Ship detection using different segmentation methods

Ákos Gángoly, Gergely Kovács, Zoltán Gyenes

***Abstract*—Nowadays deep learning is one of the most important research topics in the science life. Image segmentation plays a big role in the research topic because of the big number of the applications that use this method. There are different types of segmentation techniques. Our goal was to implement and compare different types of segmentations solving a ship detection problem named Airbus Ship Detection Challenge created by Kaggle. We tried transfer learning on VGG-19 and SegNet as models.**

***Index Terms*—deep learning, image segmentation, transfer learning, segnet**

I. Introduction

In the field of computer vision semantic segmentation is one of the key problems and it is applied both at images and videos. The most important goal is always to understand the complete scene. Because of the growing number of the computer vision techniques there was an increase of the applications that use the imagery knowledge from the neighborhood. There are several applications to this topic, e.g.: autonomous driving [1], [2], [3] or image search engines [4].

Because of the deep learning revolution, the old methods are not useable any more. At the segmentation problems usually the Convolutional Neural Networks (CNNs) are widely used [5], [6], [7], [8], because of the efficiency. In the past there were some articles to summarize the semantic segmentation techniques [9], [10] but the most expressive and most extensive work was considered the most recent datasets and provided details about deep learning techniques [11].

Our project aims to create a neural network to detect ships on satellite image. This task is important to track traffic and increase security against pirates.

There are many solutions for object detection on image, but they were not yet implemented on satellite images to detect ships. Previous solutions used classic computer vision techniques. Instead of this we implemented segmentation techniques to solve this problem.

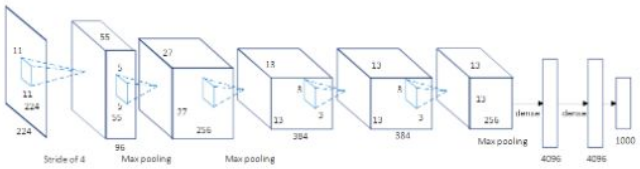
The rest of this paper is going to be organized in the following order: Section II introduces the recently common used segmentation techniques. After that Section III. presents the Dataset that was used to the task. Section IV In Section V. the conclusion of the work will be presented, and the opportunities of the further development will be defined.

II. Related Work

As it was mentioned, there are a lot of opportunities that were defined in the past to use the segmentation techniques [11]. In this section there will be overview about the AlexNet, VGG-16, GoogLeNet and ResNet methods.

A. AlexNet

The architecture of AlexNet was defined by Krizhevsky [12]. It consists of five convolutional layers: max-pooling ones, Rectified Linear Units (ReLUs) as non-linearities, three fully-connected layers and dropout as you can see in II-1. Figure.



II-1. Figure: Architechture of AlexNet [12]

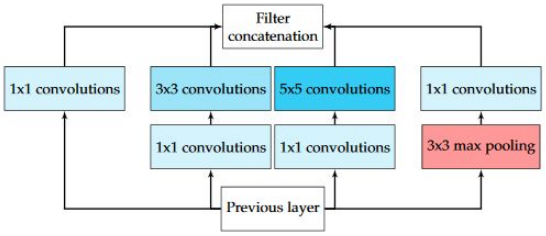
The AlexNet won the ILSVRC-2012 with on accuracy of 84.6 % against 73.8 %.

B. VGG

A group called VGG (Visual Geometry Group) from the Oxford university introduced a new CNN model called Visual Geometry Group (VGG). [13]. They submitted a model for ILSVRC-2013. This model had 16 weight layers (VGG-16) and it became very popular because of the achievement of 92.7 %. The main causes of the success were the stack of convolutional layers instead of few layers with big receptive fields.

C. GoogLeNet

GoogLeNet nework won the LSVRC-2014. It was introduced by Szegedy [14]. This CNN architecture is characterized by its complexity, emphasized by the fact that it is composed by 22 layers and a newly introduced building block called inception module, it is shown in II-2. Figure.

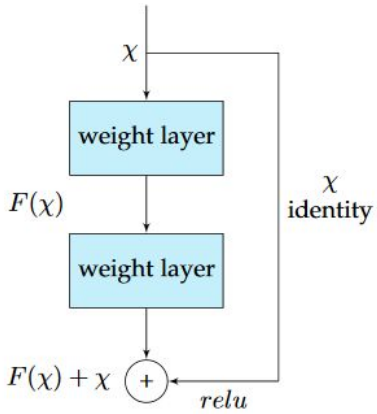


II-2. Figure: Architecture of GooLeNet

D. ResNet

Microsoft has introduced a new network called ResNet [15], and this network won the LSVRC-2016 with 96.4 % accuracy. The main difference of this network compared with the previous one was the depth of the network (152 layers) and the

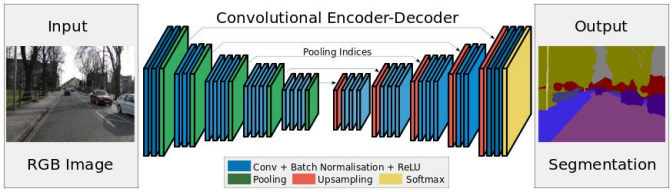
introduction of residual blocks (see in II-3. Figure). “The intuitive idea behind this approach is that it ensures that the next layer learns something new and different from what the input has already encoded (since it is provided with both the output of the previous layer and its unchanged input).”[11]



II-3. Figure: Architecture of ResNet

E. SegNet

SegNet is a convolutional neural network used for semantic image segmentation, which was introduced by Vijay Badrinarayanan, Alex Kendall and Roberto Cipolla in 2016. “The novelty of SegNet lies is in the manner in which the decoder upsamples its lower resolution input feature map(s). Specifically, the decoder uses pooling indices computed in the max-pooling step of the corresponding encoder to perform non-linear upsampling. This eliminates the need for learning to upsample.”[8]



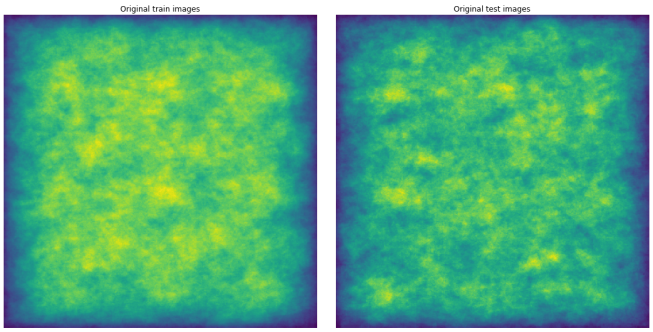
II-4. Figure Architecture of SegNet

III. DATASETS, DATA PREPROCESSING

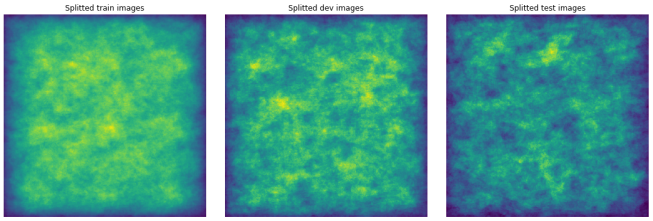
The dataset we used, was provided by Kaggle for the Airbus Ship Detection Challenge and was separated to a training and test dataset. Both of them consisted of numerous images and a .csv file. The two .csv files were used to describe the metadata of the images, namely the name of the given image and the areas of the images, which represented part of a ship. If there were multiple ships on an image, multiple lines were reserved for them in the .csv file, one line for every ship. The ship areas were given in the following form: a list of pixels, which were first pixel of a connected ship area and the size of the following connected ship area (the pixels were taken from top left to right and down, an area was connected as long as all following pixels were pixels of a ship as well).

The data was published in two parts. We have merged and split again into 3 dataset with rations 0.7-0.2-0.1

We explored the datasets and visualized them in order to understand them better and to be able to use them properly. We visualized the heatmap of ship occurrences for the original training and test set, and the newly created training and validation dataset as well. The conclusion from these images were, that the ship occurrences were aptly distributed considering the sizes of the different datasets. Furthermore, we saw, that the images didn’t contain many ships near the borders of the images.

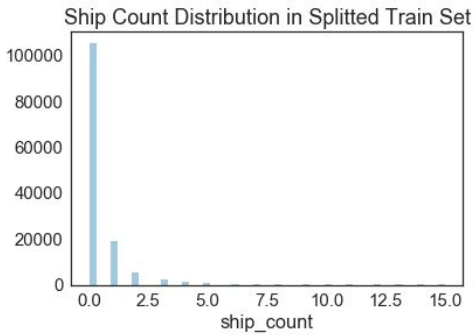


III-1. Figure Ship occurrence heatmap for original training and test dataset



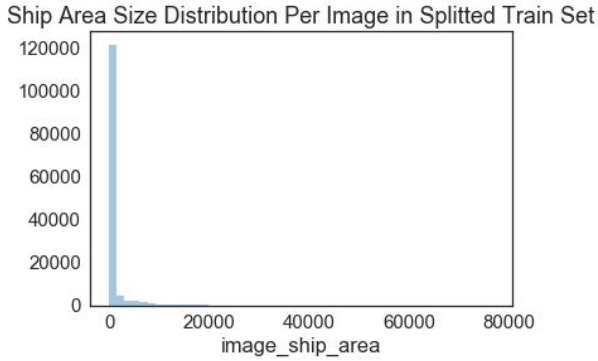
III-2. Figure Ship occurrence heatmap for redistributed training, validation(dev) and test dataset

We also explored the different statistics of the datasets: the number of ships per image and the area size of ships per image and per ship. We did this for all datasets. We calculated the area sizes both with all images and only with images containing any ships. We concluded, that the distribution of the area sizes (all versions) and the number of ships both were similar comparing the datasets, so again, the distribution of the original datasets was properly done.

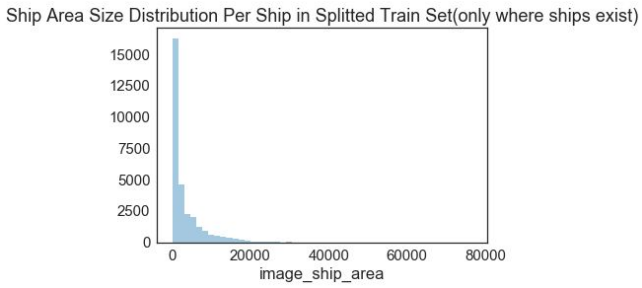


III-3. Figure Distribution of number of ships per image

Regarding the ship count, we were also able to mark, that the majority of all images did not contain any ships, which was reflected in the ship area size distribution as well, which proved to be an important feature of the dataset. That's why we also calculated the distributions without the ships containing no ships. Though the distribution changed, the peak of it still remained strongly at the near of zero, which meant, that most of the ships are very small, and most of the images only contain a small area of ships.



III-4. Figure Distribution of ship area size per image with images containing no ships



III-5. Figure Distribution of ship area size per image only for images with ships

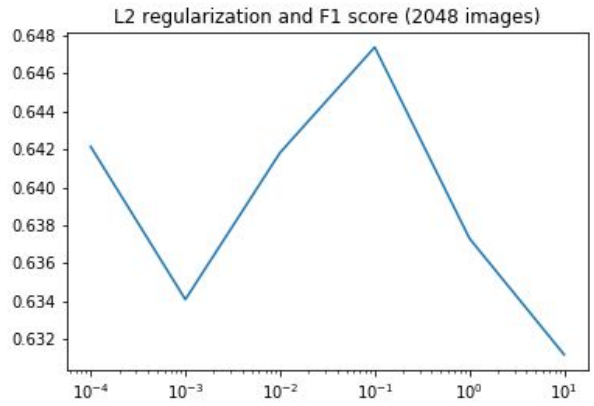
We also preprocessed the data to have the masks in a matrix format, one element of a matrix had the value 0 if it wasn't part of a ship, 1 if it was. Because of the large size of the dataset, we used generators to feed the data for training, validation and evaluation. We used different preprocessings for different models, we use binary masks and 0-1 scaled input image for SegNet. For VGG-19 we subtract the mean from the input, the output is the same.

IV. Methods and solutions

Our first approach is to use transfer learning for VGG-19, by remove the last fully connected layers, and replacing it with one convolutional layer, so the output is equal with the number of pixels on the input. The first idea was to use fully connected layer at the end, but since VGG-19 uses 224*224 images, the resulting matrix would have 224^4=2517630976 values. If we use 32 bit floats, that would have meant 9.38GB of memory requirement only for the last layer just for inference.

By simpli letting the model to learn on all the data resulted in the model learning to output 0 all the time. So we have introduced an iterative learning technique. The point is that, we order the data, based on the size of the ships on the image. So the images with the largest ships will be at the beginning of the dataset. Than we take the top (let's say 1024) images and let the model train on it using early stopping. Once the model has finished training we increase the data size to 2048. We continue this until all the data is in use. This process gave us an 0.7098 F1 score on the training set and 0.4963 on the dev set.

Seeing the gap between the training and dev performance I have decided to try L2 regularization. So I have tried various values on a 2048 subset and got the following plot:



As you can see 1e-1 seems to be ideal. So I train the model with the hole dataset with this value. This gave me 0,50. To try to further reduce overfitting a gave gaussian noise to the image with std 1 and 5. This gave me 0.519 and 0.53 respectively.

We also tried solving the problem using the model of SegNet. The basic architecture consisted out of the same amount and type of layers, which is presented in the original (shown in II-4. Figure): for the encoder 2-2-3-3-3 layers of convolutional-batch normalization-activation layer group, containing max pooling layers between them. The decoder was somewhat the opposite, 3-3-3-2-2 layers of the same

groups, but containing upsampling layers between them. Both the encoder and decoder contained relu activation. Finally at the end we used a convolutional-batch normalization-activation group with softmax activation creating 2 outputs for the 2 class prediction probabilities respectively. For the upsampling we used the help of TensorFlow function tf.nn.max_pool_with_argmax to recreate the pooling used in the original SegNet. We also tried a modified version of this network, reforming it to a residual neural network, using the Add layer of Keras. For the whole implementation we used the code found at https://github.com/ykamikawa/SegNet and modified it.

Unfortunately this model is extremely slow to train. On a Tesla K80, it took about 6 hours to train one epoch of 16384. Due to time limitations it was not possible to train the model longer. However it has produced some interesting promising results. With 0.9881 f1 score on the training set and 0.9813 on the dev set. Note in this case the dev set is 20% of the 16384 images, not the hole dataset.

V. Conclusion, further work

A simple VGG-19 network with convolutions at the and is fast to train, but did not gave very good results. SegNet is more promising and further development could result in a really useful model.

References

[1] A. Ess, T. M¨uller, H. Grabner, and L. J. Van Gool, “Segmentation-based urban traffic scene understanding.” in BMVC , vol. 1, 2009,p. 2.

[2] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for au-tonomous driving? the kitti vision benchmark suite,” in 2012 IEEE Conference on Computer Vision and Pattern Recognition , June 2012, pp. 3354–3361

[3] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3213–3223.

[4] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li,“Deep learning for content-based image retrieval: A comprehen-sive study,” in Proceedings of the 22nd ACM international conference on Multimedia . ACM, 2014, pp. 157–166.

[5] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber,“Deep neural networks segment neuronal membranes in electron microscopy images,” in Advances in neural information processing systems , 2012, pp. 2843–2851.

[6] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, “Learning hierarchical features for scene labeling,” IEEE transactions on pattern analysis and machine intelligence, vol. 35, no. 8, pp. 1915–1929, 2013.

[7] B. Hariharan, P. Arbel aez, R. Girshick, and J. Malik, “Simulta-neous detection and segmentation,” inEuropean Conference on Computer Vision . Springer, 2014, pp. 297–312.

[8] Vijay Badrinarayanan, Alex Kendall, Roberto Cipolla: SegNet: A Depp Convolutional Encoder-Decoder Architecture for Image Segmentation, arXiv:1511.00561v3 [cs.CV] 10 Oct 2016

[9] H. Zhu, F. Meng, J. Cai, and S. Lu, “Beyond pixels:A comprehensive survey from bottom-up to semantic image segmentation and cosegmentation,”Journal of Visual Communication and Image Representation , vol. 34, pp. 12 –27, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1047320315002035

[10] M. Thoma, “A survey of semantic segmentation,”CoRR , vol.abs/1602.06541, 2016. [Online]. Available: http://arxiv.org/abs/1602.06541

[11] A. Garcia-Garcia, S. Orts-Escolano, S.O. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez: A Rewiew on Deep Learning Techniques Applied to Semantic Segmentations. arXiv:1704.06857v1 [cs.CV] 22 Apr 2017, online Available: https://arxiv.org/pdf/1704.06857.pdf

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in neural information processing systems, 2012, pp. 1097–1105

[13] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint arXiv:1409.1556 , 2014.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov,

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.