Implementation of Edge Detection Process by using Supervised Convolution Neural Network

M.Jansirani, P.Sumitra

Department of Computer Science and Applications, Vivekanandha College of Arts and Science for Women, Tiruchengode, Tamilnadu, India

Corresponding author: M. Jansirani, Email: jansinagu@gmail.com

Edge detection is important in digital image processing because it contains essential information that is needed to process the specific application requirement. The feature detection and characteristic extraction processes are both directly influenced by the edge detection procedure. The image quality is determined by the retrieved edges and features. To derive the edge characteristics, several current techniques such as Canny, Laplacian of Gaussian, Prewitt, Zero-Crossing, Roberts, and Sobel are commonly used. However, when the amount of the dataset grows, this strategy requires a lot of computing time. The high running suggests that edge detection approaches have an impact on the overall efficiency of picture analysis. To address these challenges, this paper introduces an edge detection technique combined with machine learning algorithms. To reduce computing complexity, a supervised convolution neural network is used in conjunction with the edge detection procedure. The neural network employs automatic learning functions and pre-training patterns to anticipate edge-related features with minimal effort. The fully convoluted layer and max-pooling function are used in the convolution network to reduce unnecessary features and forecast edges with the highest detection accuracy. Edge prediction deviations are minimized during this procedure by back propagating the incorrect value to the preceding layer, which improves the overall recognition rate. The system's efficiency is then calculated using the MATLAB tool, along with the appropriate performance measures.

Keywords: Image processing, Edge detection, supervised convolution neural networks, Computation complexity, Fully connected layer and max-pooling function.

2023. In Saroj Hiranwal & Garima Mathur (eds.), *Artificial Intelligence and Communication Technologies*, 275–281. Computing & Intelligent Systems, SCRS, India. https://doi.org/10.52458/978-81-955020-5-9-28

1 Introduction

The process of predicting picture points, or sharp changes in the brightness of images, is known as edge detection [1, 2]. Image boundaries or edges relate to the designated sharp points. In computer vision, picture edges play an important role in recognizing image patterns. The image edges are constantly changing, resulting in discontinuities. Depth, surface orientation, material characteristics, and scene illumination are all examples of discontinuities [3]. The image is subjected to an edge computing process that identifies image structural elements and minimizes pattern analysis computation complexity. Because of the fragmentation, edge computing frequently causes misunderstanding [4]. The edges are formed from non-trivial images in which edge points may or may not be connected, resulting in uncertainty in image interpretation. To solve this problem, image edge attributes are investigated using basic techniques such as image processing, analysis, and pattern recognition. To improve the overall recognition rate, computing approaches should take edge attributes into account. The image has perspective independent and dependent qualities that are utilised to determine the image's boundary alterations [5,6]. Several edge detection techniques are used to consider these properties, including Robert's operator [7], Sobel's operator [8], Prewitt's operator [9], Second derivative operator [10], canny edge detection [11], and Laplacian of Gaussian [12] operators.

These approaches effectively assess images and pertinent boundaries, but they run into problems when the size of the output image is lowered after applying the filter to the input image. The size of the considered image is 6*6, however if the edge detection is done with a 3*3 filter, the size of the output image is modified to 4*4. In general, the output image for a n*n image with a r*r filter size edge detection method is computed as (n-r+1)*(n-r+1). When using the edge detection approach, most of the essential information may be lost due to the smaller output image size. As a result, various studies [13] have used the padding approach to address image shrinkage concerns. The input image must be padded before the edge analysis technique is applied to reduce information loss. However, when the image and dataset sizes grow, the existing system consumes more calculation time. The maximum running time has an impact on the overall efficiency of picture analysis and pattern recognition. To address this problem, a supervised learning approach based edge detection technique is used in this study. Convolution neural networks are used in this method to anticipate image edge details. The network employs a collection of convolution layers to predict a set of edge patterns with the least amount of effort. The back propagation learning algorithm is used to verify the anticipated edge efficiency, which is then utilized to reduce the error rate. The suggested system is implemented with the help of the MATLB tool and the appropriate parameters.

The main contribution of the paper is listed as follows.

- Using convolution networks to increase the accuracy of edge detection.
- Minimizing the difference between the real and computed edge information by applying back propagation learning to propagate the error value. Here, convoluted pre-training model is created to improve the overall edge detection efficiency.

The remainder of the paper is structured as follows. Section 2 delves into the viewpoints of many researchers on the edge detection technique. Section 3 examines the convolution-based edge detection technique and the system's efficiency, which is evaluated in Section 4. Section 5 describes the conclusion.

2 Related Works

Gholizadeh-Ansari, et al., 2020 [14] used a deep learning strategy to detect CT image edges. The goal of this research is to use dilated convolution networks to reduce edge computing complexity while minimizing information loss. The network identifies and extracts edge features in several directions, including vertical, horizontal, and diagonal, using non-trainable edge layers. The effective use of these layers lowers the rate of error and information loss.

Orujov, et al., 2020 [15] used a fuzzy based edge prediction system to analyze and detect blood vessels in retinal pictures. To improve image quality, fund us eye images are first gathered and processed using a histogram equalization approach. The edge relevant information is then detected using Mamdani type 2 fuzzy rules. The resulting edge information enhances total vessel detection accuracy by up to 0.95 percent. The STARE and DRIVE databases were used to assess the system's effectiveness. The new strategy takes care of the system's dependability and flexibility.

Kartik, et al., 2021 [16] used deep learning techniques to detect edges from numerical digits. The goal of this project is to generate the decode stream in relation to the graph. First, the RGB images are converted to text, and then the image is converted to canny. To determine the number of occurrences, the transformed images are evaluated using a multi-label classification and segmentation process. This technique was performed indefinitely to obtain the digit decode stream. In comparison to previous approaches, the presented system has an 89 percent prediction rate.
He, et al., 2019 [17] used bi-directional cascade neural networks to detect edges. To forecast the exact edge features, the incorporated neural network evaluates the image edges in a multi-scale viewpoint. Furthermore, the scale improvement module is used to study deeper edge information, which is more valuable for accurately identifying edges in complex images. The effectiveness of the introduced system is assessed using the Multi cue, NYUDV2, and BSDS500 databases, with the system achieving an accuracy of 0.828 percent. Deep learning approaches are effectively used in this image processing field, according to the above researcher's perspective, to improve the edge identification process. Information loss and production shrinkage, on the other hand, should be avoided at all costs. To address this problem, a convolution neural network is used to improve overall edge detection accuracy in this study.

3 Convolution neural network based image edge detection

This section discusses the edge identification technique using convolutional neural networks from various photos. The project's major goal is to reduce computation complexity while also boosting edge detection rates. Image processing and machine learning approaches are used in this study to attain this goal. The following is a detailed explanation of how image edge detection works.

Preprocessing and image conversion

Because the collected images are in RGB colour and have noise elements, the initial step of this task is image conversion and preprocessing. The compute complexity and time factor increase as a result of colour picture processing. To address this problem, photos are turned into binary images in this study to make the edge detection procedure easier. The RGB component's minimum and maximum levels are used in the conversion. After that, the brightness, hue, and saturation values are calculated, and the results are compared to the threshold value. If the pixel values are less than the threshold value, they must be kept; otherwise, they must be removed. To convert the photos, this process was repeated indefinitely. After that, the noise in the photos should be eliminated to improve the image quality. The noise pixel is replaced by a mean filter in this case. Each pixel is compared to the threshold value, and if it differs, it must be replaced with the mean value. The average value of the nearby pixels is used to calculate the mean value. To eliminate the noise from the photos, this process is repeated. When noise is removed from an image, the overall efficiency of edge computing improves.

Edge Segmentation

The next crucial stage is edge segmentation, which entails extracting the useful image boundary information that will be employed in further image analysis. In this study, a convolution neural network is used to detect edges in a picture. The method for resolving the output shrink and information loss problem with the least amount of calculation time. The convolute network is made up of numerous trained patterns that are used to calculate depth with minimal effort. Furthermore, the convolution network analyses pictures using sub sampling layers, which reduces computation time and improves edge detection accuracy. The network, on the other hand, uses a small sub-sampling factor to estimate the edges in a straightforward manner. The network uses temporal or spatial sub-sampling, weight replications, and local receptive fields during computation. These fields are used to thoroughly evaluate the supplied photos, improving overall edge detection accuracy.
The input layer takes the input images and processes them pixel by pixel using the local receptive fields. Then, using defined fields, neurons are used to anticipate image end-points, oriented edges, and corners. Finally, additional layers are placed to identify the image's features. There are two types of cells in the convolute network: simple and complicated cells. Using its receptive field, the simple cell investigates the images in order to derive edge-like patterns. The complex cell then uses the higher receptive field to determine the exact position of the edges. The network computes the relationship and correlation between the features during the edge analysis, which enhances the overall prediction rate. To improve the overall segmentation accuracy, each layer in the network is connected to the sub-units. Figure 1 shows the architecture of a convolution neural network.

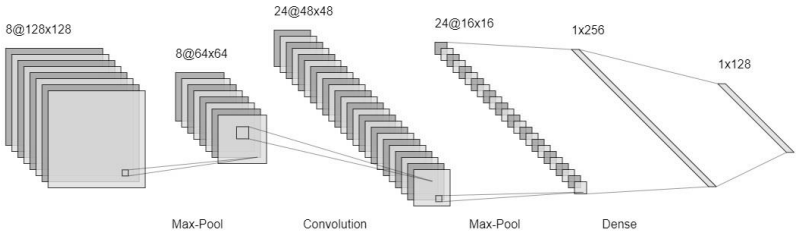

Figure 1: Convolution neural network architecture

The weight settings in each layer of the neural network are optimised for edge prediction accuracy. To increase total edge feature detection accuracy, the weights are connected to the next layer neurons. As previously said, the network has multiple layers, including input convolution layers, maximum pooling layer, sub-sampling layer, and completely convolution layer. Each layer uses a pre-defined function to do a given task. During the learning step, the images are initially presented in the shift format for extracting the edges from images. This method looks at the image edge features from various angles. The network encounters the translation invariant problem during computation, which has been solved by using the fully connected layer. After processing the input image, the down-sampling is performed using the max-pooling layer. This stage aids in the reduction of the feature set's dimensionality. The pooling method separates the images into non-overlapping regions, making it easier to spot edge-related information. The system's major goal is to reduce upper-layer computation complexity while maintaining translation invariance.

The first layer is the convolution layer, which is made up of kernels or filters that execute the convolution. The feature matrix is generated by convolving the convolution matrix with the input image. The stride or kernel of the convolution layer is applied to pictures to perform element-wise multiplication and obtain the output value. To obtain picture edge information, the kernel-based convolute operator is applied to each pixel in the image. For the output matrix, which is specified as padding, the input image matrix is combined with stripes of zero. The convolution network trains the features using the set of image patterns employed in the training step, and this technique is utilized to obtain exact edge details with little compute complexity. To determine the error value, the computed edge details are propagated to the previous layer. To train and learn the features, a back propagation technique is used. To obtain the edge information, this process is done indefinitely. The edge detected images are shown in figure 2 according to the discussion.

(a) Image conversion **(b) Noise Removed Images**

(c) Edge detected images **(d) Edge enhanced images**

Figure 2: Sample output edge detected images

To improve the entire pattern recognition process, the edge identified images are used for further image analysis. The system's effectiveness is then assessed using experimental data and discussions.

3 Results and Discussions

This section assesses the effectiveness of the edge detection procedure based on convolution neural networks. On an Intel R CoreTMi7-3770K CPU running at 4GHz, 8GB RAM, and Windows 10 Professional, the mentioned method is implemented using the MATLAB program. The collected photos are processed in this step by transforming them to binary images. The conversion procedure entails calculating brightness, hue, and saturation, and then estimating a threshold value to transform the pictures into binary. The noise in the photos is then replaced with a mean value that covers nearly all of the pixels in the image. The edge detection accuracy is improved by the noise removal technique. Finally, using the completely convolution layer, pooling layer, and activation function, convolute network layers are used to analyze picture edges in multiple directions such as vertical, horizontal, and diagonal. During the analysis, CNN network uses the 9*9 filter size kernel and, 64 numbers of channels are utilized. The PSRN, SSIM, and accuracy measures are then used to evaluate the introduced system's results. The following is how these metrics are calculated.

Peak Signal to Noise Ratio (PSNR)

The PSRN value is used to determine the quality of the anticipated edges, which is expressed in decibels (dB).

$$PSNR = 10log_{10}(\frac{R^2}{MSE}) \quad (1)$$

The MSE (difference between the actual and anticipated edge details) is given as the error rate in eq (1), and R is the input image maximum variations.

Structural Similarity Index (SSIM)

The SSIM metric determines how similar the actual and edge detected pictures are. eq is used to calculate the SSIM (2)

$$SSIM = [l(I_o, I_r)]^{\alpha}.[c(I_o, I_r)]^{\beta}.[s(I_o, I_r)]^{\gamma} \quad (2)$$

In eq (2) $\alpha = \beta = \gamma = 1$. According to the definition of SSIM, SSIM=1 if the two pictures I o and I rare are equivalent.

These performance metrics are compared to edge detection approaches like Canny, Laplacian of Gaussian, Prewitt, Zero-Crossing, Roberts, and Sobel. The quantity of photographs is used to compare the system's effectiveness. The acquired findings are shown in Figure 3.

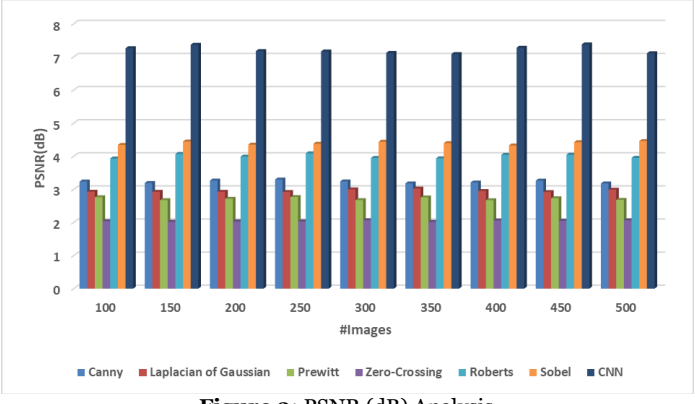

Figure 3: PSNR (dB) Analysis

In comparison to other existing edge detection methods, the presented convolution neural network (CNN) method computes picture edges with good quality, as shown in Figure 3. The convolution network in this case employs many layers with stride that convolution with the input information. The quality of the photos is improved by this convolution technique. The high PSNR value implies that the system has a high SSIM value when compared to other approaches, as seen in Figure 4.

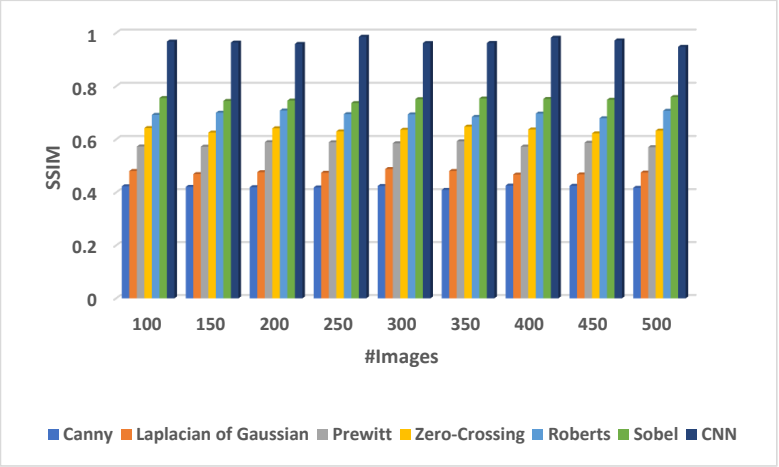

Figure 4: SSIM Analysis

The high SSIM value in figure 4 shows that the proposed CNN method achieves a high structural similarity with the actual and edge detected image. When compared to other methods, the effective usage training pattern and learning procedure improves the overall SSIM value. Table 1 shows the edge detection accuracy that was attained.

Table 1: Edge detection accuracy

Methods	Canny	Laplacian of Gaussian	Prewitt	Zero-crossing	Roberts	Sobel	CNN
Accuracy	78	79.68	82.89	83.1	87.38	89.29	94.86
PSNR	3.21	2.96	2.72	2.03	4.00	4.37	7.24
SSIM	0.420	0.476	0.581	0.635	0.69	0.749	0.966

As a result, when detecting edges from images, the proposed CNN technique achieves a 94.86 percent accuracy. The extracted edges have a higher similarity to the source photos (0.96) and are of higher quality (7.24dB). In comparison to other approaches, the extracted edges have a high efficiency value.

5 Conclusion

As a result, the research examines visual edge detection using convolution neural networks. The photos are gathered and transformed into binary images to make the edge detection algorithm easier to use. Following that, mean values are computed by comparing the pixel to a threshold value in order to remove noise information. This procedure improves the accuracy of edge detection much more. The edges from the image are then estimated using convolution layer functions such as pooling, drop out, and activation layers. The appropriate use of these layers enhances the overall accuracy of edge detection. The effectiveness of the system was examined using MATLAB findings, which showed that the system ensured 94.86 percent accuracy when compared to other methods. Optimized methods will be implemented in the future to increase overall edge detection accuracy.

6 References

[1] Mittal, Mamta, Amit Verma, Iqbaldeep Kaur, Bhavneet Kaur, Meenakshi Sharma, Lalit Mohan Goyal, Sudipta Roy, and Tai-Hoon Kim. "An efficient edge detection approach to provide better edge connectivity for image analysis." IEEE access 7 (2019): 33240-33255.

[2] Khan, Salman, Hossein Rahmani, Syed Afaq Ali Shah, and Mohammed Bennamoun. "A guide to convolutional neural networks for computer vision." Synthesis Lectures on Computer Vision 8, no. 1 (2018): 1-207.

[3] Xin, Shumian, Sotiris Nousias, Kiriakos N. Kutulakos, Aswin C. Sankaranarayanan, Srinivasa G. Narasimhan, and Ioannis Gkioulekas. "A theory of Fermat paths for non-line-of-sight shape

reconstruction." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6800-6809. 2019.

[4] May, Felix, Benjamin Rosenbaum, Frank M. Schurr, and Jonathan M. Chase. "The geometry of habitat fragmentation: Effects of species distribution patterns on extinction risk due to habitat conversion." Ecology and Evolution 9, no. 5 (2019): 2775-2790.

[5] Waldner, François, and Foivos I. Diakogiannis. "Deep learning on edge: Extracting field boundaries from satellite images with a convolutional neural network." Remote Sensing of Environment 245 (2020): 111741.

[6] Peng, Yeping, Songbo Ruan, Guangzhong Cao, Sudan Huang, Ngaiming Kwok, and Shengxi Zhou. "Automated product boundary defect detection based on image moment feature anomaly." IEEE Access 7 (2019): 52731-52742.

[7] Xie, Xin, Songlin Ge, Mingye Xie, Fengping Hu, and Nan Jiang. "An improved industrial sub-pixel edge detection algorithm based on coarse and precise location." Journal of Ambient Intelligence and Humanized Computing 11, no. 5 (2020): 2061-2070.

[8] Xiangxi, Zou, Zhang Yonghui, Zhang Shuaiyan, and Zhang Jian. "FPGA implementation of edge detection for Sobel operator in eight directions." In 2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), pp. 520-523. IEEE, 2018.

[9] Song, Yan, Baoying Ma, Wei Gao, and Shuping Fan. "Medical Image Edge Detection Based on Improved Differential Evolution Algorithm and Prewitt Operator." Acta Microscopica 28, no. 1 (2019).

[10] Topno, Preeti, and Govind Murmu. "An improved edge detection method based on median filter." In 2019 Devices for Integrated Circuit (DevIC), pp. 378-381. IEEE, 2019.

[11] Cao, JianFang, Lichao Chen, Min Wang, and Yun Tian. "Implementing a parallel image edge detection algorithm based on the otsu-canny operator on the hadoop platform." Computational intelligence and neuroscience 2018 (2018).

[12] Ghosal, Sudipta Kumar, Jyotsna Kumar Mandal, and Ram Sarkar. "High payload image steganography based on Laplacian of Gaussian (LoG) edge detector." Multimedia Tools and Applications 77, no. 23 (2018): 30403-30418.

[13] Ahmadian, Amir M., and Maryam Amirmazlaghani. "A novel secret image sharing with steganography scheme utilizing Optimal Asymmetric Encryption Padding and Information Dispersal Algorithms." Signal Processing: Image Communication 74 (2019): 78-88.

[14] Gholizadeh-Ansari, Maryam, Javad Alirezaie, and Paul Babyn. "Deep learning for low-dose CT denoising using perceptual loss and edge detection layer." Journal of digital imaging 33, no. 2 (2020): 504-515.

[15] Orujov, Farid, R. Maskeliūnas, R. Damaševičius, and W. Wei. "Fuzzy based image edge detection algorithm for blood vessel detection in retinal images." Applied Soft Computing 94 (2020): 106452.

[16] Kartik, P. V. S. M. S., Konjeti BVNS Sumanth, V. N. V. Sri Ram, and G. Jeyakumar. "Decoding of graphically encoded numerical digits using deep learning and edge detection techniques." Journal of Intelligent & Fuzzy Systems Preprint (2021): 1-8.

[17] He, Jianzhong, Shiliang Zhang, Ming Yang, Yanhu Shan, and Tiejun Huang. "Bi-directional cascade network for perceptual edge detection." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3828-3837. 2019.