FULLY CONVOLUTIONAL NEURAL NETWORKS FOR IMAGE SEGMENTATION

Andrei LEICA [1*]
Mihai Bogdan VOICESCU [2]
Răzvan-Ştefan BRÎNZEA [3]
Costin-Anton BOIANGIU [4]

ABSTRACT

Image segmentation is a Computer Vison process in which an input image is split into different and fully-disjoint parts, which are considered to possess a certain characteristic of interest (they have almost the same color, a resembling texture, they represent the same object inside a scene, etc.). In most scenarios, the key for a successfully image analysis, in which there is required a high-level interpretation of its content, may be found in a correct segmentation, but, unfortunately, in most of the real-life cases, this is a very difficult task. Our method is based on deep learning neural network architectures, which hold state of the art accuracy for pixel-wise segmentation on various challenges. We will design and train different architectures and use all of them together as a voting-based image segmentation system.

KEYWORDS: *Image Segmentation, Convolutional Neural Network, Machine Learning, Deep Semantic Segmentation, Thresholding, Region Growing, Split and Merge*

1. INTRODUCTION

FCN-s or Fully Convolutional Networks [1] are learning models which can output a pixel-wise, dense prediction, and are widely used for various image segmentation tasks. A Fully Convolutional Neural Network is similar to a usual Convolutional Neural Network (CNN), with multiple convolutional layers stacked on top of each other, mixed with nonlinearities (ReLU) and max-pooling layers, but differentiates itself by replacing the final fully connected layer with another convolutional layer having a large *"receptive field"*. The purpose is to analyze the scene in its entirety (the objects found in the image and a rough estimate of their location), in order to get an accurate prediction for each individual pixel.

[1*] corresponding author, Engineer, "Politehnica" University of Bucharest, Bucharest, Romania, leicaandrei@gmail.com
[2] Engineer, "Politehnica" University of Bucharest, Bucharest, Romania, mihai1voicescu@gmail.com
[3] Engineer, "Politehnica" University of Bucharest, Bucharest, Romania, razvan.brinzea@gmail.com
[4] Professor PhD Eng., "Politehnica" University of Bucharest, Bucharest, Romania, costin.boiangiu@cs.pub.ro

This model architecture differs from traditional models because it uses no fully-connected layers, instead relying completely on convolution and upsampling operations. This is because the last fully-connected layers are usually used for classification, thus eliminating any spatial information, which is very important in dense prediction tasks, like the pixel-wise image segmentation. As shown in *Figure 1*, the FCN model first performs many layers of convolution on the image to extract a multiscale feature representation of the image, with the dimension *(Hi, Wi, Ci)*, where *Ci* is the number of channels or kernels.

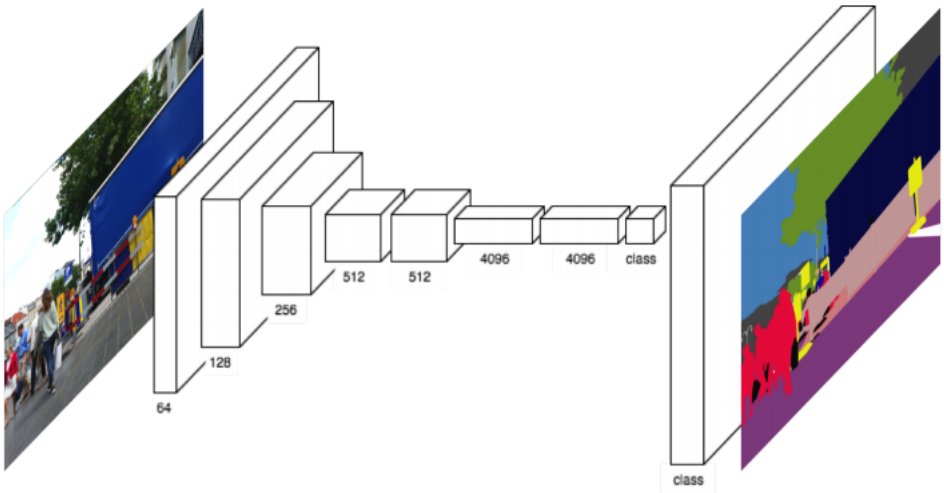

Figure 1. A fully convolutional neural network architecture

The architecture begins with a series of "shallow" layers, which are progressively replaced by "deep" layers. Shallow layers apply a low number of small filters to the image, preserving most of the image's dimensions, while the deep layers are much smaller in height and width, but contain more channels of information. Finally, the last layer performs a transposed convolution that increases the dimensions to (H, W, C0), whose height and width are the same as the input image, with the depth in each pixel being the likelihood that the pixel belongs to each of the C0 classes.

Figure 2 illustrates the actual number of convolutions and max-pooling operations that each layer has, together with the upsampling convolutional layers. The concept of Deconvolution (upsampling or transposed convolution) first appeared in [11] and is a special case of convolution capable of producing larger feature maps from smaller ones, very useful in generative models and dense prediction tasks. This idea was first introduced for segmentation in [12].

Starting with this architecture, we can easily extend our models with different CNN architectures and various type of layers and training procedures, which can be used in a voting manner to get the best accuracy on an evaluation dataset.

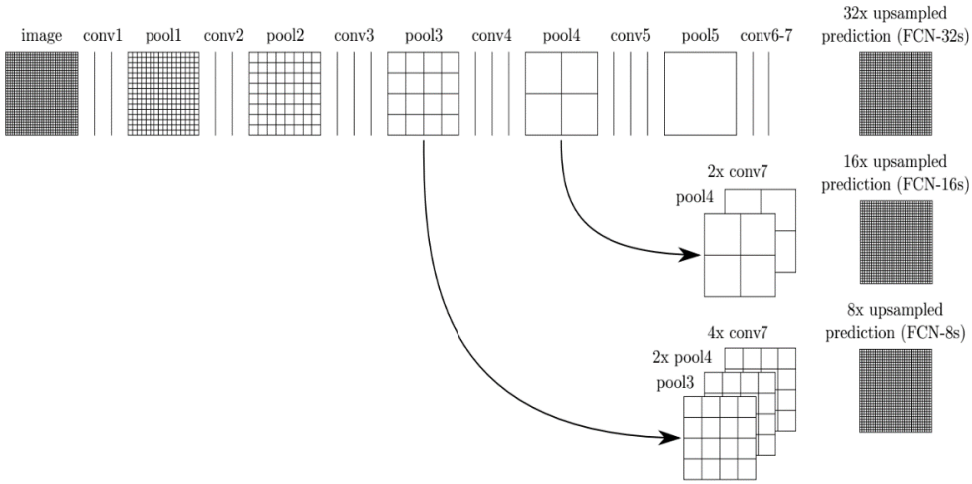

Figure 2. Upsample Convolutions

2. RELATED WORK – CLASSICAL METHODS

In the following paragraphs we describe summarily some classical methods of digital image segmentation, outlined in [5].

Thresholding is the simplest and, perhaps, the most used segmentation technique. In this method, the main idea is to find one (or more) thresholds and to build different segments from the pixels that fall into different ranges when comparing them with the thresholds.

Contextual segmentation: Region growing. Thresholding is a technique that is, somehow, suboptimal, because it groups pixels based on their individual properties without using their vicinity at all. To correct this, some may need to take into account two properties: discontinuity and similarity. A discontinuity-oriented technique will try to find boundaries (as closed as possible) for the objects using variations of the intensity in the image across the previously detected edges. A similarity-based technique, will perform the other way around, trying to group connected pixels that accomplish both individual and group criteria.

Split-and-merge segmentation. The process is presented in Figure 3, and the main data structure employed is the quad tree. The input image is *split* successively into quadrants until "homogeneous" regions are obtained, and the resulting segments are then *merged* together if the same criteria is met for their combined result. The "homogeneity" measure may be defined and used accordingly to the application needs, one may choose to measure the variance of the data in the candidate region, other may find more useful to enforce a maximum allowable difference between maximum and minimum pixel values, other may choose to compute the relation between some region statistics against the same statistics performed on whole image.

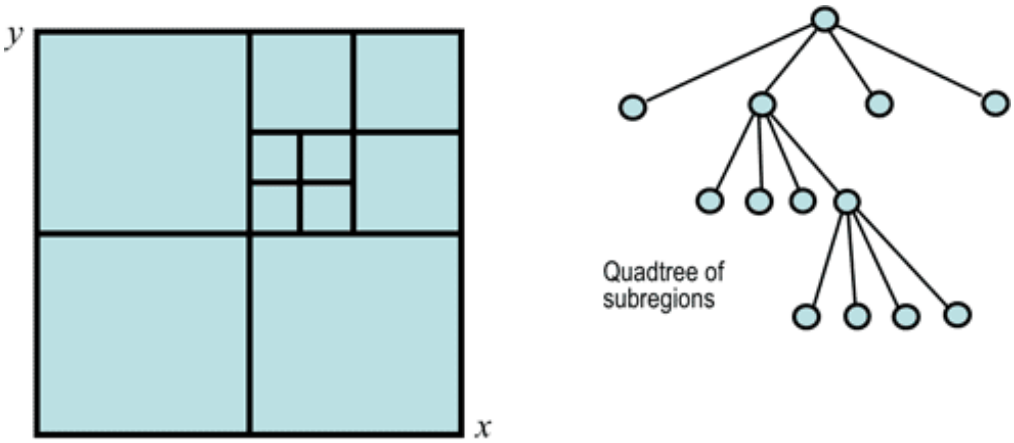

Figure 3. An illustration of the Split and Merge method. The resulting regions can be stored in a quad tree.

Other recent developments include compression-based methods [6, 7], which hypothesize that an optimal segmentation of an image also minimizes the coding length of the data, histogram-based methods, which have been found useful in video tracking [8], due to their efficiency compared to other image segmentation methods, and graph partitioning methods [9, 10], which model the image as a weighted, undirected graph.

3. RELATED WORK - DEEP SEMANTIC SEGMENTATION

Almost all models developed to-date, including those mentioned above, rely on pixel-level linear classification or other methods with limited predictive power, such as decision trees. Neural networks, by contrast, have the potential to learn highly nonlinear decision boundaries and eliminate tedious human feature engineering. Current state-of-the art methods for semantic image segmentation include basic deep convolutional networks (employed by Chen et al. [2] in conjunction with conditional random fields), fully convolutional networks (Long et al. [1]), and deconvolutional networks (Noh et al. [3]). Of great interest are also models originally intended for object detection (which are frequently applicable to segmentation to some extent) such as the R-CNN model developed by Girshick et al. [4].

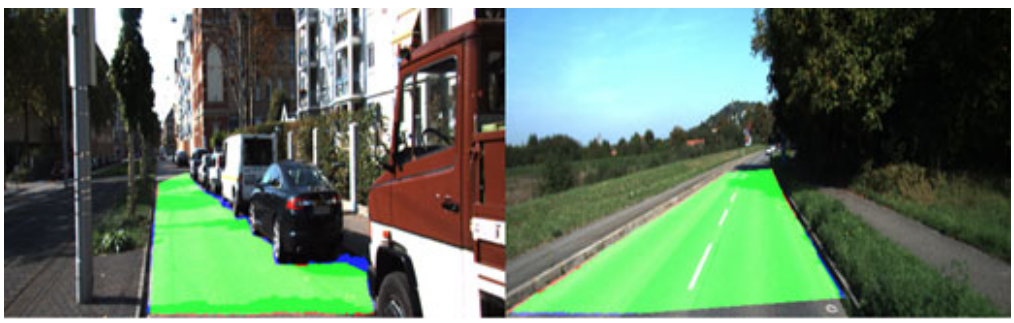

Figure 4. Images from the KITTI Road dataset, with the annotated ground-truth

4. FULLY CONVOLUTIONAL MODEL

Firstly, we focus on an existing fully convolutional neural network architecture, described by Long et al. in [1], and used successfully in image segmentation tasks. We analyze how this model can be applied to solve the difficult task of road and lane detection. This is a very important task for autonomous driving. It is also used is providing pedestrian detection and offering relevant driving assistance.

We trained a model using the KITTI Road dataset [16]. For the training process, ground-truth labels consist of images with the same size as the original input images (1280*380 for KITTI dataset), where each pixel is 1 for road and 0 for non-road area. The network learns to predict a label for each individual pixel, by outputting a probability distribution over the possible pixel values (0/1 in our binary classification task). The network is trained until converges, when the loss (softmax-cross-entropy function) no longer decreases.

There are 289 + 290, training + test images, in the urban data set, classified as below:

- Unmarked (98 + 100, training + test)
- Marked (95 + 96, training + test)
- Multiple marked lanes (96 + 94, training + test)
- Combined (unmarked, marked, multiple marked lanes)

The ground truth data has been obtained using a manual annotation process. It is available for the road area (the collection of all lanes) and the lane for the driving vehicle (marked). Manual annotation is necessary due to ground truth being provided for training images only.

5. RESULTS

Figure 5. Road sections detected in test images

We extended the KITTI training data set, by also including mirrored version of the training images. Consequently, while the original KITTI Road dataset consists of 289 training images, our dataset uses 580 training images. For evaluation, 34 images are used. The FCN architecture is implemented using the TensorFlow open-source machine learning framework.

The following measures were used for the evaluation: Precision, Recall and harmonic mean (F1-measure, $\beta = 1$). The best accuracy obtained with FCN architecture was achieved after 300 epochs, with Precision = 0.967, Recall = 0.920, F1_measure = 0.943. A sample result is shown in figure 5.

6. FURTHER WORK ON FCN

For future work, we will focus our attention on the PASCAL VOC 2011 segmentation training set. A bigger, and perhaps more relevant, collection of training images (Hariharan et al. [13]), will be used to train on models on 90 provided classes, thus moving beyond binary classification (as in the road segmentation problem). Different convolutional neural network architectures must be tested, in order to determine which one provides the most satisfying results.

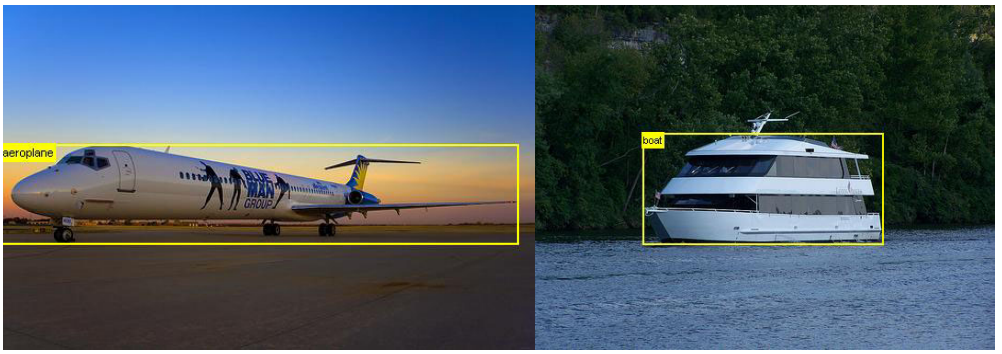

Figure 6. Sample images from the PASCAL VOC 2011 training/validation data

7. INSTANCE OBJECT DETECTION AND SEGMENTATION

In the following, we propose another use of semantic segmentation, by augmenting an Object Detection neural network pipeline with segmentation for individual instances. One of its most popular and useful functions is that of extracting semantic features from high dimensional data, such as 2D streams of data, as is the case with video streams. In the past, people have made intensive, unsuccessful efforts of manually extracting features from 2D images using various techniques. Convolutional Neural Networks (CNN) offer a way of automatically extracting semantically useful features from high dimensional data by enforcing some constraints like the fact that a filter applied to an image should be invariant to its position on the image. The CNN is a Neural Network that contains multiple convolutional layers. A convolution operation is achieved by sliding a kernel over the input and computing the dot product between the input and the filter. The convolved feature is the result of this operation. The use of convolutional networks for

extracting features led to their extensive usage in other areas as well, such as classifying, detecting and segmenting salient objects. These qualities make them essential to our task of detecting objects with the end goal of driver assisted technologies and self-driving cars.

In the following we will try to give an overview of each component of the system. However, a comprehensive and self-contained presentation of each topic used is impossible and beyond the scope of this design document. As such, we redirect the interested reader in search of a more detailed explanation to the huge amount of information available online and throughout books and research papers.

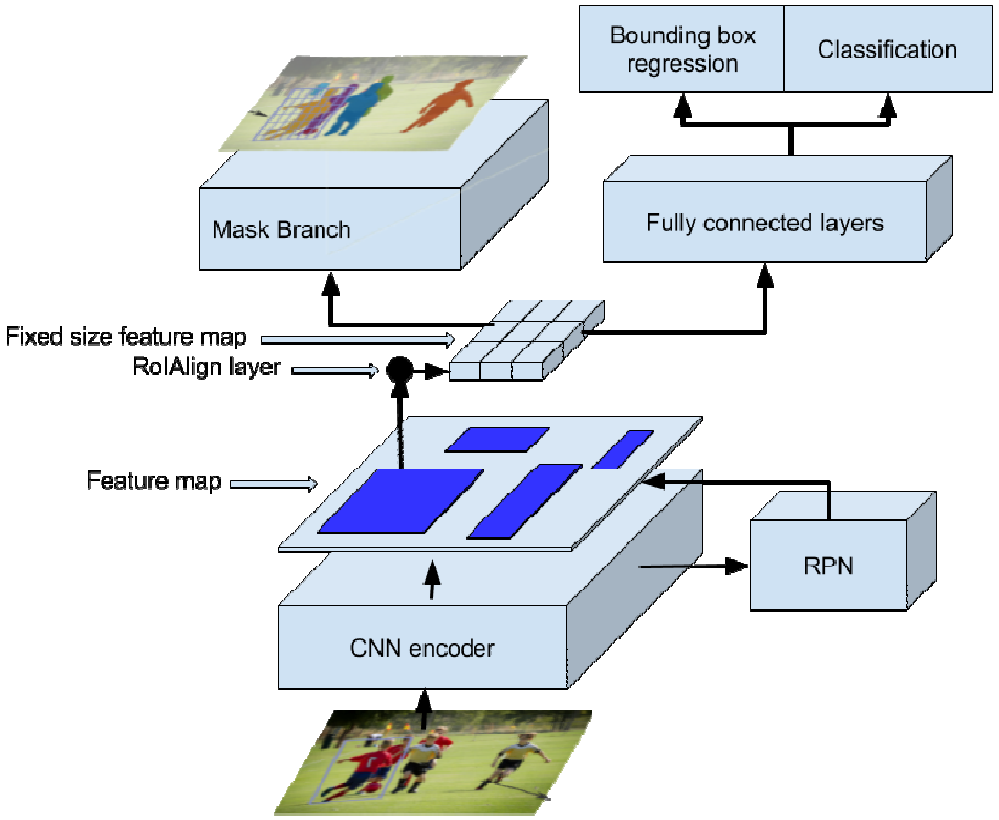

Figure 7. High-level overview of the proposed image segmentation system architecture

The Mask R-CNN [14] network builds upon the Faster R-CNN [15] network with the addition of segmentation, whose purpose is to predict the object mask.

The Faster R-CNN network is critical to the system since it is the most accurate at detecting obstacles and classifying them into classes -- cars, busses, pedestrians, etc. We shall refer specifically to Faster R-CNN as the variant from the Mask R-CNN system.

The system's architecture consists of a FCN (fully convolutional network) which acts as the backbone of the system and maps an image given as input into a lower-dimensional representation. This feature map is passed to the RPN (region proposal network), whose

job is to propose ROIs (regions of interest) for the network to analyze as potential candidates for the presence of objects.

These candidates are scored based on their projection quality onto original ground truth bounding boxes.

The network then crops out the feature map according to these proposals given by the RPN using the RoIAlign layer. The RoIAlign layer is a quantization-free layer that accurately keep the spatial locations of the features.

The crops are then resized to fix dimensions and passed to the 3 parallel branches that form the head of the network: the bounding box regression branch, the bounding box classification branch and the mask segmentation branch. The roles of each of these branches are the following:

- The bounding box regression branch - outputs deviations from the original positions of the region of interest detected by the RPN, fine-tuning the exact location of the box such that it faithfully covers the object of interest.
- The bounding box classification branch - predicts the class score of the object in question with respect the existing classes the network is training on.
- The mask segmentation branch - come up with individual instances and pixel-wise segmentation masks for each object instance.

If the tasks are complementary, then sharing the encoder can be beneficial in allowing the tasks to achieve a higher score than without sharing the convolutional encoder backbone.

At inference time, each of the tasks outputs its corresponding results which are subsequently applied to the original image for visualization purposes.

7.1. Design details

The system architecture is based upon the following elements:

- **Fully Convolutional Neural Network (FCN)** - this is the shared encoder between the three branches of the architecture (presented as the first architecture). It is responsible for transforming a high dimensional input, i.e. an image to a lower dimensional condensed representation of the image, whose space is better for class separation, and boundary detection. The result of this stage is a feature map with dimensions [w, h, c], w being the weight, h the height and c the number of filters. The following image illustrates the output of this stage.

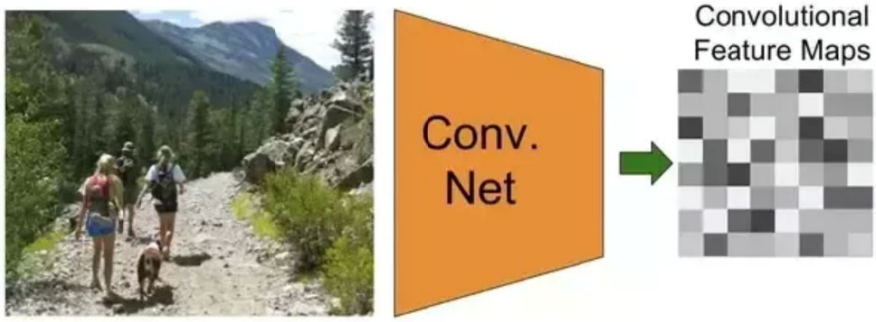

Figure 8. An example feature map derived by a Convolutional Neural Network

- **Region Proposal Network (RPN)** employs two convolutional layers with a 3x3 kernel. Their purpose is to detect those regions for which it is possible to find an object in the feature map. For each convolution position on the image a set of 9 anchors having three aspect ratios, three scales and sharing a common center. The process is depicted below. For each position in the image and for each anchor box a regressor outputs the predicted bounding box (x, y, w, h) and a classifier probability *p* that an object is inside the predicted box. Both parallel heads are trained using the original ground truth boxes and the probability threshold p uses the IoU (intersection over union metric)

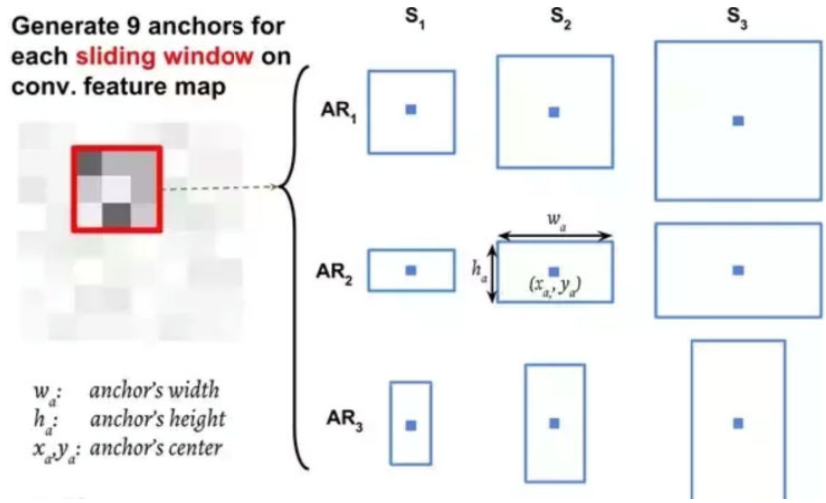

Figure 9. Anchors generated by the Region Proposal Network

- **The RoIAlign layer** - is used to crop a region of interest from the feature map using bilinear interpolation to compute the pixel values, without performing any rounding to the values and thus achieving perfect alignment between the feature map resulted from the crop_and_resize operation and the actual position in the original image. This process is illustrated in greater detail below.

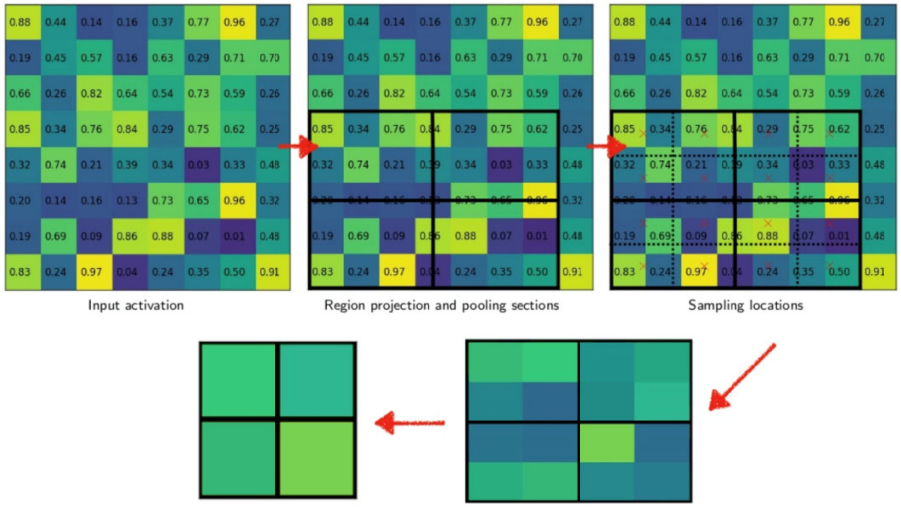

Figure 10. Bilinear interpolation performed by the RoIAlign layer

- **Bounding Box Regression Branch** - uses fully connected layers to output 4 bounding box regression coordinates (x, y, w, h) for each class.
- **Bounding Box Classification Branch** - uses fully connected layers to output a class score for each of the possible classes of the network.
- **Mask Segmentation Branch** - uses convolutional layers to output, for each class, a dense prediction for each extracted feature map, thus labeling each pixel according to its predicted membership to that class.

The system is constructed using the TensorFlow deep learning framework developed by Google, using python and numpy as additional tools. TensorFlow offers automatic gradient computation and backpropagation, thus manually implementing the training procedure is unnecessary. Training and testing require computational resources supplied by the use of multiple GPUs, specifically NVIDIA GTX 1080TI. The computations are sped up by using NVIDIA's CUDA toolkit and associated cudnn library.

ACKNOWLEDGEMENT

This work was supported by a grant of the Romanian Ministry of Research and Innovation, CCCDI - UEFISCDI, project number PN-III-P1-1.2-PCCDI-2017-0689 / „Lib2Life-Revitalizarea bibliotecilor si a patrimoniului cultural prin tehnologii avansate" / "Revitalizing Libraries and Cultural Heritage through Advanced Technologies", within PNCDI III.

REFERENCES

[1] E. Shelhamer, J. Long, and T. Darrell. "Fully Convolutional Networks for Semantic Segmentation". IEEE Pattern Analysis and Machine Intelligence (PAMI), May 2016.

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. "Semantic image segmentation with deep convolutional nets and fully connected CRFs". arXiv preprint arXiv:1412.7062, 2014.

[3] H. Noh, S. Hong, and B. Han. “Learning deconvolution network for semantic segmentation”. In Proceedings of the IEEE International Conference on Computer Vision, pages 1520– 1528, 2015.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 580–587, 2014

[5] Nick Efford. “Digital Image Processing: A Practical Introduction Using JavaTM”. Pearson Education, 2000.

[6] Hossein Mobahi, Shankar Rao, Allen Yang, Shankar Sastry and Yi Ma. “Segmentation of Natural Images by Texture and Boundary Compression”. International Journal of Computer Vision, volume 95, pages 86–98, 2011

[7] Shankar Rao, Hossein Mobahi, Allen Yang, Shankar Sastry and Yi Ma. "Natural Image Segmentation with Adaptive Texture and Boundary Encoding". In Proceedings of the 9th Asian conference on Computer Vision - Volume Part I (ACCV'09), Hongbin Zha, Rin-ichiro Taniguchi, and Stephen Maybank (Eds.), Vol. Part I. Springer-Verlag, Berlin, Heidelberg, pages 135-146, 2011

[8] P. Dunne and B. J. Matuszewski, “Histogram Based Detection of Moving Objects for Tracker Initialization in Surveillance Video”, International Journal of Grid and Distributed Computing, vol. 4, no. 3, pages 71-78, 2011.

[9] Jianbo Shi and Jitendra Malik, “Normalized Cuts and Image Segmentation”. IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 22, no. 8 (August 2000), pages 888-905. DOI=http://dx.doi.org/10.1109/34.868688, 2000

[10] Leo Grady, "Random Walks for Image Segmentation", IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 28, no. 11, pages 1768–1783, 2006

[11] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, pages 2528–2535. IEEE, 2010.

[12] F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions” published as a conference paper at ICLR 2016, arXiv:1511.07122, 2016

[13] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik, “Semantic contours from inverse detectors”, 2011 IEEE International Conference on Computer Vision (ICCV)

[14] Kaiming He, Georgia Gkioxari, Piotr Doll, Ross B. Girshick, “Mask R-CNN”, CoRR, abs/1703.06870, 2017

[15] Shaoqing Ren, Kaiming He, Ross B. Girshick, Jian Sun, “Faster {R-CNN:} Towards Real-Time Object Detection with Region Proposal Networks”, CoRR, abs/1506.01497, 2015

[16] KITTI Road dataset, URL: http:// www.cvlibs.net/ datasets/ kitti/ eval_road.php, Available: March 1, 2018.