Dense Extreme Inception Network: Towards a Robust CNN Model for Edge Detection

Xavier Soria† Edgar Riba† Angel Sappa†,‡
† Computer Vision Center - Universitat Autonoma de Barcelona, Barcelona, Spain
‡ Escuela Superior Politécnica del Litoral, Guayaquil, Ecuador
{xsoria,eriba,asappa}@cvc.uab.es

Abstract

This paper proposes a Deep Learning based edge detector, which is inspired on both HED (Holistically-Nested Edge Detection) and Xception networks. The proposed approach generates thin edge-maps that are plausible for human eyes; it can be used in any edge detection task without previous training or fine tuning process. As a second contribution, a large dataset with carefully annotated edges, has been generated. This dataset has been used for training the proposed approach as well the state-of-the-art algorithms for comparisons. Quantitative and qualitative evaluations have been performed on different benchmarks showing improvements with the proposed method when F-measure of ODS and OIS are considered.

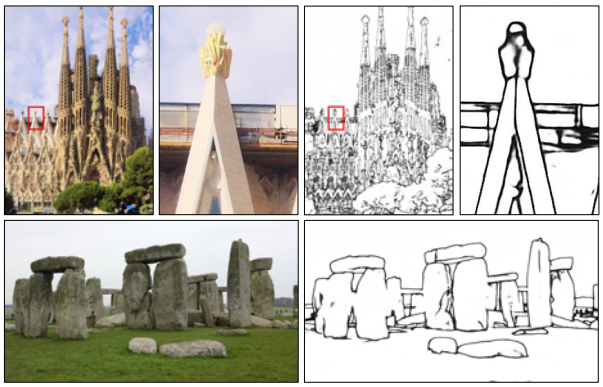


Figure 1. The edge-maps predictions from the proposed model in images acquired from internet.

1. Introduction

Edge detection is a recurrent task required for several classical computer vision processes (e.g., segmentation [39], image recognition [38, 30]), or even in the modern tasks such as image-to-image translation [41], photo sketching [18] and so on. Moreover, in fields such as medical image analysis [27] or remote sensing [16] most of their heart activities require edge detectors. In spite of the large amount of work on edge detection, it still remains as an open problem with space for new contributions.

Since the Sobel operator [33], many edge detectors have been proposed [25] and most of the techniques like Canny [5] are still being used nowadays. Recently, in the era of Deep Learning (DL), Convolutional Neural Netwoks (CNN) based edge detectors like DeepEdge [4], HED [36], RCF [20], BDCN [14] among others, have been proposed. These models are capable of predicting an edge-map from a given image just like the low level based methods [42], with better performance. The success of these methods is mainly by the CCNs applied at different scales to a large set of images together with the training regularization techniques.

Most of the aforementioned DL based approaches are trained on already existing boundary detection or object segmentation datasets [22, 31, 24] to detect edges. Even though most of the images on those datasets are well annotated, there are a few of them that contain missing edges, which difficult the training, thus the predicted edge-maps lost some edges in the images (see Fig. 1). In the current work, those datasets are used just for qualitative comparisons due to the objective of the current work is edge detection (not objects' boundary/contour detection). The boundary/contour detection tasks, although related and some times assumed as a synonym task, are different since just objects' boundary/contour need to be detected, but not all edges present in the given image.

This manuscript aims to demonstrate the edge detection generalization from a DL model. In other words, the model is capable of being evaluated in other datasets for edge detection without being trained on those sets. To the best of our knowledge, the unique dataset for edge detection shared to the community is Multicue Dataset for Boundary Detection (MDBD—2016) [23], which although mainly generated for the boundary detection study, it contains a subset of

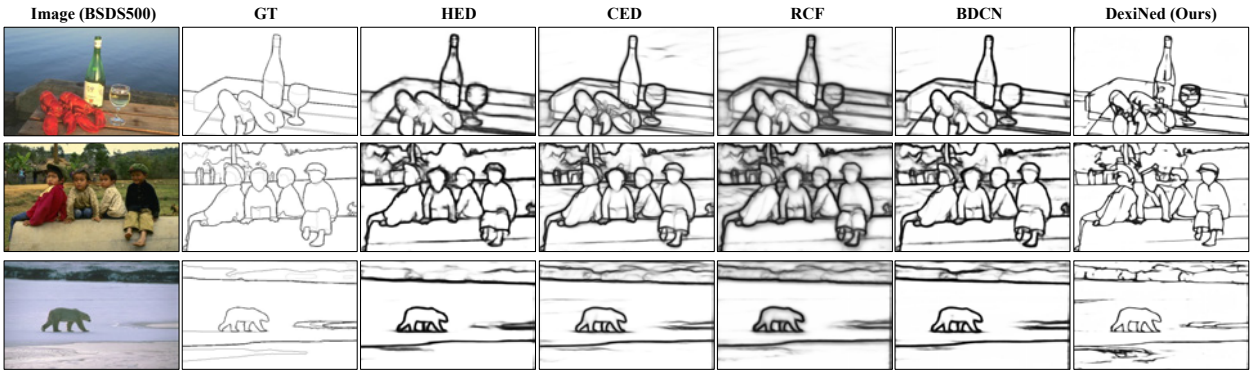


Figure 2. Edge-maps predicted from the state-of-the-art models and DexiNed on three BSDS500 [3] images. Note that DexiNed was just trained with BIPED, while all the others were trained on BSDS500.

images devoted for edge detection. Therefore, a new dataset has been collected to train the proposed edge detector. The main contributions in the paper are summarized as follow:

- A dataset with carefully annotated edges has been generated and released to the community—BIPED: Barcelona Images for Perceptual Edge Detection.[1]
- A robust CNN architecture for edge detection is proposed, referred to as DexiNed: Dense Extreme Inception Network for Edge Detection. The model has been trained from the scratch, without pretrained weights.

The rest of the paper is organized as follow. Section 2 summarizes the most relevant and recent work on edge detection. Then, the proposed approach is described in Section 3. The experimental setup is presented in Section 4. Experimental results are then summarized in Section 5; finally, conclusions and future work are given in Section 6.

2. Related Work

There are a large number of work on the edge detection literature, for a detailed review see [42, 11]. According to the technique the given image is processed, proposed approaches can be categorized as: i) Low level feature; ii) Brain-biologically inspiration; iii) Classical learning algorithms; iv) Deep learning algorithms.

Low-level feature: Most of the algorithms in this category generally follow a smooth process, which could be performed convolving the image with a Gaussian filter or manually performed kernels. A sample of such methods are [5, 28, 26]. Since Canny [5], most of the nowadays methods use non-maximum suppression [6] as the last process of edge detection.

Brain-biologically inspiration: This kind of method started their research in the 60s of the last century analyzing the edge and contour formation in the vision systems of monkeys and cats [8]. inspired on such a work, in [12] the authors proposed a method based on simple cells and Gabor filters. Another study focused on boundary detection is presented in [23]. This work proposes to use Gabor and derivative of Gaussian filters, considering three different filter sizes and machine learning classifiers. More recently, in [37], an orientation selective neuron is presented, by using first derivative of a Gaussian function. This work has been recently extended in [2] by modeling retina, simple cells even the cells from V2.

Classical learning algorithms: These techniques are usually based on sparse representation learning [21], dictionary learning [35], gPb (gradient descent) [3] and structured forest [9] (decision trees). At the time these approaches have been proposed, they outperformed state-of-the-art techniques based on low level processes reaching the best F-measure values in BSDS segmentation dataset [3]. Although obtained results were acceptable in most of the cases, these techniques still have limitations in challenging scenarios.

Deep learning algorithms: With the success of CNN, principally because of its result in [17], many methods have been proposed [10, 4, 36, 20, 34]. In HED [36] for example, an architecture based on VGG16 [32] and pre-trained with ImageNet dataset is proposed. The network generates edges from each convolutional block constructing a multi-scale learning architecture. The training process uses a modified cross entropy loss function for each predicted edge-maps. Using the same architecture as their backbone, [20] and [34] have proposed improvements. While in [20] every output is feed from each convolution from every block, in [34] a set of fusion backward process, with the data of each outputs, is performed. In general, most of the current DL based models use as their backbone the convolutional blocks of VGG16 architecture.

[1] Code + dataset: https://github.com/xavysp/DexiNed

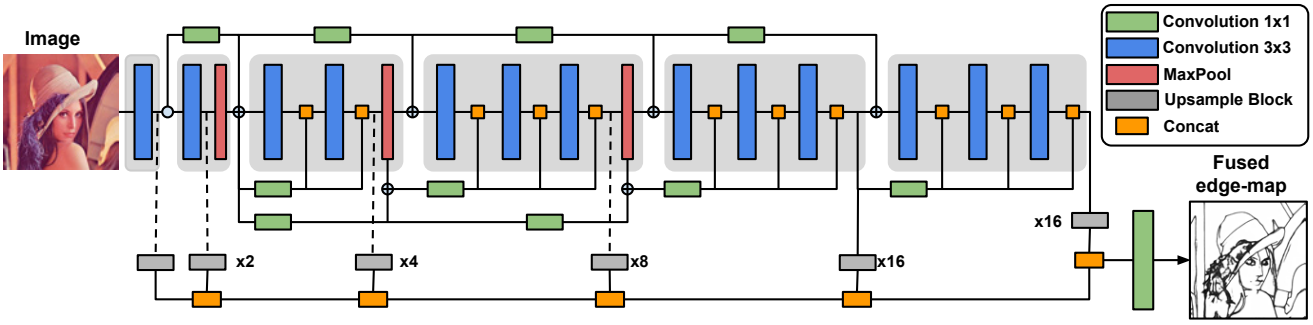


Figure 3. Proposed architecture: Dense Extreme Inception Network, consists of an encoder composed by six main blocks (showed in light gray). The main blocks are connected between them through 1x1 convolutional blocks. Each of the main blocks is composed by sub-blocks that are densely interconnected by the output of the previous main block. The output from each of the main blocks is fed to an upsampling block that produces an intermediate edge-map in order to build a Scale Space Volume, which is used to compose a final fused edge-map. More details are given in Sec. 3.

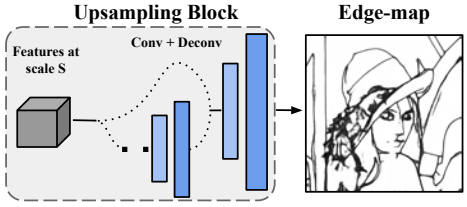


Figure 4. Detail of the upsampling block that receives as input the learned features extracted from each of the main blocks. The features are fed into a stack of learned convolutional and transposed convolutional filters in order to extract an intermediate edge-map.

3. Dense Extreme Inception Network for Edge Detection

This section presents the architecture proposed for edge detection, termed DexiNed, which consists of a stack of learned filters that receive as input an image then predict an edge-map with the same resolution. DexiNed can be seen as two sub networks (see Figs. 3 and 4): Dense extreme inception network (Dexi) and the up-sampling block (UB). While Dexi is fed with the RGB image, UB is fed with feature maps from each block of Dexi. The resulting network (DexiNed) generates thin edge-maps, avoiding missed edges in the deep layers. Note that even though without pre-trained data, the edges predicted from DexiNed are in most of the cases better than state-of-the-art results, see Fig. 1.

3.1. DexiNed Architecture

The architecture is depicted in Fig. 3, it consists of an encoder with 6 main blocks inspired in the xception network [7]. The network outputs feature maps at each of the main blocks to produce intermediate edge-maps using an upsampling block defined in Section 3.2. All the edge-maps resulting from the upsampling blocks are concatenated to feed the stack of learned filters at the very end of the network and produce a fused edge-map. All six upsampling blocks do not share weights.

The blocks in blue consists of a stack of two convolutional layers with kernel size 3×3, followed by batch normalization and ReLU as the activation function (just the last convs in the last sub-blocks does not have such activation). The max-pool is set by 3×3 kernel and stride 2. As the architecture follows the multi-scale learning, like in HED, an upsampling process (horizontal blocks in gray, Fig. 3) is followed (see details in Section 3.2).

Even though DexiNed is inspired in xception, the similarity is just in the structure of the main blocks and connections. Major differences are detailed below:

- While in xception separable convolutions are used, DexiNed uses standard convolutions.
- As the output is a 2D edge-map, there is "not exit flow", instead, another block at the end of block five has been added. This block has 256 filters and as in block 5 there is not maxpooling operator.
- In block 4 and block 5, instead of 728 filters, 512 filters have been set. The separations of the main blocks are done with the blocks connections (rectangles in green) drawn on the top side of Fig. 3.
- Concerning to skip connections, in xception there is one kind of connection, while in DexiNed there are two type of connections, see rectangles in green on the top and bottom of Fig. 3.

Since many convolutions are performed, every deep block losses important edge features and just one main-connection is not sufficient, as highlighted in DeepEdge [4], from the forth convolutional layer the edge feature loss is more chaotic. Therefore, since block 3, the output of each sub-block is averaged with *edge-connection* (orange squares in Fig. 3). These processes are inspired in ResNet

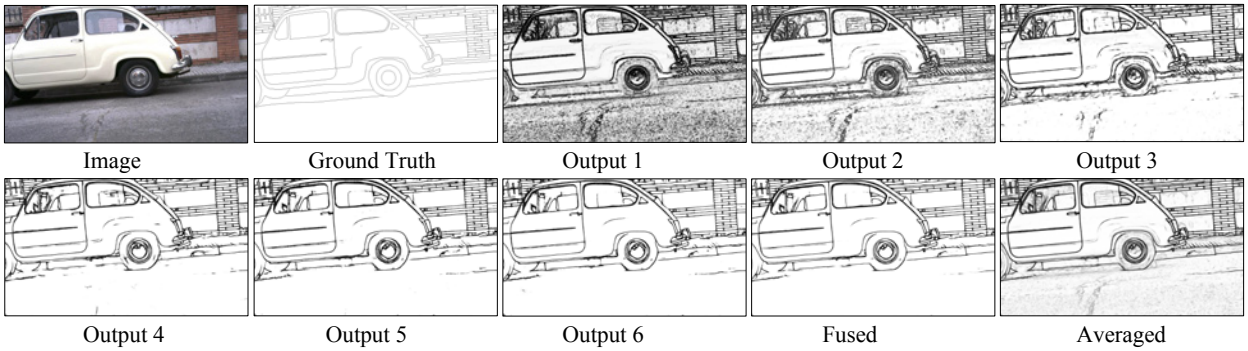


Figure 5. Edge-maps from DexiNed in BIPED test dataset. The six outputs are delivered from the upsampling blocks, the *fused* is the concatenation and fusion of those outputs and the *averaged* is the average of all previous predictions.

[15] and RDN [40] with the following notes: i) as shown in Fig. 3, after the max-pooling operation and before summation with the main-connection, the edge-connection is set to average each sub-blocks output (see rectangles in green, bottom side); ii) from the max-pool, block 2, edge-connections feed sub-blocks in block 3, 4 and 5, however, the sub-blocks in 6 are feed just from block 5 output.

3.2. Upsampling Block

DexiNed has been designed to produce thin edges in order to enhance the visualization of predicted edge-maps. One of the key component of DexiNed for the edge thinning is the upsampling block, as appreciated in Fig. 3, each output from the Dexi blocks feeds the UB. The UB consists of the conditional stacked sub-blocks. Each sub-block has 2 layers, one convolutional and the other deconvolutional; there are two types of sub-blocks. The first sub-block (sub-block1) is feed from Dexi or sub-block2; it is only used when the scale difference between the feature map and the ground truth is equal to 2. The other sub-block (sub-block2), is considered when the difference is greater than 2. This sub-block is iterated till the feature map scale reaches 2 with respect to the GT. The sub-block1 is set as follow: kernel size of the conv layer 1×1; followed by a ReLU activation function; kernel size of the deconv layer or transpose convolution $s \times s$, where s is the input feature map scale level; both layers return one filter and the last one gives a feature map with the same size as the GT. The last conv layer does not have activation function. The sub-block2 is set similar to sub-block1 with just one difference in the number of filters, which is 16 instead of 1 in sub-block1. For example, the output feature maps from block 6 in Dexi has the scale of 16, there will be three iterations in the sub-block2 before fed the sub-block1. The upsampling process of the second layer from the sub-blocks can be performed by bi-linear interpolation, sub-pixel convolution and transpose convolution, see Sec. 5 for details.

3.3. Loss Functions

DexiNed could be summarized as a regression function $\eth$, that is, $\hat{Y} = \eth(X, Y)$, where X is an input image, Y is its respective ground truth, and $\hat{Y}$ is a set of predicted edge maps. $\hat{Y} = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_N]$, where $\hat{y}_i$ has the same size as Y, and N is the number of outputs from each upsampling block (horizontal rectangles in gray, Fig. 3); $\hat{y}_N$ is the result from the last fusion layer f ($\hat{y}_N = \hat{y}_f$). Then, as the model is deep supervised, it uses the same loss as [36] (weighted cross-entropy), which is tackled as follow:

$$\begin{aligned} \ell^n(W, w^n) = &-\beta \sum_{j \in Y^+} \log \sigma(y_j = 1 | X; W, w^n) \\ &- (1-\beta) \sum_{j \in Y^-} \log \sigma(y_j = 0 | X; W, w^n), \end{aligned} \tag{1}$$

then,

$$\mathcal{L}(W, w) = \sum_{n=1}^{N} \delta^n \times \ell^n(W, w^n), \tag{2}$$

where W is the collection of all network parameters and w is the n corresponding parameter, δ is a weight for each scale level. $\beta = |Y^-|/|Y^+ + Y^-|$ and $(1-\beta) = |Y^+|/|Y^+ + Y^-|$ ($|Y^-|$, $|Y^+|$ denote the edge and non-edge in the ground truth). See Section 4.4 for hyper-parameters and optimizer details for the regularization in the training process.

4. Experimental Setup

This section presents details on the datasets used for evaluating the proposed model, in particular the dataset and annotations (BIPED) generated for an accurate training of the proposed DexiNed. Additionally, details on the evaluation metrics and network's parameters are provided.

4.1. Barcelona Images for Perceptual Edge Detection (BIPED)

The other contributions of the paper is a carefully annotated edge dataset. It contains 250 outdoor images of 1280×720 pixels each. These images have been carefully annotated by experts on the computer vision field, hence no redundancy has been considered. In spite of that, all results have been cross-checked in order to correct possible mistakes or wrong edges. This dataset is publicly available as a benchmark for evaluating edge detection algorithms. The generation of this dataset is motivated by the lack of edge detection datasets, actually, there is just one dataset publicly available for the edge detection task (MDBD [23]). Edges in MDBM dataset have been generated by different subjects, but have not been validated, hence, in some cases, the edges correspond to wrong annotations. Some examples of these missed or wrong edges can be appreciated in the ground truths presented in Fig. 8; hence, edge detector algorithms that obtain these missed edges are penalized during the evaluation. The level of details of the dataset annotated in the current work can be appreciated looking at the GT, see Figs. 5 and 7. In order to do a fair comparison between the different state-of-the-art approaches proposed in the literature, BIPED dataset has been used for training those approaches, which have been later on evaluated in ODS, OIS, and AP. From the BIPED dataset, 50 images have been randomly selected for testing and the remainders 200 for training and validation. In order to increase the number of training images a **data augmentation process** has been performed as follow: i) as BIPED data are in high resolution they are split up in the half of image width size; ii) similarly to HED, each of the resulting images is rotated by 15 different angles and crop by the inner oriented rectangle; iii) the images are horizontally flip; and finally iv) two gamma corrections have been applied (0.3030, 0.6060). This augmentation process resulted in 288 images per each 200 images.

4.2. Test Datasets

The datasets used to evaluate the performance of DexiNed are summarized bellow. There is just one dataset intended for edged detection MDBD [23], while the remainders are for objects' contour/boundary extraction/segmentation: CID [12], BSDS [22, 3], NYUD [31] and PASCAL [24].

MDBD: The Multicue Dataset for Boundary Detection has been intended for the purpose of psychophysical studies on object boundary detection in natural scenes, from the early vision system. The dataset is composed of short binocular video sequences of natural scenes [23], containing 100 scenes in high definition (1280 × 720). Each scene has 5 boundary annotations and 6 edge annotations. From the given dataset 80 images are used for training and the remainders 20 for testing [23]. In the current work, DexiNed has been evaluated using the first 20 images (the sub set for edge detection).

CID: This dataset has been presented in [12], a brain-biologically inspired edge detector technique. The main limitation of this dataset is that it just contains a set of 40 images with their respective ground truth edges. This dataset highlight that in addition to the edges the ground truth map contains contours of object. In this case the DexiNed has been evaluated with the whole CID data.

BSDS: Berkeley Segmentation Dataset, consists of 200 new test images [3] additional to the 300 images contained in BSDS300 [22]. In previous publications, the BSDS300 is split up into 200 images for training and 100 images for testing. Currently, the 300 images from BSDS300 are used for training and validation, while the remainders 200 images are used for testing. Every image in BSDS is annotated at least by 6 annotators; this dataset is mainly intended for image segmentation and boundary detection. In the current work both datasets are evaluated BSDS500 (200 test images) and BSDS300 (100 test images).

NYUD: New York University Dataset is a set of 1449 RGBD images that contains 464 indoor scenarios, intended for segmentation purposes. This dataset is split up by [13] into three subsets—i.e., training, validation and testing sets. The testing set contains 654 images, while the remainders images are used for training and validation purposes. In the current work, although the proposed model was not trained with this dataset, the testing set has been selected for evaluating the proposed DexiNed.

PASCAL: The Pascal-Context [24] is a popular dataset in segmentation; currently most of major DL methods for edge detection use this dataset for training and testing, both for edge and boundary detection purposes. This dataset contains 11530 annotated images, about 5% of them (505 images) have been considered for testing DexiNed.

4.3. Evaluation Metrics

The evaluation of an edge detector has been well defined since the pioneer work presented in [42]. Since BIPED has annotated edge-maps as GT, three evaluation metrics widely used in the community have been considered: fixed contour threshold (ODS), per-image best threshold (OIS), and average precision (AP). The F-measure (F) [3] of ODS and OIS, will be considered, where $F = \frac{2 \times Precision \times Recall}{Precision + Recall}$.

4.4. Implementation Notes

The implementation is performed in TensorFlow [1]. The model converges after 150k iterations with a batch size of 8 using Adam optimizer and learning rate of 10^{-4}. The training process takes around 2 days in a TITAN X GPU with color images of size 400x400 as input. The weights for fusion layer are initialized as: $\frac{1}{N-1}$ (see Sec. 3.3 for

Outputs	ODS	OIS	AP
Output 1 ($\hat{y}_1$)	.741	.760	.162
Output 2 ($\hat{y}_2$)	.766	.803	.817
Output 3 ($\hat{y}_3$)	.828	.846	.838
Output 4 ($\hat{y}_4$)	.844	.858	.843
Output 5 ($\hat{y}_5$)	.841	.8530	.776
Output 6 ($\hat{y}_6$)	.842	.852	.805
Fused ($\hat{y}_f$)	.857	.861	.805
Averaged	**.859**	**.865**	**.905**

(a)

Methods	ODS	OIS	AP
SED[2]	.717	.731	.756
HED[36]	.829	.847	.869
CED[34]	.795	.815	.830
RCF[19]	.843	.859	.882
BDCN[14]	.839	.854	.887
DexiNed-f	.857	.861	.805
DexiNed-a	**.859**	**.867**	**.905**

(b)

Table 1. (a) Quantitative evaluation of the 8 predictions of DexiNed on BIPED test dataset. (b) Comparisons between the state-of-the-art methods trained and evaluated with BIPED.

N). After a hyperparameter search to reduce the number of parameters, best performance was obtained using kernel sizes of 3×3, 1×1 and $s \times s$ on the different convolutional layers of Dixe and UB.

5. Experimental Results

This section presents quantitative and qualitative evaluations conducted by the metrics presented in Sec. 4. Since the proposed DL architecture demands several experiments to be validated, DexiNed has been carefully tuned till reach its final version.

5.1. Quantitative Results

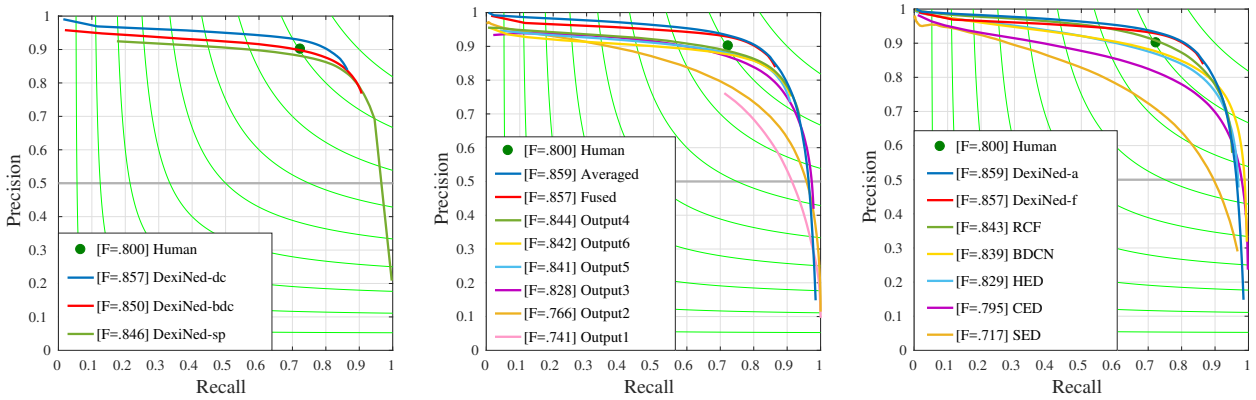
Firstly, in order to select the upsampling process that achieves the best result, an empiric evaluation has been performed, see Fig. 6(a). The evaluation consists in conducting the same experiments by using the three upsampling methods; **DexiNed-bdc** refers to upsampling performed by a transpose convolution initialized with a bi-linear kernel; **DexiNed-dc** uses transpose convolution with trainable kernels; and **DexiNed-sp** uses subpixel convolution. According to F-measure, the three versions of DexiNed get the similar results, however, when analyzing the curves in Fig. 6(a), a small difference in the performance of DexiNed-dc appears. As a conclusion, the DexiNed-dc upsampling strategy is selected; from now on, all the evaluations performed on this section are obtained using a DexiNed-dc upsampling; for simplicity of notation just the term DexiNed is used instead of DexiNed-dc.

Figure 6(b) and Table 1(a) present the quantitative results reached from each DexiNed edge-map prediction. The results from the eight predicted edge-maps are depicted, the best quantitative results, corresponding to the fused (DexiNed-f) and averaged (DexiNed-a) edge-maps are selected for the comparisons. Similarly to [36] the averaged of all predictions (DexiNed-a) gets the best results in the three evaluation metrics, followed by the prediction generated in the fusion layer. Note that the edge-maps predicted from the block 2 till the 6 get similar results to DexiNed-f, this is due to the fact of the proposed skip-connections. For a qualitative illustration, Fig. 5 presents all edge-maps predicted from the proposed architecture. Qualitatively, the result from DexiNed-f is considerably better than the one from DexiNed-a (see illustration in Fig. 5). However, according to Table 1(a), DexiNed-a produces slightly better quantitative results than DexiNed-f. As a conclusion both approaches (fused and averaged) reach similar results; through this manuscript whenever the term DexiNed is used it corresponds to DexiNed-f.

Dataset	Methods	ODS	OIS	AP
Edge detection dataset				
MDBD[23]	HED[36]	.851	**.864**	.890
	RCF[20]	.857	.862	-
	DexiNed-f	.837	.837	.751
	DexiNed-a	**.859**	**.864**	**.917**
Contour/boundary detection/segmentation datasets				
CID[12]	SCO[37]	.58	.64	.61
	SED[2]	**.65**	**.69**	.68
	DexiNed-f	.65	.67	.59
	DexiNed-a	**.65**	**.69**	**.71**
BSDS300[22]	gPb[3]	.700	.720	.660
	SED[2]	.69	.71	.71
	DexiNed-f	.707	.723	.52
	DexiNed-a	**.709**	**.726**	**.738**
BSDS500[3]	HED[36]	.790	.808	.811
	RCF[20]	**.806**	**.823**	-
	CED[34]	.803	.820	**.871**
	SED[2]	.710	.740	.740
	DexiNed-f	.729	.745	.583
	DexiNed-a	.728	.745	.689
NYUD[31]	gPb[3]	.632	.661	.562
	HED[36]	.720	**.761**	**.786**
	RCF[20]	**.743**	.757	-
	DexiNed-f	.658	.674	.556
	DexiNed-a	.602	.615	.490
PASCAL[24]	CED[34]	**.726**	**.750**	**.778**
	HED[36]	.584	.592	.443
	DexiNed-f	.431	.458	.274
	DexiNed-a	.475	.497	.329

Table 2. Quantitative results of **DexiNed trained on BIPED** and **the state-o-the-art methods trained with the corresponding datasets** (values from other approaches come from the corresponding publications).

Table 1(b) presents a comparison between the DexiNed and the state-of-the-art techniques on edge and boundary detection. In all the cases BIPED dataset has been considered, both for training and evaluating the DL based models (i.e., HED [29], RCF [20], CED [34]) and BDCN [14], the training process for each model took about two days. As can



(a) (b) (c)

Figure 6. Precision/recall curves on BIPED dataset. (a) DexiNed upsampling versions. (b) The outputs of DexiNed in testing stage, the 8 outputs are considered. (c) DexiNed comparison with other DL based edge detectors.

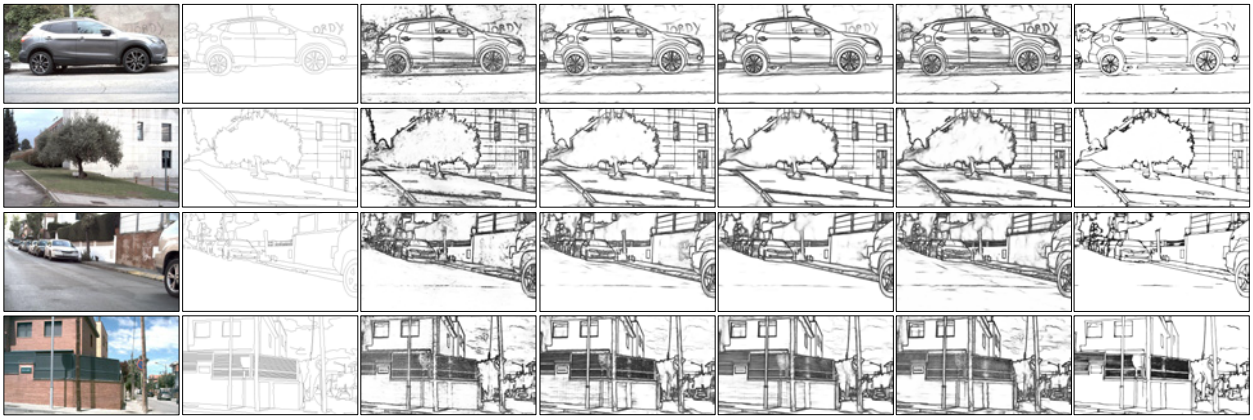


Image *GT* *CED [34]* *HED [36]* *RCF [20]* *BDCN [14]* *DexiNed*

Figure 7. Results from different edge detection algorithms trained and evaluated in BIPED dataset.

be appreciated from Table 1(b), DexiNed-a reaches the best results in all evaluation metrics. Actually both, DexiNed-a and DexiNed-f obtain the best results in almost all evaluation metrics. The F-measure obtained by comparing these approaches is presented in Fig. 6(c); it can be appreciated how for Recall above 75% DexiNed gets the best results. Illustrations of the edges obtained with DexiNed and the state-of-the-art techniques are depicted in Figure 7, just for four images from the BIPED dataset. As it can be appreciated, although RCF and BDCN obtain similar quantitative results than DexiNed, which were the second best ranked algorithms in Table 1(b), DexiNed predicts qualitative better results. Note that the proposed approach was trained from scratch without pre-trained weights.

The main objective of DexiNed is to get a precise edge-map from every dataset (RGB or Grayscale). Therefore, all the datasets presented in Sec. 4.2 have been considered, split up into two categories for a fair analysis; one for **edge detection** and the others for **contour/boundary detection/segmentation**. Results of edge-maps obtained with state-of-the-art methods are presented in Table 2. It should be noted that for each dataset the methods compared with DexiNed have been trained using images from that dataset, while DexiNed is trained just once with BIPED. It can be appreciated that DexiNed obtains the best performance in the MDBD dataset. It should be noted that DexiNed is evaluated in CID and BSDS300, even though these datasets contain a few images, which are not enough for training other approaches (e.g., HED, RCF, CED). Regarding BSDS500, NYUD and PASCAL, DexiNed does not reach the best results since these datasets have not been intended for edge detection, hence the evaluation metrics penalize edges detected by DexiNed. To highlight this situation, Fig. 8 depicts results from Table 2. Two samples from each dataset are considered. They are selected according to the best and worst F measure. Therefore, as shown in Fig. 8, when the image is fully annotated the score reaches around 100%, otherwise it reaches less than 50%.

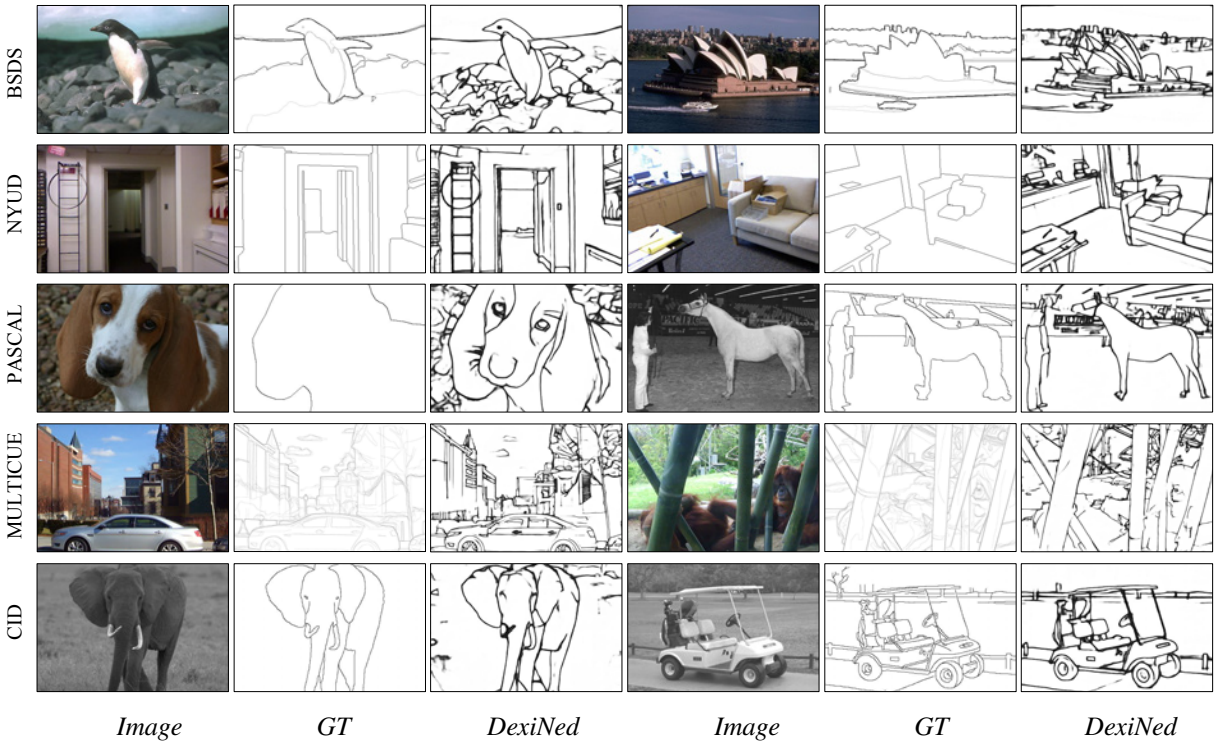


Figure 8. Results from the proposed approach using different datasets (note that DexiNed has been trained just with BIPED).

5.2. Qualitative Results

As highlighted in previous section, when the deep learning based edge detection approaches are evaluated in datasets intended for objects' boundary detection or objects segmentation, the results will be penalized. To support this claim, we present in Fig. 8 two predictions (the best and the worst results according to F-measure) from all datasets used for evaluating the proposed approach (except BIPED that has been used for training). The F-measure obtained in the three most used datasets (i.e., BSDS500, BSDS300 and NYUD) reaches over 80% in those cases where images are fully annotated; otherwise, the F-measure reaches about 30%. However, when the edge dataset (MDBD [23]) is considered the worst F-measure reaches over 75%. As a conclusion, it should be stated that edge detection and contour/boundary detection are different problems that need to be tackled separately when a DL based model is considered.

6. Conclusions

A deep structured model (DexiNed) for image's edge detection is proposed. Up to our knowledge, it is the first DL based approach able to generate thin edge-maps. A large experimental results and comparisons with state-of-the-art approaches is provided showing the validity of DexiNed. Even though DexiNed is trained just one time (with BIPED) it outperforms the state-of-the-art approaches when evaluated in other edge oriented datasets. A carefully annotated dataset for edge detection has been generated and is shared to the community. Future work will be focused on tackling the contour and boundary detection problems by using the proposed architecture and approach.

Acknowledgment

This work has been partially supported by: the Spanish Government under Project TIN2017-89723-P; the "CERCA Programme / Generalitat de Catalunya" and the ESPOL project PRAIM (FIEC-09-2015). The authors gratefully acknowledge the support of the CYTED Network: "Ibero-American Thematic Network on ICT Applications for Smart Cities" (REF-518RT0559) and the NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. Xavier Soria has been supported by Ecuador government institution SENESCYT under a scholarship contract 2015-AR3R7694.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016. 5

[2] A. Akbarinia and C. A. Parraga. Feedback and surround modulated boundary detection. *International Journal of Computer Vision*, 126(12):1367–1380, Dec 2018. 2, 6

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, May 2011. 2, 5, 6

[4] G. Bertasius, J. Shi, and L. Torresani. Deepedge: A multi-scale bifurcated deep network for top-down contour detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4380–4389, 2015. 1, 2, 3

[5] J. Canny. A computational approach to edge detection. In *Readings in computer vision*, pages 184–203. Elsevier, 1987. 1, 2

[6] J. F. Canny. Finding edges and lines in images. Technical report, Massachusetts Inst. of Tech. Cambridge Artificial Intelligence Lab, 1983. 2

[7] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017. 3

[8] J. G. Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *JOSA A*, 2(7):1160–1169, 1985. 2

[9] P. Dollár and C. L. Zitnick. Fast edge eetection using structured forests. *IEEE transactions on pattern analysis and machine intelligence*, 37(8):1558–1570, 2015. 2

[10] Y. Ganin and V. Lempitsky. n^4fields: Neural network nearest neighbor fields for image transforms. In *Asian Conference on Computer Vision*, pages 536–551. Springer, 2014. 2

[11] X.-Y. Gong, H. Su, D. Xu, Z.-T. Zhang, F. Shen, and H.-B. Yang. An overview of contour detection approaches. *International Journal of Automation and Computing*, Jun 2018. 2

[12] C. Grigorescu, N. Petkov, and M. A. Westenberg. Contour detection based on nonclassical receptive field inhibition. *IEEE Transactions on image processing*, 12(7):729–739, 2003. 2, 5, 6

[13] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from rgb-d images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013. 5

[14] J. He, S. Zhang, M. Yang, Y. Shan, and T. Huang. Bi-directional cascade network for perceptual edge detection. *arXiv preprint arXiv:1902.10903*, 2019. 1, 6, 7

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual rearning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[16] F. Isikdogan, A. Bovik, and P. Passalacqua. Rivamap: An automated river analysis and mapping engine. *Remote Sensing of Environment*, 202:88–97, 2017. 1

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2

[18] M. Li, Z. Lin, R. Mˇ ech, E. Yumer, and D. Ramanan. Photosketching: Inferring contour drawings from images. *WACV*, 2019. 1

[19] Y. Liu, M. Cheng, X. Hu, J. Bian, L. Zhang, X. Bai, and J. Tang. Richer convolutional features for edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2019. 6

[20] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai. Richer convolutional features for edge detection. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 5872–5881. IEEE, 2017. 1, 2, 6, 7

[21] J. Mairal, M. Leordeanu, F. Bach, M. Hebert, and J. Ponce. Discriminative sparse image models for class-specific edge detection and image interpretation. In *European Conference on Computer Vision*, pages 43–56. Springer, 2008. 2

[22] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423 vol.2, July 2001. 1, 5, 6

[23] D. A. Mély, J. Kim, M. McGill, Y. Guo, and T. Serre. A systematic comparison between visual cues for boundary detection. *Vision research*, 120:93–107, 2016. 1, 2, 5, 6, 8

[24] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille. The role of context for object detection and semantic segmentation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 891–898, 2014. 1, 5, 6

[25] M. A. Oskoei and H. Hu. A survey on edge detection methods. *University of Essex, UK*, 33, 2010. 1

[26] P. Perona, J. Malik, et al. Detecting and localizing edges composed of steps, peaks and roofs. 1991. 2

[27] R. Pourreza, Y. Zhuge, H. Ning, and R. Miller. Brain tumor segmentation in mri scans using deeply-supervised neural networks. In *International MICCAI Brainlesion Workshop*, pages 320–331. Springer, 2017. 1

[28] B. G. Schunck. Edge detection with gaussian filters at multiple scales. In *Proceedings of a Workshop on Computer Vision, Published by IEEE Computer Society Press, Washington, DC*, pages 208–210, 1987. 2

[29] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang. Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3982–3991, 2015. 6

[30] J. Shotton, A. Blake, and R. Cipolla. Multiscale categorical object recognition using contour fragments. *IEEE transactions on pattern analysis and machine intelligence*, 30(7):1270–1281, 2008. 1

[31] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, pages 746–760. Springer, 2012. 1, 5, 6

[32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[33] I. Sobel. Camera models and machine perception. Technical report, Computer Science Department, Technion, 1972. 1

[34] Y. Wang, X. Zhao, and K. Huang. Deep crisp boundaries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3892–3900, 2017. 2, 6, 7

[35] R. Xiaofeng and L. Bo. Discriminatively trained sparse code gradients for contour detection. In *Advances in neural information processing systems*, pages 584–592, 2012. 2

[36] S. Xie and Z. Tu. Holistically-nested edge detection. *International Journal of Computer Vision*, 125(1-3):3–18, 2017. 1, 2, 4, 6, 7

[37] K.-F. Yang, S.-B. Gao, C.-F. Guo, C.-Y. Li, and Y.-J. Li. Boundary detection using double-opponency and spatial sparseness constraint. *IEEE Transactions on Image Processing*, 24(8):2565–2578, 2015. 2, 6

[38] M.-H. Yang, D. J. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Transactions on pattern analysis and machine intelligence*, 24(1):34–58, 2002. 1

[39] K. Zhang, L. Zhang, K.-M. Lam, and D. Zhang. A level set approach to image segmentation with intensity inhomogeneity. *IEEE transactions on cybernetics*, 46(2):546–557, 2016. 1

[40] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2472–2481, 2018. 4

[41] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017. 1

[42] D. Ziou, S. Tabbone, et al. Edge detection techniques-an overview. *Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii*, 8:537–559, 1998. 1, 2, 5