To family and friends. To all of my teachers and the mentor.

CONTENTS

1. Introduction

This work is concerned with the problem of semantic segmentation of images, a rather green field of computer vision, and the subproblem of aerial imagery analysis.

The problems in the field of computer vision are usually solved using convolutional neural networks (CNNs), and all the architectures concerned in this work are based on those.

First few chapters give an overview of underlying neural networks and semantic segmentation itself in the second and third chapter respectively, datasets commonly used for work on the problem in the fourth chapter and some modern approaches to semantic segmentation of these and other datasets in the fifth.

Lastly, in the sixth and seventh chapter, the work focuses on analysis of Synthinel and Inria dataset images respectively using SwiftNet, a software initally made for semantic segmentation of urban scenery.

2. Neural networks

Neural networks are mathematical models used for many tasks in machine learning, ranging from traditional AI tasks like game playing to novel approaches such as art production and natural language processing. Although originally conceived in 1940s, neural networks were not really used for practical purposes until 1970s when the backpropagation algorithm came to light.

2.1. Perceptron

The simplest basic building block of a neural network is a perceptron, an imitation of a brain neuron. Just as a real neuron receives input throgh dendrites from several other neurons, a perceptron receives input from perceptrons in the back and sends them through axon terminals to the next layer neurons' dendrites. The strength of the signal depends on how thick the myeline sheath around the axon is, which is represented by weight parameter.

On a less abstract level, it is a simple computational unit that receives inputs from the layer before it (or raw input if it is a part of the input layer), sums them up and multiplies them by its designated weight. Perceptrons are usually used in fully connected neural networks, although it's not always the case.

The mathematical function each perceptron is a model of is rather simple one, with w being a weight, x being the input and phi the threshold function:

$$y = \varphi(\sum_{i=1}^{n} w_i x_i + b) = \varphi(\mathbf{w}^T \mathbf{x} + b)$$

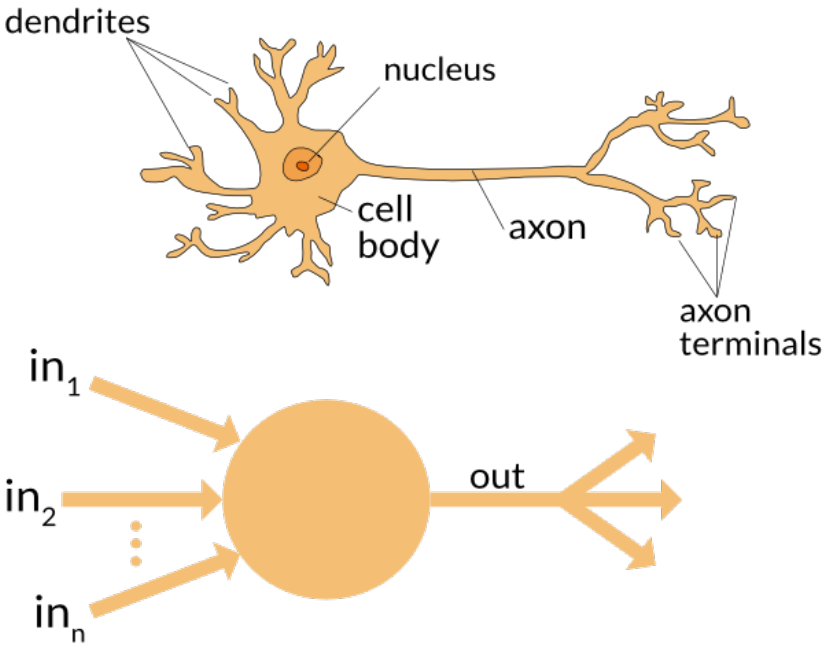


Figure 2.1: A scheme of a perceptron with inputs and outputs compared to a scheme of a natural neutron

2.2. Threshold functions

Threshold functions are used to activate or deactivate the output of a neuron and either emphasize or downplay its significance.

The oldest used function is a simple step function that either deactivates the neuron if it's value is negative or sends it on if it is positive. A nowadays much more popular function is Softmax: it maps the very high values near 1 and very small and negative values near 0, whereas the values between 0 and 1 are transferred in a nearly linear manner. Hyperbolic tangens is very similar to the softmax, but its values range from -1 to 1. Both of these functions examine the vanishing gradient problem - as the space to which they map large values is very limited, even large differences between values get mapped to very close values and changes in parametres tend to be comparatively small - effectively non - existent.

The activation function used in all of the nets examined in this work, and generally the most popular activation function choice in recent deep learning software is ReLU, rectified linear unit: f(x) = max(0, x). As values are not squashed, differences between values and outputs are comparatively large.

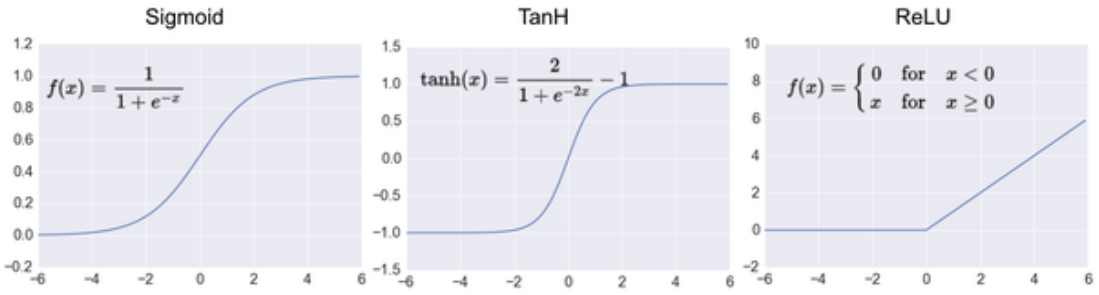


Figure 2.2: Sigmoid, hyperbolic tangens and rectified linear unit function's plot and mathematical expression

2.3. The learning process

The learning of a network is achieved by minimising the loss function, or, in the other words, describing how far is the solution of a neural network from the ground truth laid before in order for the network to learn.

Optimisation is then done by moving the solution vector in the direction opposite to the gradient's.

The main loss functions used in neural networks are L1 , also known as mean absolute error (MAE) and L2, also known as mean square error (MSE). Both functions measure the average difference between expected and actual values, but the squared version punishes the largely outlying values much more severely. The math expressions that express L1 and L2 respectively go as follows:

$$MAE = \frac{\sum_{i=1}^{n} \mid y_i - \hat{y}_i \mid}{n}$$

$$MSE = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}$$

Mean bias error is very similar to the L1 function and is only used in specific cases when showing the bias itself is particularly important.

$$MBE = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)}{n}$$

A loss function of great importance for providing a set of inputs with a single semantic mark is multi class SVM loss. It tries to push the model into making the error value of one class as small as possible regardless of the others. This loss function is very important for many uses of computer vision - it is very easily illustrated in the

cats and dogs example. The image cannot be 35% cat, and such an assumption has to be punished severely.

$$SVMLoss = \sum_{j \neq y_i} max(0, s_j - s_{y_i} + 1)$$

Another loss function used for classification problems is Cross Entropy Loss. It also punishes "undecided" differences with ground truths and is thus important for computer vision.

$$CrossEntropyLoss = -(y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i))$$

2.4. Optimisation

Loss function optimisation, the steps towards the optimal solution, is done by the method of stochastic gradient descent. The parametres go to the other side from gradient's direction in that point, as goal of the process is minimisation. Stochastic gradient descent with momentum is an implementation in which the momentum is taken into account, which means every time an optimisation step is undertaken the steps preceding it are taken into account as well.

The most modern approach to the problem almost exculsively used in the implementation shown in the rest of the work is ADAM, which stands for Adaptive Moment Estimation. The moment and learning rate are adapted according to the performance of the algorithm. The algorithm itself is based on AdaGrad and RMSprop, other advanced gradient descent algorithms.

2.5. Overfitting

One of the biggest problems in deep learning is overfitting. Overfitting is the phenomenon of networks getting "over - learned", being unable to generalize the results. A most exreme such example would be a neural network not being able to classify a dog image it was learned on moved by a single pixel to one side.

That problem can be solved in several ways, which can also be combined. First of all, simply diversifying the learning dataset and making it larger does part of the trick. Still, that may not be enough.

The introduction of separate training and evaluation datasets is another brainchild of attempts to solve that problem, as the good model is not the one that knows the

training dataset, but the one that understands a different one. The concept is the same as exam questions and practice questions being different to evaluate student's ability to generalize ideas laid before him in a course.

Another step towards the goal of getting rid of overfitting is regularisation. Some of the most important methods are L1 and L2 regularisation. Both these methods tend to reduce some or all of the weight values. That works because, as a rule of thumb, the bigger weights are the more complex and thus more fitting to the training dataset model is. L1 penalises the absolute weight values, whereas L2 penalises the squared values.

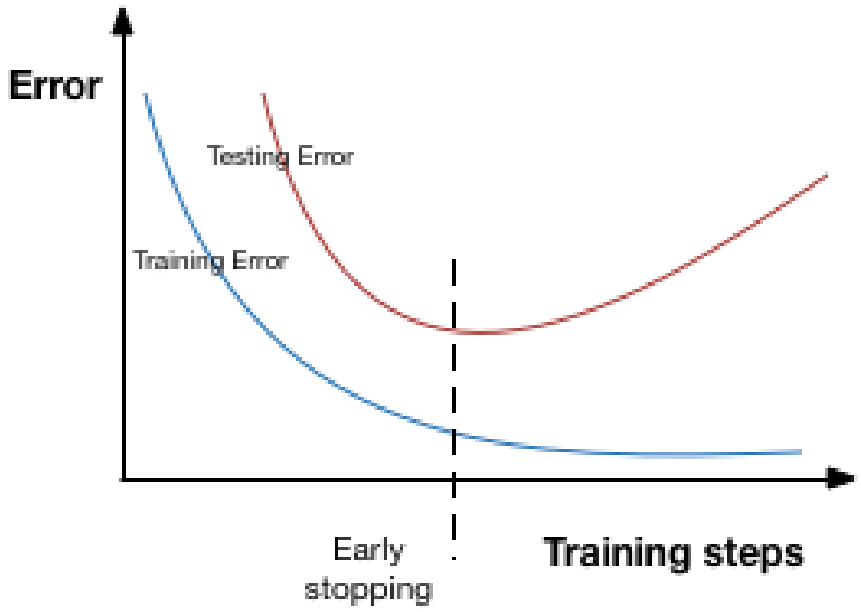


Figure 2.3: Where the early stop has to take place in a symbolical representation of a learning process

2.6. Fully connected neural network

Many of older approaches in the field of semantic segmentation have some fully connected perceptron layers as a part of their architecture. It is worth noting that fully connected layers are usually eliminated from more modern approaches in this field as their computational complexity is not backed up by state-of-the-art performance.

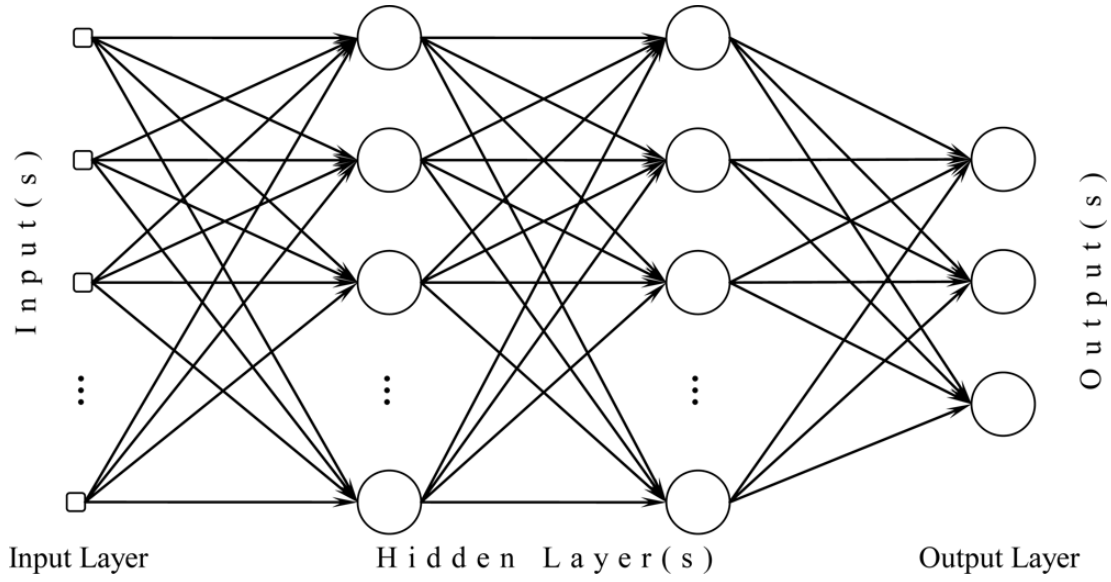


Figure 2.4: A fully connected neural network

3. Convolutional neural networks

Recognition of image's regions, that are in turn to be given a semantic label, is done with convolutional neural networks. The image division is done layer by layer: as the first layer recognizes simpler features such as straight or curved lines, the farther we go towards the network's output, the more complex features network extracts. In the end, such a network is able to distinguish between "human - level" classes such as persons, traffic signs and even emotions.

Convolutional neural networks are much different from their classical predecessors. They are esentially arrays of filters and downsampling or upsampling modules. The common convolutional network downsamples by pooling after every convolution and soon starts upsampling with some simple method, for example bilinear interpolation.

There is a number of innovative approaches used to enchance the performance of "classical" CNNs. For semantic segmentation of aerial imagery, some of the best approaches are based in ladder style architectures as shown in (Kreso et al., 2017) and (Krešo et al., 2020).

3.1. Convolution

Convolution is a mathematical operation whose operands are two functions, and result is expression of the second function's effect on the shape of the first, shifting one function along the other's domain and evaluating the value of their multiplication's integral. In the other words, in the context of CNNs, convolution is esentially mapping features from the image directly extracting them into a new one. For example, a convolution with horizontal line extraction filter would turn a raw image into an image with only its horizontal edges visible.

One of the issues of convolution is that it "cuts off" the pixels on the borders of the image, thus creating a need for upsampling at some point after the convolution.

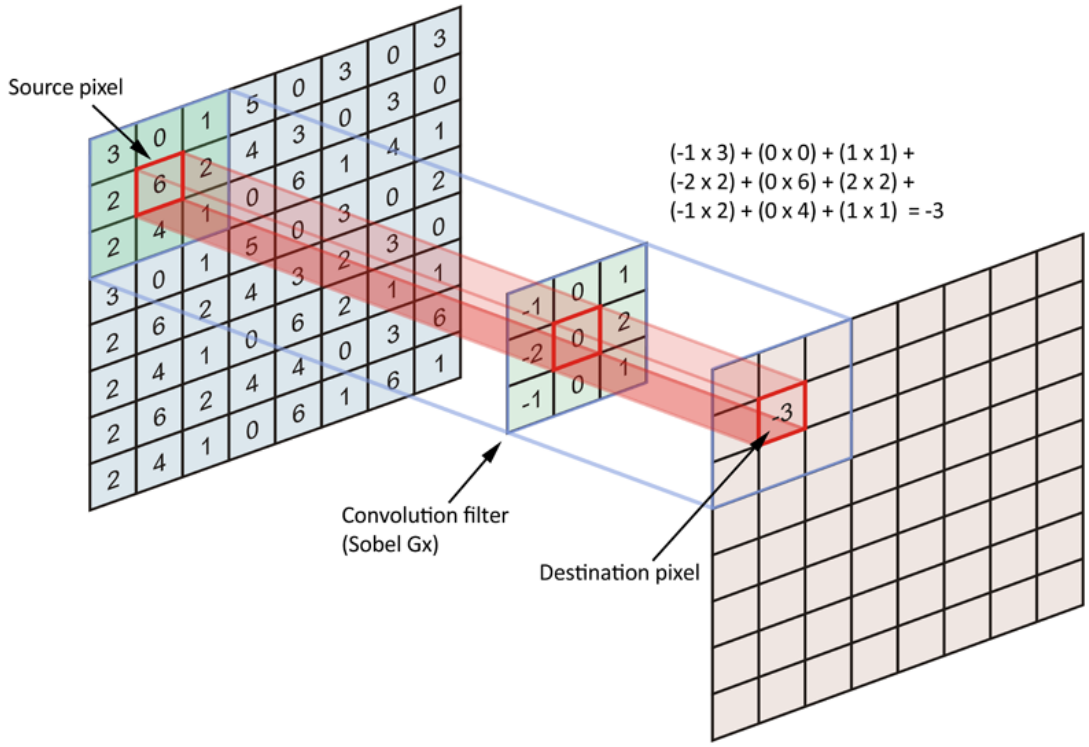


Figure 3.1: A graphic showing convolution of two matrices, the smaller one being a 3x3 filter

3.2. Atrous convolution

Atrous convolution is a procedure based on taking groups of sparse instead of adjacent pixels thus reducing the number of operations without compromising the perception power of the network.

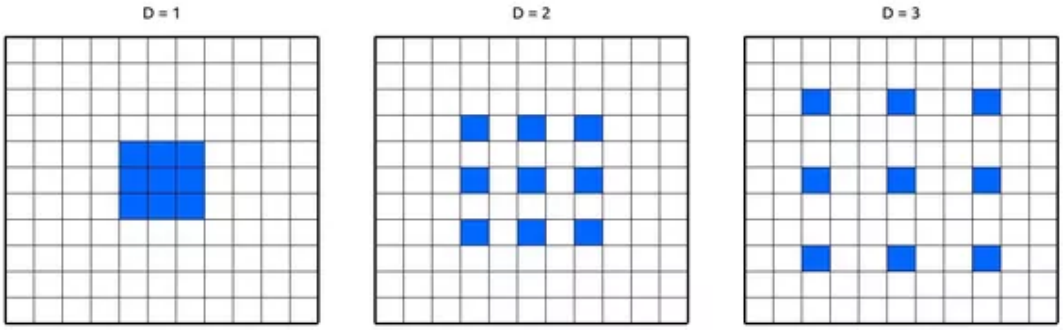


Figure 3.2: The example of atrous convolution parameter alternations, ranging from "normal" convolution to a very sparse model

3.3. Pooling

Pooling is an action that is usually performed right after the convolution. It is a way of downsampling the image by turning certain regions of an image into smaller such regions.

The most commonly used flavour of pooling is max pooling, which esentially means extracting the "strongest" value and representing the whole pooling selection with that value.

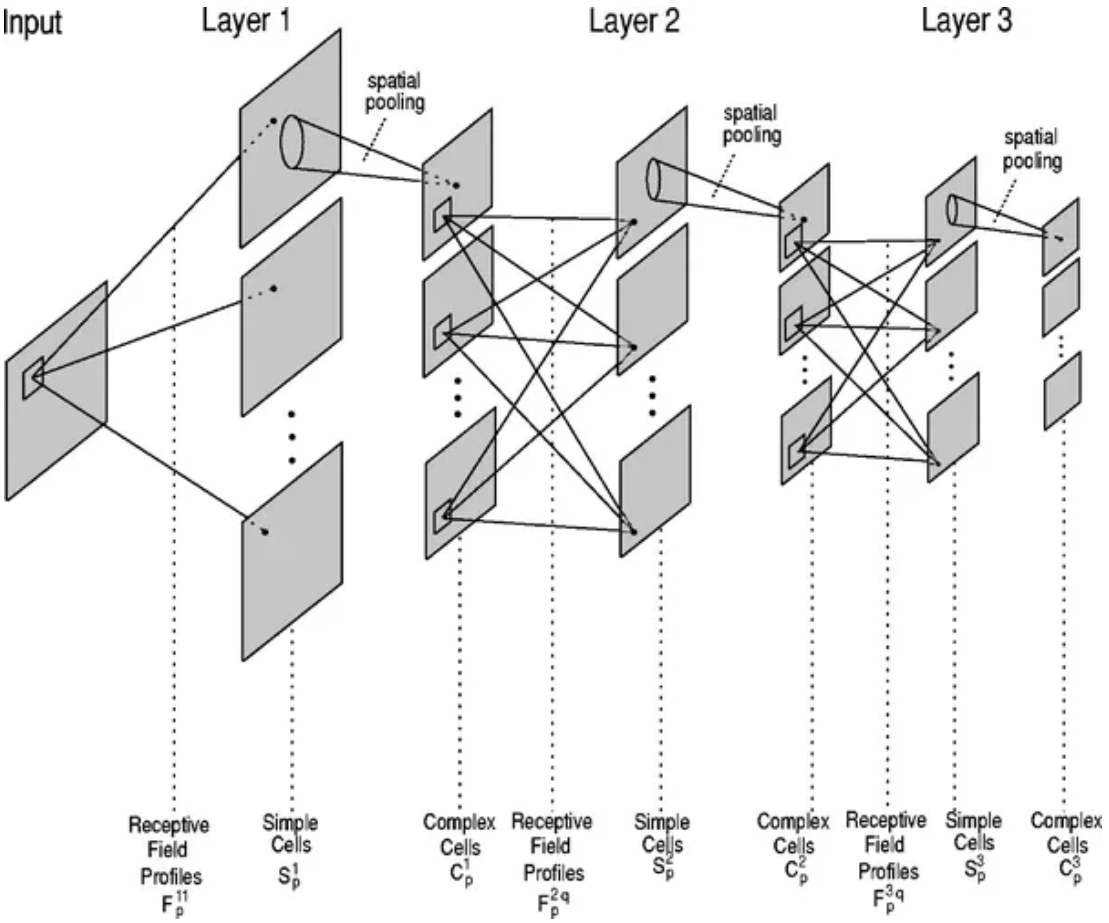


Figure 3.3: Uses of pooling in a typical CNN

3.4. Conditional Random Field

Conditional random field is a mathematical graph similar to a Markov chain split into subsets X and Y. X is a subset of atomic "raw" input units, in our case pixels or superpixels, whereas members of Y are semantic labels or classes. Conditional probablities of Y with the condition of X are then calculated and taken into account

depending on the probability of each pixel or superpixel belonging to a certain class.

CRFs are very popular in problems regarding computer vision and natural language analysis as they are able to take into account the context in which they are taken. Thus they are able to rectify the output of the rest of the convolutional network.

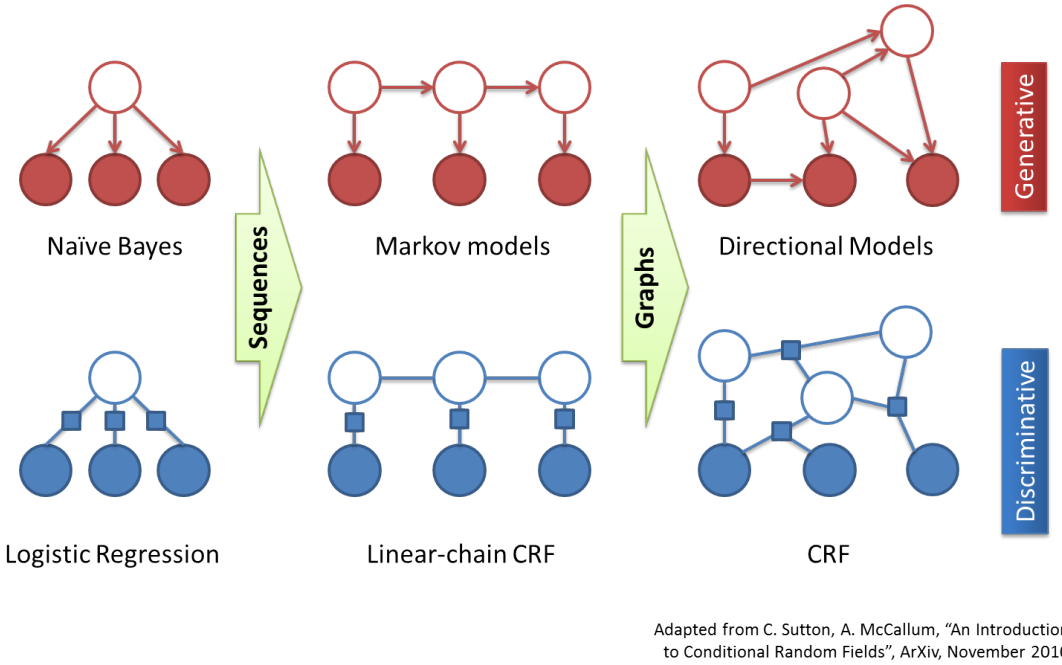


Figure 3.4: The relationship between CRF and similar mathematical structures

4. Semantic segmentation

Semantic segmentation of an image is a process of division of an image into areas - segments - of its entire composition and pairing of each segment with its semantic label. In the other words, each pixel of an image is converted into a single number representing its semantic class.

4.1. The problem

The number of semantic classes is what makes semantic segmentation different to the other problems in the field of computer vision: usually, in computer vision we view the whole image as something that should be assigned a semantic value and the solution is synthesis of image's meaning.

In this particular problem, we try to assign each pixel, and thus different parts of the picture a different role, and the solution is an analysis instead of synthesis of image's components' semantics.

4.2. Applications of semantic segmentation

Semantic segmentation's main applications can be found in autonomous agent orientation and medicine. The speed of evaluation is especially important in former case: where a patient's brain can be analyzed for days or weeks, an autonomous car's computational unit has to analyze the image in a split-second and react accordingly. As the high speed of evaluation and low complexity of the computational modules are both very important and contradictory points, making the evaluation of scenes as fast as possible is very important.

There is also a lingering ethical question: the accuracy of scene analysis cannot be compromised, as otherwise it could cause a great material damage or even loss of life. Thus, approaches to this problem can differ greatly depending on the application.

5. Datasets

Nowadays, a number of datasets are available for semantic segmentation training mainly in areas of scene analysis and medical image analysis, but also of aerial imagery.

Datasets are usually split into three parts. The largest one is the train subset, used for training the neural network.

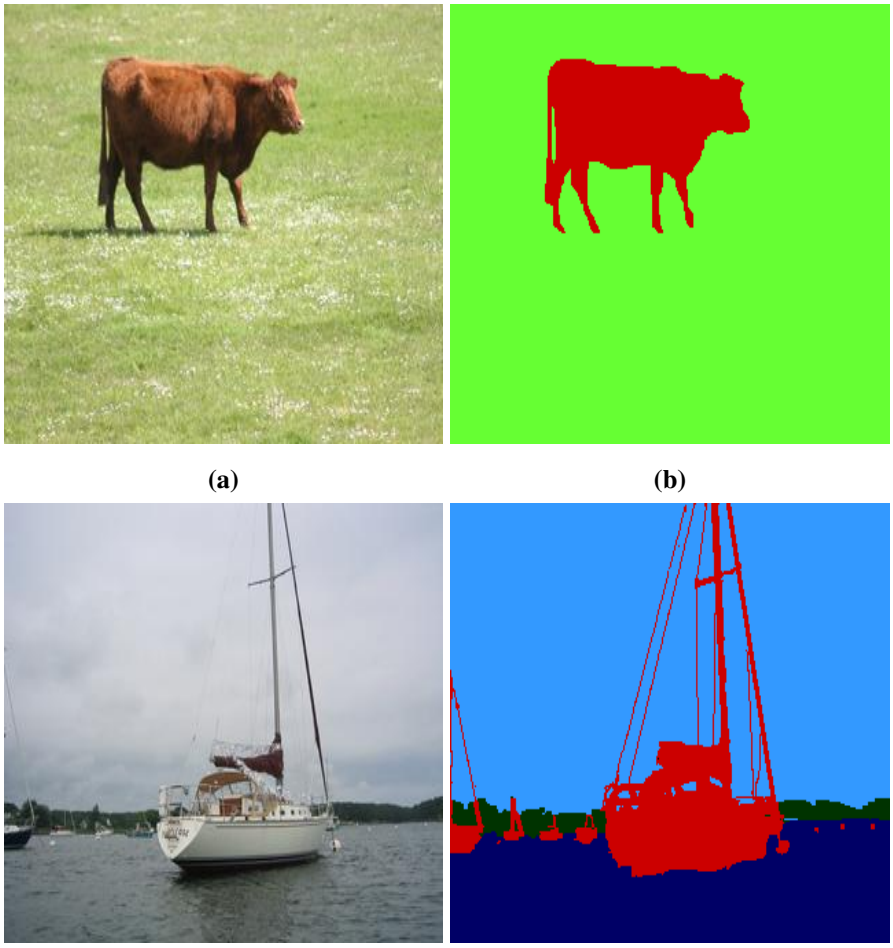
The other one is val subset used for evaluation of the dataset after each step of learning - these images are not learned on, but the accuracy of the net when evaluating those is used as quality control.

The last one is the test subset. That is the subset on which evaluation's ability of the web is finally tested. Usually, the test set contains data significantly different from the one used in evaluation and train datasets as to prove how well the net generalises.

5.1. Stanford Backgrounds

One of the oldest and most common datasets for scene analysis, *Stanford Backgrounds* (Gould et al., 2009), is a dataset that offers a rather small and comprehensible set of 715 images with dimensions of 320 by 240 pixels along with their labels in three different aspects: it is offered in versions segmented by regions of the picture (each visible object is separated), one indicating the geometric class (vertical, horizontal, sky) and one indicating precise semantic regions as seen from a human perspective, the regions being sky, tree, road, grass, water, building, mountain and foreground object. The images itself are a combination of scenes from daily life, usually of a very urban or very rural scenery with an emphasis on a clearly defined subject that stands out in the image.

In the creation of this work, Stanford Backgrounds dataset was used to serve as a "hello world" example of the software running and to understand how the software itself works. It was used because of its universal recognizability and a small number of small images that allow for a reasonably fast training and evaluation.



(a) **(b)**

(c) **(d)**

Figure 5.1: Examples of typical Stanford Backgrounds images and their semantic segmentation examples

5.2. Inria

Inria Aerial Image Labeling Dataset(Maggiori et al., 2017) is a dataset of 360 aerial imagery photographs of urban settlements that cover an area of 810 km^2 and comes with half of it (405 km^2) labelled into "building" and "not building" semantic classes so that half can be used for training. The other half of ground truths is not available. The images' resolution is very large, each of them being 5000 by 5000 pixels, and their spatial resolution is 0.3 m.

The imagery itself is very diverse, offering scenes from Californian business districts to Middle European Alpine towns. The spectre of urban settlement nature is emphasized as one of the dataset's roles is to assess the generalization power of a semantic segmentation algorithm.

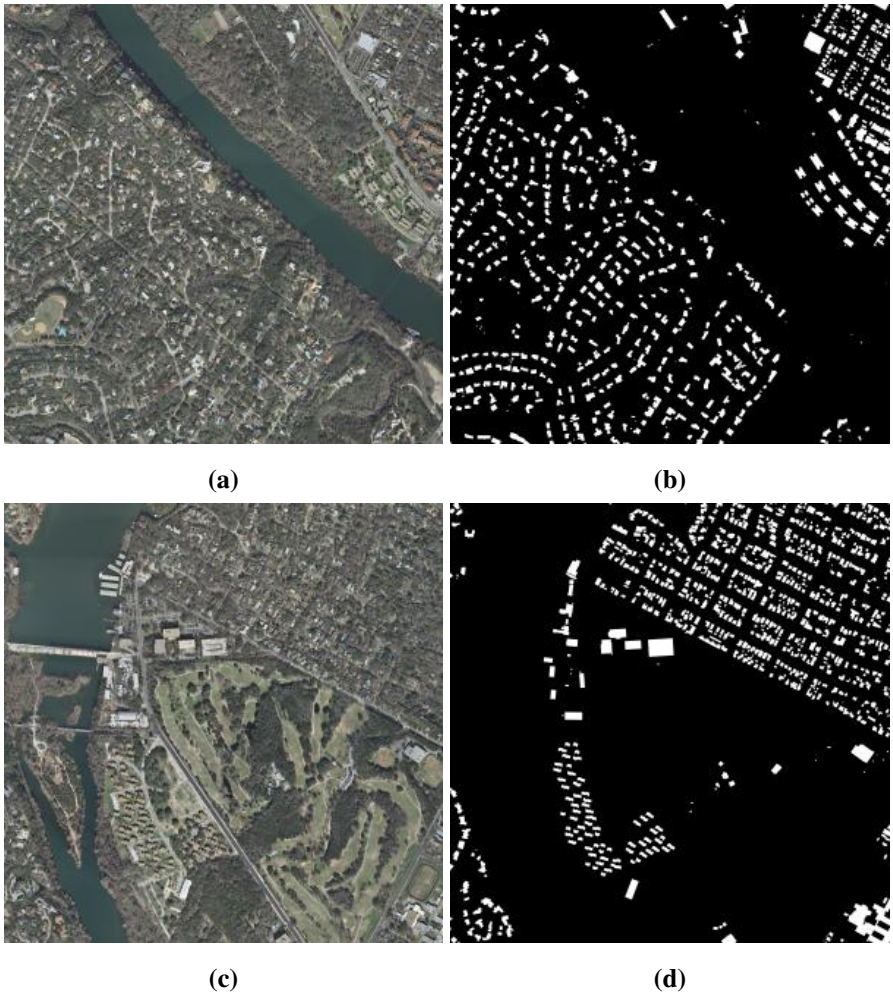


Figure 5.2: Examples of Austin aerial images from dataset and its semantic segmentations

5.3. DeepGlobe

DeepGlobe Land Cover Classification Challenge Dataset(Demir et al., 2018) contains 803 satellite images sized 2448 by 2448 pixels with the resolution of 50 cm. Every image is paired with its ground truth, meaning each image's components are separated in one of the following semantic classes: Urban land, Agriculture, Rangeland, Forrest land, Water, Barren land and Unknown.

5.4. DLR-SkyScapes

DLR-SkyScapes(Azimi et al., 2019) is a dataset for high definition mapping and as such features a dataset with a large number of semantic labels - the images are divided into 31 categories, of which 12 are different kinds of road lane markings, whereas the

Figure 5.3: A DeepGlobe image with semantic marks on buildings

others are low vegetation, paved road, non-paved road, paved parking place, non-paved parking place, bike-way, sidewalk, entrance/exit, danger area, building, car, trailer, van, truck, large truck, bus, clutter, impervious surface and tree. For practical reasons the dataset is divided in several groups where some groups are merged so the dataset can be used in different ways.

The extraordinary attention to detail and high resolution of images, 13 cm of ground for each pixel, make the DLR-SkyScapes unique in its category as it can potentially be used for a large number of uses in many fields.

5.5. Synthinel

Synthinel-1 dataset(Kong et al., 2020) is a dataset created with Synthinel software for creation of aerial imagery of synthetic, simulated nature. When examining cross - dataset evaluation between Inria and DeepGlobe (training on one and testing on another and vice-versa), both UNet and DeepLab V3 showed substantially better results when trained on datasets that included one of those and Synthintel simultaneously.

Synthinel-1 includes 1891 images of synthetic cities that resemble real urban, suburban and non-urban settlements of both very contemporary and older architecture.

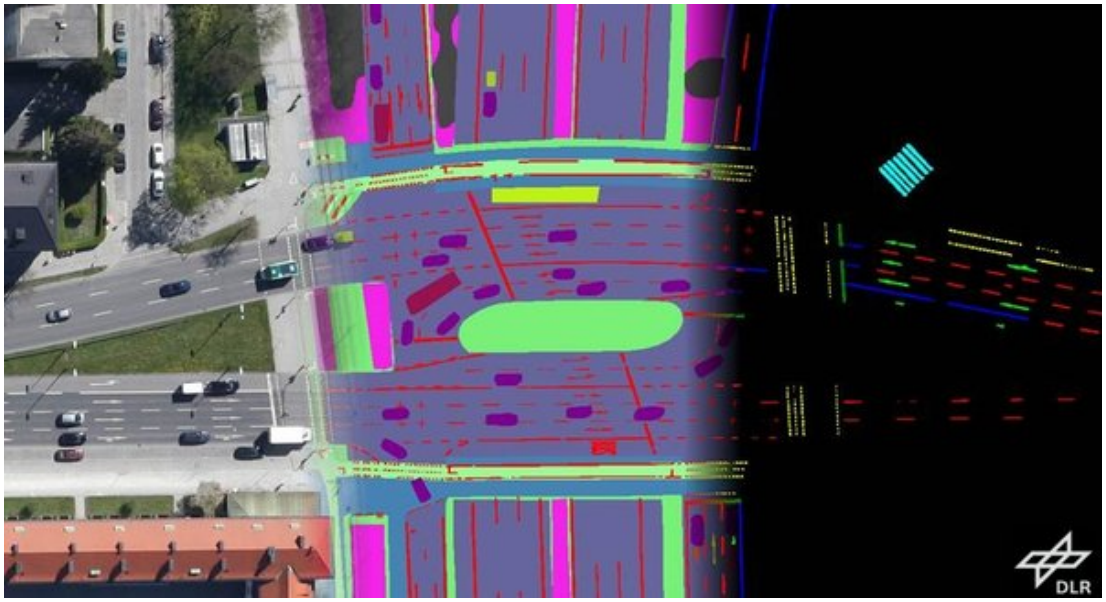


Figure 5.4: An example of DLR-SkyScapes image and its many semantic classes

Such diversity provides realistic data.

Another plus of synthetic datasets, Synthinel included, is the option of production of datasets of custom sized images as well as experimenting with different flavours of cities regarding the specific use.

6. Contemporary approaches

Nowadays, many researchers concern themselves with problems regarding semantic segmentation. Modern approaches have reached reasonably good results working for the most part with forementioned datasets. Some of these approaches include U-Net, DeepGlobe and Swiftnet.

6.1. U-Net

U-Net(Ronneberger et al., 2015) is a convolutional neural network that was suggested for use on biomedical image segmentation. The network derives its name from its U - shaped scheme, as there is a matching number of expansive and contracting layers.

U-Net is made to use large images and analyze them by parts instead of smaller ones in order to achieve better performances on the GPU. A bigger number of smaller images would result in overhead. When presented with an image selection close to the border, it fills in the missing parts of the image by mirroring its neighboring regions into them.

The revolutionary aspect of U-Net is the fact that it does not have a fully connected layer, but only uses convolution, max pooling and upsampling, thus creating a fully convolutional network. Also, instead of only using max-pooling for further abstraction, some lower-grade features are fed directly into classification by upsampling channels.

Another concept is directly channeling parts of the image throughout the network. Parts of images are fed directly to the "other side" of the network so more abstract features can be directly extracted from raw image.

6.2. DeepLab

DeepLab(Chen et al., 2018) is a web architecture currently at its third generation. Its approach is based on using fully convolutional networks with atrous convolution

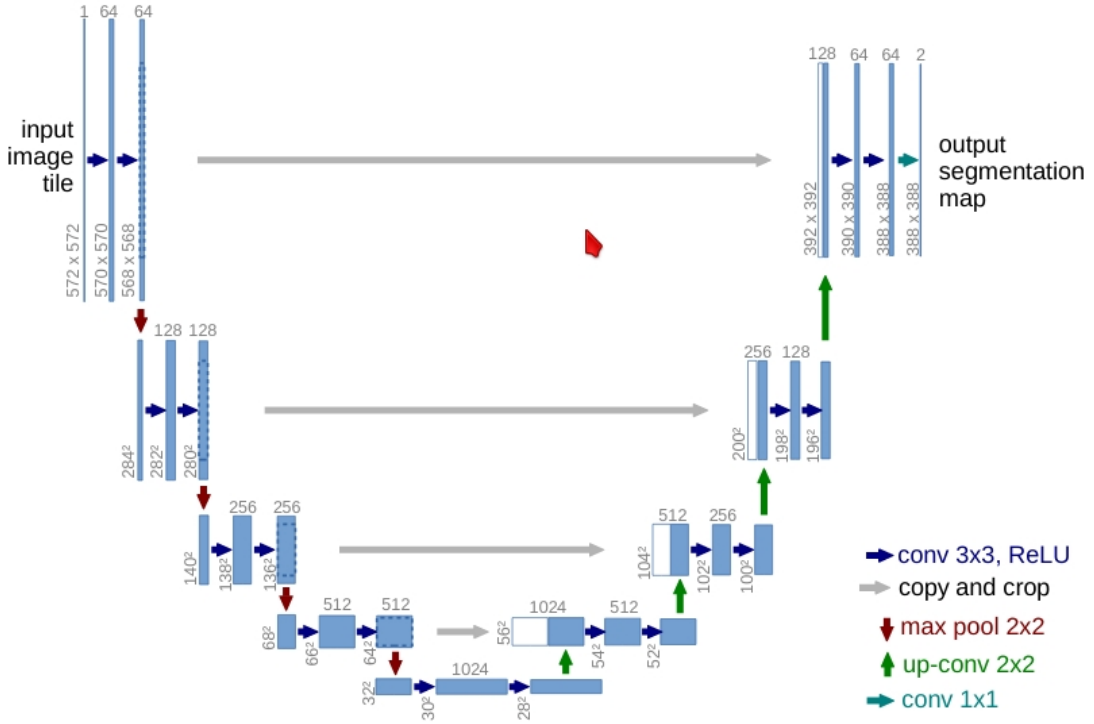


Figure 6.1: U-Net architecture scheme, with blue boxes corresponding to multi-channel feature maps, and arrows to operations

and fully connected conditonal random fields.

6.3. Swiftnet

Swiftnet(Orsic et al., 2019) is a web architecture that has been conceived for approaches related to real-time analysis of driving scenes - namely for automated vehicles such as cars, trucks and drones. The main points of the work are a focus on lightweight universal net architecture, using lateral connections for low-cost upsampling and new ways of fusing features shared at multiple resolutions for reduction of needed operations.

The Swiftnet architecture uses ResNet-18 model pre-trained on ImageNet dataset as a recognition encoder. The result of encoder's actions is a semantically rich image, i.e. image with strong semantic features. It is also worth noting that the number of feature channels increases after each step.

On the other hand, the number of features is fixed on the decoder path. That leads to a problem: when establishing lateral connections between encoder and decoder components, the lack of features on the decoder size is fixed by a 1x1 convolution

before the summation with decoder component output. The decoder is supposed to be as lightweight as possible as the best performance is achievable by using minimal computational resources and time is such an important resource when evaluating the scenery in real time.

The direct input to the decoder layer is fed through a spatial pyramid pooling module that receives the feature map resized to W/32 * H/32 size.

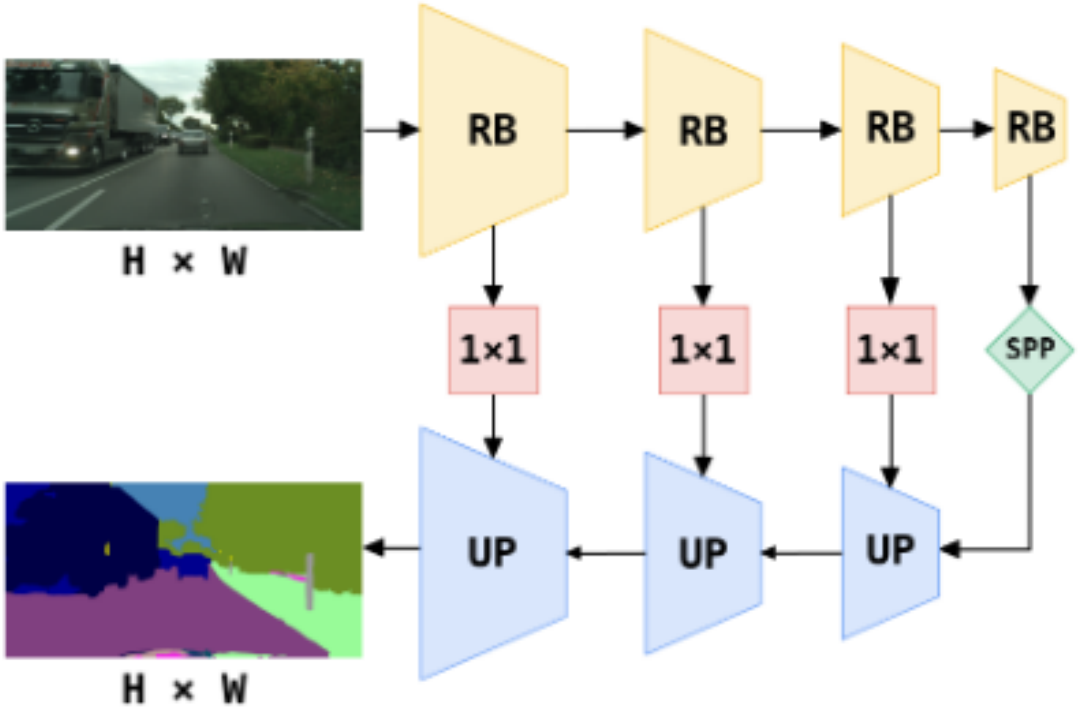


Figure 6.2: Swiftnet architecture scheme, with yellow trapezoids corresponding to encoder, feature extracting components, blue ones with decoder, upsampling components and the red squares with 1x1 convolution bottlenecks for lateral connections

7. Semantic segmentation of Synthinel with Swiftnet

There are several reasons Swiftnet was chosen as the dataset for performing this experiment. Firstly, the focus on lightweithedness lets us execute experiments even on very old, highly unspecialized hardware for relatively complex tasks. Secondly, the universal nature of the web provides us with the ability to use it for the tasks it was not made for.

The dataset was split into training, validation and test subsets with 60% of the dataset used for training, 10% for the evaluation of a model and the remaining 30% for test set. No selection was done and the subsets' members are chosen completely at random. The results of evaluation at test dataset will be used as reference points when mentioning IoU.

7.1. Results

Swiftnet proved to be an excellent tool for analysis of the dataset.

The quality of a function is measured with IoU, abbreviation for Intersection over Union. More precisely, it is a measure of area of intersection between expected and actual area of each of the semantic class instances.

The choice of hyper - parametre's experiment values are based on the original work. It was assumed these values can serve as a good starting point as problems of semantic segmentation very much resemble each others.

7.2. Number of upsampling layers

One of the most important hyper-parametres is the number of upsampling layers. The choice of experiment values was based on the original work.

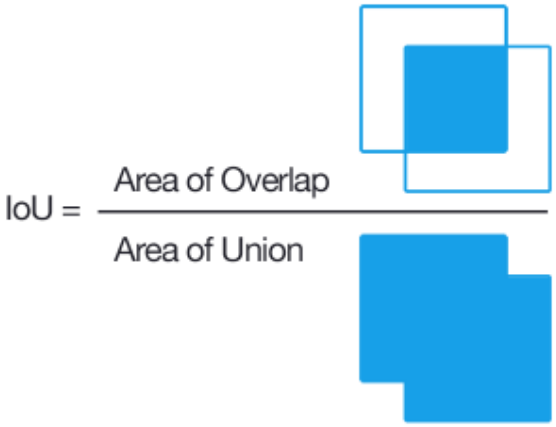


Figure 7.1: A graphical presentation of how IoU works with rectangles presenting semantic class instances

Number of upsample layers	16	32	64	128
IoU of the test dataset	87.1	87.2	87.8	88.3

However, surprisingly enough, the number of upsampling layers has not changed the performance significantly.

The number of semantic classes is very low, not to say minimal - the software only analyses if a pixel belongs to a building, or it does not. Also, the buildings themselves are usually rectangular which means their shapes are not complex and do not require many feature channels.

There is also a significant speed benefit to decreasing the number of semantic classes and the extra resources can be used for analysis of bigger datasets.

7.3. Crop size

Another potentially important parameter is how big the parts of image that will be analsed at a time will be.

When the crop size is reduced to 64, there is a drop from 88.34% to 86.62%, which is rather insignificant. The difference of execution speeds is not significant either.

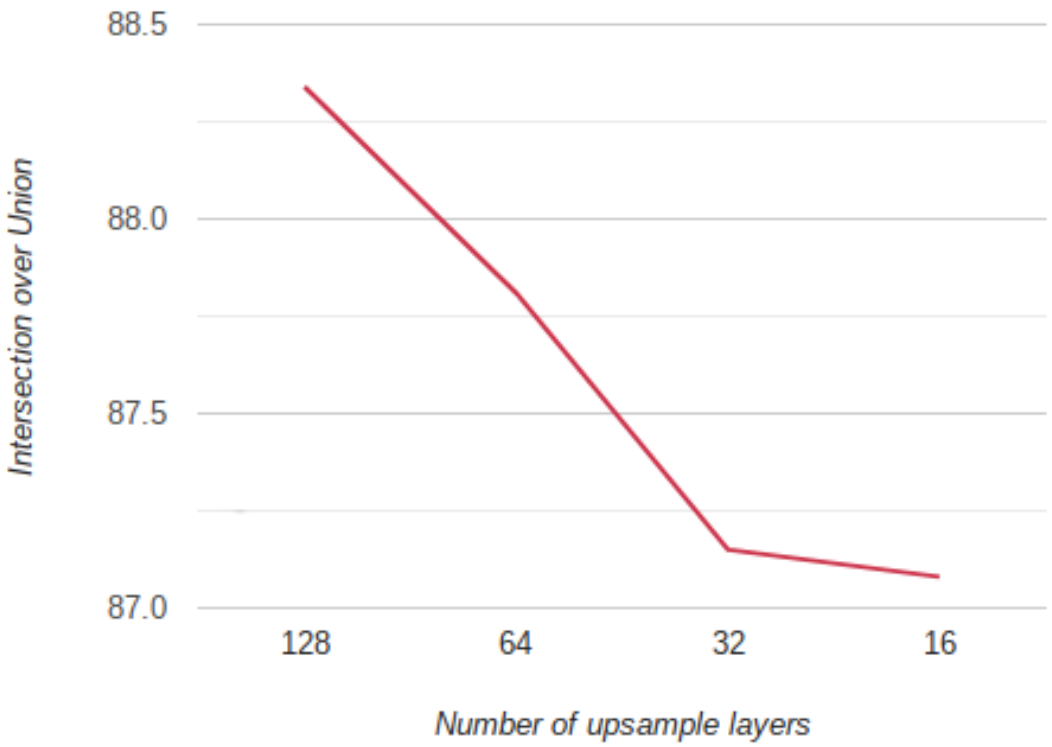
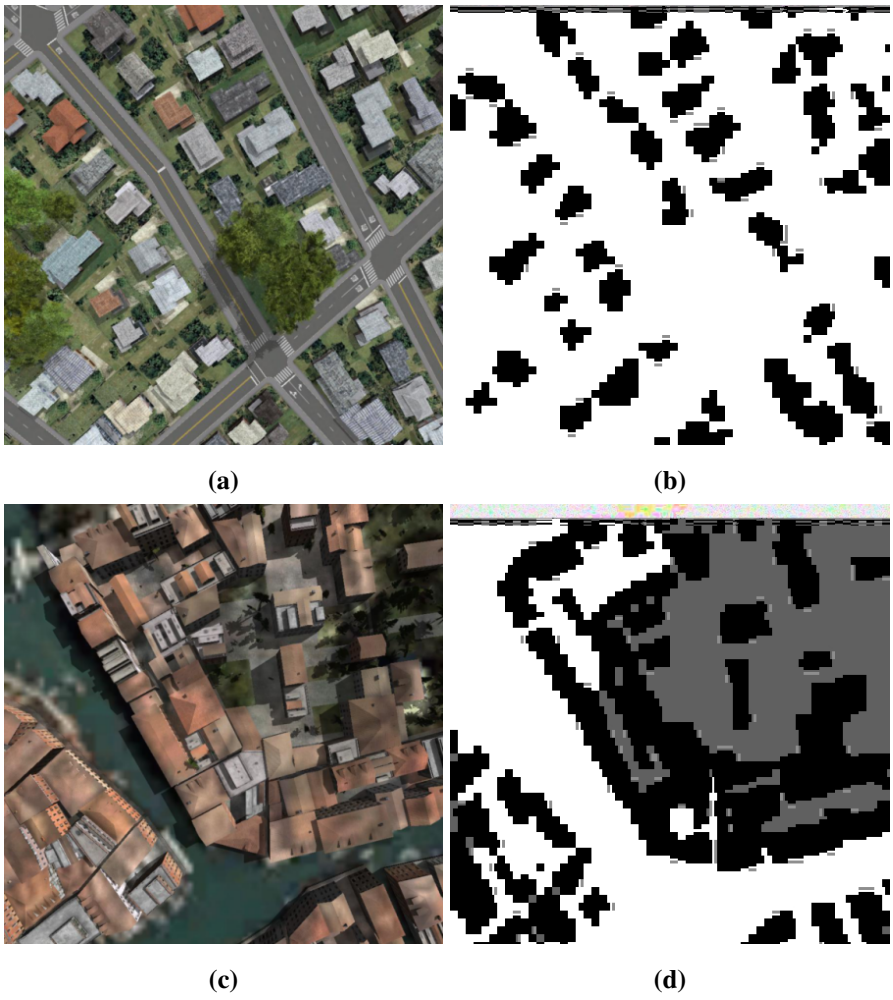


Figure 7.2: Graph showing the relation between the number of upsample layers and the IoU



(a) **(b)**

(c) **(d)**

Figure 7.3: Synthinel images analysed with default Swiftnet hyper-parametres

8. Semantic segmentation of Inria with Swiftnet

Another analysed dataset was Inria. Inria is a substantially different dataset from Synthinel, as it is not synthetic,but consists of real aerial photography.

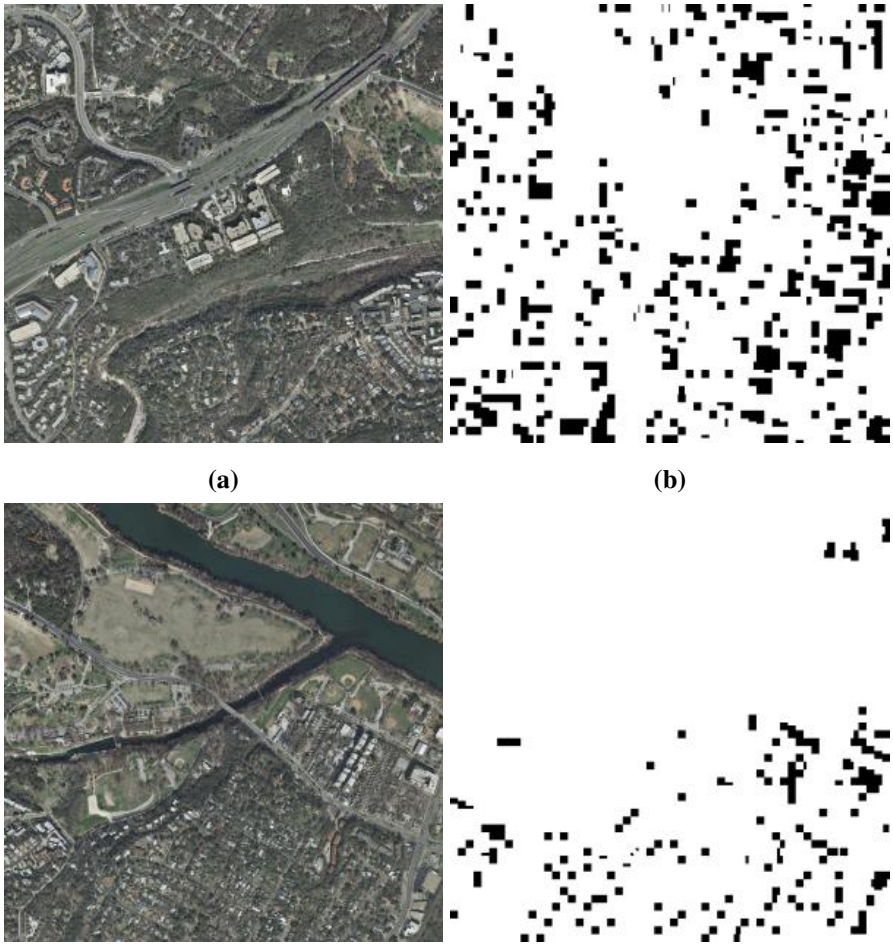
The same as Synthinel, Inria was divided into completely randomised test, train and val datasets consisting of 30, 60 and 10 percent respectively.

8.1. Results

The results proved to be worse than what was the case with Synthinel. Such results should be taken in the context of available hardware not being strong enough for the task at hand.

First of all, every image in the dataset was converted to the size of 2500 by 2500 pixels using bilinear interpolation to downsample the images. Secondly, the number of upsample layers had to be fixed at eight, which is a rather small value. Also, the crop size was reduced to 16, which is not much for images of such size. All that resulted in an IoU of 41.77%, which is substantially lower than the results on Synthinel.

Nevertheless, some limited results were achieved. Some images anayzed proved to be slightly nicer, whereas on the others it is safe to say semantic segmentation failed.



(a) **(b)**

(c) **(d)**

Figure 8.1: Examples of some better and some worse results of semantic segmentation of Inria dataset images with Swiftnet

9. Conclusion

In this work, semantic segmentation and neural networks required for its execution were described.

Convolutional neural network are definitely a way to go in semantic segmentation, and they have probably not reached their full potential yet. Right now, it is safe to say that fully convolutional network with a number of lateral connections will see their use in autonomous vehicles of many kinds very soon.

The chosen architecture for analysis of a dataset, Swiftnet, is a good choice for tasks in semantic segmentation of aerial imagery. Also, we can assume that what is good for semantic segmentation in general, is pretty much as good for the subproblem of aerial imagery segmentation. Hyper - parametres of such a net can be fine-tuned to each different dataset, but as a rule of thumb the hyper - parametres that perform well on one dataset are also optimal for another.

BIBLIOGRAPHY

S. Azimi, C. Henry, L. Sommer, A. Schaumann, i E. Vig. Skyscapes – fine-grained semantic understanding of aerial scenes, 2019.

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, i Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *CoRR*, abs/1802.02611, 2018. URL `http://arxiv.org/abs/1802.02611`.

Ilke Demir, Krzysztof Koperski, David Lindenbaum, Guan Pang, Jing Huang, Saikat Basu, Forest Hughes, Devis Tuia, i Ramesh Raskar. Deepglobe 2018: A challenge to parse the earth through satellite images, June 2018.

S. Gould, R. Fulton, i D. Koller. Decomposing a scene into geometric and semantically consistent regions, 2009. URL `http://robotics.stanford.edu/~koller/Papers/Gould+al:ICCV09.pdf`.

Fanjie Kong, Bohao Huang, Kyle Bradbury, i Jordan Malof. The synthinel-1 dataset: a collection of high resolution synthetic overhead imagery for building segmentation, 2020.

Ivan Kreso, Sinisa Segvic, i Josip Krapac. Ladder-style DenseNets for semantic segmentation of large natural images. U *Proceedings of the IEEE International Conference on Computer Vision Workshops*, stranice 238–245, 2017.

Ivan Krešo, Josip Krapac, i Siniša Šegvić. Efficient ladder-style DenseNets for semantic segmentation of large images. *IEEE Transactions on Intelligent Transportatoin Systems*, 2020.

Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat, i Pierre Alliez. Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark, 2017.

Marin Orsic, Ivan Kreso, Petra Bevandic, i Sinisa Segvic. In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images. U *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, stranice 12607–12616, 2019.

Olaf Ronneberger, Philipp Fischer, i Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. URL `http://arxiv.org/abs/1505.04597`.

Semantic Segmentation of Aerial Imagery

Abstract

Within this thesis semantic segmentation of aerial imagery on the example of Swiftnet - powered Synthinel and Inria datasets segmentation was described. The work looks into technical details of implementaton of neural networks for semantic segmentation. It also examines the different publicly available datasets used for semantic segmentation as well as different approaches close to the state-of-the-art performances. In the end there is a proposal for hyper - parameter fine - tuning of Swiftnet for the given datasets.

Keywords: neural networks, convolutional neural networks, semantic segmentation, aerial imagery, Swiftnet, Synthinel, Inria