IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 3 June 2022, accepted 27 June 2022, date of publication 1 July 2022, date of current version 8 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3187838

RESEARCH ARTICLE

Have We Solved Edge Detection? A Review of State-of-the-Art Datasets and DNN Based Techniques

MUHAMMAD MUBASHAR[1], NAEEMULLAH KHAN[2], ABDUR REHMAN SAJID[1], MUHAMMAD HASHIM JAVED[1], AND NAVEED UL HASSAN[1], (Senior Member, IEEE)

[1]Department of Electrical Engineering, Lahore University of Management Sciences (LUMS), Lahore 54792, Pakistan
[2]Department of Engineering Sciences, University of Oxford, Oxford OX1 3PJ, U.K.

Corresponding author: Naveed Ul Hassan (naveed.hassan@lums.edu.pk)

This work was supported in part by the Syed Babar Ali School of Science and Engineering (SBASSE), Lahore University of Management Sciences, Dean's Pool; and in part by the Natural
Science Foundation of China under Grant 61850410535.

ABSTRACT Recent Deep Neural Networks (DNNs) based edge detection methods claim to beat human performance on small scale datasets like BSDS500. But is the problem of edge detection really solved? To answer this question, we review the existing dataset limitations as well as the generalization capabilities of the proposed architectures. To this end, we develop a Synthetic Textured Masks Dataset (STMD) that contains 28,000 grayscale images. The performance of several edge detection methods is severely degraded on STMD. To further validate these results we propose a baseline Single Scale Feed Forward Edge Detector (SFED), which is a simple 9-layer feed-forward convolutional neural network with no pooling layers. The performance of SFED is better than most state-of-the-art architectures on BSDS500 and is superior to all the compared architectures on STMD. These results show that most of the architectural advancements of existing architectures are at the cost of generalizability where if we change the dataset set distribution (both training and testset), the performance becomes significantly degraded and therefore the problem of edge detection is still far away from being solved. There are also severe limitations of existing datasets in the field, and STMD provides a framework for designing and testing better edge detection architectures for novel application areas, such as, medical imaging and self-driving cars.

INDEX TERMS Computer vision, edge detection, segmentation.

I. INTRODUCTION

Edge detection is one of the classical problems of computer vision. It is important for several high level vision tasks like segmentation [1], detection [2], recognition [3], and photo-sketching [4]. In recent years, this problem has regained significance because of its utility in practical applications, such as, self-driving cars [5], [6], augmented reality [7], [8], image colorization [9]–[11] and medical imaging [12], [13].

In [5] edge detection is used to assist self-driving cars with lane detection. In [6] the authors propose edge detection to detect road obstacles like speed bumps, potholes, and other objects for autonomous vehicles. Edge detection can also be used to precisely identify buildings in augmented reality applications [7]. In [8] contour-based methods are used to detect and track objects in videos. Authors in [9] use edge detection to colorize grayscale images to reduce bleeding across boundaries. Similarly, authors in [10] utilize edges to generate new colored icons and to change the color of old ones.

In the medical imaging area, [12] uses edge detection for the extraction of important features like length, width, and angle of blood vessels in the retina of the human eye. These features are then used for screening and diagnosis of diseases including hypertension, arteriosclerosis, glaucoma, and diabetic retinopathy. Furthermore, the segmentation of

The associate editor coordinating the review of this manuscript and approving it for publication was Mohammad Shorif Uddin.

blood vessels in eye images can be used for computer-assisted surgery and biometric identification. In [13], authors employ edge detection for tumor marking in mammography images to classify them as benign or malignant. It is interesting to note that medical images are primarily grayscale textured mages where the change in textons of different regions encode semantically meaningful patterns. Therefore, edges hold great semantic importance for the segmentation of important features from the background. This principle has been proposed in [14] for brain tumor segmentation in MRI scans of patients.

The images in standard edge detection datasets mostly lack textures [15]. Therefore, the performance of edge detection algorithms for large textons images remains relatively understudied. In this work, we bridge this research gap and we also propose a technique to create large-scale synthetic edge detection dataset with accurately marked ground truth.

A. OUR CONTRIBUTIONS

In this paper, we undertake a study of state-of-the-art in edge detection and its performance. The objective is to verify whether the edge detection problem is indeed solved as claimed for recent DNN based methods. It is important to point out that these methods have been trained and tested on small scale datasets like BSDS500 [15]. To this end, we make the following contributions.

- We introduce a Synthetic Textured Masks Dataset (STMD) pipeline, which allows us to create a complex textured dataset of 28,000 images with known ground truth.
- We propose a baseline Single Scale Feed Forward Edge Detector (SFED) architecture, which consists of a simple 9-layer feed-forward convolutional neural network with no pooling layers (to avoid complexity).
- We train SFED and several state-of-the-art edge detection architectures on STMD and BSDS500 datasets. Various experiments are conducted, and qualitative and quantitative results are obtained to test the generalization and performance claims of available architectures.

The rest of the paper is organized as follows. In Section II we review the literature and discuss various edge detection algorithms and datasets. In Section III we provide the details of STMD and SFED. In Section IV, the quantitative and qualitative experiments are presented. The paper is concluded in Section V.

II. LITERATURE REVIEW

In this section, we review edge detection methods and datasets.

A. EDGE DETECTION METHODS

The earliest edge detection methods were based on image gradients. Various hand-crafted methods were proposed to improve the detection of edges [19]–[22]. These methods showed some promise but were unable to capture the vast array of edges present in natural scenes, especially in the presence of textured regions. In [20] Canny edge detector was proposed, which applied Gaussian smoothing to the image followed by gradient and thresholding operators. One of the most successful classical hand-crafted descriptors approaches was presented in [15], where a set of Gabor filters was used to provide the feature, which was fused to achieve the edge maps.

Another set of methods for edge detection relies on region based segmentation. In these methods, regions are segmented and the edges are detected as boundaries of these regions. Popular among the region based methods are [23]–[25]. Generally, the results for regions based methods on edge metric are not on par with that of the classical edge detection methods. However, on challenging textured datasets where the appearance of regions form an important cue for boundary detection, region based methods outperform edge based methods [26].

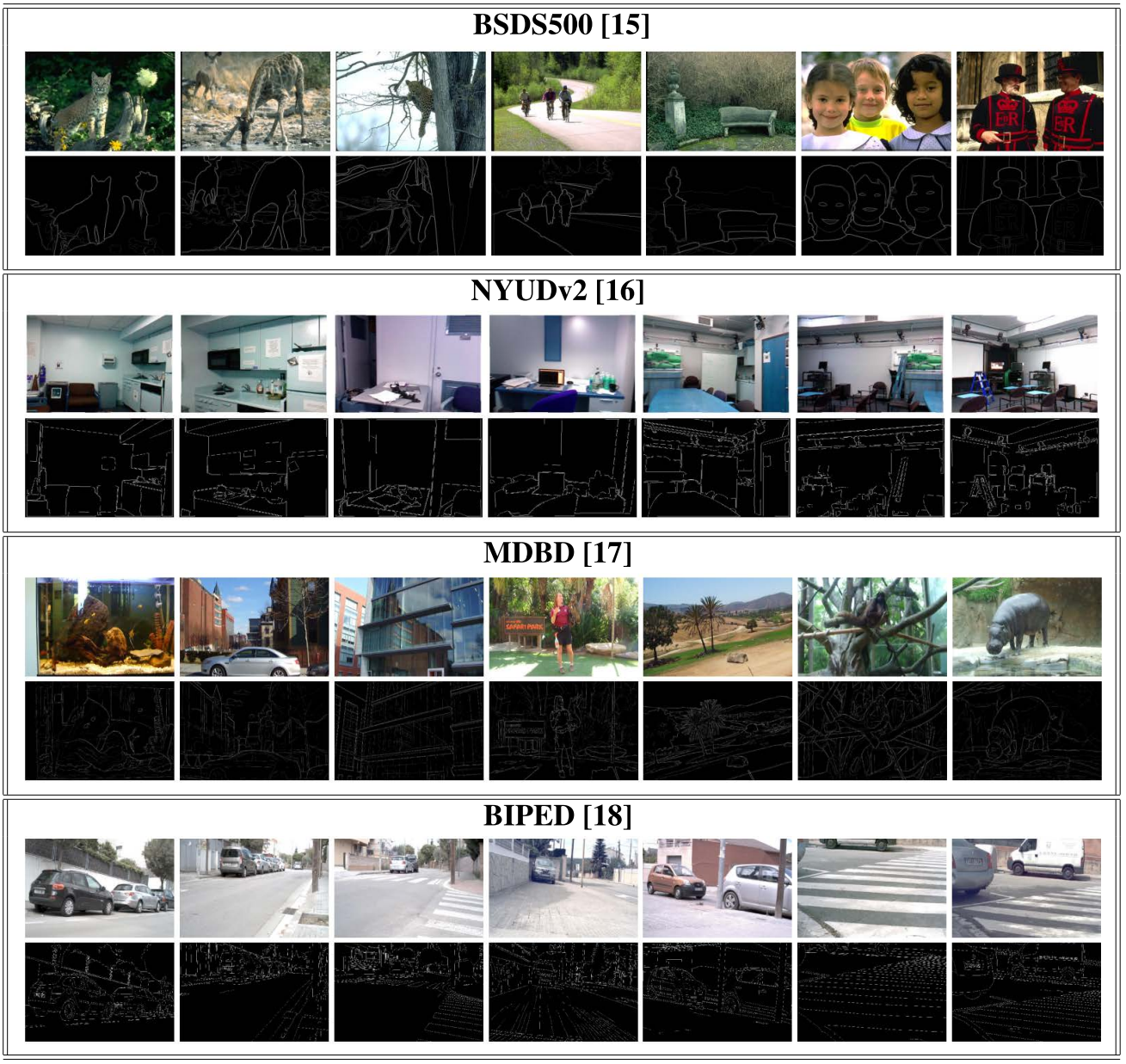
Instead of considering edges as high intensity points, some authors have also proposed frequency domain techniques where images are considered as being formed by a Fourier series and features are assumed to lie at maximum phase congruency points [27]. Normalized cuts methods have also been used for edge detection. These methods use hand crafted features like a custom Gabor filter and then employ normalized cuts followed by post processing steps like edge thinning on the edge maps [15], [28]. Prior to DNN techniques, these methods were considered state-of-the-art in the edge detection field.

The first Deep Learning method for edge detection was proposed in [29]. This method, called Holistically-Nested Edge Detection (HED), used a multi-scale model with VGG16 (Visual Geometry Group - 16 Layer CNN) backbone [30] to learn edges. Inspired by HED, the authors in [31] proposed a new model by adding deep supervision to improve the directness and transparency of the learning network. Another method that was also inspired by HED addressed the problem of blurred boundaries [28]. By adding a refinement module to HED and replacing bilinear interpolation with sub-pixel convolution they were able to obtain localized and crisp boundaries on the BSDS500 dataset. Another similar approach was proposed in [32], where the authors tried to predict more localized and crisp edges by using dice loss along with the weighted binary cross-entropy loss. In a more recent work [33] called Bi-Directional Cascade

TABLE 1. Datasets available for training and evaluation of edge detection techniques.

Dataset	Year	Images	Important Features
BSDS500 [15]	2011	500	Manually annotated by multiple annotators for segments and edges.
NYUDv2 [16]	2013	1449	Includes Depth information and multi-class labels.
MDBD [17]	2016	100	Both edge and boundary annotations included.
BIPED [18]	2020	250	Highly-detailed ground-truth.

TABLE 2. Some sample images and their ground truths from available datasets that are used for training and evaluation of edge detection techniques.



Network (BDCN), the authors introduced a Scale Enhancement Module (SEM) that uses dilated convolution to generate multi-scale features. SEM helps in generating less repetitive edges. Moreover, the BDCN model is bi-directional and it produces output at each layer to learn a multi-scale representation.

In Richer Convolutional Features (RCF) [34], the authors make use of multi-scale and multi-level knowledge to detect edges by adding convolutional features in a holistic framework. In Dense Extreme Inception Network (DexiNed) [18], which is inspired by HED [29] and Xception [35] networks, the authors presented a new dataset along with the model called inception network where RGB image is fed into the network. Then the learned feature maps were up-sampled to generate thin edge maps.

Semantic edge detection methods try to uniquely identify the edges of each semantically meaningful class. Many state-of-the-art methods use Fully Connected Networks (FCNs) but due to delicate structures of edges, FCNs are vulnerable to misaligned labels. The authors in [36] tried to address the misaligned labels issue through simultaneous learning and edge alignment. This method was further improved in [37] where authors proposed a boundary thinning layer together with a loss function to produce thin and precise edges. Feature extractor with a normalizer and adaptive weight fusion module was introduced in [38]. The authors in [39] combined both semantic boundary detection task and semantic segmentation task into a joint learning framework with an iterative pyramid context module. Duality loss was proposed which improved the boundary pixel accuracy.

The current state-of-the-art edge detection methods are based on very large and complex DNNs with a huge number of parameters in most cases (at least 15 million). Furthermore, their architectural complexity is enhanced while keeping in mind the datasets that are available for training. As a result, we believe that most of these methods are prone to performance degradation even after they are trained on challenging unseen textured datasets. In the next subsection, we discuss the edge detection datasets and their limitations.

B. EDGE DETECTION DATASETS

In this subsection, we provide an overview of datasets that are mostly used for training edge detection algorithms. A brief summary of datasets and some sample images from these datasets are given in Tables 1 & 2 respectively.

1) BERKELEY SEGMENTATION DATA SET 500 (BSDS500)

BSDS500 [15], an extension of BSDS300, is the most popular dataset in edge detection literature. It consists of 500 images, which are divided into 200 training, 100 validation, and 200 testing images. Each image is annotated by multiple annotators for both image segmentation and boundary detection. The final boundary ground truth is achieved by averaging the annotations of all annotators. We identify two main issues in this dataset. Firstly, the number and scale of edges marked are not consistent across images i.e., one ground truth may also mark the edges in texture while a similar texture is ignored in another image. Secondly, each annotator marks his own path for the edges, which results in the very untidy ground truth. Due to the very small number of images, both these problems significantly affect the training and evaluation of edge detection methods.

2) THE MULTI-CUE BOUNDARY DETECTION DATASET (MDBD)

MDBD [17] consists of high definition (1280×720) videos of 100 natural scenes from both left and right views. The last frame of each left view is annotated for both boundary and edge detection separately by six annotators resulting in a total of 100 annotated images for edge detection. This dataset was released in 2016 and is mostly used with a random split of 80-20 for training and testing respectively.

3) NEW YORK UNIVERSITY DEPTH v2 (NYUDv2)

NYUDv2 [16] comprises 1449 pairs of densely labeled RGB and depth images. It is split into 381 training, 414 validation, and 654 testing images and is widely used for scene understanding and edge detection. It also contains 407,024 unlabeled frames from 464 scenes but the labeled 1449 frames only come from 26 scene types. It also comes with class and instance labels for each object. However, the range of image scenes is very limited as it only covers indoor images and does not capture a rich variety of real-life scenes.

4) BARCELONA IMAGES FOR PERCEPTUAL EDGE DETECTION (BIPED)

BIPED [18] contains 250 high definition (1280×720) outdoor images from the city of Barcelona. The images have been annotated by experts in the computer vision field at a very fine level instead of the general public used in most datasets. It is split into 200 training & validation and 50 testing images. However, the images in this dataset come from an urban setting and mostly only cover roads, buildings, cars, and other roadside objects. Thus, this dataset fails to capture a rich variety of real-life scenes.

Most of the edge detection datasets are extremely small ($\sim 10^2$) and the images in these datasets are limited to a few scenarios like indoor and non-textured images. Most papers use data augmentation on these small datasets using image transformations, such as, rotations, translations, and scaling. However, these augmentations do not add any new semantically meaningful information to the datasets. As a result, the DNNs are over-fitted to these datasets.

It is also important to highlight that there are several large scale image segmentation datasets such as Microsoft Common Objects in COntext (MS-COCO) [40] and PASCAL Visual Object Classes (PASCAL VOC) challenge datasets [41]. However, these datasets are not too useful for edge detection problems because several images in these datasets have piece-wise constant regions, which do not present a significant challenge for edge detection. Most of the images particularly in MS-COCO are from indoor settings and contain indoor objects, which do not have a textured appearance and the edges between these objects are easy to detect. For these reasons, image segmentation datasets are not routinely used as edge detection benchmarks.

III. SYNTHETIC TEXTURED MASKS DATASET (STMD) AND SINGLE-SCALE FEED FORWARD EDGE DETECTOR (SFED)

In this section, we propose STMD, which is a fully labeled large scale dataset of textured images. We also propose a simple and small benchmark DNN for edge detection called SFED with the objective of demonstrating the overfit of existing DNN methods on non-textured datasets that are commonly used in literature. To this end, we train SFED and state-of-the-art DNN methods on STMD and BSDS500. Further details of STMD and SFED are provided in the following subsections.

A. SYNTHETIC TEXTURED MASKS DATASET (STMD)

We create a new synthetic texture dataset called STMD, which is a fully labeled dataset with no annotation bias because the images are generated from the ground truth. The technique adopted to generate STMD is novel and it can be used to create very large datasets for supervised edge detection. The method relies on a database of masks [42] and textures [43] that are used to produce challenging textured images. The method is explained in Figure 1. The major steps

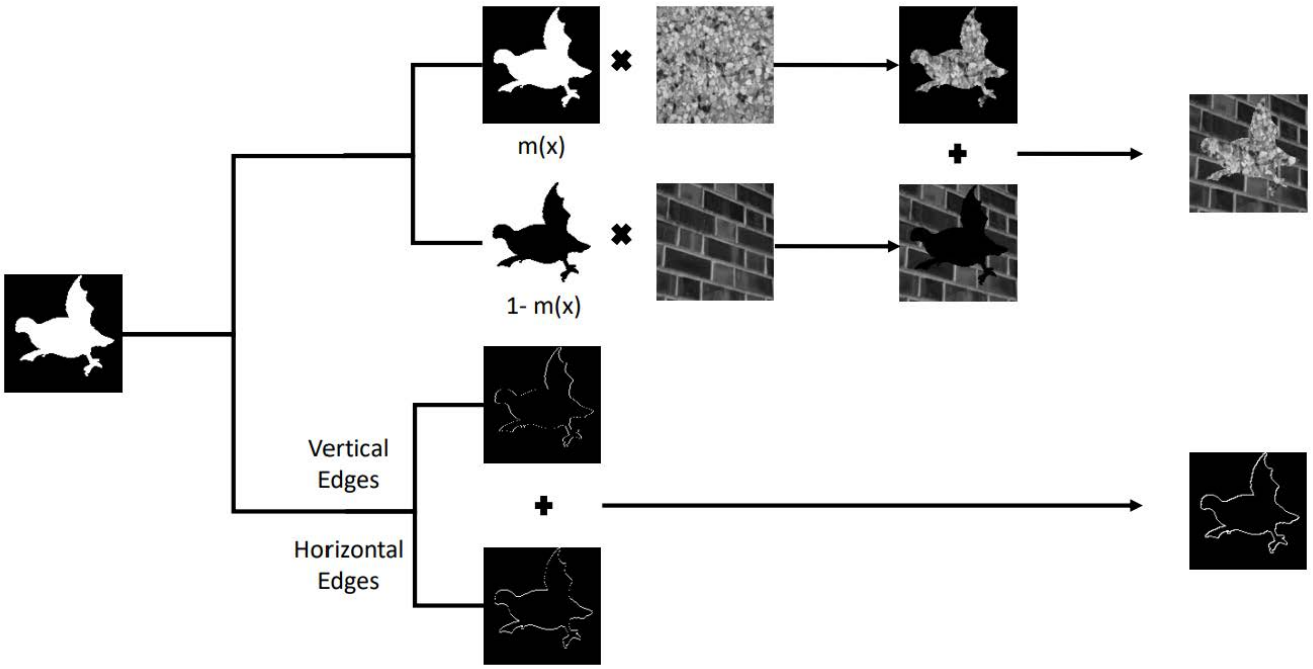


FIGURE 1. Procedure for creating a single image and its ground truth in STMD.

needed for the generation of images and their ground truths are also listed below.

1) Select a random binary mask from the masks database denoted by $m(x)$.
2) Select a random texture from the texture database and multiply it with the mask $m(x)$. This will add texture in the foreground.
3) Select another texture and multiply it with the additive inverse of the mask $(1 - m(x))$. This will add texture in the background.
4) Add the foreground and background textured masks in steps 2 and 3 to get the final image and store it in STMD.
5) Find horizontal edges in the mask by taking a gradient in the horizontal direction.
6) Find vertical edges in the mask by taking a gradient in the vertical direction.
7) Add the norm of the horizontal and vertical edges and binarize the output. This will be the ground truth for the final image obtained in step 5. Store the ground truth in STMD.

We use 1400 masks and 500 textures for the generation of 28,000 grayscale images. The total dataset is divided into 22,400 training, 2800 validation, and 2800 test images. This dataset provides a precise and consistent ground truth because it is created synthetically from masks and is not biased/dependent on humans. In many applications, finding edges between regions of unique statistics is more important than edges within the textons. This practice is ubiquitous in edge detection benchmarks, such as BSDS500, BIPED, and MDBD. Finding edges that separate different regions in a textured image is more challenging. On the other hand, finding all edges including region edges and within-textons edges can be achieved by training a network to find intensity edges. The SMTD contains images having complex textures, which allows the edge detection methods to learn the boundaries between textures while ignoring the edges of the textons within the textures.

B. SINGLE-SCALE FEED FORWARD EDGE DETECTOR (SFED)

Our proposed DNN architecture called SFED is shown in Figure 2. It is a simple 9-layer feed-forward convolutional neural network with no pooling layers (to avoid complexity) and we use it as a baseline model in our experiments. To reduce the model complexity, SFED employs single-scale approach in which the loss is calculated only on the last layer. This single-scale configuration is computationally more efficient in both training and testing. To deal with the problem of scale, we use large filters of size 7×7 in the starting layers and slowly work our way down to smaller 3×3 filters. The large filters in the early layers help the network to better capture the properties of large scale textures and somewhat alleviate the need of pooling layers and multi-scale configuration. The number of filters at each layer varies and decreases as the depth increases starting from 256 and ending at 1. The details of SFED layers are summarized in Table 3.

SFED has about 1.5 million parameters. This network is chosen because we want to test the hypothesis that current state-of-the-art methods for edge detection are overly complex and biased on training datasets, and a simple network like SFED can perform just as well or better on general datasets compared to the very complex networks with at least 10 times more parameters.

TABLE 3. SFED convolution layers details.

Layer	Filter Size	Number of Filters	Padding	Stride	WidthxHeightxDepth
Input Layer	–	–	–	–	256x256x3
Layer 1	7x7	256	3	1	256x256x256
Layer 2	5x5	128	2	1	256x256x128
Layer 3	5x5	128	2	1	256x256x128
Layer 4	3x3	128	1	1	256x256x128
Layer 5	3x3	64	1	1	256x256x64
Layer 6	3x3	64	1	1	256x256x64
Layer 7	3x3	32	1	1	256x256x32
Layer 8	3x3	32	1	1	256x256x32
Output Layer	1x1	1	0	1	256x256x1

Given an input image X_i, the proposed method outputs an image $\hat{Y}_i$ in which the value at each pixel represents the probability of that pixel being an edge. We use a single weighted binary cross-entropy loss function $L(\cdot)$ at the last layer for network learning:

$$L(\cdot) = -\frac{1}{|D|}\sum_{i}^{|D|}\sum_{j}^{|Y_i|}\left(\beta y_{ij}\log(\hat{y}_{ij}) + (1-y_{ij})\log(1-\hat{y}_{ij})\right)$$

where D is the dataset, Y_i is the ground truth of i^{th} image, y_{ij} and $\hat{y}_{ij}$ are the respective ground truth and prediction at j^{th} pixel of i^{th} image, and β is the weight for positive ground truth. We use $\beta = 5$ in our method.

To summarize, the main highlights of SFED are:

1) Single scale approach to make it computationally more efficient in training and testing.
2) Use of large filters in the beginning to capture large scale textures and to compensate for single scale.
3) Use of a small network with only 1.5 million parameters (at least 10 times less than state-of-the-art DNNs).

The ground truth for the edge detection datasets is binary. This is ubiquitous for all edge detection benchmarks (BSDS500, BIPED, MDBD). The evaluation metrics for edge detection benchmarks look at the overlap of two binary edge maps, i.e., the ground-truth and the output. All methods including ours must first convert the output map [0,1] into a binary edge map {0, 1} before evaluation of the output can be performed. This is done by selecting a threshold. All edge detection methods perform this thresholding step.

For performance quantification, we use two widely used edge detection evaluation metrics called Optimal Dataset Scale (ODS) and Optimal Image Scale (OIS) [15]. ODS represents the average score when a single edge best threshold is chosen for the entire dataset. Similarly, OIS represents the average score when the best threshold is chosen for each individual image in the dataset. The optimum threshold of ODS and OIS are selected by exhaustively going over all the values of the threshold {0, 0.01, 0.02, ...1} and selecting the best value (the value for which we get the best results). For ODS we have a single best threshold for the entire dataset. For OIS we have an individual best threshold for each individual image in the dataset. The same process of threshold selection is used in all state-of-the-art edge detection architectures.

We would also like to highlight that segmentation architectures such as [44], [45] can also be used for edge detection. However, state-of-the-art in edge detection is generally better in the task compared to these networks [29]. Moreover, in this paper, we have also proposed SFED to show that even with a very simple architecture we can outperform complex architectures, which are biased due to their training on small scale datasets. Thus, our aim is to use the simplest configuration of CNN to highlight the existence of biases in the current state-of-the-art networks being used for edge detection. It is important to note that establishing the generalization ability of SFED on unseen datasets is important. In the experimental results, we will test this aspect of SFED by training it on the BSDS500 dataset and testing it on NYUD-v2, MDBD, and BIPED datasets.

IV. EXPERIMENTS

In this section, we describe our experimental setup and discuss the quantitative and qualitative results. The objective is to show the overfit of state-of-the-art DNNs to small datasets. To show this we train and evaluate four DNNs (BDCN [33], HED [29], DexiNed [18], and RCF [34]) on STMD and BSDS500 and compare them against SFED baseline architecture.

We trained our models from scratch on each dataset until the validation loss converged. SFED was trained on 22400 STMD images for 100 epochs using Adam optimizer with a learning rate of 0.0001. For training on BSDS500, we used 200 images for 30 epochs using an Adam optimizer with a learning rate of 0.001 which was decayed to 10% every 10 epochs. All other state-of-the-art models were trained using the same images as our model with their original hyper-parameters and backbones until the validation loss converged. The corresponding training and validation loss graphs of all the models are given in the appendix.

A. QUALITATIVE RESULTS

Figure 3 presents the qualitative (visual) results of SFED, BDCN, DexiNED, RCF, and HED on BSDS500. Please note that all the architectures are trained on the BSDS500 training dataset (200 images). We can notice that even though BDCN is the current state-of-the-art on augmented BSDS500 dataset [29], its performance is subpar producing thicker and blurry edges. RCF also has more blurry edges on the BSDS500 dataset. HED produces thicker edges but overall does a better job than the rest because of its ability to navigate through some finer edges not annotated in ground truth due to

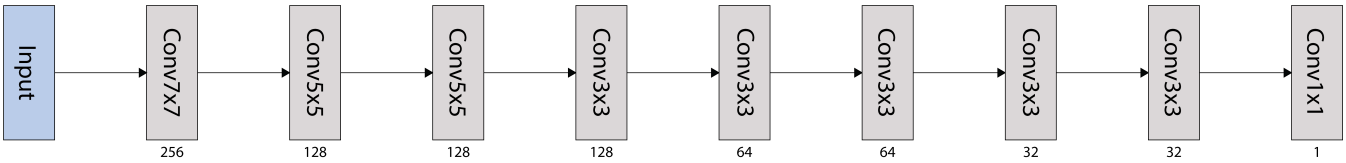


FIGURE 2. SFED Architecture – it consists of 1 input layer, 8 convolution layers and 1 output layer. The numbers inside the box donate the filter size whereas the numbers below the box donate number of filters.

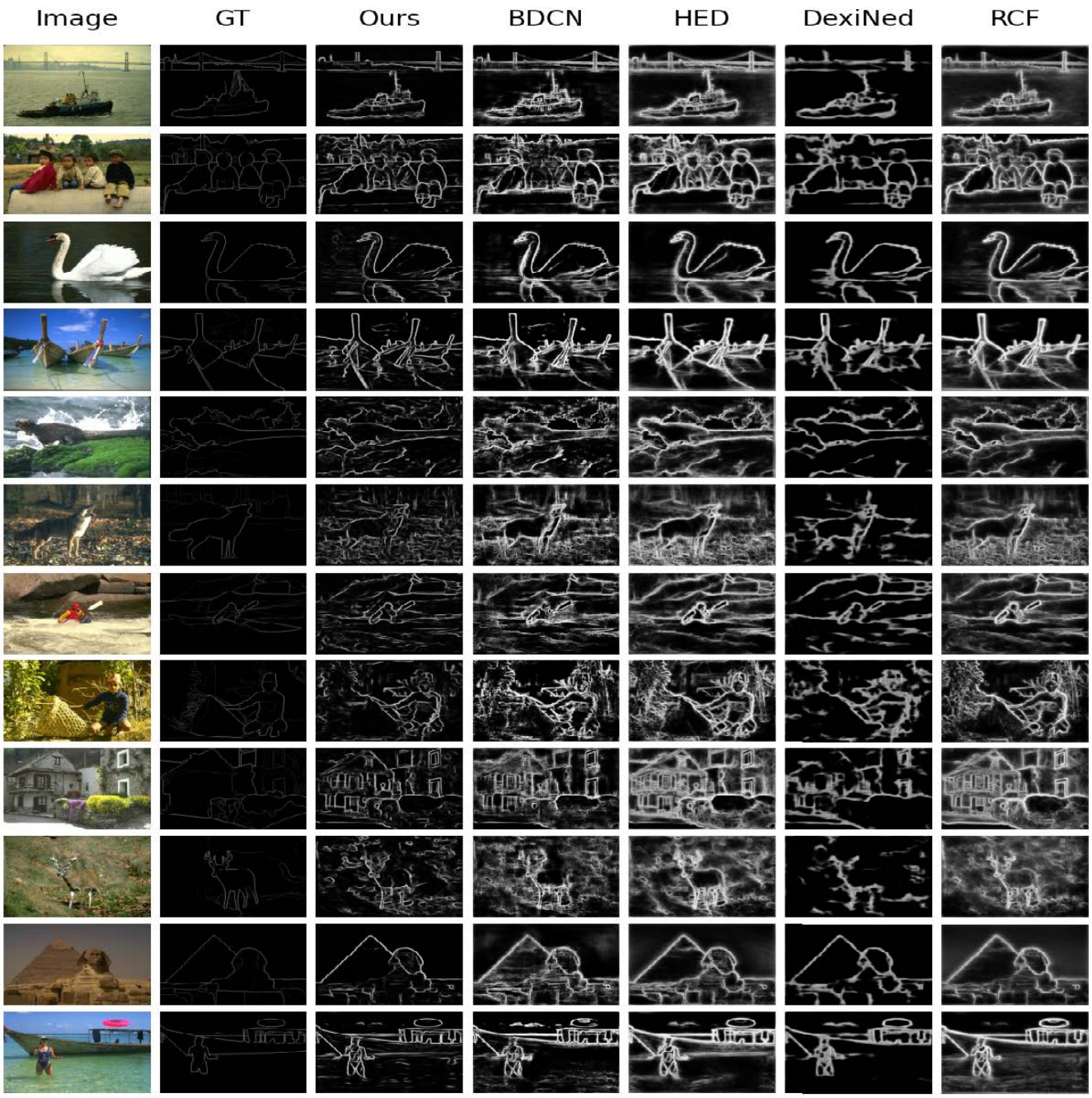


FIGURE 3. Visual results of our architecture (SFED) and existing methods on BSDS500 Test data when trained on BSDS500 training data.

its multi-scale approach BDCN and RCF also use multi-scale approach but they fail to get a higher score because their architectures are very complex and require a large amount of data for effective training. DexiNed performs the worst with smudgy edges. This can be attributed to the fact that DexiNed has a huge number of parameters (about 20 million) and unlike BDCN, HED, and RCF, it trains from scratch without employing transfer learning. Therefore, it requires a

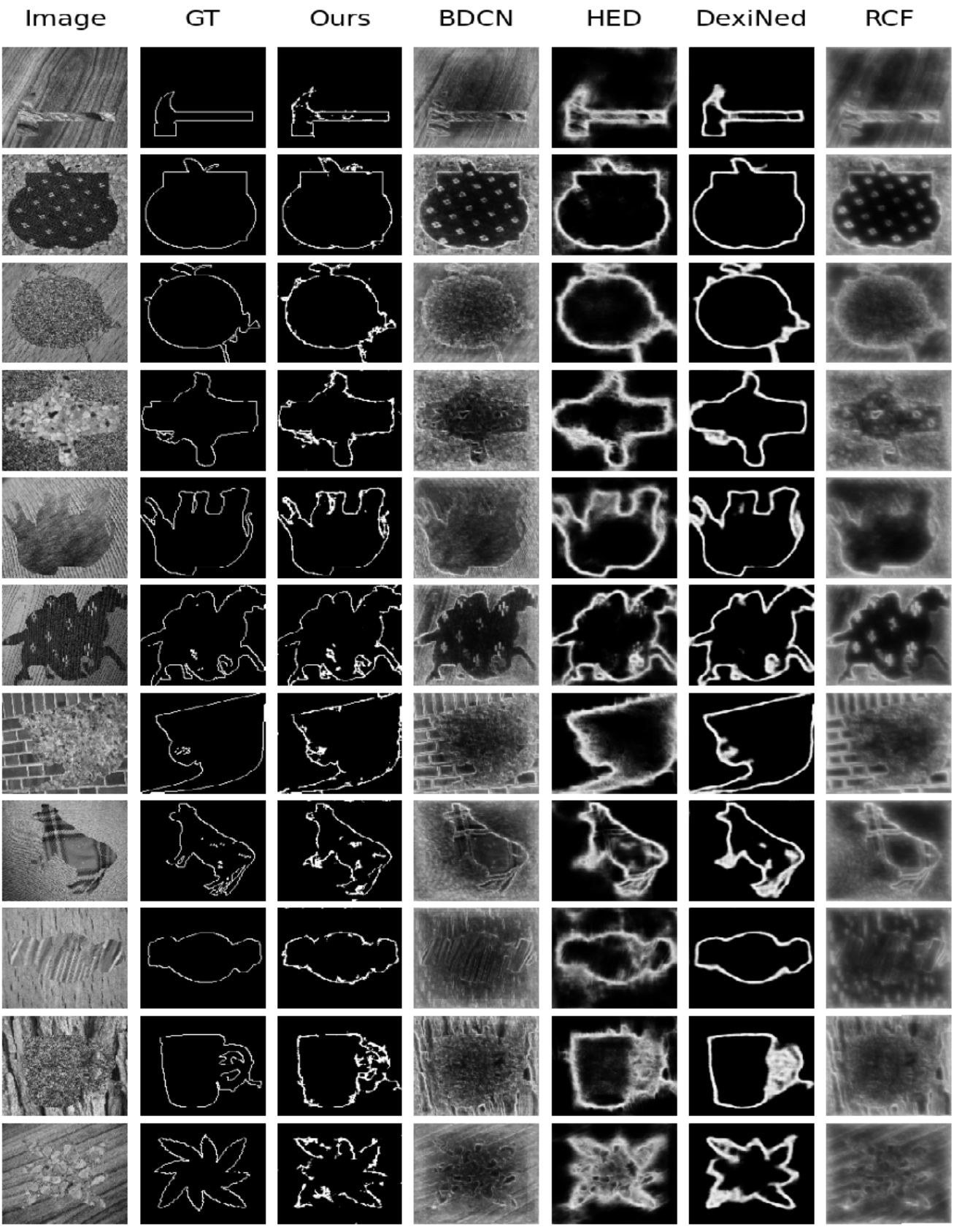


FIGURE 4. Visual results of our architecture (SFED) and existing methods on STMD Test data when trained on STMD training data.

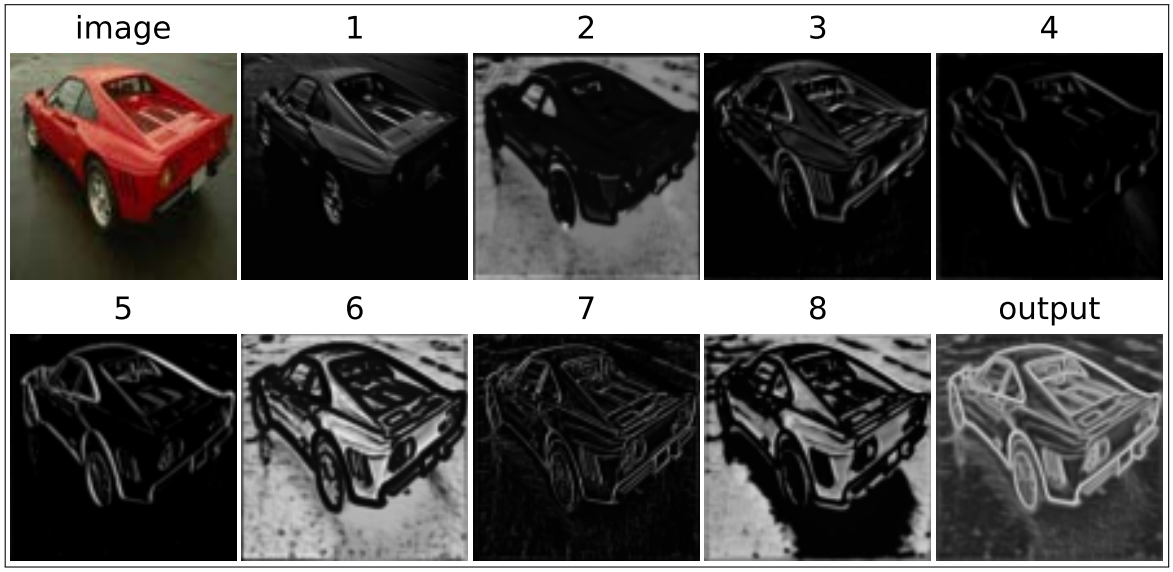


FIGURE 5. Output of a single channel for each layer of SFED. It consists of 1 input layer, 8 intermediate layers and 1 output layer.

huge dataset for effective training. Our baseline SFED architecture having a very small number of training parameters outperforms almost all state-of-the-art DNN methods.

In Figure 4 we present the qualitative results of SFED, BDCN, DexiNED, RCF, and HED on STMD. All the architectures are trained on STMD training data (22,400 images). SFED gives the crispest edges and successfully ignores the texton edges. However, BDCN, HED, and RCF are unsuccessful in ignoring texture and the visual quality of the edges is blurry. DexiNed, on the other hand, successfully navigates through texture but produces thick edges. These results expose the limitations of existing architectures and datasets where state-of-the-art methods on small scale datasets fail on complex large scale datasets.

B. QUANTITATIVE RESULTS

In this subsection, we quantify the performance of SFED and state-of-the-art architectures using ODS and OIS scores. We also compute the percentage difference in the ODS and OIS values of state-of-the-art architectures relative to SFED using the following formula.

$$\text{Difference } (\%) = \frac{\text{Other Architecture Value - SFED Value}}{\min(\text{SFED Value, Other Architecture Value})} \quad (1)$$

A negative difference would indicate the superiority of SFED architecture over the compared architecture and vice versa.

Table 4 shows the quantitative results i.e., ODS and OIS scores when we trained the architectures on BSDS500 till the validation loss converged. The comparative performance of the algorithms is also presented. We can see that our method (SFED) is the second-best after HED. The complexity of all the architectures in terms of number of parameters is also given in this table. We can see that our SFED architecture with 1.55 million parameters is almost ten times less complex than HED which has 14.7 million parameters. DexiNed has the highest complexity with around 20.8 million parameters which is almost 13.5 times higher than our SFED architecture.

TABLE 4. Quantitative results of methods trained on BSDS500 dataset. Red represents the best results, Green represents the second best results.

Models	ODS	OIS	Parameters
BDCN [33]	0.704 (-1.7%)	0.72 (-1.38%)	16,302,712
DexiNed [18]	0.617 (-16.04%)	0.63 (-15.87%)	20,798,091
HED [29]	0.75 (+4.74%)	0.77 (+5.48%)	14,716,171
RCF [34]	0.70 (-2.28%)	0.72 (-1.38%)	14,803,781
SFED (Ours)	0.716	0.73	1,554,657

We then trained state-of-the-art architectures and SFED on more challenging STMD training data that contains 22,400 textured images. The corresponding ODS and OIS scores obtained for each architecture and comparisons with SFED are presented in Table 5. On both measures, SFED has the best performance compared to all other state-of-the-art architectures. The degradation in the performance of state-of-the-art architectures on textured dataset reveals that these architectures do not generalize well to challenging images. These architectures are overfitted to small scale datasets. Thus, both quantitatively and qualitatively, SFED architecture, which is much simpler than the rest of the architectures has performed well on the challenging textured dataset. These results expose the limitations of existing architectures and datasets and validate the main hypothesis of this work.

In Table 6, we present the results on the generalization ability of SFED. In these experiments, SFED is trained on the BSDS500 dataset but then it is tested on NYUDv2, MDBD, and BIPED datasets. We can notice that SFED performs quite well on all the datasets. The best performance is achieved on MDBD (Boundaries) dataset where the ODS and OIS scores are almost similar to the ones obtained for the training dataset. On the other hand, the performance slightly degrades on NYUDv2 and BIPED datasets. However, these results

TABLE 5. Quantitative results of methods trained on synthetic textured dataset (STMD). Red represents the best results, Green represents the second best results.

Models	ODS	OIS
BDCN [33]	0.34 (-111.8%)	0.30 (-140%)
BDCN Finetuned [33]	0.30 (-140%)	0.29 (-148.2%)
DexiNed-avg [18]	0.64 (-12.5%)	0.62 (-16.12%)
DexiNed-fused [18]	0.67 (-7.46%)	0.68 (-5.88%)
HED [29]	0.52 (-38.46%)	0.52 (-38.46%)
RCF [34]	0.28 (-157.1%)	0.24 (-200%)
SFED (Ours)	0.72	0.72

TABLE 6. Performance of SFED trained on BSDS500 and tested on NYUDv2, MDBD, and BIPED datasets.

Model	Dataset	ODS	OIS
SFED	NYUDv2 [16]	0.58 (-23.44%)	0.59 (-23.72%)
	MDBD (Boundaries) [17]	0.72 (0.55%)	0.73 (0%)
	MDBD (Edges) [17]	0.66 (-8.48%)	0.67 (-7.46%)
	BIPED [18]	0.6 (-19.33%)	0.61 (-19.67%)
	BSDS500 [15]	0.716	0.73

clearly demonstrate the generalizability of SFED architecture on unseen datasets.

The output of the intermediate layers of SFED is shown in Figure 5. Our architecture consists of 9 convolution layers (including the final output layer). A single channel of the output of each intermediate layer is shown. Notice that different types of features and edges are captured by each layer. Layers 1-2 broadly capture the foreground and background in the image. Layers 3-4 capture various types of edges. As the filter size gets smaller, layers 5-8 capture more complex features (finer edges). Finally, the output layer has the full edge map of the image. These visualizations reveal the internal working of SFED.

V. CONCLUSION & FUTURE WORK

DNN techniques have significantly contributed toward solving the edge detection problem. State-of-the-art algorithms even claim to beat human-level performance. In this paper, we explored the limitations of these claims by showing that the architectural novelty of DNN based edge detection algorithms does not improve their generalization capabilities. To this end, we introduced the STMD pipeline that allowed us to generate a set of 28,000 fully labeled grayscale textured images. The images in this dataset are more challenging than the BSDS500 dataset which is commonly used for training edge detection algorithms. We also proposed SFED as a baseline DNN architecture with 9 nine feed-forward convolutional layers. Through various experiments, we showed that the performance of several state-of-the-art algorithms degraded on the STMD dataset. In comparison, SFED when trained with a standard loss function performed better on both STMD and BSDS500 datasets. These observations demonstrate that the edge detection problem still requires more research and it is not yet solved.

The future extensions of this work include the development of a large-scale labeled dataset of complex real-life images with ample textures and difficulty for benchmarking and training algorithms for emerging application areas, such as medical imaging and self-driving cars. Our work also opens up the possibility of designing novel edge detection architectures that avoid the overfitting problem and have better generalization capabilities. The shortcomings of existing architectures can also be quantified along with suitable remedial steps to improve their performance. New metrics to test and evaluate the performance and generalization capabilities of edge detection algorithms can also be developed.

APPENDIX
TRAINING AND VALIDATION LOSS GRAPHS

The training and validation loss of various edge detection algorithms on STMD and BSDS500 datasets are plotted in the following figures.

A. STMD DATASET

Training loss of BDCN, DexiNed, HED, RCF, and SFED algorithms on the STMD dataset is given in Figure 6. Similarly, the validation loss of these algorithms on the STMD dataset is given in Figure 7.

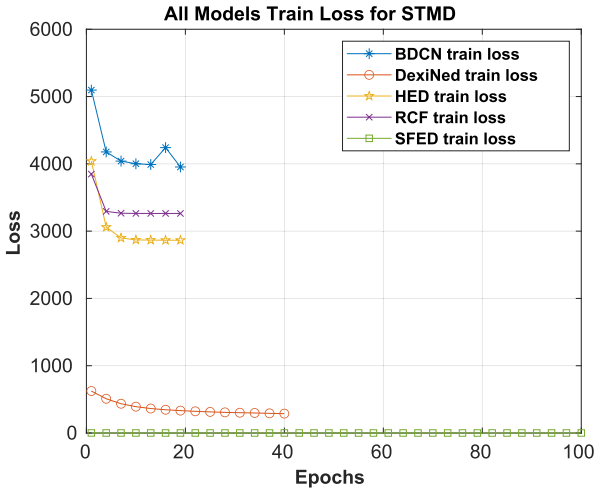


FIGURE 6. Training Loss of BDCN, DexiNed, HED, RCF and SFED algorithms on STMD dataset.

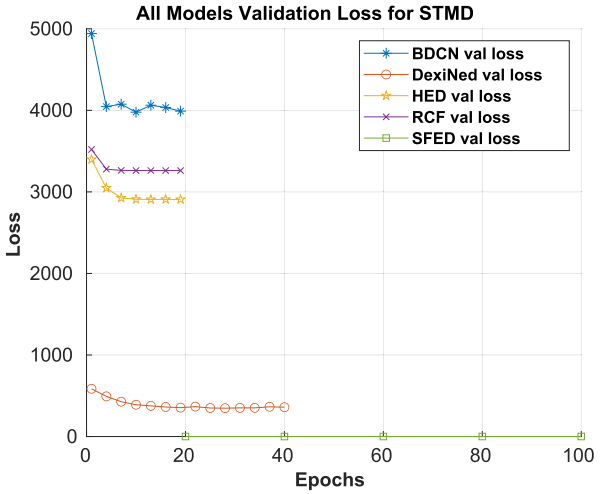


FIGURE 7. Validation Loss of BDCN, DexiNed, HED, RCF and SFED algorithms on STMD dataset.

B. BSDS500 DATASET

Training loss of BDCN, DexiNed, HED, RCF, and SFED algorithms on the BSDS500 dataset is given in Figure 8. Similarly, the validation loss of these algorithms on BSDS500 dataset is give in Figure 9.

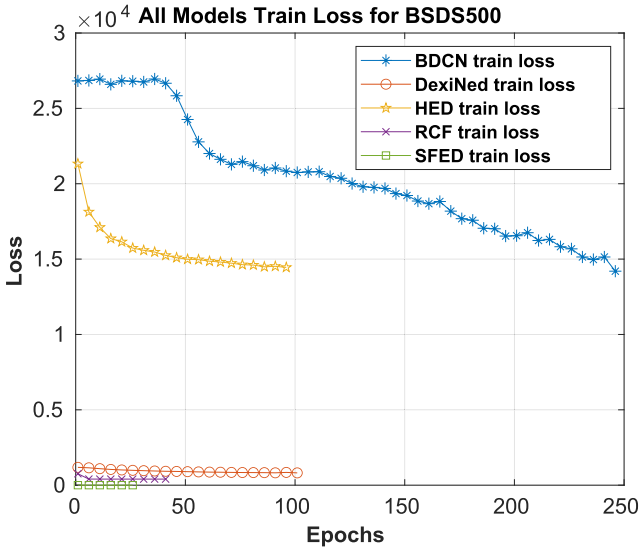


FIGURE 8. Training Loss of BDCN, DexiNed, HED, RCF and SFED algorithms on BSDS500 dataset.

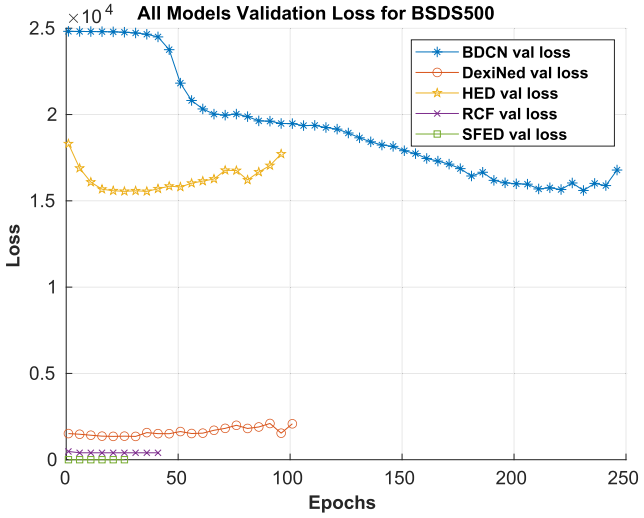


FIGURE 9. Validation Loss of BDCN, DexiNed, HED, RCF and SFED algorithms on BSDS500 dataset.

REFERENCES

[1] K. Zhang, L. Zhang, K.-M. Lam, and D. Zhang, "A level set approach to image segmentation with intensity inhomogeneity," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 546–557, Feb. 2016.

[2] M.-H. Yang, D. Kriegman, and N. Ahuja, "Detecting faces in images: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 1, pp. 34–58, Jan. 2002.

[3] J. Shotton, A. Blake, and R. Cipolla, "Multiscale categorical object recognition using contour fragments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 7, pp. 1270–1281, Jul. 2008.

[4] M. Li, Z. Lin, R. Mech, E. Yumer, and D. Ramanan, "Photo-sketching: Inferring contour drawings from images," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2019, pp. 1403–1412.

[5] R. Muthalagu, A. Bolimera, and V. Kalaichelvi, "Lane detection technique based on perspective transformation and histogram analysis for self-driving cars," *Comput. Electr. Eng.*, vol. 85, Jul. 2020, Art. no. 106653.

[6] R. A. Z. Daou, F. El Samarani, C. Yaacoub, and X. Moreau, "Fractional derivatives for edge detection: Application to road obstacles," in *Smart Cities Performability, Cognition, & Security*. Cham, Switzerland: Springer, 2020, pp. 115–137.

[7] C. Orhei, S. Vert, and R. Vasiu, "A novel edge detection operator for identifying buildings in augmented reality applications," in *Proc. Int. Conf. Inf. Softw. Technol.*, 2020, pp. 208–219.

[8] G. Kühne, S. Richter, and M. Beier, "Motion-based segmentation and contour-based classification of video objects," in *Proc. 9th ACM Int. Conf. Multimedia*, 2001, pp. 41–50.

[9] Y.-C. Huang, Y.-S. Tung, J.-C. Chen, S.-W. Wang, and J.-L. Wu, "An adaptive edge detection based colorization algorithm and its applications," in *Proc. 13th Annu. ACM Int. Conf. Multimedia*, 2005, pp. 351–354.

[10] T.-H. Sun, C.-H. Lai, S.-K. Wong, and Y.-S. Wang, "Adversarial colorization of icons based on structure and color conditions," 2019, *arXiv:1910.05253*.

[11] E. J. Wharton, K. Panetta, and S. S. Agaian, "Logarithmic edge detection with applications," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, Oct. 2007, pp. 3346–3351.

[12] F. Orujov, R. Maskeliūnas, R. Damaševičius, and W. Wei, "Fuzzy based image edge detection algorithm for blood vessel detection in retinal images," *Appl. Soft Comput.*, vol. 94, Sep. 2020, Art. no. 106452.

[13] A. T. Jamal, A. B. Ishak, and S. Abdel-Khalek, "Tumor edge detection in mammography images using quantum and machine learning approaches," *Neural Comput. Appl.*, vol. 33, no. 13, pp. 7773–7784, Jul. 2021.

[14] R. Pourreza, Y. Zhuge, H. Ning, and R. Miller, "Brain tumor segmentation in MRI scans using deeply-supervised neural networks," in *Proc. Int. MICCAI Brainlesion Workshop*, 2018, pp. 320–331.

[15] 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 *Proc. 8th Int. Conf. Comput. Vis.*, vol. 2, Jul. 2001, pp. 416–423.

[16] S. Gupta, P. Arbeláez, and J. Malik, "Perceptual organization and recognition of indoor scenes from RGB-D images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 564–571.

[17] D. A. Mély, J. Kim, M. McGill, Y. Guo, and T. Serre, "A systematic comparison between visual cues for boundary detection," *Vis. Res.*, vol. 120, pp. 93–107, Mar. 2016.

[18] X. Soria, E. Riba, and A. D. Sappa, "Dense extreme inception network: Towards a robust CNN model for edge detection," 2019, *arXiv:1909.01955*.

[19] J. Kittler, "On the accuracy of the Sobel edge detector," *Image Vis. Comput.*, vol. 1, no. 1, pp. 37–42, 1983. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0262885683900069

[20] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986, doi: 10.1109/TPAMI.1986.4767851.

[21] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-33, no. 5, pp. 898–916, Aug. 2011.

[22] J. J. Lim, C. L. Zitnick, and P. Dollár, "Sketch tokens: A learned mid-level representation for contour and object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3158–3165.

[23] N. Khan, "Shape-tailored features and their application to texture segmentation," M.S. thesis, King Abdullah Univ. Sci. Technol., Thuwal, Saudi Arabia, Apr. 2014, doi: 10.25781/KAUST-TP035.

[24] N. Khan, M. Algarni, A. Yezzi, and G. Sundaramoorthi, "Shape-tailored local descriptors and their application to segmentation and tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3890–3899.

[25] N. Khan, B.-W. Hong, A. Yezzi, and G. Sundaramoorthi, "Coarse-to-fine segmentation with shape-tailored continuum scale spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4240–4249.

[26] N. Khan, "Shape-tailored invariant descriptors for segmentation," Ph.D. dissertation, King Abdullah Univ. Sci. Technol., Thuwal, Saudi Arabia, 2018, doi: 10.25781/KAUST-EXX0N.

[27] P. Kovesi, "Edges are not just steps," in *Proc. 5th Asian Conf. Comput. Vis.*, Melbourne, VIC, Australia, vol. 8, 2002, pp. 8–22.

[28] P. Isola, D. Zoran, D. Krishnan, and E. H. Adelson, "Crisp boundary detection using pointwise mutual information," in *Computer Vision—ECCV*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham, Switzerland: Springer, 2014, pp. 799–814.

[29] S. Xie and Z. Tu, "Holistically-nested edge detection," 2015, *arXiv:1504.06375*.
[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[31] Y. Liu and M. S. Lew, "Learning relaxed deep supervision for better edge detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 231–240.
[32] R. Deng, C. Shen, S. Liu, H. Wang, and X. Liu, "Learning to predict crisp boundaries," 2018, *arXiv:1807.10097*.
[33] J. He, S. Zhang, M. Yang, Y. Shan, and T. Huang, "Bi-directional cascade network for perceptual edge detection," 2019, *arXiv:1902.10903*.
[34] Y. Liu, M.-M. Cheng, X. Hu, J.-W. Bian, L. Zhang, X. Bai, and J. Tang, "Richer convolutional features for edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1939–1946, Aug. 2019.
[35] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," 2016, *arXiv:1610.02357*.
[36] Z. Yu, W. Liu, Y. Zou, C. Feng, S. Ramalingam, B. V. K. V. Kumar, and J. Kautz, "Simultaneous edge alignment and learning," 2018, *arXiv:1808.01992*.
[37] D. Acuna, A. Kar, and S. Fidler, "Devil is in the edges: Learning semantic boundaries from noisy annotations," 2019, *arXiv:1904.07934*.
[38] Y. Hu, Y. Chen, X. Li, and J. Feng, "Dynamic feature fusion for semantic edge detection," 2019, *arXiv:1902.09104*.
[39] M. Zhen, J. Wang, L. Zhou, S. Li, T. Shen, J. Shang, T. Fang, and Q. Long, "Joint semantic segmentation and boundary detection using iterative pyramid contexts," 2020, *arXiv:2004.07684*.
[40] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 740–755.
[41] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010.
[42] B. S. Manjunath, P. Salembier, and T. Sikora, *Introduction to MPEG-7: Multimedia Content Description Interface*. Hoboken, NJ, USA: Wiley, 2002.
[43] S. Lazebnik, C. Schmid, and J. Ponce, "A sparse texture representation using local affine regions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1265–1278, Aug. 2005.
[44] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, 2015, pp. 234–241.
[45] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

MUHAMMAD MUBASHAR received the B.S. degree in computer science from the Lahore University of Management Sciences (LUMS), Lahore, Pakistan, in 2021. From 2020 to 2021, he was involved in research with the LUMS and the Torr Vision Group, Oxford. He is currently working as a Machine Learning Engineer. His main research interests include computer vision and reinforcement learning.

NAEEMULLAH KHAN received the B.S. degree in electronic engineering from the GIK Institute, Pakistan, in 2011, and the M.S. and Ph.D. degrees in electrical engineering from the King Abdullah University of Science and Technology, Thuwal, Saudi Arabia, in 2014 and 2018, respectively. In 2018, he joined the Torr Vision Group, University of Oxford, as a Postdoctoral Research Scientist. He joined Oxford Brookes University, as a Research Fellow, in 2021. He has been a Junior Research Fellow at the Lady Margaret Hall, University of Oxford, since 2020. His research interests include robustness and uncertainty quantification in deep neural networks.

ABDUR REHMAN SAJID received the B.S. degree in computer science from the Lahore University of Management Sciences, Lahore, in 2021. From 2020 to 2021, he did research in deep learning to find how state-of-the-art models work on edge detection and how it can be improved. He is currently working as a full-time Software Developer in a startup company. His research interest includes machine learning combined with software engineering to solve real-life problems.

MUHAMMAD HASHIM JAVED received the B.S. degree in computer science from the Lahore University of Management Sciences (LUMS), Lahore, Pakistan. He worked as a Teaching Assistant with the Department of Computer Science, LUMS, for programming and machine learning courses. From 2020 to 2021, he did research in DNN-based edge detection methods at LUMS. His research interests include machine learning and deep learning.

NAVEED UL HASSAN (Senior Member, IEEE) received the B.E. degree from the College of Aeronautical Engineering, Risalpur, Pakistan, in 2002, and the M.S. and Ph.D. degrees from the Ecole Superieure d'Electricite, Gif-sur-Yvette, France, in 2006 and 2010, respectively. He is currently an Associate Professor with the Department of Electrical Engineering, Lahore University of Management Sciences (LUMS), Pakistan. His research interests include machine learning applications, wireless communications, smart energy systems, blockchain technology, and indoor positioning systems.