Pattern Recognition 139 (2023) 109461

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Dense extreme inception network for edge detection

Xavier Soria[a,b,1,*], Angel Sappa[c,a,2], Patricio Humanante[b,3], Arash Akbarinia[d,4]

[a] Computer Vision Center, Autonomous University of Barcelona, Barcelona, Spain
[b] UMAYUK Research Group, National University of Chimborazo, Riobamba, Ecuador
[c] ESPOL Polytechnic University, FIEC, CIDIS, Guayaquil, Ecuador
[d] Department of Experimental Psychology, Justus-Liebig University, Giessen, Germany

A R T I C L E   I N F O

Article history:
Received 3 December 2021
Revised 13 February 2023
Accepted 20 February 2023
Available online 24 February 2023

Keywords:
Edge detection
Deep learning
CNN
Contour detection
Boundary detection
Segmentation

A B S T R A C T

Edge detection is the basis of many computer vision applications. State of the art predominantly relies on deep learning with two decisive factors: dataset content and network architecture. Most of the publicly available datasets are not curated for edge detection tasks. Here, we address this limitation. First, we argue that edges, contours and boundaries, despite their overlaps, are three distinct visual features requiring separate benchmark datasets. To this end, we present a new dataset of edges. Second, we propose a novel architecture, termed Dense Extreme Inception Network for Edge Detection (DexiNed), that can be trained from scratch without any pre-trained weights. DexiNed outperforms other algorithms in the presented dataset. It also generalizes well to other datasets without any fine-tuning. The higher quality of DexiNed is also perceptually evident thanks to the sharper and finer edges it outputs.

1. Introduction

Edges provide important clues in visual information processing, from classical computer vision algorithms in image recognition [2] to modern techniques in generative adversarial networks (GAN) [3]. Despite their importance, robust edge detection remains an open problem. To demonstrate this, let's examine four example images in Fig. 1. The third to fifth rows illustrate the results of three state-of-the-art models (RCF [4], BDCN [5], and CATS [6]) that are trained on the BSDS dataset [1]. It is qualitatively visible that neither of these models faithfully detects the edges of tiny details. For instance, while the internal edges of the bell sculpture or the helicopter are fully annotated in the ground truth, the edge output of those models does not capture these details. We can observe a similar phenomenon in the stripes of the building and the zebra. In this case, although these details are also excluded in the ground truth, the edges are clearly visible in the image. Grounded on this, we argue that current models of edge detection require further improvement to robustly detect edges and generalize well to new scenes independent of their training set.

Following this, we argue that one of the main limitations of current deep learning (DL) approaches is the annotated ground truth in the training set. BSDS is widely accepted within the community as the benchmark dataset to train and evaluate edge-detection algorithms. However, this dataset was originally designed and annotated for scene segmentation. Therefore the annotated ground truth corresponds mainly to high-level object boundaries rather than low-level edges. Given the DL-based models are strongly shaped by their training data, we need to acquire independent datasets for three tasks: edge-, contour, and boundary-detection [7,8]. This is of great importance for both training and evaluation. It is challenging for a network to learn these three concepts from a dataset that blends edges, contours and boundaries. It is also difficult to judge the fitness of a network, whether it is performing better or worse in one of those tasks.

To showcase this importance at a glance, we can look at the bottom two rows of Fig. 1. The only difference between those two rows is the training set. When our network—Dense Extreme Inception Network for Edge Detection (DexiNed)—is trained on BSDS, it suffers from the same set of problems as other models. However, when we train DexiNed on our dataset (BIPEDv2), it generalizes well to BSDS images in fine-scale edges. The image content of these two datasets qualitatively differs. The BIPEDv2 mainly contains images of urban settings. The accurately detected edges in the zebra and bell sculpture images suggest that the DexiNed-BIPEDv2 is robust to novel scenes. This robustness is

* Corresponding author.
E-mail addresses: xavier.soria@unach.edu.ec (X. Soria), sappa@ieee.org (A. Sappa), phumanante@unach.edu.ec (P. Humanante), Arash.Akbarinia@psychol.uni-giessen.de (A. Akbarinia).
[1] [orcid=0000-0003-2997-2439]
[2] [orcid=0000-0003-2468-0031]
[3] [orcid=0000-0003-2632-2051]
[4] [orcid=0000-0002-4249-231X]

https://doi.org/10.1016/j.patcog.2023.109461
0031-3203/© 2023 Elsevier Ltd. All rights reserved.

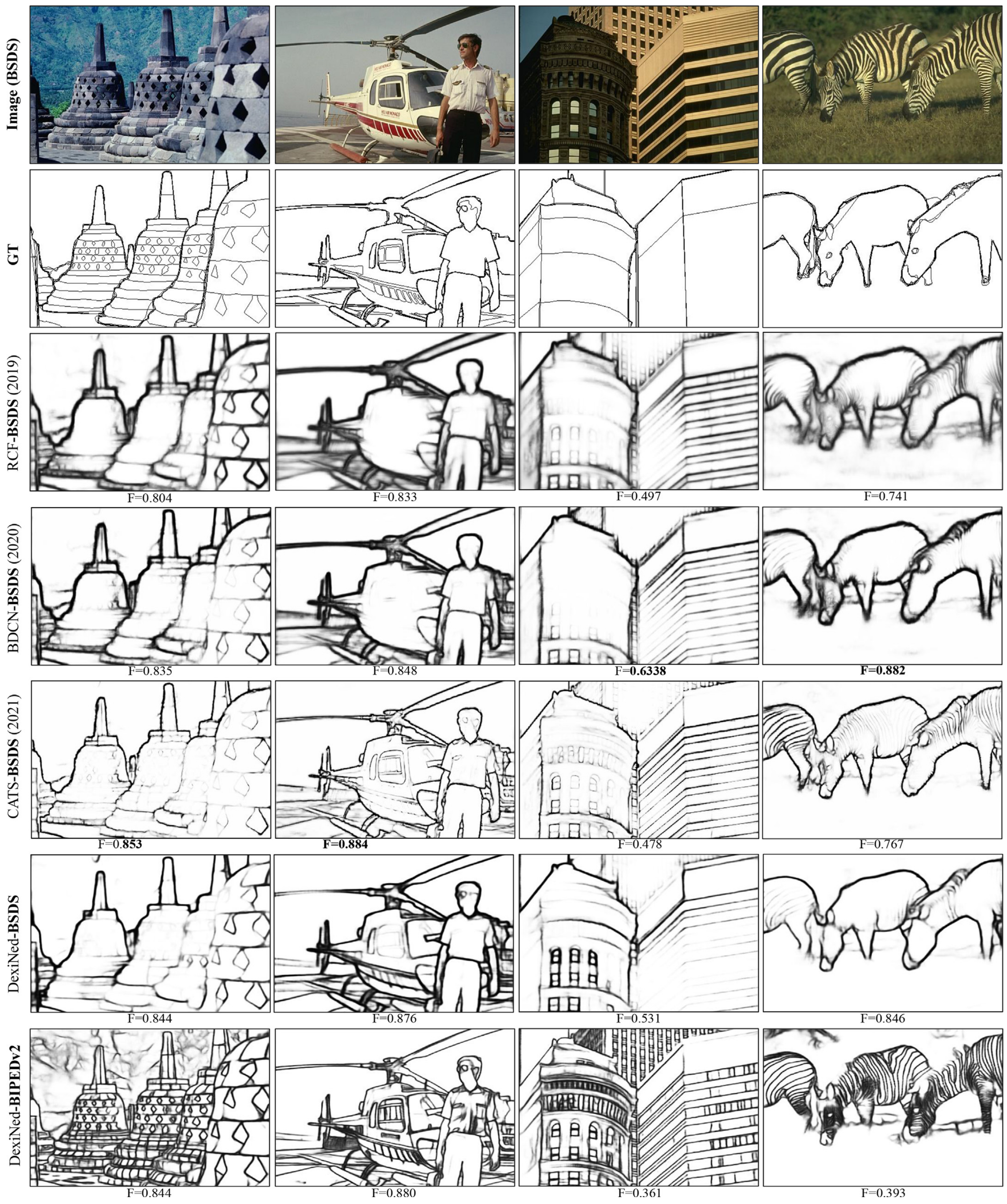

Fig. 1. The results of several algorithms on the BSDS dataset [1]. The suffix in model names (i.e., BSDS and BIPEDv2) indicates the training set of that model.

thanks to the careful manual annotation of edges in the proposed dataset. We ensured that all the edges, within or across objects, are reflected in the ground truth data.

1.1. Edges, contours and boundaries

The following three terms, edge, contour, and boundary detection are often used interchangeably despite their differences. This is a potential cause of confusion in the interpretation and evaluation of models. Thus, we start by reviewing the origin of each term. In the 1980s, edge detection was defined as the intensity changes in a vicinity originated by discontinuities along the surface, reflectance, or illumination [9]. This was revisited by emphasizing the properties of objects, such as their photometrical, geometrical and physical characteristics [10]. Accordingly, contours and boundaries are a subset of edges and are associated with semantically meaningful entities [11], for example, the silhouette and outline of an object [12]. Boundary refers to the object borders in the image plane, corresponding to pixel ownership within a scene [13,14]. On the contrary, contour refers to the borders of a region within a given object [7].

To summarize, given an image, an edge detection algorithm aims to capture all the meaningful intensity discontinuities. It does not concern itself with pixel ownership or open and closed shapes. These are the domain of contour and boundary detection to eliminate edges that do not correspond to salient features of objects and shapes. Hence, as edge, contour and boundary detection are primary operations, each process will improve different computer vision or image processing tasks. For instance, edge-maps are more suitable for applications with a requirement for a finer level of detail (e.g., image enhancement [15], super-resolution [16], image generation and reconstruction [17], etc.). In other tasks like scene /semantic segmentation and saliency detection, boundary maps would be more beneficial. In this article, we focus on edges.

1.2. Structure of data matters

Currently, deep learning is the primary approach in the task of edge detection. It is well established that the quality of the training dataset is a critical factor within this framework [18]. To this end, state-of-the-art heavily relies on the Berkeley segmentation dataset (i.e., BSDS300 [14] and BSDS500 [1]) even though image segmentation was the original objective of this dataset and its refined version turned into a benchmark dataset of boundary detection. Most studies train their models on BSDS300 and validate them on BSDS500.

Each image contains boundary annotations drawn by at least five different participants. In many cases, there is a large discrepancy among different human-drawn boundaries (see Fig. 2). This issue is partially overcome by computing the loss function over a consensus ground truth, e.g., [4,5,19]. In the consensus stage, pixels that are annotated by more than two participants are considered true edges, otherwise discarded. This facilitates the convergence of the training process [19]. Nevertheless, this consensus stage also filters out a large portion of true edges. This is visually evident in Fig. 2. For instance, all the zebra stripes vanish even with a moderate consensus of two participants. A similar observation can be made in the case of the red car. This problem exists in many images of the BSDS.

Hence, the networks trained on the BSDS learn to become a *boundary-detector* rather than an *edge-detector*. The distinction between the two can be of great importance, for example, when edges are the building blocks of another computer vision application, as in [20]. This issue is addressed in this study. We present a benchmark dataset of edges that allows for an accurate evaluation of edge detection algorithms.

1.3. Contributions

In this paper, we address the aforementioned issues by proposing an end-to-end deep learning approach for the task of edge detection. Our primary objective is to identify all true edges in any given image. To demonstrate this, we train a network only on one dataset and evaluate it on several publicly available datasets. We show that our model obtains a higher degree of generalization in comparison to the other algorithms of state-of-the-art. We conduct an exhaustive quantitative evaluation on two datasets with annotated edges (i.e., MDBD [7] and BIPED). On other datasets with contour/boundary annotation, we qualitatively show the robustness of our model. In the current work, we extend our previous conference paper [21] in the following aspects:

- Presenting a detailed description of the proposed architecture, DexiNed (Dense eXtreme Inception Network for Edge Detection), and thoroughly analysing the impact of its different components.
- Boosting the degree of annotation, specifically for fine-scale edges, in the proposed benchmark dataset of edge detection, referred to as BIPED (Barcelona Images for Perceptual Edge Detection).[5]
- Establishing a common evaluation benchmark among four state-of-the-art networks by interchanging the training and validation set between two datasets of edge detection (BIPED and MDBD).
- Improving the loss functions from BDCN [5] by modifying the λ values to better balance the portion of positive and negative samples in each DexiNed output and incorporating averaging at the level of pixels.

The rest of the paper is organized as follows. Section 2 summarizes the most relevant works on edge detection. Then, the proposed approach is described in Section 3. The acquired dataset and ground truth generated for training and testing, as well as the datasets used for validating the proposed DexiNed architecture, are presented in Section 4. In Section 5, quantitative and qualitative details are summarized. Finally, conclusions are given in Section 7.

2. Related work

There is a large body of literature on edge detection (for review see [10] and [11]). In this section, a set of representative algorithms are detailed. They can be broadly categorized into four groups: *i*) driven by low-level features; *ii*) brain-inspired; *iii*) classical learning-based; and *iv*) deep learning.

Algorithms driven by low-level features: The early edge detection algorithms such as Sobel [22], Robert and Prewitt [23] are based on the first-order derivative. The input image is smoothed by the linear local filters, normally, a set of two orientations (horizontal and vertical). In the end, edges are detected by thresholding. These operators advanced by considering the second-order derivative, where edges are detected by the extraction of zero-crossing points. In the mid-eighties Canny [9] proposed an edge detector by grouping three different key processes. Despite being antiquated, these approaches are still used in some modern computer vision applications. Many variants of the above-mentioned algorithms have been present in the literature.

Brain-inspired algorithms: This set of edge detection algorithms relies in known mechanisms of the biological visual system. Since the 1960s, experiments on monkeys and cats have significantly advanced our understanding of the Primary Visual Cortex (V1). For instance, it was discovered that a group of simple cells are responsive to edges [24]. The authors in [24] develop a mathematical

[5] Dataset and code: https://github.com/xavysp/DexiNed.

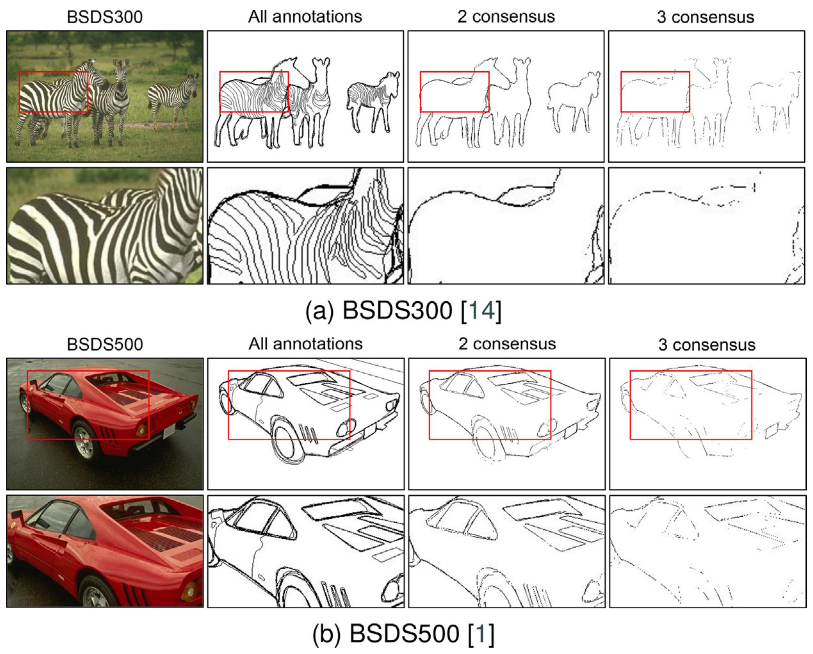

Fig. 2. A careful examination of the BSDS benchmark datasets for two sample images and their corresponding ground truths. There is a large discrepancy among human-drawn "edges". Enforcing consensus among two/three annotations results in the elimination of true edges. This issue demonstrates that BSDS is a suitable dataset for evaluating boundaries but **not** edges.

model to simulate the human retinal vision by Gaussian derivatives; since then, many algorithms have been implemented resembling the edge detection of the biological neurons in real-world images. For example, in [25] a special line weight function is introduced for edge enhancement, which consists of a combination of zero and second-order Hermite functions. Later on, in [13] Gabor energy maps have been proposed to recreate the non-classical receptive field of V1 for contour detection. This proposal has been evaluated with 40 images—this dataset could be considered the first annotated ground truth proposed in the literature to validate contour detection algorithms. This approach has been later on improved by [26] by modeling spatial facilitation and surround inhibition with local grouping functions and Gabor filters, respectively. Recently, in [27], a modulation of double opponent cells is performed to extract more complex edge properties from color and texture. This was complemented by incorporating the second visual area and accounting for feedback connections [28]. Subsequent experiments demonstrated that modeling various surround modulations, typical in neurons of the visual cortex, boosts the performance of edge detection [28]. A combination of three Gabor filters at different scales was proposed in [7], followed by a PCA and a machine-learning classifier. This leads us to the next group, which is learning-based.

Classical learning-based algorithms: The challenge of edge detection in natural scenes has motivated the development of the learning-based algorithm. One of the first learning approaches for boundary detection has been presented in [14], which uses different filters to extract gradients from brightness, color and texture. The authors then train a logistic regression classifier to generate a Probabilistic boundary (Pb). Later on, different variants of Pb detectors have been proposed, like the one from [29]. These new proposals focus on global information besides using local information like the original approach. They are referred to as globalized Probability of boundary (gPb). In these approaches, the gradients are computed by three scales for each image channel individually. In [30], a conditional random field-based approach has been proposed. Its maximum likelihood parameters are trained given the image edge annotations. This model captures the continuity and frequency of different junctions by a continuity structure. On the contrary, the usage of a sparse code gradient has been proposed in [31]. In this case, the patches are classified by a support vector machine. Lastly, the usage of a Bayesian model has been considered in [32]. edge-maps have been predicted by a Sequential Monte-Carlo-based approach using different gradient distributions. The random forest framework is also considered for edge detection, where each tree is trained independently to capture complex local edge structures [33].

Deep learning algorithms: During the last decade, CNNs have become the standard model in computer vision. It has been shown that the internal representation of object classification networks relies on edges [34]. The first CNN-based model for edge detection has been proposed in [35]. In this approach, like in most DL models, given an annotated edge-map dataset, the network learns to predict edges from an RGB image. Since this original work, several methods have been proposed [11]; the backbone of most architectures is VGG16 [36]. From this architecture, just the convolutional layers are used (13 convolutional layers). The parameters of this architecture are obtained by training the network in the ImageNet dataset as a classification problem. Then, the network is fine-tuned by training it on the datasets conceived for boundary detection. This is the case of HED [19], RCF [4], BDCN [5], CATS [6], Chrnet [37] and many others, our proposal does not follow this procedure for the training stage, therefore, DexiNed uses less time for the training and the testing but still reaches the state-of-the-art results. In HED [19] a multi-scale network together with a deep supervision technique is configured. Hence, each VGG block has as an output an edge prediction, which can be considered as a space scale representation. RCF [4] uses the same configuration as HED, but instead of getting edges from each VGG block, it extracts edges from each layer on the blocks. On the contrary to previous approaches, in Chrnet [37] different refinement blocks are used to merge the outputs from the third backbone block till the block five; it also considers two outputs from the two last refinement blocks to predict the final edge-map. With this procedures the edge-map of Chrnet is sharper and cleaner than the previous approaches that use the same training procedure and architecture as

HED. Even though the quantitative performance of the aforementioned methods overcomes the state-of-the-art of classical learning algorithms, the predicted edge-maps are not as sharp as expected, and somehow detected edges are coarse. Hence, Context-Aware Tracing Strategy (CATS) [6] is proposed to improve the edge-map crispness. Moreover, with the last advances of transformer learning models EDTER—Edge Detection with TranformER is proposed [38], the main drawback of this proposal is that it has more than 400M parameters, which will require more computation resources.

Although most of the DL-based models for edge detection have been based on the usage of VGG16, there are a few approaches that are based on other models, like ResNet50 [39] which is used in [4]. With these variants, tinny improvements have been obtained, about 1%. Despite that, in the current work VGG16 is going to be described as the standard architecture used in the edge detection approaches. Most of the models based on VGG16 outperform traditional edge detection methods in the standard edge detection datasets such as BSDS [1], NYUD [40], PASCAL-CONTEXT [41] and MDBD— Multicue Dataset for Boundary Detection [7]. Despite this, even after obtaining high-ranked performance, some of these methods still present drawbacks or lack of generalization or coarser predicted edge-maps, see results in Fig. 1. For instance, it can be observed that in some cases the generated edge-maps do not predict the human perceptual edges. Contrary to those models, our proposal predicts cleaner, thinner and more accurate edges using the same fusion strategy and slightly modified loss function from HED, RCF, and BDCN. Additionally, new lighter models like LDC (Lightweight Dense CNN for Edge Detection) [42] have being proposed inspired on the Dexined approach.

3. Proposed approach

This section presents the proposed architecture and the loss function used to train the model proposed in the manuscript.

3.1. Dense extreme inception network for edge detection

DexiNed is designed to allow end-to-end training without the need to weight initialization from pre-trained object detection models, like in most DL-based edge detectors. In our previous work [43] we observed that the edge features computed in shallow layers are often lost in the deeper layers. This inspired us to design an architecture similar to Xception [44] but with two parallel skip-connections that materialize on all the edge information computed across different layers. DexiNed can be interpreted as a collection of two sub-networks (see Fig. 3): the dense extreme inception network (Dexi) and the upsampling network (USNet). Dexi receives an RGB image as input processing it in different blocks, whose feature-maps are fed to USNet.

3.2. Dexi

The Dexi architecture contains six blocks acting similarly to an encoder. Each block is a collection of smaller sub-blocks with a group of convolutional layers. Skip-connections couple the blocks, as well as their sub-blocks (depicted in light gray and blue rectangular shapes in Fig. 3). The feature-maps generated at each of the blocks, are fed to a separate USNet to create intermediate edge-maps. These intermediate edge-maps are concatenated to form a stack of learned filters. At the very end of the network, these features are fused to generate a single edge-map.

Each sub-block (blue rectangles in Fig. 3) constitutes two convolutional layers (the number of kernels is specified on the right side of the blue rectangles). All kernels are of size 3×3. In the very first block, convolutions are with a stride of 2, hence $s2$ in its name. Each convolutional layer is followed by batch normalization

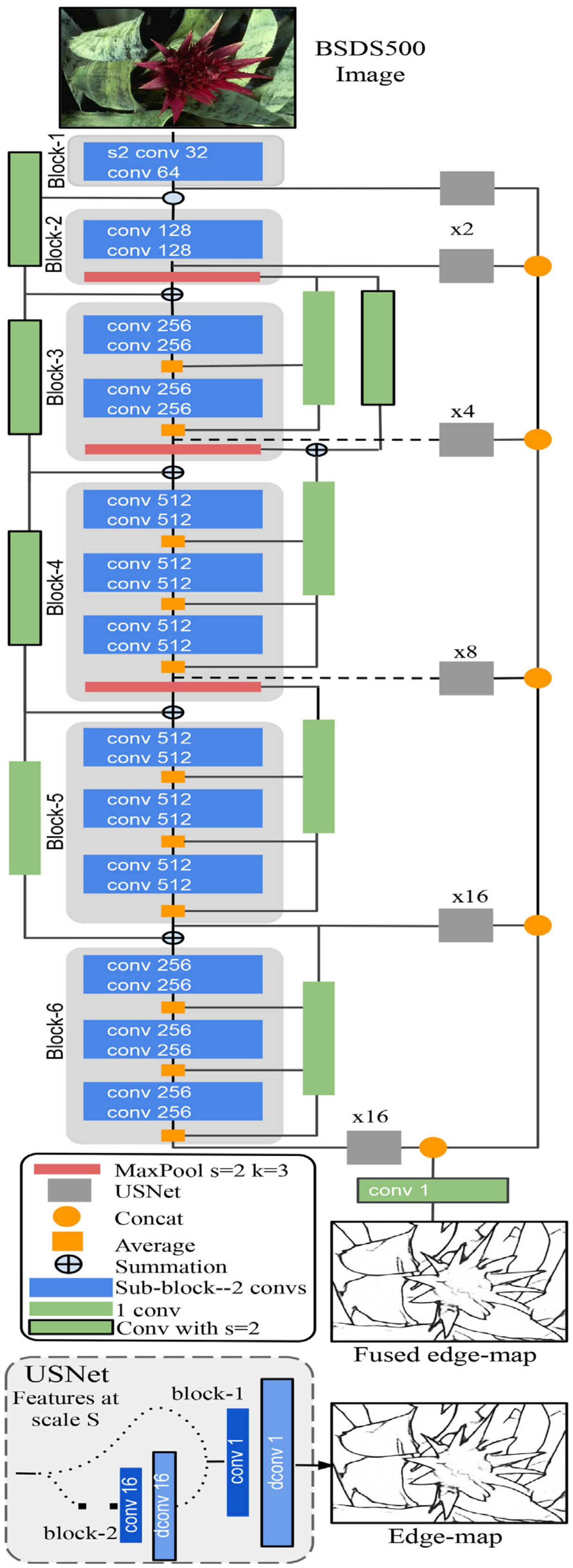

Fig. 3. Flowchart of proposed architecture (DexiNed). It consists of two building blocks a *Dense Extreme Inception Network* and an upsampling Net (*USNet*).

and a rectified linear unit (ReLU). From Block 3 (light grey rectangles) the last convolutional, of the last sub-block, does not contain the ReLU function. Rectangles in red are max-polling operators with a 3×3 kernel size and stride of 2.

The multitude of convolutional operations over the depth of processing causes important edge features to vanish [43,45]. To avoid this issue we introduce parallel skip-connections, inspired by He et al. [39] and Zhang et al. [46]. From the third block (Block-3) forward, the output of each sub-block is averaged with another skip-connection termed *second skip-connections—SSC* (green rectangles on the right side of Fig. 3). After the max-pooling operation, these SSC average the output of connected sub-blocks before summation with the *first skip-connection—FSC*, green rectangles on the left side of Fig. 3. In parallel to this, the output of max-pooling layers is directly fed to subsequent sub-blocks. For instance, the sub-blocks of block-3 receive input from the first max-pooling; and the sub-blocks of block-4 receive input with a summation of the first and second max-pooling.

3.3. USNet

The upsampling network (USNet) is a conditional stack of two blocks. Each one of then consists of a sequence of one convolutional and deconvolutional layer that up-sample features on each pass. The block-2 gets activated only to scale the input feature-maps from the Dexi network. This block is iterated until the feature-map reaches a scale twice the size of the GT. Once this condition is met, the feature-map is fed to the block-1. The block-1 processes the input with a kernel of size 1×1, followed by a ReLU activation function. Next, it performs a transpose convolution (deconvolution) with a kernel of size $s \times s$, where s is the input feature-map scale level. With the last deconvolution of block-1, the feature-map reaches the same size as the GT. The last convolutional layer does not have an activation function.

We considered three strategies for upsampling blocks: bi-linear interpolation, sub-pixel convolution and transpose convolution. This is an influential factor in generating thin edges, a desired feature that, for example, enhances the visualization of edge-maps. We considered this point in the design of DexiNed and a detailed evaluation of it is presented in Section 5.

3.4. Loss functions

The DexiNed can be formalized as a regression mapping function $\eth(:)$; in other words, $\hat{Y} = \eth(x, y)$, where the input image is $x \in \mathbb{R}^{m \times d \times c}$ and $y \in \{0,1\}^{m \times d \times 1}$ is the corresponding ground truth edge-map, m, d, and c are image sizes—c is set to 3 due to the RGB channels used in the model's training. The output $\hat{Y}$ corresponds to a set of predicted edge-maps, $\hat{Y} = [\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_N]$, $\hat{y}_n$ is the edge-map predicted from the n DexiNed predictions, N is the number of outputs from DexiNed (rectangles in gray, Fig. 3) and the last fused edge-map. $\hat{y}_N$ corresponds to the result of the last convolutional layer, which fuse the concatenation of all $\hat{y}_n$ (rectangles in gray), when the weight is initialized by $1/(N-1)$. Note that $\hat{y}_N$ has the same size as y. The loss function in our preliminary work [21] was considered from HED [19] (weighted cross-entropy). In the current work loss functions from HED, RCF and BDCN have been evaluated; after such evaluation the BDCN loss function, $\mathcal{L}$, has been selected with a little modification as depicted below:

$$w_{(y>0)} = 1\frac{y_-}{y_+ + y_-};\, w_{(y<=0)} = 1.1\frac{y_+}{y_+ + y_-},$$
$$l_n = \frac{1}{m \cdot d}(-w[y \cdot \log\theta(\hat{y}_n) + (1-y)\cdot \log(1-\theta(\hat{y}_n))]) \tag{1}$$

then,

$$\mathcal{L} = \sum_{n=1}^{N} \lambda_n \cdot l_n \tag{2}$$

where w is the weight of the cross-entropy loss, y_- and y_+ denote negative and positive edge samples in the given GT, respectively. θ is the sigmoid function. The λ is a set of hyper-parameters used to balance the number of positive and negative samples. Those values will be given in the implementation details section.

4. Datasets

This section details the datasets used for training and evaluation of the proposed approach as well as compares it with state-of-the-art approaches.

4.1. Training with BIPED dataset

The second contribution of the current work is the update of the original benchmark dataset, referred to as Barcelona Images for Perceptual Edge Detection (BIPED), which has been initially presented in our previous work [21]. The update presented in the current work is detailed below. All the images contained in the seminal work have been processed again. The BIPED dataset contains 250 real-world images, of 1280×720 pixels, of urban environments. Ground truth (GT) edge-maps have been generated using the crowdsourcing tool Labelbox[6], by a manual annotation; as stated in [8], there are three methods to generate GT for edge detection datasets: *i*) the synthetic images generated from a previously created edge-map, *ii*) manual edge annotation from an image, and *iii*) GT preparation from a consensus and agreement criterion given edge-maps generated from different edge-detectors. Figure 4 presents an illustration from the BIPED dataset together with its annotated edges. The pink lines in Fig. 4(*b*), which are polylines, correspond to edges used to draw open contours in the scene; on the contrary, blue lines, which are polygonal lines, correspond to closed contours in the scene. Figure 4(*c*) shows the edge-map corresponding to the Fig. 4(*a*). The annotation process has been applied to the whole dataset. To generate high-precision edge-maps, each image was processed following four steps:

1. Computer vision experts annotate all the edges on each image, just one annotation per image.
2. The administrator (also a computer vision expert) reviews the output of the previous step.
3. The obtained GTs, are used to train the HED model [19] and validate the sanity of predicted edges.
4. Considering the results of HED, the administrator crosscheck the entire dataset correcting mistakes and adding new annotations. This version was presented in [21].
5. With the updated version of LabelBox, improvements in Zoom tool make us to see additional details that were not considered in the previous version, due to the lack of deep appreciation on BIPED images. This make us to add more annotation in almost the whole images. The differences between initial annotations (BIPEDv1) and current one (BIPEDv2) can be appreciated in Fig. 5; new annotation tools allow to draw very tinny edges from the given images.

The administrator remains the same person throughout the whole process to ensure the same set of criteria is applied to all annotated images. The BIPED is a suitable dataset to benchmark the results of edge detection algorithms. To this end, we have released it publicly for the benefit of the research community.

[6] https://labelbox.com/

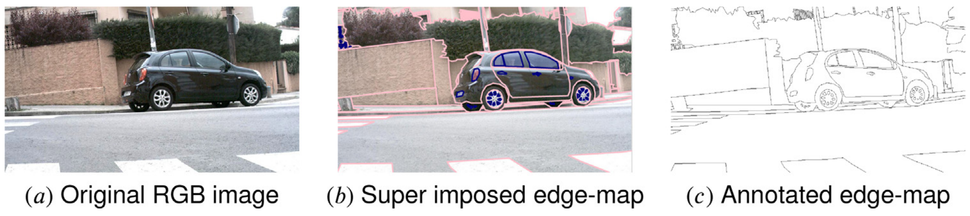

Fig. 4. A sample image from the BIPED dataset.

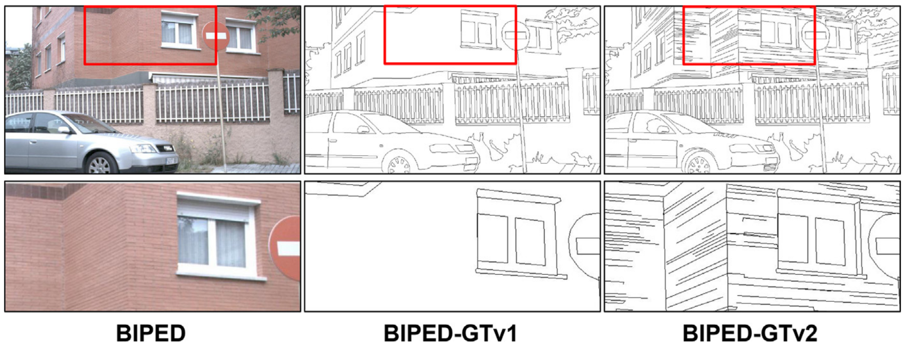

Fig. 5. Difference of BIPED GT previous annotation and the proposed version, BIPED-GTv2.

4.2. Testing with benchmark datasets

A comprehensive qualitative evaluation of the proposed model has been conducted on the datasets most widely used in the literature for edge, contour, and boundary detection, namely MDBD [7], BSDS500 [1], BSDS300 [14], NYUD [40], PASCAL-CONTEXT [41] (referred hereinafter to as PASCAL), CID [13], and our proposed BIPED. Within this list, MDBD is the most relevant dataset to the presented model, given its annotations correspond to true edges. The other datasets are more appropriate for the task of boundary and contour detections. Despite that, results on all these datasets are presented to show the effectiveness of our approach and have a better comparison to the state-of-the-art.

MDBD: The Multicue Dataset for Boundary Detection was collected for psychophysical studies of object boundary perception in complex images. Multiple cues such as luminance, color, motion and binocular disparity have been considered in its design [7]. The MDBD is composed of short binocular video sequences of natural scenes (containing 10 frames per scene). Hundred high-definition images of size 1280×720 were extracted from these videos. Each image has been annotated by several participants (six times for edge detection and five times for boundary detection). The proposed DexiNed architecture has been evaluated with the edge annotations of this dataset. Usually, in this dataset, 80% of images are used for training, while the remaining are used to evaluate the learning-based algorithm. Following this, we randomly selected 20% of images to evaluate the performance of DexiNed. The models considered for quantitative evaluations have been trained again for a fair comparison with the 80% selected in the current work.

CID: The Contour Image Database is a set of 40 gray-scale images with their corresponding ground truth contours [13]. The size of the images in this dataset is 512×512. The main limitation of this dataset is related to the small number of annotated images. The proposed DexiNed architecture has been evaluated on the entire dataset, similar to previous works in the literature (e.g., [26,28]). This dataset is difficult for DL-based approaches due to its gray-scale nature and missed annotations in some of the provided images.

BSDS500: Berkeley Segmentation DataSet (BSDS), the first version has been published in 2001 [14] and consists of 300 images split up into 200 for training and 100 for validation, termed BSDS300; the last version [1], adds 200 new images for the testing. Every image in BSDS is annotated at least by 5 subjects, this dataset contains images of 481×321. This dataset is mainly intended for image segmentation and boundary detection, therefore, as it will be illustrated in the next sections, for the edge detection purpose some images are not well annotated. Generally, to evaluate the performance of a DL model in BSDS500, the new 200 images are used for testing while the BSDS300 is used for network training purposes. As DexiNed is trained only on the BIPED dataset, the qualitative evaluation depicted in Section 5 split up the results into two parts: BSDS300 and BSDS500. The images considered for qualitative comparison are taken from the test part of BSDS500 and the validation part from BSDS300.

NYUD: New York University Dataset is a set of 1449 RGBD images from 464 indoor scenarios, intended for segmentation purposes. This dataset is split up into three subsets—i.e., training, validation and testing sets. The testing set contains 654 images, while the remaining images are used for training and validation purposes. In the current work, just the testing set has been selected for evaluating the proposed model, since DexiNed has been trained just with BIPED. Although most of the images in NYUD are fully annotated for segmentation, there are a few of them with poor annotations. This fact (missing edges in some images) affects the quality of DL-based edge detection approaches.

PASCAL: The PASCAL [41] is a popular dataset used for segmentation with a wide variety of object categories. Currently, most of the major DL methods for edge detection use PASCAL for training and testing (e.g., [19,4]), due to its ground-truths corresponding to scenes different to the ones depicted in BSDS. This dataset contains 11,530 annotated images, however, just around 5% of randomly selected images (505) have been considered for testing DexiNed. Although the images in PASCAL have more diverse labeled data, most of the images are annotated only for a couple of objects even though the scene has a vast number of features.

5. Experimental results

This section first describes the metrics used for the evaluations; then, details about the implementation and settings of the proposed approach are provided. Finally, a large set of experimental results is presented together with comparisons with state-of-the-art approaches. Again, most of the non-edge detection datasets in the literature are used just for qualitative evaluation and, for edge detection, BIPED and MDBD approaches, are used for quantitative evaluations and comparisons with DexiNed.

5.1. Evaluation metrics

Edge detection algorithms can be evaluated following two approaches. *Indirectly* through their impact on other computer vision tasks [10]. *Directly* in comparison to human-drawn edges. In this work, we opted for the latter, which is a common practice in the evaluation of benchmark datasets. These evaluation metrics are as follow:

1. Optimal Dataset Scale (ODS) computed by using a global threshold for the entire dataset;
2. Optimal Image Scale (OIS) computed by using a different threshold on every image;
3. Average Precision (AP). The F-measure—$F = \frac{2 \times Precision \times Recall}{Precision + Recall}$.

5.2. Implementation notes

The proposed DexiNed architecture has been trained from scratch without relying on pre-trained weights. This is a unique feature of the proposed model. Most state-of-the-art networks depend on pre-trained weights of the ImageNet dataset. On average, DexiNed converges after 9 epochs (15 epochs in [21]) with a batch size of 8 using Adam optimizer and learning rate of 10^{-4}—decreasing at 10 and 15 epochs by a factor of 0.1, the weight decay considered for the training was 10^{-8}. The training procedure takes around 1 day in a TITAN X GPU input with color images of size 352 × 352, 480 × 480 for MDBD. The weights for fusion layer are initialized as: $\frac{1}{N-1}$ (see Section 3.4 for details on N). After a hyper-parameter search to optimize DexiNed, the best performance was obtained using kernels of size 3 × 3, 1 × 1 and s×s on the different conv and deconv layers, with Xavier initializer [47] in Dexi and normal distribution in the last conv and deconv layers of USNet. The λ of the loss function is set to [0.7, 0.7, 1.1, 1.1, 0.3, 0.3, 1.3].

We randomly selected 200 images of BIPED to train and validate DexiNed. The remaining 50 images were used for testing. To increase the number of training images, a data augmentation process has been performed as follows: i) given the high-resolution nature of BIPED images, each image is split in half along its width; ii) similarly to HED, each of the resulting images is rotated by 15 different angles and crop by the inner axis oriented rectangle; iii) images are horizontally flipped, and finally iv) two gamma corrections have been applied (0.3030, 0.6060). This augmentation process resulted in 288 images per each of the given images.

5.3. Architecture setup

This section presents evaluations on different DexiNed configurations. The first sub-section presents details on the upsampling methods and the merging process selected for estimating the edge-maps. In the second sub-section, the advantage of using skip connections are presented.

5.3.1. Upsampling methods and DexiNed predictions

Regarding the selection of the best upsampling method, in our preliminary work [21] three approaches were evaluated. In that evaluation it is shown that the upsampling performed by the transpose convolution with trainable kernels gives the best results. Hence, in the current work it is also selected as upsampling method. On the other hand, regarding the strategy used to merge the different outputs—Output1, Output2, Output3, Output4, Output5, Output6, DexiNed-f, DexiNed-a—the DexiNed architecture (Fig. 3) has been empirically evaluated as performed in [21] showing that the best results are obtained from DexiNed-f and DexiNed-a. DexiNed-f corresponds to the result obtained by the fusion process at the end of the DexiNed architecture (see Fig. 3), while DexiNed-a corresponds to the edge-maps obtained from the average of all DexiNed predictions (DexiNed-f included). It is shown in [21] that both merging strategies reach similar quantitative results; hence hereinafter results from both merging strategies are presented; in some cases, due to space limitations, just results from DexiNed-f are depicted, in those cases results are named as DexiNed.

5.3.2. Ablation study

This subsection presents a quantitative study of the critical DexiNed parts and settings. As our model is composed of two types of skip connections (rectangles in green on the left and right sides in Fig. 3) a study of different numbers of skip-connections is performed. It is presented in Fig. 6($left$)—$DexiNed0C$ does not use skip-connections, $DexiNed1C$ uses just one skip connection (left side in Fig. 3); $DexiNed2C$ uses the two connections. The usage of two skip-connections (DexiNed2C) improves the performance of the proposed architecture.

The proposed architecture has been also trained with different loss functions to evaluate its performance. Loss functions from the following approaches have been considered: HED [19], RCF [4], BDCN [5], and BDCNloss2 (it is a slightly modified function from BDCN [5]). As illustrated in Fig. 6(*right*) the modified BDCN loss function outperforms its counterparts with almost 1%. Finally, to choose the best DexiNed performance, we have evaluated its prediction in different epochs, up to 25 epochs. As it can be appreciated in Fig. 6(*middle*) the best performance is reached when 11 epochs are considered. In the following sections of this manuscript, the comparisons of DexiNed performance on the different edge detection datasets correspond to the architecture DexiNEd2C, with 11 epochs, and using the BDCNloss2 function, termed just as DexiNed.

5.4. Quantitative comparison

This section presents comparisons of DexiNed with the state-of-the-art approaches on edge detection datasets. Additionally, a comparison of the generalization capability in two datasets (i.e., MDBD and BIPED) is studied; in other words, a study that shows results of generalization from one dataset to another dataset is presented. As introduced in Section 1, BSDS [1] is not considered since this dataset cannot generalize results on edge domain. The state-of-the-art approaches for edge, contour, and boundary detection [11] have been selected for comparison with DexiNed; these approaches are the following: RCF [4], BDCN [5], and CATS [6]. In order to perform a fair comparison, these approaches have been trained on two datasets intended for edge detection—the MDBD and our BIPED datasets. It should be noticed that the same augmentation processes has been applied in all the cases; additionally, in the case of MDBD, all models have been trained with the same training set, instead of randomly selecting images from the MDBD dataset, as performed in most of the publications.

Table 1 and Fig. 7 show different evaluations for DexiNed (DexiNed-f and DexiNed-a) and the approaches from the state of the art mentioned above. Results from both scenarios are presented—trained and tested on the same dataset and cross-evaluations (i.e., trained in a dataset and evaluated in the other dataset). It can be appreciated that DexiNed reaches the best performance (in all three metrics ODS, OIS, and AP) when trained and tested in the same dataset. Furthermore, DexiNed reaches also the best result if trained in BIPED but evaluated in the MDBD dataset. On the other hand, regarding the dataset generalization capability, considering the ODS evaluation metric, we know that the best

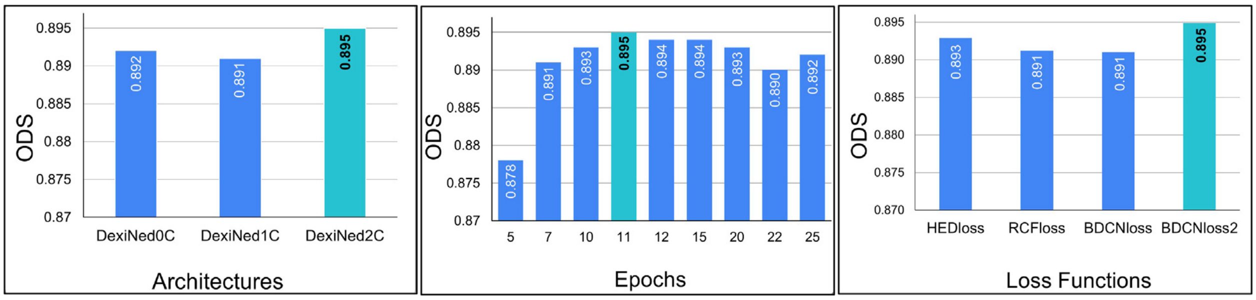

Fig. 6. (*left*) Evaluation of DexiNed architecture with (DexiNed1C: 1 connection; DexiNed2C: 2 connections) and without (DexiNed0C) skip-connections. (*middle*) DexiNed performance evolution during training. (*right*) Evaluation with different loss functions.

Table 1
Quantitative results: The performance of DexiNed model and BIPED dataset are compared to the last DL based models and the edge detection based dataset, MDBD [7].

Methods	Trained on	Tested on	ODS	OIS	AP	Tested on	ODS	OIS	AP
RCF [4] (2019)	MDBD [7]	MDBD [7]	.879	.888	.926	BIPED	.814	.828	.871
BDCN [5] (2020)			.887	.891	.793		**.854**	**.863**	.768
CATS [6] (2022)			.891	.899	.809		.813	.831	.791
DexiNed-f (Ours)			.891	.896	.930		.787	.807	.844
DexiNed-a (Ours)			**.894**	**.902**	**.951**		.789	.813	**.875**
RCF [4] (2019)	BIPED	BIPED	.849	.861	.906	MDBD [7]	.839	.854	.865
BDCN [5] (2020)			.890	.899	.934		.855	.864	.692
CATS [6] (2022)			.887	.892	.817		.837	.840	.496
DexiNed-f (Ours)			**.895**	**.900**	.927		**.863**	.871	.867
DexiNed-a (Ours)			.893	.897	**.940**		.862	**.874**	**.919**

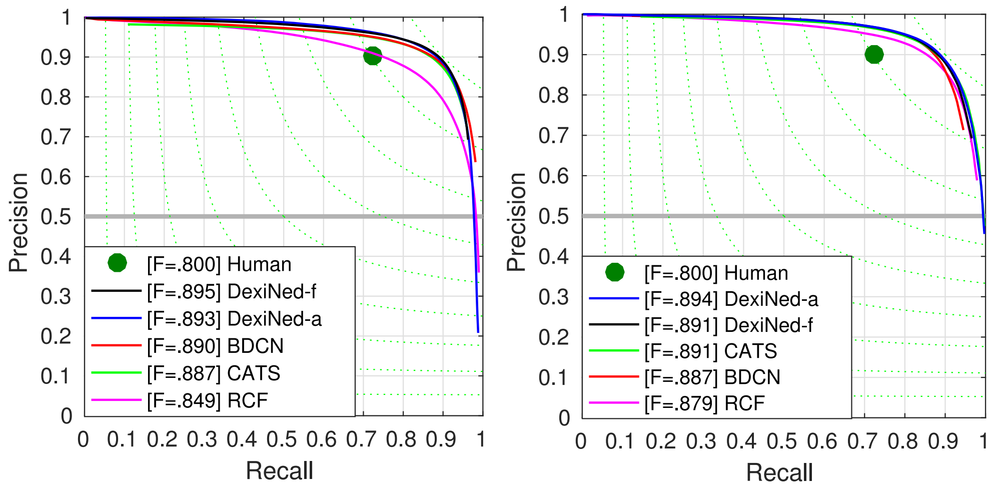

Fig. 7. (*left*) Precision-Recall curve of BIPED test set. (*right*) Precision-Recall curve of MDBD [48].

performance is reached when trained and evaluated in the same dataset. Hence, we propose to analyze the loss in performance when evaluated in another dataset different to the one used for training. This loss in performance is smaller if the different approaches are trained in BIPED, on average just 3,74% of the decrease in performance is appreciated when evaluated in MDBD; on the contrary, this loss in performance reaches an average of 7,14% if trained in MDBD and evaluated in BIPED. Our empirical observation suggests that the bad performance of the model trained in MDBD largely depends on the datasets used for training—i.e., amount of data, variability of data, and accuracy of annotations. In the case of the MDBD dataset, the main problem lies in the lack of consensus of the provided annotations as detailed in Section 1; the same problem happens during the annotation of BSDS—see illustration in Fig. 2. This consensus is required in order to be sure about the annotations. In order to have accurate information, datasets with multiple annotations require 2 or 3 consensus annotations, which enforce the removal of some edges. These missed edges affect the quantitative evaluations of trained models—since some true edges are not contained in the ground truth but, on the contrary, they may have been estimated by the networks. This bad performance is not only appreciated with DexiNed trained on MDBD but also in other architectures.

5.5. Qualitative comparison

This section presents just some illustrations as qualitative comparisons of the edge-maps predicted from all the models considered during the quantitative comparison in the previous section. Note that all these models have been trained on BIPED; Fig. 8 shows edge-maps of the evaluated architectures (trained on BIPED) when used in the datasets detailed in Section 4.2. Qualitative results, similar to those presented in Fig. 8, but trained on MDBD [7], are presented as supplementary material. Some comments and

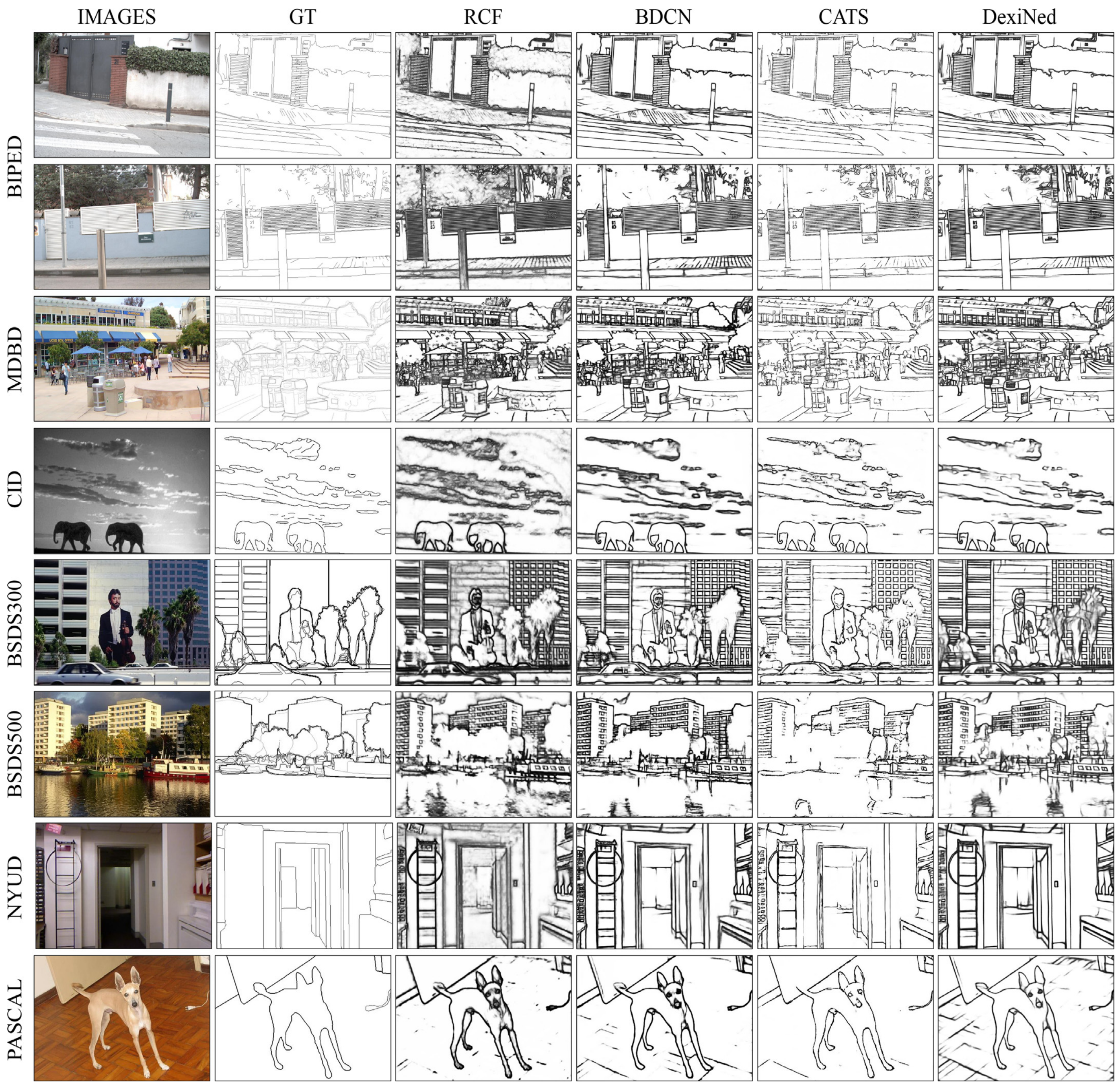

Fig. 8. The results of a few state-of-the-art architectures trained on the proposed BIPED dataset. Note that the ground truth of these datasets, except for BIPED and MDBD, correspond to boundary and segmentation tasks.

conclusions from the analysis of the obtained results are presented below:

- *BIPED:* For this dataset two illustrations are presented since this dataset is used for training the selected models considered in the comparisons. These two illustrations correspond to the best and the worst ODS results from DexiNed. As shown in Fig. 8, edge-maps predicted by BDCN, CATS, and DexiNed are clean and accurate representations. Perceptually, we can say that edge-maps predicted by BDCN and DexiNed contains more correctly detected edges. On the contrary, edge-maps predicted by CATS are cleaner but containing less predictions. As a conclusion, while searching for a model that predicts more edges but also less artifacts, DexiNed provide us those requirements without compromise them.
- *Other datasets: MDBD, CID, BSDS300, BSDS500, NUYD, and PASCAL.* As shown in Fig. 8, results in these datasets are also similar to predictions from the test set on BIPED. The CATS model shows cleaner edge-maps but with less edges than DexiNed. A more perceptual difference of this claim can be observe in the predictions of MDBD, CID, NYUD, and PASCAL. Looking at the last row in Fig. 8, which corresponds to PASCAL dataset, several edges on the floor of the scene have been detected by DexiNed, but none of the other models are able to detect them.

To finish, the second version of BIPED gives the generalization robustness to the models considered for comparison. Overall, in all the datasets presented in Fig. 8, thanks to the unique characteristics on BIPED, there are most perceptual edges predicted in the images. Concerning to the DexiNed architecture, the predictions given by this model are cleaner than from its counterparts and without compromising true edge lost, the training procedure is simpler that in any other DL based models considered for comparison. DexiNed does not need pre-training weights, it can converge in less time than other models but still reach the state-of-the-art results.

6. Discussion and limitations

DexiNed is an edge detection dedicated architecture. The principal objective of this article is to demonstrate that with a proper dataset there is no need for pretrained weights or hyper-parameters tuning at the level of the network's layers, as most of the SOTA models do. Table 1 shows that DexiNed reaches the best result, training from scratch in a short period of training time. With fewer hyper-parameters tuning, DexiNed reaches its peak performance in epoch 11, while previous studies report peak performance only after epoch 20. Although our proposal has more than double the number of parameters than BDCN [5] or CATS [6], its computational time is less in training and testing. This is possible thanks to a significant reduction of hyper-parameters search and the specific design of DexiNed's architecture.

DexiNed contains 35M parameters, despite its deep architecture, the edges are not eliminated in the feature-maps of the deeper layers (a common problem for deep edge detectors), and thanks to the contribution of USNet, the output of DexiNed contains a large number of true edges with cleaner outlines in comparison to the SOTA models. Another contribution of this manuscript is the slightly modified loss function from BDCN [5] and HED [19]. This modification helps to adapt and get a better cost of the training process while using the mini-batch size in terms of considering the mean loss instead of a sum. The final version allows for reaching the best result in MDBD and BIPED. As stated in the previous sections, we cannot quantitatively compare with BSDS, or other similar datasets used by the community, since those datasets do not have edge-level annotations since they are intended for contour or boundary detection.

As mentioned in Section 5.4, DexiNed has a good performance when trained in BIPED but evaluated in MDBD, somehow this show the good generalization of the proposed model. Unfortunately, there are not more edge detection datasets to evaluate this generalization feature.

The input color space of DexiNed (like most other deep networks) is RGB, whose channels are highly correlated in natural images (the average correlation for R, G and B channels in ImageNet is about 0.85 [49]). Due to this high intra-axes correlation, edges detect in each channel largely overlap. Contrary to this, our visual system decorrelates the input space into color-opponency, which facilitates detecting information (e.g., edges) in both chromatic and luminance channels. We propose this as future work to investigate the impact of input color space on the performance of DexiNed or other edge detection models.

7. Conclusion

This paper proposes a robust edge detection model that exhibits a great degree of generalization to new scenes. To this end, first, we present a benchmark dataset carefully designed for the task of edge detection named Barcelona Images for Perceptual Edge Detection (BIPEDv2). Second, we design a network with parallel skip-connections (DexiNed) that learns to detect edges without the need for ImageNet pre-trained weights. We show the generalization power of our approach by training the network on a single dataset and evaluating it on other benchmark datasets. In other words, by training DexiNed with only BIPEDv2 we can get competitive results in the MDBD dataset for edge detection. DexiNed is more than double in the number of parameters compared to the state-of-the-art architectures based on VGG16, nevertheless, it reaches similar FPS and does not lose edge feature maps in the deeper layers of the network. Overall, our results show the possibility of training a deep-learning model of edge detection from scratch in an end-to-end fashion. These findings open the future work in the exploration of smaller networks for the task of edge detection by reducing the number of hyper-parameters settings.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

https://github.com/xavysp/DexiNed

Acknowledgment

This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-22-1-0261; and partially supported by the Grant PID2021-128945NB-I00 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe"; the "CERCA Programme / Generalitat de Catalunya"; and the ESPOL project CIDIS-12-2022. A.A. was funded by Deutsche Forschungsgemeinschaft SFB/TRR 135 (Grant Number 222641018). X.S. was funded by Ecuador government institution SENESCYT (No. 2015-AR3R7694).

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, Contour detection and hierarchical image segmentation, Trans. Pattern Anal. Mach.Intell. 33 (5) (2011) 898–916, doi:10.1109/TPAMI.2010.161.
[2] M. Nabti, A. Bouridane, An effective and fast iris recognition system based on a combined multiscale feature extraction technique, Pattern Recognit. 41 (3) (2008) 868–879, doi:10.1016/j.patcog.2007.06.030.
[3] J.-Y. Zhu, T. Park, P. Isola, A.A. Efros, Unpaired image-to-image translation using cycle-consistent adversarial networks, in: International Conference on Computer Vision, 2017.
[4] Y. Liu, M. Cheng, X. Hu, J. Bian, L. Zhang, X. Bai, J. Tang, Richer convolutional features for edge detection, IEEE Trans. Pattern Anal. Mach. Intell. 41 (8) (2019) 1939–1946, doi:10.1109/TPAMI.2018.2878849.
[5] J. He, S. Zhang, M. Yang, Y. Shan, T. Huang, BDCN: Bi-directional cascade network for perceptual edge detection, Trans. Pattern Anal. Mach.Intell. (2020), doi:10.1109/TPAMI.2020.3007074. 1–1
[6] L. Huan, N. Xue, X. Zheng, W. He, J. Gong, G.-S. Xia, Unmixing convolutional features for crisp edge detection, IEEE Trans. Pattern Anal. Mach. Intell. 44 (10) (2022) 6602–6609, doi:10.1109/TPAMI.2021.3084197.
[7] D.A. Mély, J. Kim, M. McGill, Y. Guo, T. Serre, A systematic comparison between visual cues for boundary detection, Vision Res. 120 (2016).
[8] O. Li, P.-L. Shui, Noise-robust color edge detection using anisotropic morphological directional derivative matrix, Signal Process. 165 (2019) 90–103.
[9] J. Canny, A computational approach to edge detection, Readings in Computer Vision, Elsevier, 1987.
[10] D. Ziou, S. Tabbone, et al., Edge detection techniques-an overview, Pattern Recognit. Image Anal. C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii 8 (1998).
[11] X.-Y. Gong, H. Su, D. Xu, Z.-T. Zhang, F. Shen, H.-B. Yang, An overview of contour detection approaches, Int. J. Autom. Comput. (2018), doi:10.1007/s11633-018-1117-z.
[12] Y. Ming, H. Li, X. He, Contour completion without region segmentation, Trans. Image Process. 25 (8) (2016) 3597–3611, doi:10.1109/TIP.2016.2564646.
[13] C. Grigorescu, N. Petkov, M.A. Westenberg, Contour detection based on non-classical receptive field inhibition, Trans. Image Process. 12 (7) (2003) 729–739, doi:10.1109/TIP.2003.814250.
[14] D. Martin, C. Fowlkes, J. Malik, Learning to detect natural image boundaries using local brightness, color, and texture cues, Trans. Pattern Anal. Mach.Intell. 26 (5) (2004) 530–549, doi:10.1109/TPAMI.2004.1273918.
[15] M. Zhu, P. Pan, W. Chen, Y. Yang, EEMEFN: low-light image enhancement via edge-enhanced multi-exposure fusion network, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 13106–13113.

[16] H. Gupta, K. Mitra, Pyramidal edge-maps and attention based guided thermal super-resolution, in: European Conference on Computer Vision, Springer, 2020, pp. 698–715.

[17] P. Isola, J.-Y. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1125–1134.

[18] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444, doi:10.1038/nature14539.

[19] S. Xie, Z. Tu, Holistically-nested edge detection, Int. J. Comput. Vis. 125 (2017) 3–18, doi:10.1007/s11263-017-1004-z.

[20] R. Pourreza, Y. Zhuge, H. Ning, R. Miller, Brain tumor segmentation in MRI scans using deeply-supervised neural networks, in: International Conference on Medical Image Computing and Computer Assisted Intervention-Workshop, Springer, 2017.

[21] X. Soria, E. Riba, A. Sappa, Dense extreme inception network: Towards a robust CNN model for edge detection, in: Winter Conference on Applications of Computer Vision, 2020.

[22] I. Sobel, Camera Models and Machine Perception, Technical Report, Computer Science Department, Technion, 1972.

[23] A. Rosenfeld, A.C. Kak, Digital Picture Processing: Volume 1, 2nd ed., Morgan Kaufmann Publishers Inc., 1982.

[24] R. Young, The gaussian derivative model for spatial vision. I- retinal mechanisms, Spat. Vis. 2 (4) (1987).

[25] M. Basu, Gaussian derivative model for edge enhancement, Pattern Recognit. 27 (11) (1994) 1451–1461, doi:10.1016/0031-3203(94)90124-4.

[26] Q. Tang, N. Sang, T. Zhang, Extraction of salient contours from cluttered scenes, Pattern Recognit. 40 (11) (2007) 3100–3109, doi:10.1016/j.patcog.2007.02.009.

[27] K.-F. Yang, S.-B. Gao, C.-F. Guo, C.-Y. Li, Y.-J. Li, Boundary detection using double-opponency and spatial sparseness constraint, Trans. Image Process. 24 (8) (2015) 2565–2578, doi:10.1109/TIP.2015.2425538.

[28] A. Akbarinia, C.A. Parraga, Feedback and surround modulated boundary detection, Int. J. Comput. Vis. 126 (12) (2018) 1367–1380, doi:10.1007/s11263-017-1035-5.

[29] M. Maire, P. Arbeláez, C. Fowlkes, J. Malik, Using contours to detect and localize junctions in natural images, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2008.

[30] X. Ren, C.C. Fowlkes, J. Malik, Scale-invariant contour completion using conditional random fields, in: International Conference on Computer Vision, Vol. 2, IEEE, 2005.

[31] R. Xiaofeng, L. Bo, Discriminatively trained sparse code gradients for contour detection, Advances in Neural Information Processing Systems, 2012.

[32] N. Widynski, M. Mignotte, A multiscale particle filter framework for contour detection, Trans. Pattern Anal. Mach.Intell. 36 (10) (2014) 1922–1935, doi:10.1109/TPAMI.2014.2307856.

[33] P. Dollár, C.L. Zitnick, Fast edge detection using structured forests, Trans. Pattern Anal. Mach.Intell. 37 (8) (2015) 1558–1570, doi:10.1109/TPAMI.2014.2377715.

[34] A. Akbarinia, R. Gil-Rodríguez, Deciphering image contrast in object classification deep networks, Vision Res. 173 (2020) 61–76.

[35] Y. Ganin, V. Lempitsky, n^4fields: Neural network nearest neighbor fields for image transforms, in: Asian Conference on Computer Vision, Springer, 2014.

[36] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556(2014).

[37] O. Elharrouss, Y. Hmamouche, A.K. Idrissi, B. El Khamlichi, A. El Fallah-Seghrouchni, Refined edge detection with cascaded and high-resolution convolutional network, Pattern Recognit. 138 (2023) 109361, doi:10.1016/j.patcog.2023.109361.

[38] M. Pu, Y. Huang, Y. Liu, Q. Guan, H. Ling, EDTER: edge detection with transformer, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 1402–1412.

[39] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Conference on Computer Vision and Pattern Recognition, 2016.

[40] N. Silberman, D. Hoiem, P. Kohli, R. Fergus, Indoor segmentation and support inference from RGBD images, in: European Conference on Computer Vision, Springer, 2012.

[41] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, A. Yuille, The role of context for object detection and semantic segmentation in the wild, in: Conference on Computer Vision and Pattern Recognition, 2014.

[42] X. Soria, G. Pomboza-Junez, A.D. Sappa, LDC: lightweight dense CNN for edge detection, IEEE Access 10 (2022) 68281–68290, doi:10.1109/ACCESS.2022.3186344.

[43] X. Soria, A.D. Sappa, Improving edge detection in RGB images by adding NIR channel, in: International Conference on Signal-Image Technology and Internet-Based Systems, 2018.

[44] F. Chollet, Xception: deep learning with depthwise separable convolutions, in: Conference on Computer Vision and Pattern Recognition, 2017.

[45] W. Shen, X. Wang, Y. Wang, X. Bai, Z. Zhang, DeepContour: a deep convolutional feature learned by positive-sharing loss for contour detection, in: Conference on Computer Vision and Pattern Recognition, 2015.

[46] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, Y. Fu, Residual dense network for image super-resolution, in: Conference on Computer Vision and Pattern Recognition, 2018.

[47] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Y.W. Teh, M. Titterington (Eds.), International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research, Vol. 9, PMLR, Chia Laguna Resort, Sardinia, Italy, 2010, pp. 249–256.

[48] J. Shen, S. Castan, An optimal linear operator for step edge detection, Graph. Models Image Process. 54 (2) (1992).

[49] A. Akbarinia, R. Gil-Rodríguez, Color conversion in deep autoencoders, J. Percept. Imaging (2021).

Xavier Soria: received his BS degree in Educational Informatics from National University of Chimborazo in 2009, Ecuador, and the PhD degree in Computer Science from Autonomous University of Barcelona in 2019, Spain. Currently, He is an Adjunct Professor at National University of Chimborazo, Ecuador. His research interest includes computer vision and its applications.

Angel Sappa: received the Electromechanical Engineering degree from National University of La Pampa, Argentina, in 1995, and the PhD degree from the Polytechnic University of Catalonia, Spain, in 1999. Currently he is a Senior Researcher at the Computer Vision Center, Spain, and a Full Professor at the ESPOL Polytechnic University, Ecuador.

Patricio Humanante: received the Systems Engineer degree from the Polytechnic School of Chimborazo, Ecuador, in 1998, and the PhD degree in Training in the Knowledge Society from the University of Salamanca, Spain, in 2016. Since 1999, he is a Full Professor and Researcher at the National University of Chimborazo, Ecuador.

Arash Akbarinia: received BSc in Software Engineering from Göteborgs Universitet and MSc in Computer Vision jointly from Université de Bourgogne, Universitat de Girona, and Heriot-Watt University. He obtained his PhD from Universitat Autònoma de Barcelona. He is currently a research scientist at Justus-Liebig-Universität Giessen. His research interests include artificial and biological vision.