International Journal of Basic and Applied Sciences, 14 (5) (2025) 45-57

International Journal of Basic and Applied Sciences

Website: www.sciencepubco.com/index.php/IJBAS
https://doi.org/10.14419/zgt6cp83
Research paper

Assessment of Deep Learning Models for Image Edge Detection

Wazir [1], Rajeshwar Dass [2] *

[1] *Research Scholar, Department of Electronics & Communication Engineering, Deenbandhu Chhotu Ram University of Science & Technology, Murthal, Haryana (India)-131039*
[2] *Associate Professor, Department of Electronics & Communication Engineering, Deenbandhu Chhotu Ram University of Science & Tech-nology, Murthal, Haryana (India)-131039*
**Corresponding author E-mail: rajeshwardass.ece@dcrustm.org*

Received: June 24, 2025, Accepted: August 23, 2025, Published: September 1, 2025

Abstract

Edge detection is a crucial technique in image processing, essential for various applications, including feature extraction, object identification, and segmentation. Conventional methods, such as Sobel and Canny filters, are not suitable for complex image structures and different lighting conditions in the image. Deep learning-based edge detection algorithms perform better in these situations and become an essential component of edge detection.
This study presented a deep learning-based approach for detecting edges in objects and complex backgrounds using the U-Net and its variants. This study used the BSDS500 dataset to compare eight deep learning-based edge detection algorithms for object and background out-lines. The testing outcomes show that the residual structure based on Resnet50-Unet outperformed in detecting the outlines of objects and backgrounds, achieving 95% accuracy, 0.62 precision, 0.89 recall, and 0.71 F1 score without augmentation of the BSDS500 dataset. The performance of Resnet50-Unet is enhanced when the augmented BSDS500 dataset is utilised, and it achieved 97% accuracy, 0.72 precision, 0.98 recall, and 0.83 F1 score. The results obtained are also cross-validated using the BIPED dataset. The proposed algorithm efficiently detects edges and object boundaries in real time.

Keywords: *Unet; ResNet50-Unet; BSDS500 & BIPED Datasets; Edge Detection; Encoder-Decoder.*

1. Introduction

Edge detection is a fundamental tool in computer image processing and computer vision. This focuses on locating boundaries in an image that exhibit a notable shift in colour or intensity [1]. These edges, or boundaries, are crucial structural components that specify the forms, textures, and divisions between various sections of an item [2]. Edge detection reduces data while retaining important information, simplifying image analysis [3]. This is crucial for object recognition, image segmentation, and feature extraction [4]. Accurately detecting edges makes interpreting visual data and identifying objects within an image easier, making this technique widely used in various domains [1-5]. Traditional edge detection methods primarily rely on mathematical techniques that analyse intensity variations in an image [6]. The Canny, Laplacian of Gaussian (LoG), Sobel, and Prewitt operators are among the most widely used techniques [4-6]. The Sobel and Prewitt operators apply convolutional filters to detect gradients in horizontal and vertical directions [7]. At the same time, the Log method uses second-order derivatives to detect zero-crossings in an image [8]. Traditional edge detection methods are unsuitable for complex image structures that require the detection of both background and object outlines [9].
Deep learning (DL)-based edge detection methods are essential for achieving high accuracy, robustness to noise, and adaptability. They learn features automatically from data, capturing both local details and global context. It is to learn edge features directly from large datasets [4-9]. Holistically nested Edge Detector (HED) is a well-known DL-based model for edge detection, utilising fully convolutional networks (FCNs) to detect edges at multiple scales [7-10]. Furthermore, based on the brief literature survey, it is concluded that Unet and its variants play a crucial role in detecting object and background outlines [11]. DeepLabV3+ is another advanced deep learning model that employs Atrous (dilated) convolutions to detect fine-grained edges, making it practical for high-resolution edge detection tasks [6-12]. Specialized networks, such as DexiNed (Deep Extreme Inception Network), have been developed to achieve highly accurate edge detection results [11-15].

1.1. Problem statement

Computer vision requires edge detection for object recognition, image segmentation, and image comprehension. The primary objective is to identify the edges in images where significant changes in colour, texture, or intensity occur. Conventional edge detection algorithms,

such as the Canny and Sobel operators, employ manually designed filters to recognize these transitions. Although these traditional methods perform effectively in controlled environments, they are highly susceptible to noise and often fail in complex environmental contexts. Their robustness and generalization ability are constrained by their tendency to underperform in low-contrast environments, cluttered backgrounds, or indistinct edges. Recent deep learning advances allow models to learn hierarchical features directly from data, improving edge detection. Deep convolutional neural networks (CNNs) can extract semantically significant features, enhancing edge location and recognition. Nonetheless, these models frequently encounter a compromise between semantic comprehension and geographic accuracy. Deep convolutional neural networks downsample geometrical details, and spatial resolution models may lack the contextual understanding to distinguish object boundaries and extraneous textures.

The UNet architecture has become popular for its symmetric encoder-decoder design and the incorporation of skip links, which aid in maintaining spatial information during reconstruction. Nonetheless, the conventional UNet encoder is somewhat shallow, constraining its capacity to capture intricate semantic information essential for differentiating genuine edges from noise in complicated images. This research proposes a hybrid architecture, ResNet50-UNet, which integrates a deep residual network (ResNet-50) as the encoder within the UNet framework. ResNet-50 enhances the network's capacity to extract deep, semantically rich features, while the UNet decoder and skip connections ensure the retention of high-resolution spatial information. The goal is to achieve robust edge detection with improved localization and semantic relevance, particularly in challenging datasets such as BSDS500, which is characterized by high variability in texture, object scale, and boundary clarity. This study uses ResNet-50 and UNet to create a model that can detect continuous, precise, and semantically meaningful edges in diverse and natural image scenarios. Significant contributions of the research are:

1) The ResNet-50 encoder is embedded into the UNet framework to provide a customized deep learning architecture that allows the network to learn rich semantic characteristics while maintaining fine-grained spatial details that are essential for precise edge detection.
2) Using skip connections and multi-level feature fusion within the UNet decoder, the model enhances the capacity to find continuous and well-aligned edges in natural images, even in situations with complicated textures or weak borders.
3) The network is effectively trained for pixel-wise binary classification of edge and non-edge regions using binary cross-entropy loss. This decision encourages precise learning of sparse edge distributions in datasets such as BSDS500 and offers a steady training regime.
4) The BSDS500 benchmark is used to assess the suggested model systematically. The outcomes validate the efficacy of the hybrid architecture and BCE loss configuration by showing competitive or better performance in standard edge detection criteria when compared to baseline approaches.

1.2. Key innovations

The key innovations of this work are as follows:

1) Integration of Deep Semantic Encoding with Spatial Decoding: In an inventive way, the model combines the spatially aware UNet decoder with a deep residual encoder (ResNet-50). Together, they improve the model's capacity to learn abstract features from deeper layers and restore the spatial resolution required for accurate edge localisation.
2) Enhanced Feature Fusion via Multi-Level Skip Connections: Hierarchical skip connections between matching encoder and decoder layers are used in this architecture, as opposed to conventional CNNs or pure UNet models. Through these links, low-level texture details and high-level semantic signals can be fused, greatly enhancing edge continuity and localisation over a range of visual complexities.
3) Efficient Binary Edge Learning Using BCE Loss: Binary Cross-Entropy (BCE) loss solves the edge detection pixel-wise binary classification problem. This loss function ensures stable convergence and is ideal for handling imbalance and sparsity in edge maps from natural image datasets.

1.3. Limitations of the work and solution implications

1) The BSDS500 dataset contains only 500 distinct images; however, it includes a wide range of scene types, including natural, urban, indoor, outdoor, and synthetic environments.
2) The 481×321-pixel standard for all images may limit the ability to recognize fine-grained boundaries or run high-resolution models.
3) Multiple ground truths (5–6) are used to annotate each image, which results in contours that are weak or inconsistent. Pointwise approaches may find it challenging to acquire accurate boundary definitions as a result.
4) It's possible that some contours drawn by a small number of annotators don't accurately depict the borders of an object.
5) The borders of an object may not be adequately depicted by some contours created by a limited number of annotators.
6) Cross-validating a model with an external dataset, which tests the model outside of the training distribution, improves the model's generalizability and reliability.

These restrictions are eliminated in the present work by using an external dataset (BIPED). Testing of an external dataset has been used to investigate cross-validation. Using an external dataset to test a model is a crucial initial step in determining its generalisation ability. When the model is trained and validated on one dataset, it is tested on another. This ensures that the model is not biased toward the original training distribution and can handle changes in real-world scenarios. In fields of external testing, it is crucial to prove reliability across a variety of demographics, devices, and situations. Reporting results on an external dataset supports the established model's dependability and suitability for practical use.

2. Related work

Edge detection has recently advanced from conventional methods to deep learning algorithms. Although they rely on gradient-based techniques, classical systems like the canny edge detector frequently encounter difficulties in complicated scenarios. Machine learning techniques like SVMs and Random Forests have been used to overcome these restrictions by using hand-crafted features for edge categorisation. Since the introduction of deep learning, edge detection has undergone significant changes. To conduct end-to-end edge forecast and capture multi-scale and multi-level features, Xie and Tu (2015) presented Holistically-Nested Edge Detection (HED), which makes use of fully convolutional neural networks under deep supervision. Building on this, Liu et al. (2019) developed Richer Convolutional Features (RCF), which attains state-of-the-art results on benchmarks such as BSDS500 by aggregating hierarchical attributes from all convolutional layers. In this study, eight deep learning-based edge detection algorithms were compared for background and object outline detection using F1-

score, precision, recall, and accuracy. The experimental findings concluded that residual-based ResNet50-UNet performed better in detecting boundaries. Table 1 briefly reviews the literature on deep learning-based edge detection techniques.

Table 1: A Brief Review of the Literature on Deep Learning-Based Edge Detection Techniques

Investigator(s)	Year	Dataset(s)	Edge Detection Method	Assessment Parameter
Changbao Wen et al. [15]	2018	BSDS500	VGG16	ODS, F-score
Nan Xue et al. [16]	2020	Wireframe and York Urban dataset	HAWP	mAP and F-score
Xavier Soria et al. [17]	2020	BSDS500 and BIPED	CNN	F-measure, ODS, and OIS
Ruoxi Deng et al. [18]	2020	BSDS500 and multi-cue datasets	CNN	ODS and OIS
Zhen et al. [19]	2020	Cityscapes dataset	ResNet 101	ODS and AP metrics
Orujov et al. [20]	2020	STARE dataset/ Digital Retinal Images	(Type-2) fuzzy rules	Accuracy
Yang Liu et al. [21]	2020	BSDS500 and NYUD	Edge proportion statistics	ODS and F-measure metric
Xiaokang Yu et al. [22]	2021	The agricultural products' Images	Morphological filter	PSNR
Mario Versaci et al. [23]	2021	Thermal Infrared, Images.	Fuzzy divergence and fuzzy entropy minimisation	MSE, MAE, and SSIM
Hongliang Zhao et al. [24]	2021	BSDS500,	Strong Semantics and Weak Supervision (SSWS)	mIoU
Zhuo Su et al. [25]	2021	BSDS500, NYUD, and Multi-cue	CNN-PiDiNet	ODS and OIS
O. Elharrouss et al. [26]	2022	BSDS500	Multi-scale representation and refined Network	SI
F. Wang et al. [27]	2023	BSDS500	Bi-Directional Cascade Network	ODS, OIS, and AP
O. Elharrouss et al. [28]	2023	BSDS500, NYUD	CHRNet	SI
D. Jing et al. [29]	2023	BSDS500	EDD	SI
Al-Amaren et al. [30]	2023	BSDS500	DCNNs	SI
Fuzhang Li et al. [31]	2024	BSDS500,	UHNet	ODS, OIS, and AP
J. Zhang et al [32]	2024	BSDS500	PCNNs	PSNR, SSIM
Firas Abedi et al. [33]	2024	BSDS500	DRNet	PSNR
Ying An et al. [34]	2024	BSDS500 and NYUDv2	U-net	ODS F-score
K. Muntarina et al. [35]	2025	BSDS500	HED, FCN, DeepEdgeUnet, MultiResEdge	Entropy, MSE, PSNR, SSIM and FSIM
Ming Wang [36]	2025	BSDS500	PiDiNet	SI
WenlinLiet al. [37]	2025	BSDS500,	CNN	ODS F-score
Xiaodiao Chen et al. [38]	2025	BSDS500 and NYUDv2	GCN	SI
Xi Yanget al. [39]	2025	BSDS500, BIPED, NYUD	TANET	SI

Note: OSD- Optimal_Dataset_Scale, OIS-Optimal_Image_Scale, AP- Average Precision, msAP- Mean Structural Average Precision, PiDiNet -Pixel Difference Network, DCNNs-Deep Convolutional Neural Networks, SI-Similarity Index, PCNNs-Parallel Convolutional Neural Networks, DRNet-Dense Residual Network Structure (DRNet), GCN-Graph Convolutional Network.

2.1. Research trends and gaps

From Table 1, the following research trends and gaps have been identified:

1) Research studies have shown that by integrating both spatial and semantic features at different levels, encoder-decoder models can detect edges more accurately than fully convolutional networks (FCNs).
2) Literature consistently shows that U-Nets with skip connections make a good boundary detector.
3) Recent research has shown that to detect sharp edges, it is necessary to combine low-level and high-level features.
4) Literature reports that class imbalance (few edge pixels vs. many background pixels) reduces model performance.
5) Few studies focus on cross-dataset validation, so generalisation of ResNet50-U-Net across domains is still a challenge.

Table 1 concludes that deep learning-based U-Net has been widely used to get contextual information & hierarchical features from input data [34]. It also concludes that traditional edge detection techniques are not suitable for edge detection. Furthermore, the BSDS500 dataset is frequently used for edge detection [15-39]. F1-score and similarity index are widely utilised as assessment parameters for edge detection. This research paper is organised to present the suggested deep learning-based edge detection method utilising the ResNet50-UNet model straightforwardly and logically. It starts with an introduction outlining the significance of edge detection in computer vision as well as the rationale for using a hybrid deep learning model. The key innovations part, which describes the exceptional contributions of this work, including architectural decisions and performance evaluation techniques, follows the problem statement section, which explains the current difficulties in producing accurate and continuous edge maps. Related work emphasises the gaps this study fills by reviewing prior research. Fundamental methodology is described in Framework Adopted for Edge Detection Using Deep Learning-Based Algorithms. Model Optimisation technique is also included to increase training accuracy and efficiency. The experimental setup section details the dataset, data preparation and enhancement, and experimental setting. To provide a summary of the implementation, a brief pseudo-code is supplied. The criteria used to examine model execution accuracy, precision, recall, and F1-score are described in the assessment parameters section. The efficiency of the suggested strategy is then confirmed by presenting and discussing the experimental results. The conclusion, which comes last, delivers a summary of the results and recommendations for additional research gaps.

3. Framework adopted for edge detection using the deep learning-based algorithms

The framework adopted for edge detection using the encoder-decoder-based algorithm is given below in Fig. 1.

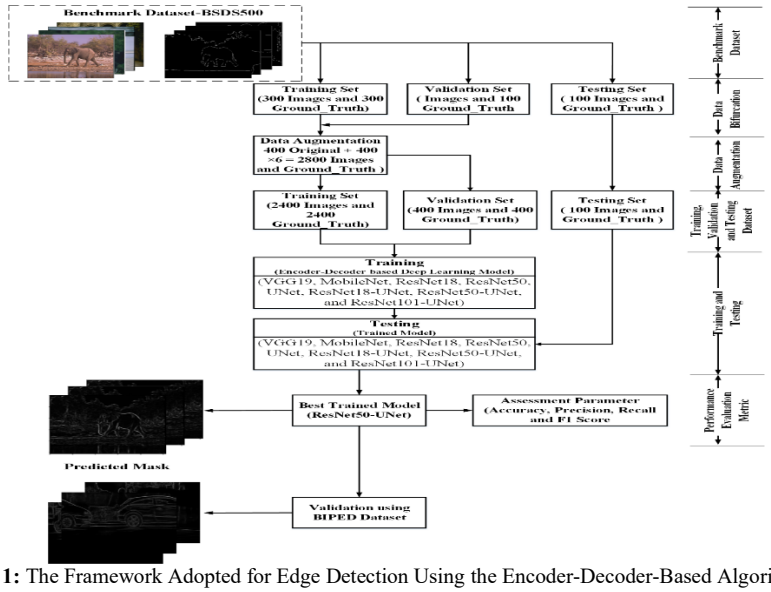

Fig. 1: The Framework Adopted for Edge Detection Using the Encoder-Decoder-Based Algorithm.

Deep learning models are crucial to autonomous navigation [16]. These networks use an encoder-decoder architecture. CNNs extract hierarchical features from an image, reducing spatial resolution while preserving edge information [16-19]. The decoder then reconstructs the edge map by progressively upsampling and refining feature maps. Skip connections in the U-Net help to retain fine-grained edge details. Training the edge detection model requires a diverse dataset containing images with labelled edges, such as the BSDS500 and a combination of the BSDS500 & BIPED datasets [15-39]. Data augmentation plans, such as rotation, flipping, and scaling, improve generalisation and robustness [24-35]. By leveraging deep learning, the encoder-decoder-based framework provides a powerful and efficient solution for high-quality edge detection across various domains [27-33].

3.1. Model optimization

This work utilises eight encoder-decoder-based deep architectures, VGG19, MobileNet, ResNet18, ResNet50, UNet, Res18-UNet, Res-Net50-UNet, and Res101-UNet, for edge detection. The benchmark dataset, BSDS500, has been used in this work, and the original image size was resized to 256 × 256. The obtained results were cross-validated using an external dataset (BIPED).
The proposed work's problem definition and mathematical modelling of UNet's residual network variant for edge detection are found here. The network learns a feature mapping of three under height (H), width (W), and channel (C) in the image for edge detection. Here, Residual is an encoder consisting of multiple convolutional layers and residual. UNet is used as a decoder to reconstruct the spatial resolution using transposed convolutions and skip connections from encoder layers. The model is trained using Binary-Cross-Entropy (BCE) loss, as shown in Equation 1, where A (I, j) and A((i,j)) are the real and predicted images.

$$L_{BCE} = -\sum_{i=1}^{H}\sum_{j=1}^{W}\left[A(i,j) * \log\left(\widehat{A(i,j)}\right)\right) + \left(1 - V_{ij}\right)\log\left(1 - \widehat{A(i,j)}\right)\right] \quad (1)$$

4. Experimental setup

The edge detection deep learning models have been trained on the BSDS500 dataset and tested on the BIPEDs dataset. Before training, the dataset has been preprocessed, including data augmentation and normalization [15-39].

4.1. Benchmark dataset

Berkeley Segmentation Dataset (BSDS500) is a prominent baseline dataset for image segmentation and edge detection [15–39]. The Berkeley Vision Group developed it, an expansion of the BSDS300, its predecessor [20-26]. BSDS500 dataset uses 500 natural images annotated with human-drawn ground truth segmentations to evaluate image edge detection &segmentation algorithms [21-27]. The BSDS500 dataset has five hundred images with multiple ground truths. Three hundred images are used for training, one hundred for validation, and one hundred for testing [25-30]. Images are 321 x 481 pixels [26]. The multiple human annotators provided the manually marked benchmarked images as ground truth [24]. It is also observed that the dataset has been collected from natural scenes and helps to assess the generalisation of edge detection algorithm capability [15-30]. The Benchmark BSDS500 image dataset (a-d), sample image (a1-d1), and sample image with annotation as ground truth images are shown in Fig. 2.

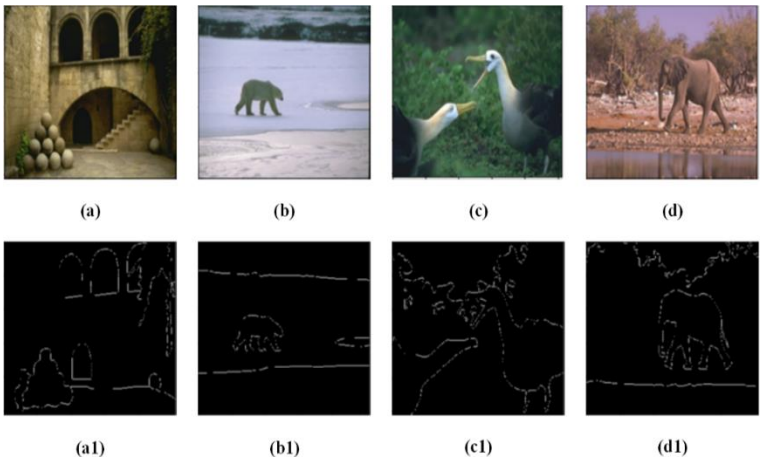

Fig. 2: The Benchmark BSDS500 Image Dataset (A-D), Sample Image (A1-D1), and Sample Image with Annotation as Ground Truth Images.

The second dataset, BIPED (Barcelona-Images for Perceptual Edge-Detection), is a high-quality benchmark for perceptual edge detection algorithms. It includes 250 high-resolution outdoor images taken with high-end cameras under various real-world scenarios. It is a fine-grained ground truth that several people have manually annotated to minimise noise from unimportant textures and reflect edges that are perceptually significant to humans. BIPED has established itself as a benchmark for contemporary edge detection studies. This work uses the BIPED dataset, an external dataset, to validate the performance of the proposed model. The BSDS500 and BIPED datasets have been used to check the performance of the model. Table 4 provides a concise overview of the BSDS500 & BIPED dataset.

4.2. Phase i: dataset preparation module

The BSDS500 dataset is standard for edge detection research because it provides high-quality human-annotated edges, allowing researchers to compare algorithms under consistent conditions [15-39]. The following steps have been adopted for the dataset preparation module to facilitate edge detection from images.

1) Resize the dataset image to a 256 × 256 resolution.
2) Conversion of ground truth into JPG format from the .mat file of the BSDS500 dataset
3) Prepared Benchmark Dataset.

4.3. Phase ii: data augmentation module

The detailed, exhaustive review of the literature on data augmentation techniques used in earlier studies is presented in Table 2.

Table 2: The Detailed, Exhaustive Review of Literature on Data Augmentation Techniques Used in Earlier Studies

Investigator(s)	Year	Augmentation Technique
Zhuo Su et al. [25]	2021	Flipping, Scaling& Rotation
O. Elharrouss et al. [26]	2022	Flipping, Scaling & Rotation
Omar Elharrouss et al. [40]	2023	Flipping, Rotation & Translation
Hao Shu et al. [40]	2024	Rotation, Flipping &Cropping
Fuzhang Li et al. [31]	2024	Flipping, Scaling & Rotation
Ming Wang [36]	2025	Flipping, Scaling & Rotation

Table 2 analyses the research on the data augmentation technique employed in previous work [25], [26], [31], [40]. It concludes that investigators widely employ data augmentation plans, such as flipping, scaling, and rotation, in their studies [40-44]. It shows the details of the BSDS500 dataset after augmentation [25], [26]. The augmentation technique and parameters that enable data augmentation are displayed in Table 3.

Table 3: Technique and Parameters Enabling Data Augmentation

Data Augmentation Technique(s)	Parameter Value	Pixel Information	Augmented images
-	Original Size of Image	256×256 Pixels	Original
Rotation	90º and 180º	256×256 Pixels	2 Times
Flipping	Horizontal and vertical	256×256 Pixels	2 Times
Scaling	Scaling	256×256 Pixels	2 Times
Total Augmentation Technique Used		Six augmented techniques with the original dataset = 7 Times	

Table 4 summarizes the features of the experimental and external datasets. The carefully selected subset of 50 images has been taken from the external dataset for external validation to provide an accurate and insightful assessment. The selection method considered anatomical regions, relevance, and image quality to ensure sample diversity. 50 images can effectively and computationally evaluate the model's performance. This subset served as a separate testing standard to confirm that the model may generalize outside of the training dataset. Without having to evaluate the complete external dataset, this selection method offers a fair, objective, and functional evaluation of robustness. Two experiments have been prepared. The model was trained and tested on BSDS500 in the first experiment. The best-trained network from the first experiment was tested on BIPED in the second experiment.

Table 4: Features of the Experimental and External Datasets (BSDS500 & BIPED)

Dataset	Total Image	Without Augmentation		After Augmentation			
		No. of Original Images	Testing	Number of images after Augmentation			Testing
				Total Images	Training	Validation	
BSDS500	500	400 for Training & Validation (400 images and ground truth for Training and Validation)	100	2800	2400	400	100
BIPED (External Dataset)	250	50 external BIPED images were used to test the proposed model.					

4.4. Phase iii: experimental setting

This work uses the Adam optimizer and exponential attenuation to adjust learning rate and mini-batch size. Consequently, the potential problem of gradient fading during training is minimized. Experimental parameters and model parameters were selected through a series of experiments. Table 5 presents the selected training parameters to avoid overfitting.

Table 5: Selected Training Parameters to Avoid Overfitting

S. No.	Model Parameter	Selected parameter value
1	Number of epochs	20
2	Learning Rate	10^{-3}
3	Mini Batch Size	16
4	Optimizer	Adam

Several strategies were employed to prevent overfitting while training deep-learning models on the BSDS500 dataset [40]. Regularization techniques included dropout (0.3 - 0.5) to deactivate random neurons, L2 weight decay (1e-3) to prevent large weights, and batch normalization for stable training [25-30]. Data augmentation increased dataset diversity, such as random cropping, flipping, rotation, and noise injection. Early stopping monitors validation loss, stopping training if overfitting has been detected. These strategies ensured better generalisation, improved edge detection accuracy, and reduced model overfitting during training. Table 6 outlines the computational settings.

Table 6: Computational Settings

Resources	Details
RAM	16 GB
CPU	IntelCorei7-12650K@2.3GHz
GPU	NVIDIA GEFORCE RTX 4060 with 8 GB RAM
Platform	Python

The layers of a deep learning model determine its number of parameters. Equation 2 illustrates the result of the feature map.

$$H_{out} = \frac{H_{in}-K+2\times P}{S} + 1 \quad (2)$$

Note –H_{out}- Output feature size (Output), H_{in} - Input size (M × N), M- Width of input, N – Height of input
K- Kernel or channel size, P- Padding, S – Stride

$$\widehat{H_{out}}[\text{Normalized Output}] = \frac{H_{out}-\mu_i}{\sqrt{\sigma_i{}^2+\epsilon}} \quad (3)$$

$\widehat{H_{out}}$ – Normalised output, H_{out} – Feature value, μ_i- Mean of the batch, $\sigma_i{}^2$- Variance of the batch, $\in$ - Small constant for numerical stability (approx. value [10^{-5}])
Transpose Convolution output

$$H_{out} = (H_{in}\text{-}1) \times \text{Stride} - 2\times\text{Padding} + \text{kernel size} \quad (4)$$

Batch Normalization (BN) is a method that normalizes activations inside a mini-batch to stabilize and speed up deep neural network training [40]. It helps to reduce internal covariate shifts, allowing the model to converge faster and generalise better. Table 7 presents the ResNet50-UNet's encoder and decoder side layer architectures.

Table 7: The Encoder and Decoder Side Layer Architecture of Resnet50-Unet

Layer (sort)	Output Pattern [C, M, N]	Parameter [K, P, S]	Layer (sort)	Output Pattern [C, M, N]	Parameter [K, P, S]
Encoder Side					
Input Size - [256 256 3]			Conv2d-100	[256, 32, 32]	[3, 1, 1]
Conv2d	[64, 128, 128]	[3, 1, 2]	BatchNorm-101	[256, 32, 32]	-
BatchNorm.-2	[64, 128, 128]	-	ReLU-102	[256, 32, 32]	-
ReLU-3	[64, 128, 128]	-	Conv2d-103	[256, 32, 32]	[3, 1, 1]
Conv2d-4	[64, 128, 128]	[3, 1, 1]	BatchNorm-104	[256, 32, 32]	-
BatchNorm.-5	[64, 128, 128]	-	ReLU-105	[256, 32, 32]	-
ReLU-6	[64, 128, 128]	-	Conv2d-106	[1024, 32, 32]	[3, 1, 1]
Conv2d-7	[64, 128, 128]	[3, 1, 1]	BatchNorm-107	[1024, 32, 32]	-
BatchNorm.-8	[64, 128, 128]	-	ReLU-108	[1024, 32, 32]	-
ReLU-9	[64, 128, 128]	-	Bottleneck-109	[1024, 32, 32]	
Conv2d-10	[256, 128, 128]	[3, 1, 1]	Conv2d-110	[256, 32, 32]	[3, 1, 1]
BatchNorm 13	[256, 128, 128]	-	BatchNorm-111	[256, 32, 32]	-
Conv2d-12	256, 128, 128]	[3, 1, 1]			

Batch Norm-13	[256, 128, 128]	-	ReLU-112	[256, 32, 32]	-
ReLU-14	[256, 128, 128]	-	Conv2d-113	[256, 32, 32]	[3, 1, 1]
Bottleneck-15[256, 128, 128]			BatchNorm-114	[256, 32, 32]	-
Conv2d-16	[64, 128, 128]	[3, 1, 1]	ReLU-115	[256, 32, 32]	-
BatchNorm-17	[64, 128, 128]	-	Conv2d-116	[1024, 32, 32]	[3, 1, 1]
ReLU-18	[64, 128, 128]	-	BatchNorm-117	[1024, 32, 32]	-
Conv2d-19	[64, 128, 128]	[3, 1, 1]	ReLU-118	[1024, 32, 32]	-
BatchNorm-20	[64, 128, 128]	-	Bottleneck-119[1024, 32, 32]		
ReLU-21	[64, 128, 128]	-	Conv2d-120	[256, 32, 32]	[3, 1, 1]
Conv2d-22	[256, 128, 128]	[3, 1, 1]	BatchNorm-121	[256, 32, 32]	-
BatchNorm-23	[256, 128, 128]	-	ReLU-122	[256, 32, 32]	-
ReLU-24	[256, 128, 128]	-	Conv2d-123	[256, 32, 32]	[3, 1, 1]
Bottleneck-25[256, 128, 128]			BatchNorm-124	[256, 32, 32]	-
Conv2d-26	[64, 128, 128]	[3, 1, 1]	ReLU-125	[256, 32, 32]	-
BatchNorm-27	[64, 128, 128]	-	Conv2d-126	[1024, 32, 32]	[3, 1, 1]
ReLU-28	[64, 128, 128]	-	BatchNorm-127	[1024, 32, 32]	-
Conv2d-29	[64, 128, 128]	[3, 1, 1]	ReLU-128	1024, 32, 32]	-
BatchNorm-30	[64, 128, 128]	-	Bottleneck-129[1024, 32, 32]		
ReLU-31	[64, 128, 128]	-	Conv2d-130	[256, 32, 32]	[3, 1, 1]
Conv2d-32	[256, 128, 128]	[3, 1, 1]	BatchNorm-131	[256, 32, 32]	-
Batch Norm-33	[256, 128, 128]	-	ReLU-132	[256, 32, 32]	-
ReLU-34	[256, 128, 128]	-	Conv2d-133	[256, 32, 32]	[3, 1, 1]
Bottleneck-35[256, 128, 128]			BatchNorm-134	[256, 32, 32]	-
Conv2d-36	[64, 128, 128]	[3, 1, 1]	ReLU-135	[256, 32, 32]	-
Batch Norm-37	[64, 128, 128]	-	Conv2d-136	[1024, 32, 32]	[3, 1, 1]
ReLU-38	[64, 128, 128]	-	BatchNorm-137	[1024, 32, 32]	-
Conv2d-39	[64, 128, 128]	[3, 1, 1]	ReLU-138	[1024, 32, 32]	-
Batch Norm-40	[64, 128, 128]	-	Bottleneck-139[1024, 32, 32]		
ReLU-41	[64, 128, 128]	-	Conv2d-140	[512, 32, 32]	[3, 1, 1]
Conv2d-42	[512, 64, 64]	[3, 1, 2]	BatchNorm-141	[512, 32, 32]	-
Batch Norm-43	[512, 64, 64]	-	ReLU-142	[512, 32, 32]	-
Conv2d-44	[512, 64, 64]	[3, 1, 1]	Conv2d-143	[512, 16, 16]	[3, 1, 1]
Batch Norm-45	[512, 64, 64]	-	BatchNorm-144	[512, 16, 16]	-
ReLU-46	[512, 64, 64]	-	ReLU-145	[512, 16, 16]	-
Bottleneck-47[512, 64, 64]			Conv2d-146	[2048, 16, 16]	[3, 1, 1]
Conv2d-48	[64, 128, 128]	[3, 1, 1]	BatchNorm-147	[2048, 16, 16]	-
Batch Norm-49	[64, 128, 128]	-	Conv2d-148	[2048, 16, 16]	[3, 1, 1]
ReLU-50	[64, 128, 128]	-	BatchNorm-149	[2048, 16, 16]	-
Conv2d-51	[128, 64, 128]	[3, 1, 1]	ReLU-150	[2048, 16, 16]	-
Batch Norm-52	[128, 64, 128]	-	Bottleneck-151[2048, 16, 16]		
ReLU-53	[128, 64, 128]	-	Conv2d-152	[512, 16, 16]	[3, 1, 1]
Conv2d-54	[512, 64, 64]	[3, 1, 1]	Batch Norm-153	[512, 16, 16]	-
Batch Norm-55	[512, 64, 64]	-	ReLU-154	[512, 16, 16]	-
ReLU-56	[512, 64, 64]	-	Conv2d-155	[512, 16, 16]	[3, 1, 1]
Bottleneck-57[512, 64, 64]			BatchNorm-156	[512, 16, 16]	-
Conv2d-58	[128, 64, 64]	[3, 1, 1]	ReLU-157	[512, 16, 16]	-
Batch Norm-59	[128, 64, 64]	-	Conv2d-158	[2048, 16, 16]	[3, 1, 1]
ReLU-60	[128, 64, 64]	-	BatchNorm-159	[2048, 16, 16]	-
Conv2d-61	[128, 64, 64]	[3, 1, 1]	ReLU-160	[2048, 16, 16]	-
Batch Norm-62	[128, 64, 64]	-	Bottleneck-161[2048, 16, 16]		
ReLU-63	[128, 64, 64]	-	Conv2d-162	[512, 16, 16]	[3, 1, 1]
Conv2d-64	[512, 64, 64]	[3, 1, 1]	BatchNorm-163	[512, 16, 16]	-
Batch Norm-65	[512, 64, 64]	-	ReLU-164	[512, 16, 16]	-
ReLU-66	[512, 64, 64]	-	Conv2d-165	[512, 16, 16]	[3, 1, 1]
Bottleneck-67[512, 64, 64]			Batch Norm-166	[512, 16, 16]	-
Conv2d-68	[128, 64, 64]	[3, 1, 1]	ReLU-167	[512, 16, 16]	-
Batch Norm-69	[128, 64, 64]	-	Conv2d-168	[2048, 16, 16]	[3, 1, 1]
ReLU-70	[128, 64, 64]	-	BatchNorm-169	[2048, 16, 16]	-
Conv2d-71	[128, 64, 64]	[3, 1, 1]	ReLU-170	[2048, 16, 16]	-
Batch Norm-72	[128, 64, 64]	-	Bottleneck-171[2048, 16, 16]		
ReLU-73	[128, 64, 64]	-	Decoder Side		
Conv2d-74	[512, 64, 64]	[3, 1, 1]	ConvTranspose-172	[1024, 32, 32]	[4, 1, 1]
Batch Norm-75	[512, 64, 64]	-	ReLU-173	[1024, 32, 32]	-
ReLU-76	[512, 64, 64]	-	BatchNorm2d-174	[1024, 32, 32]	-
Bottleneck-77[512, 64, 64]			ConvTranspose-175	[512, 64, 64]	[4, 1, 1]
Conv2d-78	[256, 64, 64]	[3, 1, 1]	ReLU-176	[512, 64, 64]	-
BatchNorm-79	[256, 64, 64]	-	BatchNorm2d-177	[512, 64, 64]	-
ReLU-80	[256, 64, 64]	-	ConvTranspose-178	[256, 128, 128]	[4, 1, 1]
Conv2d-81	[256, 32, 32]	[3, 1, 2]	ReLU-179	[256, 128, 128]	-
BatchNorm-82	[256, 32, 32]	-	BatchNorm2d-180	[256, 128, 128]	-
ReLU-83	[256, 32, 32]	-	ConvTranspose-181	[64, 256, 256]	[4, 1, 1]
Conv2d-84	[1024, 32, 32]	[3, 1, 1]	ReLU-182	[64, 256, 256]	-
BatchNorm-85	[1024, 32, 32]	-	BatchNorm2d-183	[64, 256, 256]	-
Conv2d-86	[1024, 32, 32]	[3, 1, 1]	ConvTranspose-184	[64, 256, 256]	[4, 1, 1]
BatchNorm-87	[1024, 32, 32]	-	ReLU-185	[64, 256, 256]	
ReLU-88	[1024, 32, 32]	-	BatchNorm2d-186	[64, 256, 256]	
Bottleneck-89[1024, 32, 32]			ConvTranspose-187	[64, 256, 256]	[1, 1, 0]
Conv2d-90	[256, 32, 32]	[3, 1, 1]	ReLU-188	[64, 256, 256]	
BatchNorm-91	[256, 32, 32]	-	BatchNorm2d-189	[64, 256, 256]	
ReLU-92	[256, 32, 32]	-	Conv2d-190	[1, 256, 256]	[1, 1, 0]

Conv2d-93	[256, 32, 32]	[3, 1, 1]	Output size of Predicted Mask – [256 256 1]
BatchNorm-94	[256, 32, 32]	-	Total Trainable Parameter - 34,720,705
ReLU-95	[256, 32, 32]	-	
Conv2d-96	[1024, 32, 32]	[3, 1, 1]	
BatchNorm-97	[1024, 32, 32]	-	
ReLU-98	[1024, 32, 32]	-	
Bottleneck-99	[1024, 32, 32]		

The variant of UNet is a hybrid deep learning model that combines the encoder and UNet's upsampling structure as the decoder for edge detection [15-39]. The encoder extracts hierarchical features using a pre-trained network with residual connections, enabling deeper feature learning [40]. The decoder consists of transposed convolutions and skips connections, progressively upsampling the feature maps while restoring spatial details [25-26, 45-48]. The final layer applies a 1×1 convolution to generate pixel-wise predictions. Batch normalization and ReLU activation improve training stability. This architecture efficiently balances feature extraction and precise localisation (UNet), making it suitable for edge detection tasks. The basic architecture of Unet based on residual is presented in Fig.3.

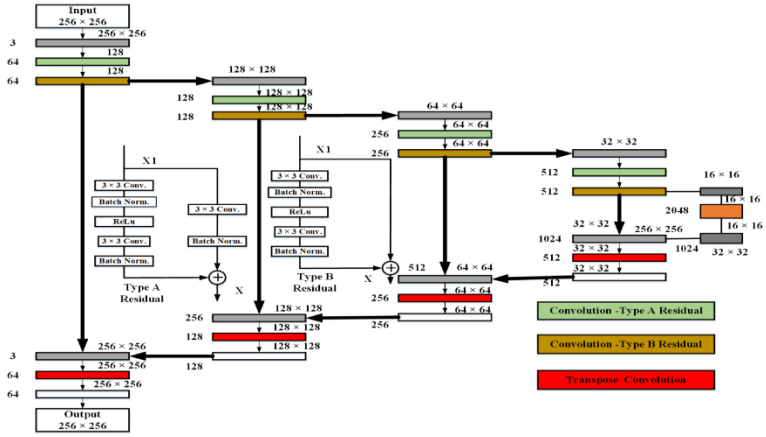

Fig. 3: The Basic Architecture of Unet Based on Residual.

4.5. Pseudo code for ResNet50-UNet edge detection

The entire training and validation procedure for the ResNet50-UNet model, which is utilised in edge detection tasks, is described by the pseudo code. First, the model, optimizer, and loss function are initialized. Next, the dataset is divided into training, validation, and test sets. To increase generalisation, data augmentation methods including rotation, cropping, and random flipping are applied to the training data. The network processes picture batches to produce predicted edge maps during each training epoch, and the Adam optimizer is used to calculate and minimise the Binary Cross-Entropy loss. The model is verified on an unaugmented dataset after every epoch. By contrasting the binary outputs with ground truth labels, essential performance metrics, including accuracy, precision, recall, and F1-score, are computed. These measures aid in evaluating the model's edge detection performance, guaranteeing that it learns across data distributions with accuracy and consistency. Pseudo code is given below:

Initialize:

- Load ResNet50-UNet model
- Define Binary Cross-Entropy loss
- Set optimizer to Adam with learning rate = 0.001
- Set batch size = 16
- Set input image size = 256 × 256
- Total epochs = 20

Prepare data: BSDS500, and external dataset BIPED

- Resize all images to 256 × 256
- Apply data augmentation to the training set:
- Rotation, Random flips, scaling
- Normalize input images
- Normalize validation and test sets (no augmentation)

For each epoch from 1 to 20:

- Train the model:

For each batch in the training set:

- Apply augmentation
- Pass 256 × 256 images through model → predicted edge maps
- Compute loss
- update model weights
- Validate the model:

Initialize TP, FP, FN, TN = 0

For each batch in the validation set:

- Pass 256 × 256 images through model → predictions
- Apply threshold to convert prediction to binary edges
- Compute loss
- Compare with ground truth to update TP, FP, FN, TN
- Compute metrics:

Accuracy = (TP + TN) / (TP + FP + FN + TN)

Precision = TP / (TP + FP)
Recall = TP / (TP + FN)

F1-score = 2 * (Precision * Recall) / (Precision + Recall)

- Log training loss, validation loss, and metrics

Save the trained model after the final epoch.

4.6. Assessment parameters for performance evaluation indicators

To demonstrate the approach's effectiveness in the current endeavour, the edge detection algorithm derived from traditional image processing methods and a variant of the Unet technique has been selected for comparison [15-39]. Precision (Pre), Recall (Rec), and F1 Score parameters have been used as evaluation indicators to assess the algorithms' edge detection capability [25], [26]. The assessment parameters for performance evaluation indicators are shown in Equation [insert number]. 5-8.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{5}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{6}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{7}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision+ Recall}} \tag{8}$$

Accuracy (Acc), Precision (Pre), Recall (Rec), and F1 Score are key evaluation metrics in deep learning for image processing and edge detection [15-27]. Precision measures the accuracy of predicted edges, while recall measures the effectiveness of detecting actual edges [28-40].

5. Experimental results

The outcomes of investigations using different deep learning models for edge detection are evaluated using Accuracy (Acc), Precision (Pre), Recall (Rec), and F1 Score. Traditional CNN-based models struggle with fine details, while the advanced architecture, ResNet50-UNet, achieves better accuracy. ResNet50-UNet shows a higher F1 score, indicating a balance between precision and recall. Overall, models that integrate residual connections and multi-scale learning outperform others regarding detection precision and image quality restoration. The experimental results of various deep-learning models, in terms of performance evaluation metrics using the BSDS500 dataset, are shown in Table No. 8.

Table 8: The Experimental Results of Various Deep Learning Models in Terms of Performance Evaluation Metric

Dataset → / P.E.M → / Model (s) ↓	Without Augmentation (Original Dataset)				With Augmentation			
	Acc.	Pre	Rec.	F1 Score	Acc.	Pre	Rec.	F1 Score
VGG19	81%	0.59	0.50	0.54	83%	0.45	0.74	0.56
MobileNet	87%	0.49	0.64	0.59	89%	0.52	0.73	0.61
ResNet18	89%	0.55	0.69	0.61	91%	0.56	0.72	0.63
ResNet50	90%	0.59	0.68	0.63	93%	0.61	0.70	0.65
UNet	91%	0.51	0.81	0.66	92%	0.55	0.89	0.68
ResNet18-UNet	93%	0.60	0.76	0.67	94%	0.63	0.82	0.71
ResNet50-UNet	95%	0.62	0.89	0.71	97%	0.72	0.98	0.83
ResNet101-UNet	92%	0.61	0.79	0.69	93%	0.63	0.82	0.71

Note –P.E.M. - Performance Evaluation Metric, Acc- accuracy, Pre-precision, Rec- recall

An external testing set, the BIPED dataset, has been employed to assess the suggested model's capacity for generalization. Since the training data do not influence BIPED, it provides an objective standard by which to evaluate performance. The dataset offers high-quality edge maps that have been manually annotated, enabling precise comparison of predicted outcomes with ground truth. Testing on BIPED validates the model's accuracy, resilience, and suitability for a variety of natural settings. This external assessment shows that the model is not restricted to the original dataset and can successfully adjust to distributions of images that are not visible. The testing of the external dataset BIPED by the best-performing network, ResNet50-Unet, is shown in Table 9.

Table 9: The Testing of the External Dataset BIPED by the Best Performing Network, Resnet50-Unet

Dataset → / P.E.M → / Model (s) ↓	Testing of BIPED Dataset			
	Acc.	Pre	Rec.	F1 Score
VGG19	79%	0.55	0.45	0.49
MobileNet	80%	0.56	0.45	0.50
ResNet18	82%	0.57	0.46	0.52
ResNet50	84%	0.61	0.48	0.55
UNet	85%	0.62	0.50	0.56
ResNet18-UNet	85%	0.61	0.49	0.58
ResNet50-UNet	87%	0.52	0.70	0.59
ResNet101-UNet	86%	0.60	0.48	0.57

When the ResNet50-UNet model trained on BSDS500 is validated on the BIPED dataset, its performance often degrades initially due to differences in dataset characteristics. BSDS500 provides only 500 relatively low-resolution images with subjective human annotations, while BIPED offers high-resolution natural images with more precise and consistent edge annotations specifically designed for perceptual edge detection. Because of this domain shift, the model may struggle to maintain the same accuracy, leading to lower scores in metrics such as Boundary F1-score, Precision, and Recall. However, validating on BIPED is essential to assess model effectiveness and generalizability. The performance drop indicates that training solely on BSDS500 limits robustness, but fine-tuning or transfer learning with BIPED data significantly improves results. This process not only enhances edge localization but also demonstrates the model's adaptability to different datasets. Thus, BIPED serves as a better benchmark to confirm the effectiveness of Res50-UNet beyond BSDS500. Training and validation loss measure a deep learning model's performance during learning. Training loss quantifies the error on the training dataset, while validation loss assesses generalisation on unseen data. A well-trained model shows decreasing training and validation loss, indicating improved learning. If training loss decreases but validation loss increases, then the model is going towards overfitting. Early stopping, dropout, and data augmentation help to prevent overfitting. A small gap between training and validation loss suggests good generalisation. The training and validation loss value is shown in Figure 4 (a) & (b) for the non-augmented and augmented datasets (BSDS500), respectively, for the proposed model (ResNet50-Unet).

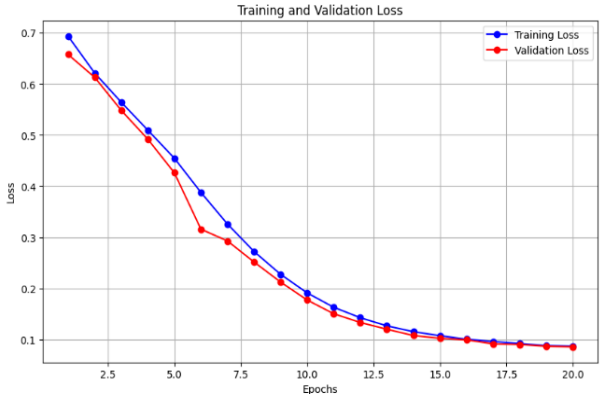

Fig. 4: A) Training and Validation Loss Value Without Augmentation for the Resnet50-Unet Model.

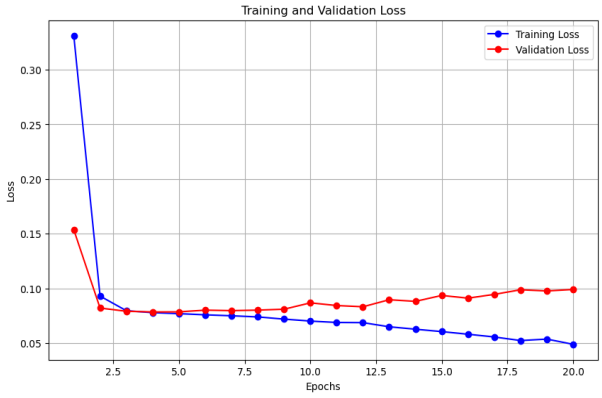

Fig. 4: B): Training and Validation Loss Value with Augmentation for the ResNet50-Unet Model.

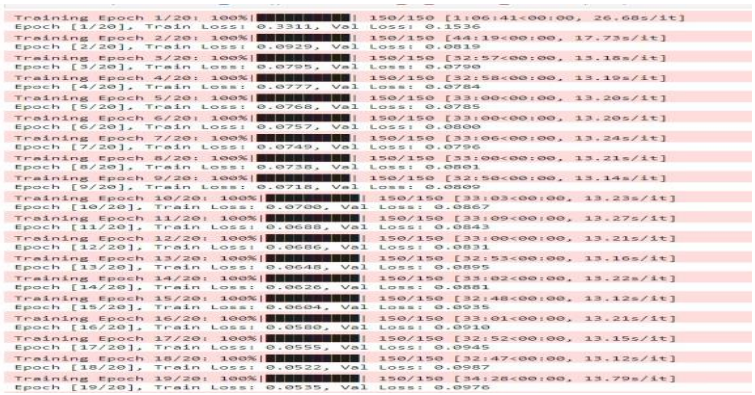

Fig. 5: The Training of Resnet50-Unet with Varying Numbers of Epochs.

Table 8 indicates that the ResNet50-Unet model leads the others in terms of edge detection and structure. Figure 6 shows the result of edge detection:(a) test input images, (b) ground truth edges, (c) predicted edges using the ResNet18-Unet model, and (d) predicted edges using the ResNet50-Unet model.

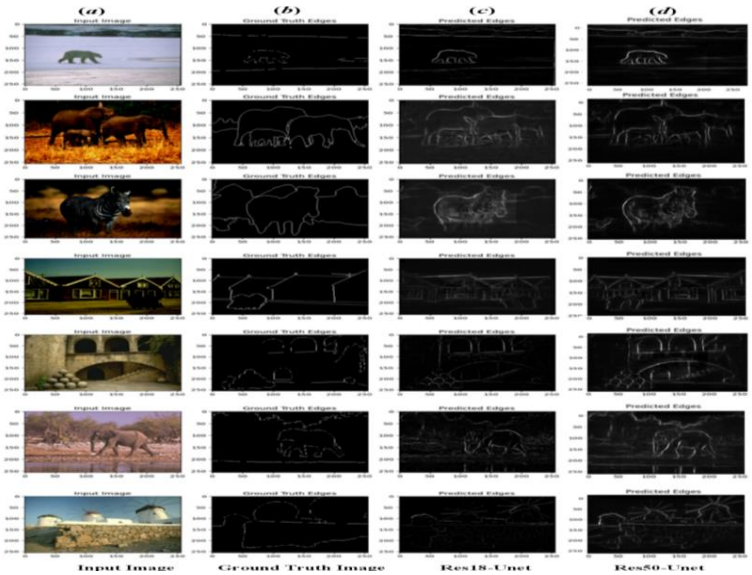

Fig. 6: The Result of Edge Detection (A) Test Input Images, (B) Ground Truth Edges, (C) Predicted Edges Using the Res18-Unet Model, (D) Predicted Edges Using the Resnet50-Unet Model.

5.1. Discussion

Overfitting has been prevented in this study by introducing substantial variety in the training data using a comprehensive data augmentation method. As a result of these augmentations, the model is less dependent on patterns in the original data, which improves generalisation. Depending on the intensity and execution of augmentation, total training duration may rise by 10-30%. Despite this expense, the trade-off is worthwhile because augmentation decreases overfitting and enhances generalisation without the requirement for new labelled data. As a result, the computational cost is reasonable and justifiable given the performance advantages, particularly for small datasets like BSDS500. High performance on the training dataset does not necessarily indicate generalizability; therefore, it is essential to consider the possibility of overfitting, even when the model achieves high accuracy and F1 Scores. To address this, the model was tested on an independent external dataset. The results demonstrated consistent performance, suggesting that the model's predictive potential extended beyond the initial training distribution. This illustrates that the model is not just memorizing features, but also learning discriminative representations. Such cross-dataset validation boosts confidence in the model's resilience, lowers the risk of overfitting, and demonstrates its viability for use in real-world applications. Overfitting is a significant issue when training deep learning models on small datasets like BSDS500, since the model memorizes training data rather than learning generalizable features. This produces high accuracy on training data but poor performance on unseen data. To address this issue, data augmentation techniques have been applied to the BSDS500 dataset. These transformations generate new, diverse versions of the original images, thereby increasing the training dataset size and introducing variability, resulting in high accuracy.

The ResNet50-UNet design is appropriate for edge detection tasks where both global context and precise localization are crucial because it provides a convincing balance between semantic understanding and spatial accuracy. By using deep residual connections to extract high-level semantic features, the encoder, which is based on ResNet-50, enables the model to identify intricate structures and contextual relationships in the image. However, sharp edges may get blurred due to the deep encoding's tendency to lower spatial resolution. To counteract this, ResNet50-UNet recovers spatial features and improves boundary localisation by implementing skip connections from the encoder's early layers to comparable decoder layers. ResNet50-UNet offers better contextual interpretation than the conventional U-Net, which maintains spatial precision well but does not have deep semantic abstraction because of its shallow encoder. An ImageNet-pretrained ResNet-50 encoder and a UNet decoder are combined in ResNet50-UNet. Strong semantic understanding is an upside: even in complex scenes, residual blocks and deeper receptive fields can capture context at the class level. There is a trade-off in spatial precision due to ResNet-50's multiple stride downsamples, which distort edges and small structures. Skipping connections improves things, but you lose

some boundary detail. ResNet50-UNet improves semantics, but improving spatial fidelity usually necessitates changes to the architecture or training that take edge cases.

Ethical considerations are critical when using deep learning models such as ResNet50-UNet, especially in sensitive areas such as surveillance. In this application, poor edge predictions may lead to key misinterpretations, such as defective border identification in real-time navigation systems, and potentially result in severe outcomes. To reduce risks, model projections must be validated by experts and linked with decision-support systems. The experiments on the external BIPED dataset demonstrate that the suggested approach is highly generalizable. When evaluated on this independent dataset, the model performed consistently with good precision, recall, and F-measure, proving its efficacy in edge detection tasks. The precision with which the ground truth annotations and predicted edges align demonstrates the method's robustness. These results indicate the model's capacity to react to unseen natural images while avoiding overfitting to training data. Testing on BIPED clearly demonstrates the model's durability and suitability for practical edge detection applications. Figure 7 presents the results of edge detection with an external dataset.

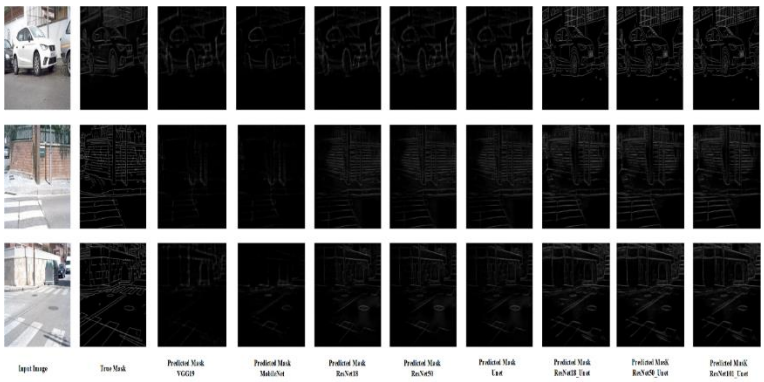

Fig. 7: The Result of Edge Detection by Using the Testing of an External Dataset.

6. Conclusion

Experimental results on the BSDS500 dataset show that ResNet50-UNet achieves the highest F1 Score, Precision, and Recall, demonstrating its ability to balance edge accuracy and image quality. Using batch normalization and residual connections ensures stable training, preventing gradient vanishing and improving convergence. However, challenges such as overfitting can arise, requiring data augmentation and regularization techniques. The ResNet50-UNet architecture effectively enhances edge detection by combining ResNet-50's deep feature extraction with UNet's precise localisation capabilities. The model captures multi-scale contextual information by leveraging a pre-trained ResNet-50 as the encoder, while the decoder, equipped with transposed convolutions and skip connections, restores spatial accuracy. This hybrid approach produces superior edge maps with reduced noise and sharper boundaries. Compared to other DL-based edge detection models, such as VGG19, MobileNet, ResNet18, ResNet50, UNet, ResNet18-UNet, and ResNet101-UNet, the proposed model provides better structural preservation and fewer false detections. Its ability to generalise well across different types of datasets makes it suitable for various applications, such as medical imaging, object segmentation, and remote sensing. In conclusion, ResNet50-UNet significantly improves edge detection performance, striking a balance between feature richness and spatial precision, making it an efficient deep-learning solution for complex image analysis.

The future of edge detection lies in the integration of AI, real-time processing, 3D imaging, and edge computing. As technology advances, edge detection will play a vital role in medical diagnostics, autonomous systems, industrial automation, and environmental monitoring. Further research has focused on adaptive, noise-resistant, and hardware-optimized edge detection techniques.

References

[1] N. S. Dagar and P. K. Dahiya, "Edge detection technique using binary particle swarm optimization," *Procedia Computer Science*, vol. 167, pp. 1421–1436, 2020. https://doi.org/10.1016/j.procs.2020.03.353.

[2] N. S. Dagar and P. K. Dahiya, "A comparative investigation into edge detection techniques based on computational intelligence," *Int. J. Image, Graphics and Signal Processing*, no. 7, pp. 58–68, 2019. https://doi.org/10.5815/ijigsp.2019.07.05.

[3] R. Maskeliunas, R. Damaševičius, D. Vitkute-Adzgauskiene, and S. Misra, "Pareto optimized large mask approach for efficient and background humanoid shape removal," *IEEE Access*, vol. 11, pp. 33900–33914, 2023. https://doi.org/10.1109/ACCESS.2023.3253206.

[4] Y. Zhao, W. Gui, and Z. Chen, "Edge detection based on multi-structure elements morphology," in *Proc. 6th World Congr. Intell. Control Autom.*, vol. 2, pp. 9795–9798, Jun. 2006. https://doi.org/10.1109/WCICA.2006.1713908.

[5] G. M. H. Amer and A. M. Abushaala, "Edge detection methods," in *Proc. 2nd World Symp. Web Appl. Netw. (WSWAN)*, pp. 1–7, Mar. 2015. https://doi.org/10.1109/WSWAN.2015.7210349.

[6] B. Chander, K. Guravaiah, B. Anoop, and G. Kumaravelan, Eds., *Handbook of AI-Based Models in Healthcare and Medicine: Approaches, Theories, and Applications*. Boca Raton, FL, USA: CRC Press, 2024. https://doi.org/10.1201/9781003363361.

[7] V. M. Dharampal, "Methods of image edge detection: A review," *J. Electr. Electron. Syst.*, vol. 4, no. 2, pp. 2332–0796, 2015.

[8] R. Sun, T. Lei, Q. Chen, Z. Wang, X. Du, W. Zhao, and A. K. Nandi, "Survey of image edge detection," *Frontiers in Signal Processing*, vol. 2, p. 826967, 2022. https://doi.org/10.3389/frsip.2022.826967.

[9] N. M. Sáez-Gallego, Y. Segovia, and J. Abellán, "Codesigning a motor story for promoting inclusion and critical thinking in pre-service teachers: A participatory action research approach in physical education," *J. Teaching Phys. Educ.*, vol. 1, no. aop, pp. 1–12, 2024. https://doi.org/10.1123/jtpe.2023-0337.

[10] Y. Lijun, L. Mengbo, W. Tongxin, B. Youfeng, L. Junhui, and J. Yi, "Geo-information mapping improves Canny edge detection method," *IET Image Processing*, vol. 17, no. 6, pp. 1893–1904, 2023. https://doi.org/10.1049/ipr2.12764.

[11] J. Huang, B. Bai, and F. Yang, "An effective salient edge detection method based on point flow with phase congruency," *Signal, Image and Video Processing*, vol. 16, no. 4, pp. 1019–1026, 2022. https://doi.org/10.1007/s11760-021-02048-4.

[12] Z. Wu, X. Lu, and Y. Deng, "Image edge detection based on local dimension: A complex networks approach," *Physica A: Statistical Mechanics and its Applications*, vol. 440, pp. 9–18, 2015. https://doi.org/10.1016/j.physa.2015.07.020.

[13] Z. Lv, Z. Lu, K. Xia, L. Zhang, H. Zuo, Y. Xu, and K. Liu, "Real-time detection system for polishing metal surface defects based on convolutional feature concentration and activation network," *Expert Systems with Applications*, vol. 257, 125041, 2024. https://doi.org/10.1016/j.eswa.2024.125041.

[14] N. S. Patil and E. Kannan, "An efficient corn leaf disease prediction using Adaptive Color Edge Segmentation with Resnext101 model," *Journal of the Saudi Society of Agricultural Sciences*, 2024.

[15] C. Wen, P. Liu, W. Ma, Z. Jian, C. Lv, J. Hong, and X. Shi, "Edge detection with feature re-extraction deep convolutional neural network," *Journal of Visual Communication and Image Representation*, vol. 57, pp. 84–90, 2018. https://doi.org/10.1016/j.jvcir.2018.10.017.

[16] N. Xue, T. Wu, S. Bai, F. Wang, G. S. Xia, L. Zhang, and P. H. Torr, "Holistically-attracted wireframe parsing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 2788–2797, 2020. https://doi.org/10.1109/CVPR42600.2020.00286.

[17] X. S. Poma, E. Riba, and A. Sappa, "Dense extreme inception network: Towards a robust CNN model for edge detection," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, pp. 1923–1932, 2020.

[18] R. Deng and S. Liu, "Deep structural contour detection," in *Proc. 28th ACM Int. Conf. Multimedia*, pp. 304–312, Oct. 2020. https://doi.org/10.1145/3394171.3413750.

[19] M. Zhen, J. Wang, L. Zhou, S. Li, T. Shen, J. Shang, … and L. Quan, "Joint semantic segmentation and boundary detection using iterative pyramid contexts," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 13666–13675, 2020. https://doi.org/10.1109/CVPR42600.2020.01368.

[20] F. Orujov, R. Maskeliūnas, R. Damaševičius, and W. Wei, "Fuzzy based image edge detection algorithm for blood vessel detection in retinal images," *Applied Soft Computing*, vol. 94, 106452, 2020. https://doi.org/10.1016/j.asoc.2020.106452.

[21] Y. Liu, Z. Xie, and H. Liu, "An adaptive and robust edge detection method based on edge proportion statistics," *IEEE Transactions on Image Processing*, vol. 29, pp. 5206–5215, 2020. https://doi.org/10.1109/TIP.2020.2980170.

[22] X. Yu, Z. Wang, Y. Wang, and C. Zhang, "Edge detection of agricultural products based on morphologically improved canny algorithm," *Mathematical Problems in Engineering*, vol. 2021, no. 1, p. 6664970, 2021. https://doi.org/10.1155/2021/6664970.

[23] M. Versaci and F. C. Morabito, "Image edge detection: A new approach based on fuzzy entropy and fuzzy divergence," *International Journal of Fuzzy Systems*, vol. 23, no. 4, pp. 918–936, 2021. https://doi.org/10.1007/s40815-020-01030-5.

[24] H. Zhao, B. Wu, Y. Guo, G. Chen, and D. Ye, "SSWS: An edge detection algorithm with strong semantics and high detectability for spacecraft," *Optik*, vol. 247, p. 168037, 2021. https://doi.org/10.1016/j.ijleo.2021.168037.

[25] Z. Su, W. Liu, Z. Yu, D. Hu, Q. Liao, Q. Tian, et al., "Pixel difference networks for efficient edge detection," in *Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV)*, pp. 5117–5127, 2021. https://doi.org/10.1109/ICCV48922.2021.00507.

[26] O. Elharrouss, Y. Akbari, S. Al-Maadeed, and A. Bouridane, "Edge detection with multi-scale representation and refined network," in *IET Conf. Proc. CP816*, vol. 2022, no. 14, pp. 247–251, Oct. 2022. https://doi.org/10.1049/icp.2022.2465.

[27] F. Wang and M. Zhang, "Deep learning-based edge detection algorithm for noisy images," in *Proc. 2023 6th Int. Conf. Artificial Intelligence and Pattern Recognition (AIPR)*, pp. 465–472, Sept. 2023. https://doi.org/10.1145/3641584.3641653.

[28] O. Elharrouss, Y. Hmamouche, A. K. Idrissi, B. El Khamlichi, and A. El Fallah-Seghrouchni, "Refined edge detection with cascaded and high-resolution convolutional network," *Pattern Recognition*, vol. 138, p. 109361, 2023. https://doi.org/10.1016/j.patcog.2023.109361.

[29] D. Jing, B. Li, S. Wang, J. Lin, and Y. Jiao, "Edge detection in dark industrial environments," in *Proc. 2023 9th Int. Conf. Computer and Communications (ICCC)*, pp. 1689–1693, Dec. 2023. https://doi.org/10.1109/ICCC59590.2023.10507668.

[30] A. Al-Amaren, M. O. Ahmad, and M. N. S. Swamy, "A low-complexity residual deep neural network for image edge detection," *Applied Intelligence*, vol. 53, no. 9, pp. 11282–11299, 2023. https://doi.org/10.1007/s10489-022-04062-6.

[31] F. Li and C. Lin, "UHNet: An ultra-lightweight and high-speed edge detection network," *arXiv preprint arXiv:2408.04258*, 2024.

[32] J. Zhang, W. Wang, and J. Wang, "Edge detection in colored images using parallel CNNs and social spider optimization," *Electronics*, vol. 13, no. 17, p. 3540, 2024. https://doi.org/10.3390/electronics13173540.

[33] F. Abedi, "Dense residual network for image edge detection," *Multimedia Tools and Applications*, vol. 83, no. 42, pp. 90227–90242, 2024. https://doi.org/10.1007/s11042-024-19264-y.

[34] Y. An, J. Jing, X. Li, J. Zhang, and J. Bao, "An exclusive U-net for fine and crisp edge detection," *Multimedia Tools and Applications*, vol. 83, no. 18, pp. 54657–54672, 2024. https://doi.org/10.1007/s11042-023-17706-7.

[35] K. Muntarina, R. Mostafiz, S. B. Shorif, and M. S. Uddin, "Deep learning-based edge detection for random natural images," *Neuroscience Informatics*, vol. 5, no. 1, p. 100183, 2025. https://doi.org/10.1016/j.neuri.2024.100183.

[36] M. Wang, "AttnEdge: An enhanced edge detection method based on self-attention mechanism," in *Proc. 4th Int. Conf. Computer Vision, Application, and Algorithm (CVAA)*, vol. 13486, pp. 438–446, Jan. 2025. https://doi.org/10.1117/12.3055875.

[37] W. Li, W. Zhang, Y. Liu, C. Liu, and R. Jing, "Pixel-patch combination loss for refined edge detection," *International Journal of Machine Learning and Cybernetics*, pp. 1–14, 2024.

[38] X. Chen, W. Yang, W. Wu, X. Tao, and X. Mao, "Sufficient learning: Mining denser high-quality pixel-level labels for edge detection," *Neural Computing and Applications*, pp. 1–16, 2025. https://doi.org/10.1007/s00521-024-10661-w.

[39] X. Yang, L. Cheng, G. Yuan, and H. Wu, "Textureness-aware neural network for edge detection," in *Chinese Conf. Pattern Recognition and Computer Vision (PRCV)*, Singapore: Springer, pp. 254–268, Oct. 2024. https://doi.org/10.1007/978-981-97-8505-6_18.

[40] H. Shu, "More precise edge detections," *arXiv preprint arXiv:2407.19992*, 2024.

[41] N. Yadav, R. Dass, and J. Virmani, "Assessment of encoder-decoder based segmentation models for thyroid ultrasound images," *Medical & Biological Engineering & Computing*, vol. 61, pp. 2159–2195, 2023, https://doi.org/10.1007/s11517-023-02849-4.

[42] N. Yadav, R. Dass, and J. Virmani, "Objective assessment of segmentation models for thyroid ultrasound images," *Journal of Ultrasound*, vol. 26, pp. 673–685, Oct. 2022, https://doi.org/10.1007/s40477-022-00726-8.

[43] N. Yadav, R. Dass, and J. Virmani, "A systematic review of machine learning based thyroid tumor characterisation using ultrasonographic images," *Journal of Ultrasound*, vol. 27, pp. 209–224, 2024, https://doi.org/10.1007/s40477-023-00850-z.

[44] R. Dass, Priyanka, and S. Devi, "Image segmentation techniques," *International Journal of Electrical Communication Technology*, vol. 3, no. 1, pp. 1–5, 2012.

[45] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. Wells, and A. Frangi, Eds. Cham: Springer, 2015, Lecture Notes in Computer Science, vol. 9351, pp. 234–241, https://doi.org/10.1007/978-3-319-24574-4_28.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778, https://doi.org/10.1109/CVPR.2016.90.

[47] N. Yadav, R. Dass, and J. Virmani, "Deep learning-based CAD system design for thyroid tumor characterization using ultrasound images," Multimedia Tools and Applications*, vol. 83, no. 4, Oct. 2023. https://doi.org/10.1007/s11042-023-17137-4.

[48] N. Yadav, R. Dass, and J. Virmani, "Despeckling filters applied to thyroid ultrasound images: a comparative analysis," *Multimedia Tools and Applications*, vol. 81, no. 6, pp. 8905–8937, 2022, https://doi.org/10.1007/s11042-022-11965-6.