction 7
(d) Image 547 (e) Ground-truth 547 (f) Prediction 547

Figure 4: Prediction of the best dense U-net on hard images.

B. Comparing architectures

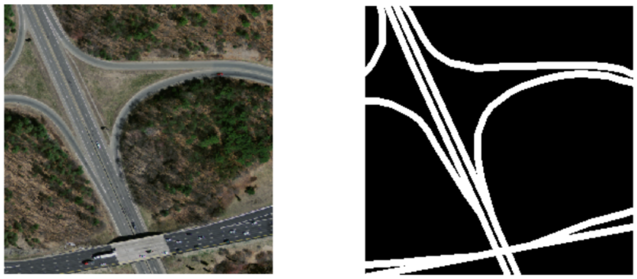
Once we found a good loss on the base model, we compared the different versions of the U-net to find the most appropriate. Our findings are summarized in the following table. We used the same data set as in the loss experiment here. The table contains the result on the test set.

F1-score of each architecture	
Classic U-net	0.851
U-net w/ channel att.	0.843
U-net w/ grid att.	0.845
Dense U-net	0.853
Dense U-net w/ channel att.	0.86
Dense U-net w/ grid att.	**0.863**

It seems like the attention mechanisms we used do not work well with the base U-net. However, for the dense version, we obtained a 1% performance increase in the F1-score by adding the grid attention mechanism. Nonetheless, one must be careful not to over-interpret those results. The difference is not significant enough to draw definitive conclusions on whether or not the dense architecture is strictly better than the original U-net. Indeed, one can observe that we obtained a different result in this experiment (with basic U-net) than in the experiment with the loss. The only difference between those runs is the seed used, therefore caution is advised.

C. Measuring impact of larger data set

The last experiment investigates whether or not adding a different data set can improve the performance of our model. To that end, we downloaded the aerial pictures used in [11].



(a) Training Image

(b) Ground-truth mask

Figure 5: Picture extracted from [11]

These images where quite different from ours. Most notably, they contained a lot of empty areas, that is, large landscapes without any roads. Moreover, the streets where often not labelled on the full width, i.e. it only indicated the center of the road. Nonetheless, [12] shows that weakly labelled data can improve the results obtained with hand-labelled pictures only. Therefore, we wanted to try this solution as well. For this experiment, we used 80% of the provided images as training examples and 20% as validation. We again performed the data augmentation independently on those two. Then, we split the images from [11] into patches of size 256x256 and kept only the section with at least 13% of roads. This way, we obtained 1427 new samples. Note that the validation set used to select the model with the best performance (checks performance after each epoch) contains only images provided by the teachers. Indeed, in the end we do not try to label roads from the Massachusset's dataset.

F1-score on Aicrowd	
Original U-net	0.901
Original U-net (+ Massach.)	0.905
Dense U-net w/ grid	0.902
Dense U-net w/ grid (+ Massach.)	**0.907**

IV. Discussion

Our model obtained a good F1-score. Hence, we believe that the proposed architecture fits the problem. However, we expected more improvement in the F1-score when including dense blocks in the original architecture. We were also quite surprised to see that attention mechanisms worsened the performance of the original U-net. Fig. 4 shows the prediction on hard images. In particular, the road on image 7 is hidden by the bridge. Therefore, the model fails there. Image 547 is difficult because almost everything is gray, even the building in the left. Nonetheless, the model is not fooled. Note that these samples were not used during training as they are part of the test set. We chose not to perform data augmentation on the Massachusetts pictures from [11] because the roads are already very diverse. See Fig 5 for an example of training sample. It could have been nice to try different losses when training the model on both (provided + Massachusetts). Indeed, pictures from [11] are clearly unbalanced, hence using the Dice loss in this case makes sense. Observe that we did not use any post processing on the predictions. Conditional random fields would be a possible extension to such a project. For instance, it was implemented in [13] as a recurrent network and could be embedded at the end of the U-net.

V. Summary

To conclude, we tried different kinds of fully convolutional neural network to perform road segmentation on aerial images. All those models are variants of the U-net architecture that we combined with the results of different research papers. We are proud of what we were able to achieve, as we had almost no experience with computer vision and deep learning. As we said before, we cannot make strict claim on whether or not dense U-net are better than the original one in general. In the end, our best prediction involved a dense U-net, but the score increase likely came rather from the new pictures, as we can observe a similar behaviour for the original U-net. This project made us want to learn more about computer vision problems in general, so thanks to the teaching team for this project proposal.

References

[1] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. [Online]. Available: https://arxiv.org/pdf/1505.04597.pdf

[2] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2016. [Online]. Available: https://arxiv.org/pdf/1608.06993.pdf

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017. [Online]. Available: https://arxiv.org/pdf/1706.03762.pdf

[4] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.

[5] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Robust visual tracking via hierarchical convolutional features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, 07 2017.

[6] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015. [Online]. Available: https://arxiv.org/pdf/1502.03167.pdf

[7] N. Camgoz, S. Hadfield, O. Koller, and R. Bowden, "Subunets: End-to-end hand shape and continuous sign language recognition," 08 2017.

[8] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, “Squeeze-and-excitation networks,” 2017. [Online]. Available: https://arxiv.org/pdf/1709.01507.pdf

[9] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, B. Glocker, and D. Rueckert, “Attention u-net: Learning where to look for the pancreas,” 2018. [Online]. Available: https://arxiv.org/pdf/1804.03999.pdf

[10] R. Deng, C. Shen, S. Liu, H. Wang, and X. Liu, “Learning to predict crisp boundaries,” 2018. [Online]. Available: https://arxiv.org/pdf/1807.10097.pdf

[11] V. Mnih, “Machine learning for aerial image labeling,” Ph.D. dissertation, University of Toronto, 2013.

[12] 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*, vol. 55, no. 11, pp. 6054–6068, 2017. [Online]. Available: http://infoscience.epfl.ch/record/230055

[13] S. Wang, L. Yi, Q. Chen, Z. Meng, H. Dong, and Z. He, “Edge-aware fully convolutional network with crf-rnn layer for hippocampus segmentation,” in *2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, 2019, pp. 803–806.