A Deeply-Supervised Deconvolutional Network for Horizon Line Detection

Lorenzo Porzi
Fondazione Bruno Kessler
Trento, Italy
University of Perugia
Perugia, Italy
porzi@fbk.eu

Samuel Rota Bulò
Fondazione Bruno Kessler
Trento, Italy
rotabulo@fbk.eu

Elisa Ricci
Fondazione Bruno Kessler
Trento, Italy
University of Perugia
Perugia, Italy
eliricci@fbk.eu

ABSTRACT

Automatic skyline detection from mountain pictures is an important task in many applications, such as web image retrieval, augmented reality and autonomous robot navigation. Recent works addressing the problem of Horizon Line Detection (HLD) demonstrated that learning-based boundary detection techniques are more accurate than traditional filtering methods. In this paper we introduce a novel approach for skyline detection, which adheres to a learning-based paradigm and exploits the representation power of deep architectures to improve the horizon line detection accuracy. Differently from previous works, we explore a novel deconvolutional architecture, which introduces intermediate levels of supervision to support the learning process. Our experiments, conducted on a publicly available dataset, confirm that the proposed method outperforms previous learning-based HLD techniques by reducing the number of spurious edge pixels.

Keywords

Horizon detection; Convolutional Neural Network; Edge detection

1. INTRODUCTION

In the last decade several research efforts [10, 12] have been made to automatically analyze pictures of natural scenes. Mining images depicting natural landscapes is of interest in many applications, such as augmented reality, image retrieval and robot navigation. In this context, the analysis of pictures of mountainous landscapes represents an inherently challenging task. Indeed, the large variability of environmental and illumination conditions requires robust computer vision and image processing algorithms. Moreover, often traditional methods do not suffice and *ad hoc* solutions need to be developed (*e.g.* standard keypoints detectors and bag-of-words techniques are not appropriate for matching mountain profiles [3, 2]).

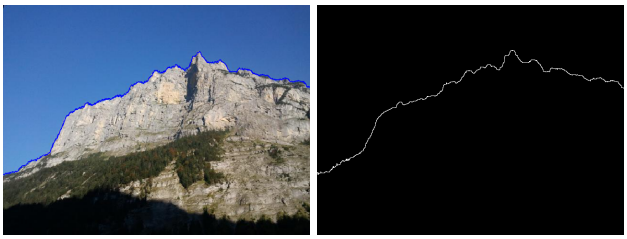

Figure 1: (left) Picture depicting a mountainous landscape and (superimposed) the horizon line detected with the proposed approach. (right) Ground truth.

Recently, the overwhelming success of deep learning architectures [17] has opened new opportunities for research on natural scene analysis. In particular, Convolutional Neural Networks (CNNs) have been effectively applied to many tasks, such as scene recognition [21], prediction of visual attributes of places [10] and semantic segmentation [11]. Similarly, CNN architectures have been shown to be particularly effective in addressing low level tasks, such as contour detection [19, 8]. As demonstrated by previous works [1, 13], learning-based techniques for contour detection are significantly more accurate than traditional filtering methods and can be successfully used for the analysis of mountains images. In particular, the effectiveness of CNN-based solutions to detect mountains profiles have been shown in [1].

Inspired by these recent successes, in this paper we propose a novel approach for detecting the horizon line in mountainous pictures. Similarly to previous works [1], our approach is based on two-steps. First, a learning-based contour detection algorithm is applied to compute the boundary between sky and non-sky regions. Then, this initial estimate is refined via dynamic programming to find the most likely horizon line. Differently from previous methods, a novel neural network architecture is proposed to extract mountain profiles, which successfully combines ideas from recent deconvolutional [11] and deeply-supervised [9] nets. Our experimental evaluation, conducted on the publicly available CH1 dataset [14], demonstrates that our approach outperforms state-of-the-art, learning-based methods for HLD. Fig.1 shows a sample mountain image and the horizon line detected by our method.

MM '16, October 15-19, 2016, Amsterdam, Netherlands

DOI: http://dx.doi.org/10.1145/2964284.2967198

The contribution of this paper is twofold. First, we demonstrate that deep deconvolutional networks can be effectively applied to the HLD task, outperforming previous learning-based solutions. Second, we show that by supporting deconvolutional networks with deep supervision, more accurate detections can be achieved compared to standard deconvolutional architectures with single supervision.

The paper is structured as follows. Section 2 discusses previous works on the analysis of mountains pictures and on learning-based approaches for contour detection. Section 3 introduces our deconvolutional architecture for horizon line detection, and in Section. 4 we report the results of our experimental evaluation. Finally, Section 5 outlines the main contributions of this paper and discusses future work.

2. RELATED WORKS

Most previous works on the automatic analysis of mountain pictures tackled the problem by designing robust methodologies for extracting peaks' profiles. Several works considered publicly available digital elevation maps (DEM), and proposed different solutions to the image-to-DEM matching problem. In fact, aligning images to synthetic profiles represents an essential step for mountain recognition and orientation identification. Baboud *et al.* [3] used edge maps computed with the Compass edge detector and a robust matching metric to calculate the rotation mapping an image to the terrain. As traditional edge detectors provides noisy estimates of mountain profiles, more recent works exploited learning-based solutions. Porzi *et al.* [13] proposed a two-steps method for detecting mountain profiles. First, the Sobel filter is applied to find putative pixels which belong to mountains edge. Then, a classifier based on Random Ferns is used to verify these hypothesis. Recently, Ahmad [1] exploited both CNN and SVM classifiers to detect pixels belonging to the horizon line. The initial estimates obtained with the learned classifier are then refined using a dynamic programming technique. A closely-related problem to HLD, *i.e.* sky segmentation for automatic picture geo-localization, was tackled in [14, 2]. Similarly, Tao *et al.* [16] focused on the identification of the sky seen in an image and proposed Skyfinder, a system to search for images with a specific sky appearance. Among previous works, the most related to ours is [1]. Similarly to [1], our approach is based on deep learning. However, our proposed network architecture is radically different from the one described in [1] and, as demonstrated in our experiments, significantly more accurate in detecting skyline pixels.

Learning-based edge detection has been explored in several previous works [5, 6, 18, 19, 8]. Dollar *et al.* [5] used a boosting-based approach to independently classify each pixel from its surrounding image patch. More recently in [6] improved results were obtained considering a Structured Random Forest classifier, such as to take into account correlations between adjacent output pixels. Ren *et al.* [18] proposed an approach based on sparse coding, which, however, suffers from high computational cost. Particularly relevant to our work are recent approaches using CNN architectures for detecting contours [19, 8].

Inspired by recent successful approaches for semantic segmentation [11], we propose a deconvolutional network, which however distinguishes from the other approaches by integrating a deeply-supervised training strategy [9, 19].

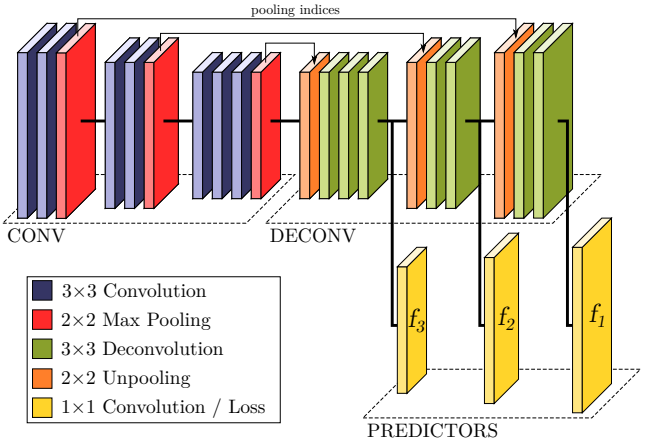

Figure 2: Schematic depiction of the proposed network architecture.

3. METHOD

A horizon line detector is a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ that takes an image $X \in \mathcal{X}$ and returns a probability mask $Y \in \mathcal{Y}$ having the same size of the input image, each entry providing the probability of belonging to the horizon line. Accordingly, the input space $\mathcal{X}$ consists of h × w rgb images, *i.e.* $\mathcal{X} = \mathbb{R}^{\mathrm{h}\times\mathrm{w}\times 3}$, while the output space is given by h × w probability masks, *i.e.* $\mathcal{Y} = [0,1]^{\mathrm{h}\times\mathrm{w}}$.

3.1 Network architecture

In this paper we propose to model the detector h as a deep neural network that combines convolutional layers with deconvolutional layers[20] to produce the *entire* detection mask in output given the entire image in input. The actual topology that we consider is shown in Fig. 2 and described below.

CONV block. The first block of layers denoted by "CONV" is the well-known VGG network [15] truncated at the 3rd max pooling layer. This part of the network is composed by convolutional layers with RELU activation and max pooling layers arranged as depicted in Fig 2.

DECONV block. The second block of layers denoted by "DECONV" is composed by deconvolutional layers with RELU activation and unpooling layers (see Fig. 2 for the actual arrangement). Following the approach of [20], we use the locations of the maxima in the pooling layers to place the outputs of the corresponding unpooling layers. In the figure, this is denoted by arrows connecting each pooling layer in the "CONV" block to the corresponding unpooling layer in the "DECONV" block.

PREDICTORS block. Instead of considering a single output layer, we add also intermediate outputs in the deconvolutional block, denoted as "PREDICTORS" in Fig. 2, in order to provide a deeper supervision during the training phase. This positively impacts both the accuracy of the method, by introducing an implicit regularization, and the convergence of the network during the training phase [9]. Accordingly, the network incorporates three predictors that we denote by $f_j : \mathcal{X} \rightarrow \mathcal{Y}_j$, $j = 1, 2, 3$. The output dimensionality of each predictor is in general lower than the input dimensionality, excepting the last one. This motivates the use of the special notation $\mathcal{Y}_j$ to represent the output space of the jth predictor. Each output layer is composed by a

1×1 convolutional layer with a sigmoid activation function in order to produce probabilistic predictions for the output mask entries. The predictors are numbered in reversed order such that f_1 becomes the predictor that comprises the entire network (see Fig. 2).

3.2 Training

We train our network from labeled images $\{(X_i, Y_i)\}_{i=1}^{\mathrm{n}}$, where $X_i \in \mathcal{X}$ is the ith training image with the corresponding ground-truth mask given by $Y_i \in \mathcal{Y}$. Our training objective is the minimization of the empirical risk $R(\theta)$ given below with respect to the parameters of the network θ:

$$R(\theta) = \frac{1}{3\mathrm{n}} \sum_{i=1}^{\mathrm{n}} \sum_{j=1}^{3} L(f_j(X_i;\theta), Y_i \downarrow_{\mathcal{Y}_j}) \,. \tag{1}$$

The empirical risk has separate loss terms for each predictor f_j. Due to the different dimensionality between the ground-truth mask $Y \in \mathcal{Y}$ and the mask predicted by f_j, we project Y into $\mathcal{Y}_j$ by downsampling Y to the proper size before evaluating the loss function. This operation is denoted by $Y_i \downarrow_{\mathcal{Y}_j}$ in (1). The loss function $L(\hat{Y}, Y)$ that measures the discrepancy between a predicted mask $\hat{Y} \in \mathcal{Y}$ and a ground-truth mask $Y \in \mathcal{Y}$ is given in terms of the following weighted cross-entropy:

$$L(\hat{Y}, Y) = -\sum_{u,v} [\beta_1 y_{uv} \log \hat{y}_{uv} + \beta_0 (1 - y_{uv}) \log(1 - \hat{y}_{uv})] \,. \tag{2}$$

Here, the sum runs over all elements of Y, each denoted by y_{uv} (and similarly $\hat{y}_{uv}$ for $\hat{Y}$), β_0 is the fraction of edges in Y, while β_1 is the fraction of non-edges in Y. The same balancing scheme of the importance of edge/non-edge terms has been used in [19].

The minimization of the empirical risk is carried out via Stochastic Gradient Descent (SGD) with mini-batches and weight decay. More details about the parametrization of the optimizer are given in the experimental section.

3.3 Horizon detection

Once the network has been trained there are different ways to combine the three predictors f_1, f_2, f_3 to obtain the final horizon detector h. The one that is more consistent with the proposed empirical risk simply averages the outcome of the different predictors, by taking care of projecting the result to the right dimensionality first, *i.e.*

$$h_{\mathrm{avg}}(X) = \frac{1}{3} \sum_{j=1}^{3} f_j(X;\theta^*) \uparrow_{\mathcal{Y}} \,, \tag{3}$$

where θ^* is the parametrization of the network obtained after the training procedure, and $\cdot \uparrow_{\mathcal{Y}}$ is the upsampling operator that projects the preceding argument into $\mathcal{Y}$. In addition to h^{avg} we evaluate also the performance of the single predictors. For each $j = 1, 2, 3$ we denote by h_j the horizon detector constructed from f_j, which is given by

$$h_j(X) = f_j(X;\theta^*) \uparrow_{\mathcal{Y}} \,. \tag{4}$$

Post-processing. As a final step, we post-process the output of the horizon detector from the neural network, which is a probabilistic mask, to obtain a more accurate estimate of the horizon curve. Following [1], we construct a weighted graph on the elements of the horizon detector's output and find the final horizon curve by solving a shortest path problem. Differently from [1], we connect each node (u, v) to the nodes $\{(u', v+1) \mid u' \in \{u-3, \ldots, u+3\}\}$, in order to account for steeper mountain profiles. Furthermore, we do not perform the "node reduction" step. The final output is a sequence of detected horizon points $P_v, v = 1, \ldots, \mathrm{w}$, one for each column of the input image.

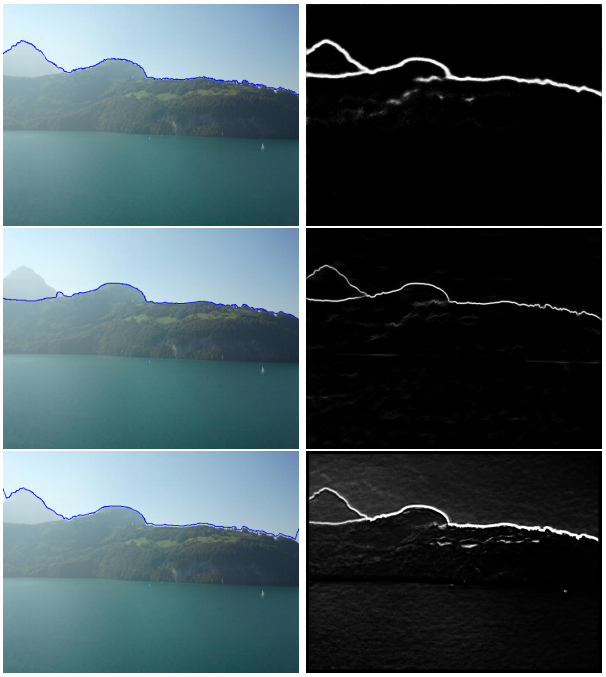

Figure 3: Detected horizon lines (left) and horizon probability maps (right) obtained with (from top to bottom): DCNN–DS1, Structured Forest [6] and the CNN in [1].

4. EXPERIMENTAL RESULTS

In this section we evaluate the HLD accuracy of the proposed method and variants thereof against other state-of-the-art approaches. The pixelswise detection mask obtained from each of the tested methods was fine-tuned with the dynamic programming approach described in Subsection 3.3 (post-processing). For each image, the HLD accuracy is evaluated using the error measure given in [1], *i.e.*

$$E = \sum_{v=1}^{\mathrm{w}} \left\| P_v - P_v^{\mathrm{gt}} \right\| , \tag{5}$$

where P_v and P_v^{gt} are the predicted and ground-truth horizon points at the v-th column of the image, respectively. The results in the following are given in terms of the mean and standard deviation of the error in (5) on the test images.

4.1 Dataset

In our evaluation we consider the publicly-available CH1 dataset[1], first presented in the work of Saurer *et al.* [14]. This dataset comprises 203 photographs of mountainous landscapes taken from various locations in the Swiss Alps, depicting a wide variety of seasonal and weather conditions.

[1]https://cvg.ethz.ch/research/mountain-localization/

Table 1: HLD error when using our best CNN configuration compared to previous methods.

Method	Error (px) mean	std
DCNN–DS1	1.53	2.38
Structured Forest [6]	8.07	18.92
Random Ferns [13]	10.21	20.41
CNN in [1]	5.78	14.73
SVM in [1]	36.69	60.59

Table 2: Horizon line detection error when using different CNN configurations.

Method	Error (px) mean	std
DCNN–DSavg	2.03	2.32
DCNN–DS1	1.53	2.38
DCNN–DS2	2.25	2.47
DCNN–DS3	2.50	2.29
DCNN–No DS	3.94	8.76
CNN–DS	3.33	9.28

Each image is annotated with a ground truth sky/terrain segmentation, from which we derive the ground-truth horizon line. In our experiments we rescale each image to a resolution of 640×480 and randomly split the dataset into training and test subsets, comprising 60% and 40% of the images, respectively.

4.2 Implementation and training details

All the network architectures considered in the experimental evaluation are trained using the same setup and implemented using the Caffe library. Data augmentation is used to enrich the training set, which comprises only 121 images. In particular, from each image we select 16 patches of 256×256 pixels, each sampled after performing random scale, rotation, flip and translation transformations. At test time, thanks to the fully-convolutional nature of the architecture, we are able to evaluate the images in their original size (*i.e.* 640×480).

The training phase is performed using mini-batch SGD with the Adam [7] method. The "CONV" layers are initialized from a publicly available implementation of VGG16 [15] pre-trained on ImageNet. Each "DECONV" layer is initialized from a zero-mean Gaussian distribution with variance $\sqrt{2/fin}$, where *fin* is the layer's fan-in. Finally, the parameters of the output layers are initialized to zero. The training meta-parameters are as follows. The batch-size is set to 48. We perform 15 training epochs with a learning rate of 10^{-3} followed by 15 epochs with a learning rate of 10^{-4}. The weight decay is set to 0.0002.

4.3 Results and discussion

In a first set of experiments, we compare the proposed deeply-supervised deconvolutional network (DCNN-DS1) using the best-performing detector $h_1(\cdot)$ (see Sec. 3) with previous learning approaches used for detecting mountains profiles, *i.e.* structured forest [6], random ferns [13], and the CNN and SVM classifiers in [1]. The results in Table 1 indicate a clear advantage of our method, which outperforms the others by a considerable margin. Interestingly, the second best result is given by the CNN classifier in [1], which employs a very simple architecture with less than 1000 parameters. This result, together with the comparatively poorer performance obtained with traditional classifiers, suggests that the use of learned features is very important for the HLD task.

In our experiments, we also observe that the HLD pipeline is significantly less robust when traditional classifiers (not CNN-based) are employed. Figure 3 shows an example of this. Here, a peak in the background is covered by a light mist, which makes it look much fainter than the mountains in the foreground. In this case, the Structured Forest is not able to differentiate between the background peak-to-sky edge and the foreground mountain-to-background peak edge, producing an equally strong response for both. This forces the dynamic programming step to choose the second, straighter edge as the horizon. Conversely, our approach and CNN correctly give a stronger response for the first edge, producing the correct horizon line.

In a final set of experiments we compare several variants of our network architecture, specifically: i) the four detectors $h_1, h_2, h_3, h_{\text{avg}}$, denoted as DCNN–DSx, where x is one of 1, 2, 3 or avg; ii) a version of the net trained using only the f_1 predictor and its corresponding loss (*i.e.* with a single supervision), denoted as DCNN–No DS; iii) a deeply-supervised non-deconvolutional version of the net, denoted as CNN–DS. With reference to Fig. 2, CNN–DS is obtained by discarding the "DECONV" block and connecting each predictor to one of the convolutional layers immediately preceding a pooling layer in the "CONV" block. The final output is given by the average of the three predictions, as in DCNN-DSavg. This architecture is similar to the HED network in [19]. The results, summarized in Table 2, show a clear advantage of the deeply-supervised deconvolutional architecture compared to the other ones. Interestingly, DCNN–DS1 gives the best result among the DCNN–DSx nets, while DCNN–No DS gives the worst result overall. This fact, together with the relatively lower performance of DCNN–DSavg compared to DCNN–DS1, suggests that the beneficial effects of a deep supervision in our case are more related to its regularizing effects than to the availability of multiple outputs.

5. CONCLUSION

We presented a new method for horizon line detection. The main novelty of the proposed approach is a deeply-supervised deconvolutional network to detect horizon pixels. Our results demonstrate that the proposed approach outperforms previous learning-based HLD solutions. Future works include investigating alternative deeply-supervised architectures (*e.g.* to reduce the computational cost) and exploiting the proposed approach for image geo-localization [4].

6. ACKNOWLEDGMENTS

This research has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 687757.

7. REFERENCES

[1] T. Ahmad, G. Bebis, M. Nicolescu, A. Nefian, and T. Fong. An edge-less approach to horizon line detection. In *ICMLA*, 2015.

[2] G. Baatz, O. Saurer, K. Köser, and M. Pollefeys. Large scale visual geo-localization of images in mountainous terrain. In *ECCV*. 2012.

[3] L. Baboud, M. Čadík, E. Eisemann, and H.-P. Seidel. Automatic photo-to-terrain alignment for the annotation of mountain pictures. In *CVPR*, 2011.

[4] M. Bansal, H. S. Sawhney, H. Cheng, and K. Daniilidis. Geo-localization of street views with aerial image databases. In *ACM MM*, 2011.

[5] P. Dollár, Z. Tu, and S. Belongie. Supervised learning of edges and object boundaries. In *CVPR*, 2006.

[6] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *ICCV*, 2013.

[7] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[8] I. Kokkinos. Surpassing humans in boundary detection using deep learning. *ICLR*, 2016.

[9] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. *AISTATS*, 2015.

[10] S. Lee, H. Zhang, and D. J. Crandall. Predicting geo-informative attributes in large-scale image collections using convolutional neural networks. In *WACV*, 2015.

[11] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *ICCV*, 2015.

[12] V. Ordonez and T. L. Berg. Learning high-level judgments of urban perception. In *ECCV*. 2014.

[13] L. Porzi, S. Rota Bulò, P. Valigi, O. Lanz, and E. Ricci. Learning contours for automatic annotations of mountains pictures on a smartphone. In *ICDSC*, 2014.

[14] O. Saurer, G. Baatz, K. Köser, M. Pollefeys, et al. Image based geo-localization in the alps. *IJCV*, pages 1–13, 2015.

[15] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[16] L. Tao, L. Yuan, and J. Sun. Skyfinder: attribute-based sky image search. In *ACM TOG*, volume 28, page 68, 2009.

[17] W. Wang, G. Chen, A. T. T. Dinh, J. Gao, B. C. Ooi, K.-L. Tan, and S. Wang. Singa: Putting deep learning in the hands of multimedia users. In *ACM MM*, 2015.

[18] R. Xiaofeng and L. Bo. Discriminatively trained sparse code gradients for contour detection. In *NIPS*, 2012.

[19] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*, 2015.

[20] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV*. 2014.

[21] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014.