Machine Learning - Project 2
Road Segmentation

Orest Gkini, Theofilos Belmpas, Dimitrios Chalatsis
EPFL, Switzerland

***Abstract*—In this project, we present our solution to the problem of road segmentation from aerial satellite images. More specifically, the goal is to classify segments of a given image as either road or background. We implement and compare three convolutional neural network architectures, with our best one achieving a F1 score of 90.3% on the testing dataset of the competition.**

I. Introduction

In computer vision, semantic image segmentation is the process of classifying segments of an image (usually pixels) as belonging to a particular class. In our case, we are given a set of aerial satellite images and their ground truth images, and are required to classify each segment between two possible classes: (i) *road* (labelled 1) or (ii) *background* (labelled 0). However, the segments that we want our models to finally classify and output are not pixels, but patches of 16x16 pixels; that is, sets of contiguous pixels in a rectangular shape.

Our report is structured as follows: in Section II we describe the provided dataset and the techniques we employed to augment it, in Section III we present the implemented models, in Section IV we discuss our methodology for training and making predictions, in Section V we present the results of our experimental evaluation, and finally in Section VI we provide some final discussion.

II. Dataset

In this section, we provide a description of the given dataset for the task, explain why it needs to be augmented and enumerate the techniques we employed to perform the augmentation.

A. Data Exploration and Analysis

Our dataset consists of 100 aerial images and their respective ground truth images. The images have a size of 400×400 pixels which are represented by their three RGB (Red, Green, Blue) values. In the ground truth images every pixel has a value of 1 or 0, representing whether that pixel belongs to a road or not in the original image. An example image and its ground truth are shown in Figure 1.

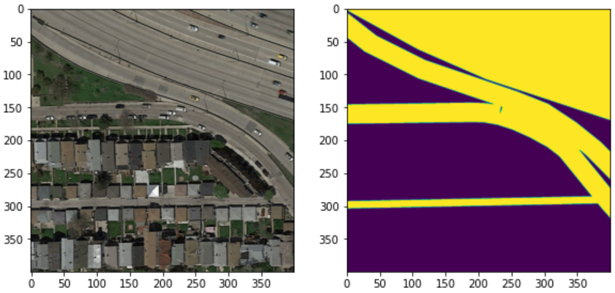

Figure 1. An example aerial satellite image from the provided dataset (left) and its ground truth image (right).

From that image we can immediately infer the complexity of the task. Firstly, roads can have various characteristics, i.e. they can have small or large width, they can have curves or be straight, they can have different orientations, they can have well defined borders or not. Secondly, in a lot of images trees, buildings, or cars do not allow the model (or even a human) to clearly distinguish roads from its surroundings. Finally, in many cases the rooftops of houses and parking spaces have the same color as the roads, and also train railways have identical shapes and appearance.

B. Data Augmentation

It is, therefore, obvious that a training set of 100 images is not enough to train a powerful convolutional neural network as it lacks the variety to prevent it from overfitting and provide it with enough information to learn all the different road patterns. Based on our observations, the images of the training set are heavily biased against diagonal roads, so the model could not accurately predict them on unseen data. Thus, to make our model more generalisable we performed the following data augmentation techniques. For every image, we crop it into 5 smaller ones of size 256×256 (located at the four corners and the center) to expand our dataset. We also rotate the initial image across 3 different angle degrees ([25, -30, 45]), which we observed to give the best results, in order to generate more images with diagonal roads. Then, we crop the center of each rotated image, resulting in an image with a cropped size of 256×256 pixels. So, we have effectively transformed our dataset from an initial size of 100 to 800 images.

C. Data Preprocessing

As a preprocessing step, we standardise the RGB values of every image by subtracting the mean and dividing by

Table I
ARCHITECTURE OF THE BASELINE CNN.

Layer	Kernen size	Stride	Padding	Output size
Input	-	-	-	16×16×3
Conv2d	5×5	1	2	16×16×32
BatchNorm2d	-	-	-	16×16×32
ReLU	-	-	-	16×16×32
MaxPool2d	2×2	2	-	8×8×32
Conv2d	5×5	1	2	8×8×64
BatchNorm2d	-	-	-	8×8×64
ReLU	-	-	-	8×8×64
MaxPool2d	2×2	2	-	4×4×64
Linear	-	-	-	1×1×512
ReLU	-	-	-	1×1×512
Output	-	-	-	1×1×2

the standard deviation of each of the three channels, respectively. We experimented with other methods, too, such as adding Gaussian blur to the image, but did not observe any improvements on the performance of the systems.

III. MODELS

Convolutional neural networks (CNN) have been widely applied to computer vision tasks with huge success [1] achieving state-of-the-art performance. More, recent applications include the task of image segmentation, such as in our problem. We have implemented and compared three convolutional network model architectures that we describe below.

A. *Baseline CNN*

The architecture of our baseline CNN model is shown in Table I. The model works on patches of images, for which common sizes are 8×8 or 16×16 pixels. In Table I we have assumed that the patch size is 16×16, thus, the input of the model has size 16×16×3 because of the 3 RGB channels of each pixel. The network also receives the ground truth label of the patch (one per patch) which classifies it as belonging to a road (label = 1) if at least 25% of the pixels in the patch belong to a road.

We apply two successive 5x5 convolutions to the input feature map, each followed by a batch normalization [2], a rectified linear unit (ReLU) activation function, and a 2×2 max pooling operation with stride 2. Then, the feature map with dimensions 4×4×64 is flattened and passed through a series of linear neural networks that output two single values, which when passed through a softmax function give us the probabilities of classifying the patch as road or background.

B. *UNet*

The architecture of UNet [3] consists of a *contracting* path and a symmetric *expansive* path, which yields a U-shaped architecture (Figure 4). The former consists of the repeated application of two 3×3 convolutions, each followed by a batch normalization and a rectified linear unit (ReLU) activation function (Figure 2).

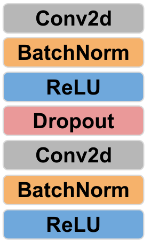

Figure 2. A double convolution block in Unet.

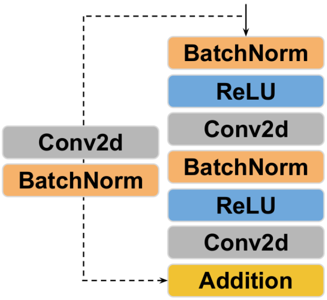

Figure 3. A double convolution block in ResUnet.

In the contracting phase, a 2×2 max pooling operation with stride 2 is used for "downsampling" after each double convolution block, which reduces the size of the feature map in half. Also, the number of feature channels is doubled. On the other hand, in the expansive phase the feature map is "upsampled" and its number of feature channels is halved, before concatenating it with the corresponding feature map from the contracting path. Then, a double convolution block follows. At the output layer, a 1×1 convolution is used to map each 64-component feature vector to our 2 classes.

We have modified the original architecture described in the paper by adding padding to the feature map in order to get an output that is the same size as the input. Therefore, we can feed a tensor of any size to the network without requiring any modifications. We have also added dropouts in the double convolution block as depicted in Figure 2, with probability 0.2 in the downsampling phase and 0.5 in the upsampling phase, as we observed that they improve the performance of the model.

C. *ResUNet*

ResUNet [4] builds on top of the UNet architecture, but modifies the double convolution block of the network as shown in Figure 3. It performs the double convolution after the batch normalization and ReLU operations, and passes the input from another convolution in parallel before adding the two outputs. We observed that adding dropout, as in

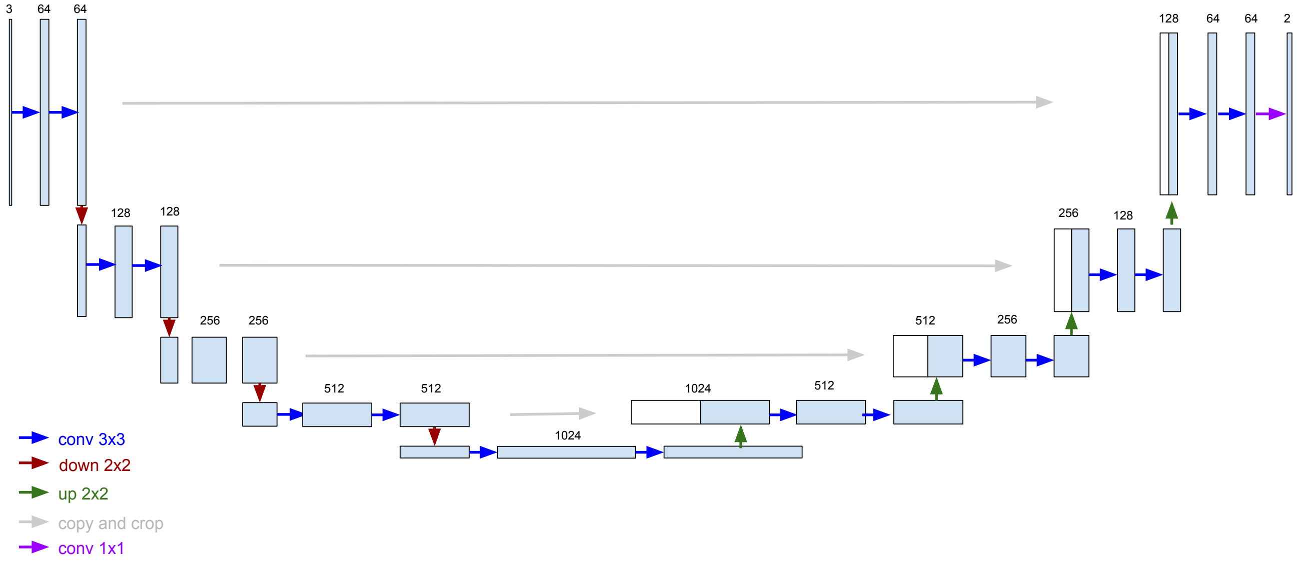

Figure 4. The Unet architecture.

the case of the simple Unet, had a negative impact on the performance of the system, so we omit it.

IV. Methodology

A. Training

As mentioned in section II-C our augmented dataset consists of 800 images and their ground truth images after applying all the described transformations.

1) For the baseline CNN: we segment each original image into patches of a fixed size (e.g. 8×8, 16×16) and feed those into the network as input along with the ground truth for each patch.

2) For Unet and ResUnet: we pass the whole image as input as a tensor of size 3×256×256 after the cropping transformations are applied.

In all cases, we split our dataset into a 90/10 ratio of training and validation data, and train our models for 100 epochs. For the optimization we use stochastic gradient descent with a learning rate of 0.01, momentum of 0.9, and regularization of 0.0001. As our loss function, we use the cross entropy loss, as proposed in [3].

B. Predictions

All of our models produce an output of 2 channels, which we pass through a softmax function that essentially gives us the two probabilities of classifying each segment as a road or background. The difference among the models lies in the granularity of the segments.

1) Predicting with the baseline CNN: The output of the baseline CNN has size equal to the batch size of patches provided as input, as it makes predictions on the patch level.

2) Predicting with Unet and ResUnet: These models make pixel-wise predictions, so the size of the output is equal to the size of the input image (but with 2 channels).

As required by the challenge description, the predictions of every model have to be in the form of 16×16 patches, where we assign a label to each patch as described before. An example output is shown in Figure 5.

V. Experiments

A. Implementation Details

We implemented our models using the PyTorch [5] framework. Unet, which is our heaviest model, takes approximately one hour to train on a Tesla T4 GPU provided by Google Colab for 100 epochs.

B. Results

The results of our experiments are shown in Table II. We perform the experiments on a 90/10 train/validation split (720 images for training and 80 for validation) of our augmented dataset and report the accuracy, precision, recall, and F1-score (for the best f1-score across 100 epochs).

The best model is Unet with a F1-score of 0.89, which is also the model that gave us the best submission score on the challenge platform (F1-score = 0.903). We observed that in some cases it fails to connect roads at intersections or detect them when their borders are not clear enough due to the existence of trees or parking spaces by the side of the road. The performance of the model improved significantly after training it on the augmented data, and marginally after adding dropouts. On the other hand, even though ResUnet

Table II
RESULTS OF OUR EXPERIMENTS.

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**
CNN8	0.80	0.62	0.70	0.64
CNN16	0.82	0.69	0.75	0.70
CNN25	0.82	0.75	0.68	0.69
UNet	0.96	0.89	0.88	**0.89**
ResUNet	0.95	0.84	0.80	0.82

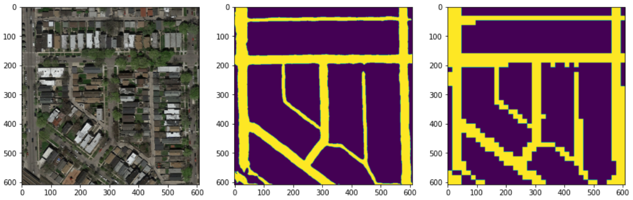

Figure 5. An example image from the test set, UNet's prediction, and its output converted to patches of size 16×16.

ranked second among the implemented systems, it did not achieve scores as close to those of Unet as we expected. The reported scores do not include dropouts for ResUnet as they hindered its performance.

We train the Baseline CNN model for patches of size 8×8, 16×16, and 25×25 in order to observe the impact of the patch size on the model's performance. We see that we achieve better results for patches of size 16, which agrees with what we generally observed in our experiments, although the difference with the case of patch with size 25 is marginal. Moreover, we see that a larger patch size leads to better predictions, which can be attributed to the fact that most roads are wider that 8 pixels, so a patch of size 8×8 is not large enough to provide the model with enough information, as it may not even contain both sides of a road.

C. *Observations*

During the training and testing of the models we analyzed their predictions and observed some important challenges. The main one is that roads are often obscured by objects, such as trees or buildings, which sometimes resulted in "holes" in the predictions. It also became apparent that without proper augmentation of the initial dataset, the models heavily overfitted to the training images and could not adapt to unseen shapes and patterns, e.g. training mainly on narrow streets would not allow them to be able to predict wide motorways, and training mainly on straight roads would not allow them to detect curved ones. Last but not least, parking spots and private driveways are in many cases virtually indistinguishable from road areas, since not only do they have the same appearance, but are also always connected to roads.

VI. CONCLUSION

In this project, we implemented and evaluated three convolutional neural network architectures on the problem of road segmentation from aerial satellite images. We experimented with different data preprocessing and augmentation techniques to properly train our models and improve their predictions on unseen data. Apart from the mentioned models, we experimented with other ideas, too, such as training two UNet models with different kernel sizes in parallel and concatenating their results, or training a second UNet model with the predictions of the first one in order to correct its mistakes, but neither of them yielded any improved results compared to the original UNet.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, vol. 25, 2012, pp. 1097–1105.

[2] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 37, 2015, pp. 448–456.

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015. [Online]. Available: http://arxiv.org/abs/1505.04597

[4] Z. Zhang, Q. Liu, and Y. Wang, "Road extraction by deep residual u-net," *CoRR*, vol. abs/1711.10684, 2017.

[5] 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., 2019, pp. 8024–8035.