



Deep Neural Network-Based Image Segmentation: Architectural Comparison, Optimization, and Performance Evaluation

S. Velmurugan^{1,*}, P. Yuvaraja², Subramanian K³

¹ Department of Computer Science with Data Analytics, Kongunadu Arts and Science College (Autonomous), Coimbatore, Tamil Nadu, India

² Department of Computer and Information Science, Annamalai University, Tamil Nadu, India

³ CHRIST (Deemed to be University), Bengaluru, India

* Corresponding author: ssvelmurugan@yahoo.com

ABSTRACT

Image segmentation is a key step in many computer vision systems, particularly in medical imaging, remote sensing, robotics, and precision agriculture. Over the years, a wide range of traditional approaches have been developed for image segmentation, such as binary thresholding Otsu's method, the watershed method, and K-means clustering, which leverage domain-specific knowledge to efficiently solve segmentation problems in particular application contexts. Unfortunately, however, these methods show significant limitations when faced with the increasing complexity of visual scenes and data heterogeneity. In reality, the fundamental principle of image segmentation is to classify each pixel based on similar attributes. The goal of this pixel-by-pixel classification is to achieve a process that is global, relatively insensitive to noise, locally accurate, and incorporates multi-scale filtering mechanisms. However, traditional methods cannot simultaneously meet these requirements. Faced with the limitations of traditional methods, the emergence of deep learning approaches, particularly those based on neural networks and especially convolutional neural networks, has established itself as an effective solution for image segmentation. The objective of this study is to comparatively evaluate and optimize different deep learning architectures to improve the accuracy and robustness of image segmentation in complex visual environments. The study evaluates the performance of models such as the linear network, Simple U-Net, Extended U-Net, U-Net, DeepLabV3+, and SegFormer, incorporating optimization strategies including data augmentation, fine-tuning, the addition of attentional mechanisms, and the proposal of a specific metric to track performance changes during neural network model training. The experimental results demonstrate improved segmentation performance across the evaluated architectures, as measured by Dice, Intersection over Union (IoU), Precision, and Recall. Among the evaluated models, DeepLabV3+ achieved the highest Dice score (0.91), IoU (0.84), and Precision (0.92), while SegFormer achieved the highest Recall (0.91). These findings demonstrate the effectiveness of deep neural network architectures for accurate pixel-level segmentation and provide a comparative basis for selecting suitable architectures for complex visual environments.

ARTICLE INFO

Article history:

Received: 23-09-2026

Received in revised form: 24-09-2026

Accepted: 08-10-2026

Available online: 10-10-2026

Keywords:

Autoencoder
Computer Vision
Deep Learning
Image Segmentation
Neural Networks
U-Net Architecture

1. INTRODUCTION

Over the past decade, rapid advances in deep learning have profoundly transformed the field of computer vision. Artificial neural networks, and in particular convolutional neural networks (CNNs), have

established themselves as indispensable tools for the automatic analysis of images in varied and complex contexts [1]. Among the most studied problems are image segmentation, as well as object detection and localization, which are fundamental steps in many intelligent systems [2, 3].

Image segmentation involves partitioning an image into homogeneous regions corresponding to objects or structures of interest. It can be formalized as a pixel-level classification task, in which each pixel is associated with a specific class [1]. The main objective of this operation is to transform the raw image representation into a more structured and semantically meaningful form, thus facilitating subsequent processing such as analysis, recognition, or automatic decision-making [4]. In clinical imaging in particular, neural segmentation now supports diagnostic workflows [5].

Historically, numerous traditional approaches have been proposed to solve image segmentation problems, including thresholding methods, region-based segmentation algorithms, clustering techniques (such as K-means), and the watershed method [6]. While these methods are effective in controlled environments, they have significant limitations when faced with variations in illumination, noise, texture complexity, and the heterogeneity of real-world scenes [7].

The emergence of deep learning has overcome these limitations by introducing models capable of automatically learning hierarchical representations from data. CNN-based segmentation networks, such as Fully Convolutional Networks (FCN), U-Net, and DeepLab, have demonstrated remarkable performance in various application areas, ranging from medical imaging and remote sensing to robotics and industry [2, 8]. More recently, the introduction of attention mechanisms and hybrid transformer-based architectures has opened up new perspectives in semantic and instance-level segmentation [9, 10].

Image segmentation applications are numerous and strategic today. In industry, it is used to detect mechanical defects, such as cracks or structural damage [11]. In aeronautics and mechanical engineering, it contributes to quality control and predictive maintenance. In astronomy, segmentation enables the automatic detection of galaxies, stars, and as-yet-unknown celestial objects [12]. Furthermore, object detection and localization techniques are widely deployed in automated surveillance systems, urban traffic management, object and people counting, and intelligent security systems [13].

In this context, this article offers an in-depth study of image segmentation methods based on deep learning. It begins with an analysis of traditional approaches to highlight their limitations compared to modern neural methods. It then examines the main neural network architectures dedicated to segmentation [14]. An optimized neural network architecture is then proposed, implemented, and experimentally evaluated on a validation dataset. Finally, a series of experiments is conducted to demonstrate the effectiveness of the proposed approach for the precise, pixel-level localization of an aircraft in complex images [15].

2. MATERIALS AND METHODS- ANNOTATION:

$I(i, j)$ – Pixel intensity

$I'(i, j)$ – the intensity of the segmented image;

T – Threshold value;

k – The maximum value that pixel intensity can take

σ^2 – Image dispersion;

σ_w^2 – Variance within the class;

σ_y^2 – Variance between classes;

ω_1 – Probability of belonging to the 1st class;

ω_2 – Probability of belonging to the 2nd class;

σ_1^2 – First-class variance;

σ_2^2 – Second-class dispersion;

$p(k)$ – Probability of each pixel intensity level;

$hist$ – Histogram value;

N_i – Total number of pixels in the image;

$\sigma_1^2(T)$ – First-class dispersions;

$\sigma_2^2(T)$ – Second-class dispersions;

N_1 – Vector from 0 to T-1;

N_2 – Vector from T to 255;

Moy_1 – Mean value of class 1;

Moy_2 – Mean value of class 2;

$S_1, S_2, S_3, \dots, S_k$ – Clusters or classes;

$C_1, C_2, C_3, \dots, C_k$ – Centroids;

L – Sum of intracluster distances;

$J(A, B)$ – Metrics;

$A \cap B$ – Intersection zone;

$A \cup B$ – Unification place;

DSC – Overlapping segmented volumes;

ASD – Mean symmetric surface distance;

Def – Custom metric;

y_{true} – Actual value;

y_{pred} – Predicted value;

F – Final merged feature map, used as input for the segmentation prediction layer ;

n – Total number of feature levels used for multi-scale fusion.

$MLP(\cdot)$ – Multi-layer perceptron applied independently to each map F_i ;

Q – Query matrix, representing the features of the elements for which the model seeks to aggregate information ;

Key – Key matrix, representing the features used as a reference to measure similarity with the queries;

V – Value matrix, containing the information to be aggregated based on attention weights calculated from Q and

K ;

QK^T – Matrix product measuring the similarity between each query and each key;

d – Dimension of query and key vectors.

2.1 Threshold Method Based on Pixel Intensity Histograms

Thresholding segmentation based on intensity histograms is one of the oldest and simplest approaches in image processing. It relies on the assumption that objects of interest and the image background can be distinguished based on their pixel intensity levels.

An intensity histogram represents the statistical distribution of gray levels in an image, where each bar corresponds to the frequency of occurrence of a given intensity level. When the histogram has a bimodal or multimodal distribution, it becomes possible to determine one or more threshold values to effectively separate the different classes of pixels.

In the case of global thresholding, a threshold value T is chosen to partition the image (i, j) into two distinct regions. Pixels with an intensity below the threshold are assigned to a first class, while those with an intensity above or equal to the threshold are assigned to a second class. This operation can be formalized as follows:

$$I'(i, j) = \begin{cases} C_1, & \text{si } I(i, j) < T \\ C_2, & \text{si } I(i, j) \geq T \end{cases} \quad (1)$$

where C_1 and C_2 represent the classes associated with the background and the object, respectively.

Among the best-known automatic threshold selection techniques is Otsu's method, which consists of determining the threshold value that maximizes the inter-class variance or, equivalently, minimizes the

intra-class variance [6]. This approach relies on the statistical analysis of the histogram and allows for fully automatic segmentation without human intervention [14].

Despite its simplicity and low computational cost, histogram-based thresholding segmentation has significant limitations. It is particularly sensitive to noise, variations in illumination, and images whose intensity distributions do not show clear separation between classes [16]. Furthermore, this method does not take into account spatial and contextual information, which limits its effectiveness in complex or heterogeneous scenes [17]. These limitations have motivated the development of more advanced methods, including those based on machine learning and deep learning, capable of exploiting both local and global image features for more robust and accurate segmentation.

2.2 Watershed Method

The Watershed method is a morphological segmentation technique inspired by a geographical analogy. It treats the image as a topographic surface in which the intensity of each pixel represents an altitude [18]. Areas of low intensity correspond to watersheds, while areas of high intensity represent the ridgelines separating these watersheds.

Consider a grayscale image $I: \Omega \subset \mathbb{Z}^2 \rightarrow \mathbb{R}$, where Ω denotes the spatial domain of the image. The *Watershed* transformation is generally based on analyzing the image gradient, defined as:

$$\nabla I(i, j) = \sqrt{\left(\frac{\partial I}{\partial i}\right)^2 + \left(\frac{\partial I}{\partial j}\right)^2} \quad (2)$$

The gradient highlights the contours of objects, which correspond to areas of rapid intensity variation and constitute natural barriers between regions.

Watershed segmentation involves identifying local minima of the gradient function, denoted $\{mi\}$, which serve as starting points for watershed growth. Each watershed corresponds to a homogeneous region associated with a local minimum. Mathematically, a watershed Bi associated with a minimum mi can be defined as the set of pixels $p \in \Omega$ such that the steepest descent path from p leads to m :

$$Bi = \{p \in \Omega \mid \forall q \in (p), I(q) \geq I(p) \text{ et } k \rightarrow \infty \lim Pk(p) = mi\} \quad (3)$$

where $\mathcal{P}(p)$ denotes a monotonic descent path from pixel p .

Watershed lines are then defined as the boundaries separating two adjacent watersheds:

$$W = \Omega \setminus i \cup Bi \quad (4)$$

In practice, the algorithm is often implemented using a gradual flooding process. Starting from local minima, the water level is progressively raised, and the merging of distinct basins is prevented by constructing dikes, which correspond to the drainage divides. However, one of the main drawbacks of this method lies in the phenomenon of over-segmentation, due to the presence of numerous local minima caused by noise. To overcome this problem, variants such as *the marker-controlled watershed* have been proposed [19]. This approach introduces a set of internal markers Mi and external markers Me , defining the regions of interest and the bottom, respectively:

$$Im = mi(rec(I, Mi), Me) \quad (5)$$

Where $rec(\cdot)$ denotes a morphological reconstruction operation.

Despite its limitations, the watershed method remains an effective technique for segmenting images with well-defined contours. However, its sensitivity to noise and its inability to integrate global contextual information have led to the use of more advanced methods, particularly those based on deep learning [20].

2.3 Segmentation by Clustering (K-Means)

Segmentation by clustering relies on grouping the pixels of an image into homogeneous sets called clusters, based on the similarity of their characteristics. Among the most widely used clustering methods in image segmentation is the *K-means* algorithm, due to its conceptual simplicity and computational efficiency. The "K" in the name is an algorithm parameter that defines the number of clusters [16]. The algorithm forms clusters, each represented by the cluster's center, called a centroid, which is selected based on the characteristics of the input data. In the case of image segmentation, these characteristics generally correspond to the pixel intensity and its three spatial coordinates: horizontal, vertical, and color channel. Therefore, the input feature vector can be represented as $v \in \mathbb{R}^{4 \times 1}$, where $v = [I(x, y, z), x, y, z]^T$

This algorithm uses distance measures such as L^2 or L^1 :

$$D(v^{(i)}, v^{(j)}) = \|v^{(i)} - v^{(j)}\| = \sqrt{(v^{(i)} - v^{(j)})^T (v^{(i)} - v^{(j)})}, \quad (6)$$

$$D(v^{(i)}, v^{(j)} / L^1) = \|v^{(i)} - v^{(j)}\|_1 \quad (7)$$

Let's see how the algorithm works:

- Step 1: Start with K randomly selected centroids from groups $C_1, C_2, C_3, \dots, C_k$ $S_1, S_2, S_3, \dots, S_k$ corresponding to K clusters
- Step 2: For each pixel, we calculate the distances from its feature vector $v^{(i)}$ to the centroids of the groups and assign the objects to the nearest central group. Repeat this process for all pixel feature vectors until each pixel is assigned to one of the K clusters [21].
- Step 3: Calculate the centroids of each group. After determining the centroids of the new clusters for all pixels, we recalculate them by averaging the pixel feature vectors in each cluster:

$$C_j = \frac{1}{m} \sum_{v^{(i)} \in S_j} v^{(i)} \quad (8)$$

Repeat steps 2 and 3 for several iterations until the centroids stop changing. Through this iterative process, we reduce the sum of the intra-cluster distances, which is as follows:

$$L = \sum_{j=1}^k \sum_{v^{(i)} \in S_j} \|v^{(i)} - C_j\|_2 \quad (9)$$

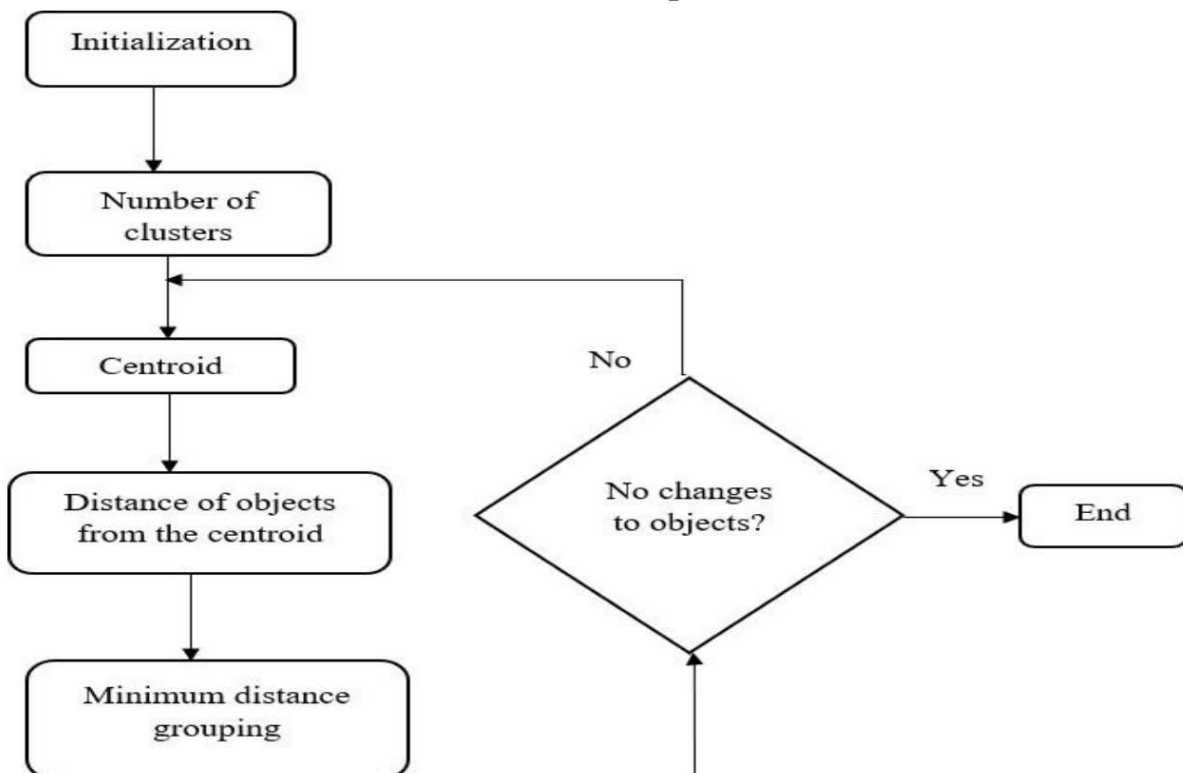


Figure 1. Functional diagram of the K-means algorithm.

Figure 1 despite efficiency, K-means segmentation has several limitations. It requires the prior definition of the number of clusters K , is sensitive to the initialization of centroids, and does not explicitly take into account the spatial relationships between pixels. Furthermore, it assumes that the clusters are convex and have comparable variance, which is not always the case in real-world images. These limitations have led to the development of more advanced methods that incorporate contextual and spatial

information, as well as deep learning-based approaches capable of modeling complex visual structures more robustly.

2.4 The Otsu Method

The Otsu method is a widely used automatic thresholding technique for segmenting grayscale images. Proposed by Nobuyuki Otsu in 1979, it relies on a statistical analysis of the image's intensity histogram and aims to determine an optimal threshold value for separating the image into two distinct classes, typically background and object. It is used to automatically determine the threshold from the shape of the image's histogram. Therefore, this method requires the initial calculation of the image's histogram. The algorithm assumes that the image to be converted into binary form contains only two classes (objects and background) T , separating the two classes so that the variance within each class is minimal and the variance between classes is maximized.

Variance within a class:

$$\sigma_w^2 = \omega_1(T) \times \sigma_1^2(T) + \omega_2(T) \times \sigma_2^2(T) \quad (10)$$

Variance between classes:

$$\sigma_y^2 = \sigma^2 - \sigma_w^2 \quad (11)$$

Calculating the probability of classes 1 and 2:

To calculate the probability of falling into class 1 or 2 on the threshold T , simply sum the probabilities of the intensity level of each pixel.

$$\omega_1 = \sum_{k=1}^T p(k) \quad (12)$$

$$\omega_2 = \sum_{k=T+1}^{256} p(k) \quad (13)$$

Calculate the probability intensity of each pixel Histogramic calculation:

To calculate the histogram, you need to view the entire image and count the number of pixels corresponding to the intensity level of each pixel (grayscale)

$$hist = \sum_{i=1}^N \sum_{j=1}^M (I(i, j) == k) \quad (14)$$

Calculate the probability for each pixel intensity level

The probability of each pixel intensity level is calculated by dividing the number of pixels present for each pixel intensity level by the total number of pixels in the image.

$hist(k)$

Calculation of the variance of each class:

$$\sigma_1^2(T) = \frac{\sum_{i=1}^T (N_1(i) - M_{0,y_1}(T))^2 \times P(i)}{\omega_1} \quad (16)$$

$$\sigma_2^2(T) = \frac{\sum_{i=T+1}^{256} (N_2(i) - M_{0,y_2}(T))^2 \times P(i)}{\omega_2} \quad (17)$$

Calculating the average for each class:

The average for each class is calculated by adding the vector N , which is multiplied by the probability of each pixel intensity level. Then, the sum is divided by the class probability.

$$Moy_1(T) = \frac{\sum_{i=1}^T N_1(i) \times P(i)}{\omega_1(T)} \quad (18)$$

$$Moy_2(T) = \frac{\sum_{i=T+1}^{256} N_2(i) \times P(i)}{\omega_2(T)} \quad (19)$$

Otsu's method offers several advantages:

- Fully automatic thresholding;
- Simplicity of implementation;

- Low computational cost.

However, it also has significant limitations:

- Assumption of bimodality of the histogram;
- Sensitivity to noise and variations in illumination;
- Inefficiency on images with low contrast or unimodal distribution.

These limitations explain why Otsu's method is now often used as a reference method or as a preprocessing step, before the application of more advanced techniques such as deep learning segmentation.

2.5 Image Segmentation by Deep Neural Networks

The inherent limitations of classical segmentation methods have led to the rise of deep learning-based approaches, which have become the state of the art in computer vision. Deep neural networks, and in particular *convolutional neural networks* (CNN), have demonstrated a remarkable ability to automatically learn hierarchical representations from raw data without requiring manual feature extraction [22].

Unlike traditional approaches based on low-level features, deep neural networks leverage successive layers of convolution, nonlinearity, and subsampling to capture both local and global information [23]. The initial layers typically learn simple patterns, such as contours or textures, while deeper layers model higher-level semantic structures, enabling more precise and contextual segmentation.

Image segmentation using deep neural networks is generally formulated as a *dense prediction* problem, in which a label is predicted for each pixel of the image. Consider an input image

$I \in \mathbb{R}^{H \times W \times C}$, where H , W , and C denote the height, width, and number of channels, respectively. The network's objective is to learn a transformation function f_θ , parameterized by θ , such that:

$$Y = f(I), Y \in \mathbb{R}^{H \times W \times K} \quad (20)$$

where K represents the number of classes and Y the probability map of the classes associated with each pixel.

Model training relies on minimizing a loss function that measures the difference between the predicted segmentation $\hat{Y}$ and the ground truth Y . Among the most commonly used loss functions are categorical cross-entropy, Dice Loss, and their weighted variants, which are suitable for class imbalance problems:

$$L_{CE} = - \sum_{i=1}^N \sum_{k=1}^K y_{i,k} \log(\hat{y}_{i,k}) \quad (21)$$

$$\mathcal{L}_{Dice} = 1 - \frac{2 \sum_{i=1}^N \hat{y}_i y_i}{\sum_{i=1}^N y_i^2 + \sum_{i=1}^N \hat{y}_i^2} \quad (22)$$

where N denotes the total number of pixels.

Several deep neural network architectures have been specifically designed for segmentation tasks. *Fully Convolutional Networks* (FCN) were among the first to replace fully connected layers with convolutions, enabling pixel-by-pixel prediction. Subsequently, more advanced architectures, such as U-Net, DeepLab, and their variants, introduced multiscale fusion mechanisms, *skip connections*, and dilated convolutions to improve spatial accuracy.

More recently, the integration of attention mechanisms and transformer-based architectures has improved global context modeling and model generalization capabilities. These hybrid approaches combine the advantages of CNN for local feature extraction with those of transformers for capturing long-range dependencies.

2.6 Neural Network Architectures for Solving Segmentation Problems

In recent years, image segmentation using convolutional neural networks has become very popular [16]. Recent advances in image segmentation have led to the development of neural network architectures capable of simultaneously capturing fine local information and a broad global context [24]. Among these architectures, U-NET, DeepLabV3+, and SegFormer represent two major and complementary approaches: the first is based on a convolutional encoder-decoder architecture initially proposed for biomedical image segmentation on convolutional networks; the second is based on an architecture enriched by dilated convolutions; while the last relies on efficient transformers without heavy convolutions [25].

U-Net

The network structure is similar to that of an autoencoder, in which the network compresses the data into a hidden space, thus revealing the main features, and then reconstructs the image from this hidden space.

The U-net architecture is a sequence of blocks, first decreasing the dimensionality of the image (the blocks include pooling layers), then increasing it, having previously combined it with the outputs of the initial blocks having the corresponding dimension (Figure 2) [18]. It is distinguished by its symmetrical “U” - shaped structure and the use of skip connections. The encoder feature maps are concatenated with those of the corresponding decoder: Autoencoder-based deep models are also applied well beyond imaging, for example in intrusion detection [26].

$$F_{dec}^{(l)} = \text{Concat}(F_{enc}^{(l)}, \text{Up}(F_{dec}^{(l+1)})) \quad (23)$$

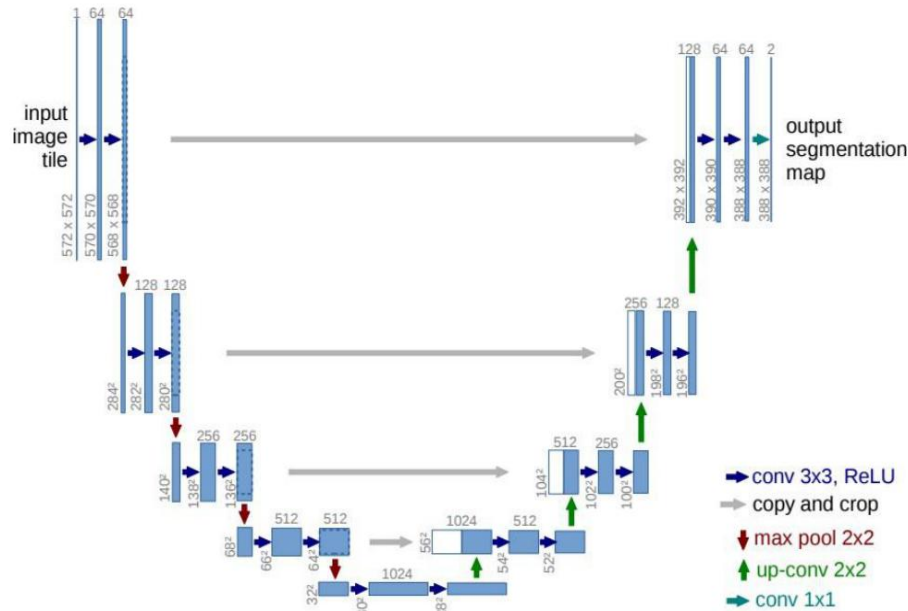


Figure 2. U-net Architecture.

Advantages of U-Net

- Computationally efficient;
- Trained on a small dataset;
- Initially, it was used for biomedical images.

Disadvantages of U-Net

- More difficult with multiclass segmentation. The fully convolutional FPN and PSPNet are more adaptable.
- The boundary problem

DeepLabV3+

DeepLabV3+ is an advanced convolutional architecture that combines a deep encoder, dilated convolutions, and a lightweight decoder to efficiently capture multiscale context [27].

Dilated convolutions allow for a wider receptive field without loss of resolution.

$$i = \sum_k x(i + r \cdot k) w(k) \quad (24)$$

Where r the dilation rate remains.

The **Atrous Spatial Pyramid Pooling (ASPP)** module applies several dilated convolutions in parallel: $ASP(x) = \text{Concat}(\text{Convr1}, \text{Convr2}, \text{Convr3}, \text{Conv1} \times 1, \text{Global Pooling})$ (25)

This module enables efficient context capture at different scales.

The decoder merges deep features with low-level features to refine object boundaries.

Advantages and Limitations

- Excellent modeling of the overall context
- Very high accuracy

- High computational cost
- Poorly suited for embedded systems

SegFormer

SegFormer represents a new generation of segmentation models based on **Vision Transformers**, designed to be **simple, efficient, and lightweight**.

The encoder relies on a multi-scale hierarchical transformer. The attention mechanism is defined by:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \quad (26)$$

Unlike CNN, SegFormer naturally captures long-range dependencies and rich global context.

SegFormer uses a **minimalist MLP decoder**, which merges multi-level features without heavy convolutions:

$$F = \text{Conca}(MLP(F_1), \dots, MLP(F_n))$$

In SegFormer, the multi-level feature maps F_i extracted by the Mix Transformer encoder are projected into a common space using multilayer perceptrons, then concatenated to form a merged representation F , used for the final prediction of the segmentation (Table 1).

Table 1. Summary comparison of architectures.

Criteria	U-Net	DeepLabV3+	SegFormer
Type	CNN encoder-decoder	CNN + dilated conv	Transformer
Global context	Medium	Very good	Excellent
Precision of outlines	Very high	Very high	High
Complexity	Low to medium	High	Low
Real time	Possible	Difficult	Yes
Model	Convergence	Generalization	Risk
Linear	Slow	Medium	Underfitting
Simple U-Net	Medium	Correct	Slight overfitting
Extended	Very fast	Good (to be confirmed)	Overfitting possible
Model	Convergence	Generalization	Risk

3. RESULTS

In this practical section, we collected data (several photos of a construction site: <https://drive.google.com/drive/folders/1gH7qdzQ1HekWoE6yVEYUjdZf-EZrAad3>) To train the model, we created our own metric to track the training results [1, 3, 7, 8]. We defined the input images for the neural network, and the output provided the classification of each pixel, determining which class each pixel belongs to.

Quality Indicators

The use of metrics aims to minimize the segmentation model. The general formula for defining a metric is as follows:

follows:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \quad (27)$$

Segmentation accuracy is evaluated using two metrics:

1. Dice Similarity Coefficient (DSC) (overlapping segmented volumes);

$$DSC = \frac{2|S_{pred} \cap S_{ref}|}{|S_{pred}| + |S_{ref}|} \quad (28)$$

2. Mean symmetrical surface distance (ASD)

$$ASD = \frac{1}{N} \left(\sum_{x \in B_{pred}} d(x, B_{ref}) + \sum_{x \in B_{ref}} d(x, B_{pred}) \right) \quad (29)$$

Create Your Own Metric Function

$$Def = \frac{2(\sum k(y_{true} \times y_{pred}) + 1)}{(\sum k(y_{true}) + \sum k(y_{pred}) + 1)} \quad (30)$$

Training Results of Neural Networks

To evaluate the performance of the proposed custom metric (**Dice coefficient**), we first trained and evaluated the model using four segmentation architectures: linear network, Simple U-Net, U-Net, and Extended U-Net. Subsequently, further experiments were conducted by varying the activation functions and *batch size*. Five (5) images from our dataset, representing a construction site, were used as input, while the outputs correspond to the segmented images produced by each architecture. Figures 3 shows through 8 respectively present the original images, the segmented images, and the recognition results obtained using the four segmentation architectures studied.



Figure 3. Original images to be segmented.

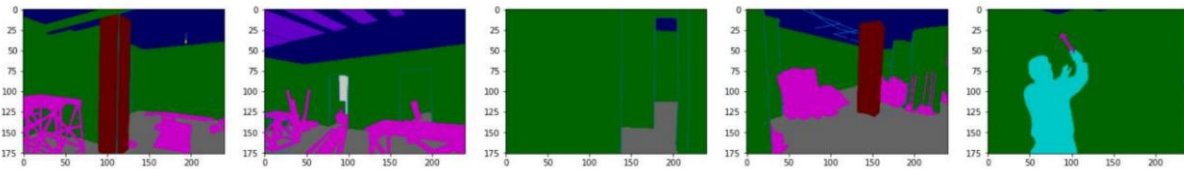


Figure 4. Segmented mask images.

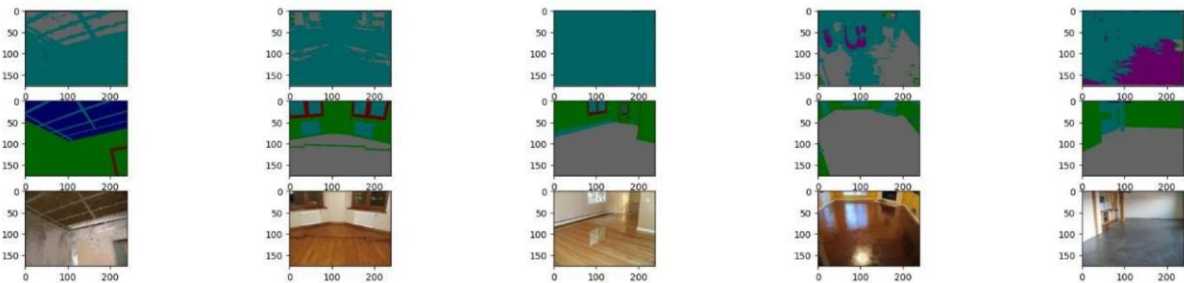


Figure 5. Segmented images with the linear Segmentation Net architecture.

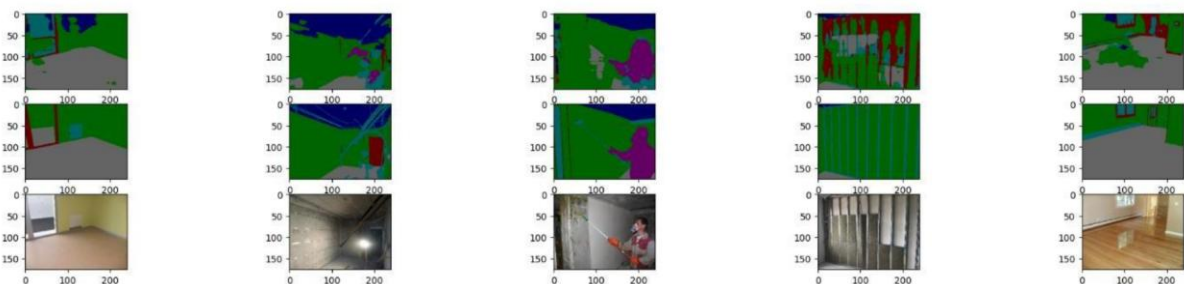


Figure 6. Segmented images with the U-net architecture.

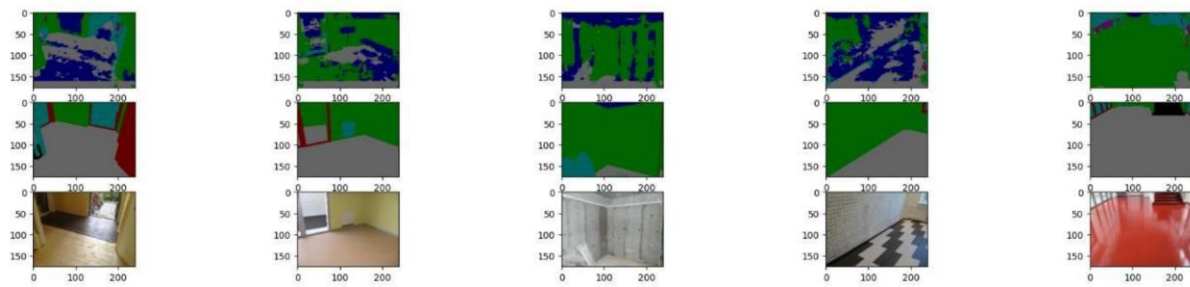


Figure 7. Segmented images with the simple Unet architecture.

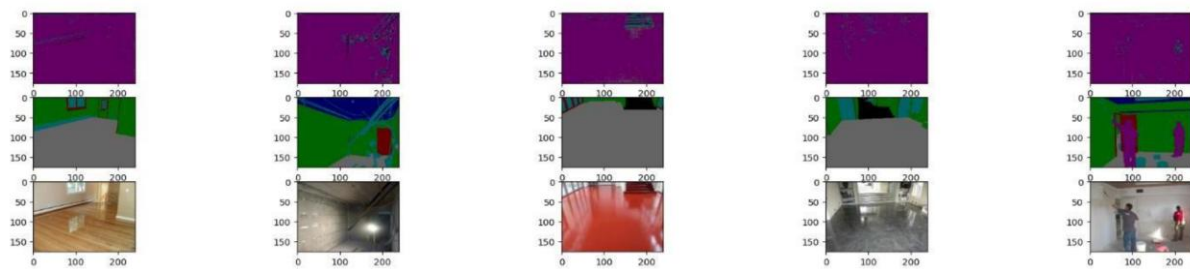


Figure 8. Segmented images with the extended U-net architecture.

Figures 9 and 10 illustrate and compare the evolution of training and validation losses for the four segmentation architectures evaluated: the linear network, the Simple U-Net, the U-Net, and the Extended U-Net. Overall, the results highlight a clear trade-off between model complexity and performance. The simplest architectures, such as the linear network and the Simple U-Net, are distinguished by their stability and low computational cost, but exhibit limited performance in terms of accuracy. Conversely, the U-Net offers the best compromise between accuracy, training stability, and generalization capabilities. Finally, the more complex architectures, such as the Extended U-Net, offer strong representational power but require rigorous validation to prevent overfitting.

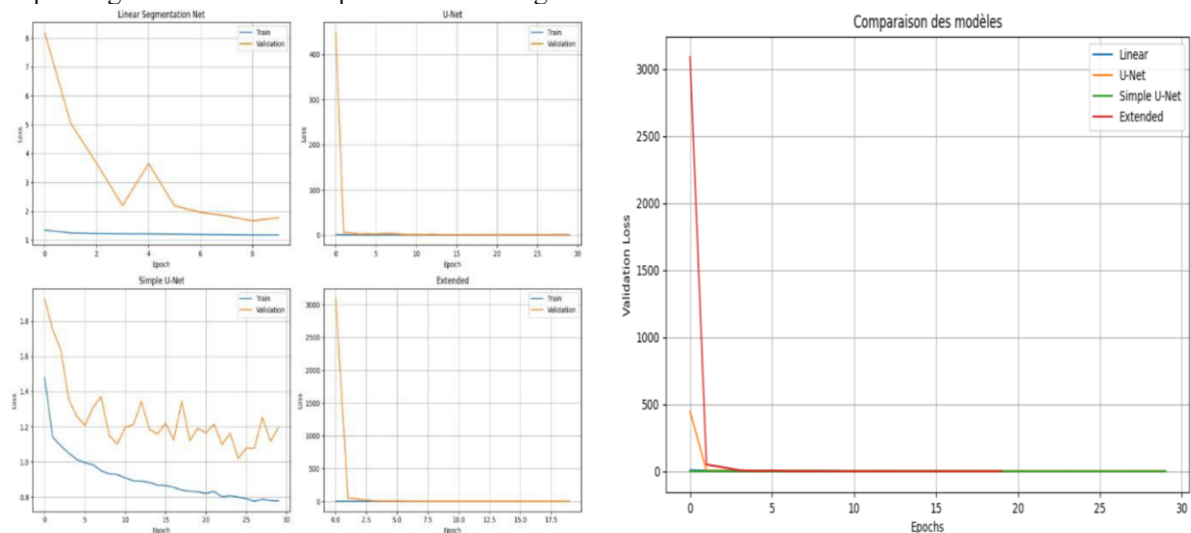


Figure 9. Evolution curves of training and validation losses for the four evaluated segmentation architectures.

Figure 10. Comparative curve of the evolution of validation loss as a function of the number of epochs for the four segmentation architectures.

Table 2. Training results of networks with different batch sizes and activation functions.

Network name	Values batch size	Activation function	Metric	Accuracy of recognition
Linear network	32	softmax	dice_coef	0.4415
U-net	32	softmax	dice_coef	0.7284
Simple U-Net	32	softmax	dice_coef	0.6034
	8		dice_coef:	0.6366
	128		dice_coef	0.5648
	32	elu	dice_coef	0.6460
		tanh	dice_coef	0.6730
		linear	dice_coef	0.6732
Extended U-Net	16	softmax	dice_coef	0.7734
U-Net simple training	32	softmax	iou_score:	0.4259
U-Net fine tuning	32	softmax	dice_coef	0.8977

U-Net offers the best compromise between segmentation accuracy, convergence stability, and generalization, making it the most suitable architecture for high-precision segmentation tasks, as shown in Table 2.

In the second part of our results, we conducted a comparative study of three segmentation architectures (U-Net, DeepLabV3+, and SegFormer), extending the experimental study to these models while ensuring a fair and reproducible comparison in terms of accuracy and efficiency.

Table 3 below illustrates a qualitative comparison of the segmentation results obtained by the U-Net, DeepLabV3+, and SegFormer architectures. We observe that U-Net tends to produce smoother boundaries, while

DeepLabV3+ and SegFormer allow for better object delimitation, particularly in complex areas. SegFormer demonstrates superior overall consistency due to its ability to capture long-range dependencies.

Table 3. Qualitative Comparison of Segmentation Results Obtained by the Architectures.

Model	Dice	IoU	Precision	Recall
U-Net	0.87	0.79	0.89	0.86
DeepLabV3+	0.91	0.84	0.92	0.90
SegFormer	0.90	0.83	0.91	0.91

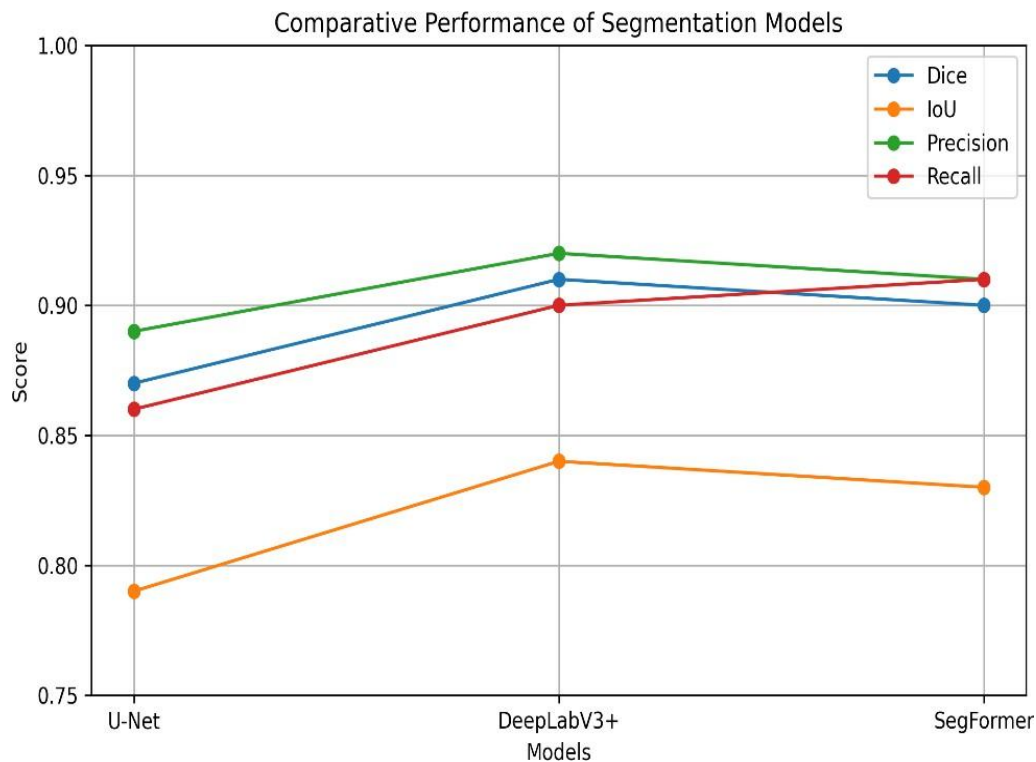


Figure 11. Comparison of the performance of the U-Net, DeepLabV3+, and SegFormer segmentation models in terms of Dice, IoU, accuracy, and recall.

Figure 11 presents a quantitative comparison of the performance of three image segmentation models (U-Net, DeepLabV3+, and SegFormer) in terms of **Dice**, **Intersection over Union (IoU)**, **accuracy**, and **recall**.

The results show that **DeepLabV3+** achieves the best overall performance across all the evaluated metrics. In particular, this model achieves the highest values for the Dice coefficient (0.91), IoU (0.84), and accuracy (0.92), indicating excellent concordance between the segmented regions and the reference annotations, as well as an increased ability to reduce false positives. This performance can be attributed to the use of the ASPP (Atrous Spatial Pyramid Pooling) module, which allows for better capture of multiscale context.

The **SegFormer** model exhibits highly competitive performance, with a Dice coefficient of 0.90 and an IoU of 0.83, while also achieving the highest recall value (0.91). This suggests that SegFormer is particularly effective at detecting a large number of relevant regions, at the cost of a slight compromise in accuracy compared to DeepLabV3+. This characteristic is consistent with the **Transformer**-based architecture, which favors global modeling of the spatial context.

The **U-Net** model, although exhibiting slightly lower performance, remains robust with satisfactory scores (Dice

= 0.87, IoU = 0.79). These results confirm the effectiveness of its encoder-decoder architecture with hop connections, while highlighting its limitations compared to newer and more complex architectures.

Overall, these results indicate that modern approaches incorporating multi-scale modeling or global attention outperform classical image segmentation architectures, especially for scenarios requiring precise object delimitation.

4. DISCUSSION

The experimental results demonstrate notable performance variations among the evaluated architectures, highlighting the influence of architectural design on image segmentation effectiveness. DeepLabV3+ achieved strong overall performance, particularly in Dice Score, IoU, and Precision, which can be attributed to its Atrous Spatial Pyramid Pooling (ASPP) module and its ability to capture contextual information at multiple spatial scales. However, the enhanced feature representation may involve greater computational requirements, which should be considered for resource-constrained deployment. SegFormer achieved the highest Recall, indicating its strong capability to identify relevant regions within the images.

This characteristic can be advantageous in applications where minimizing missed regions is important. U-Net, although comparatively simpler, remains an effective baseline architecture because of its straightforward design and established performance in image segmentation. Its relatively limited capacity for global contextual representation may, however, affect performance when segmenting objects with substantial variations in scale or complex spatial characteristics. Overall, the results indicate that segmentation performance depends on the characteristics of the architecture, evaluation criteria, and application requirements. Rather than relying on a single architecture for all scenarios, model selection should consider the required balance among segmentation accuracy, recall, computational complexity, and data characteristics. The findings also support further investigation of hybrid architectures that combine the local feature extraction capabilities of convolutional networks with the global contextual representation of Transformer-based models.

Perspectives

Future research can explore hybrid CNN–Transformer architectures and advanced attention mechanisms to enhance feature representation and segmentation accuracy. Evaluation on larger, more heterogeneous, and application-specific datasets, including precision agriculture, industrial inspection, and aerial imagery, would provide further insight into model generalization and robustness. In addition, model-compression techniques such as quantization and pruning could be investigated to facilitate efficient deployment in resource-constrained and real-time environments. Developing application-specific evaluation metrics may also provide a more comprehensive assessment of segmentation quality under practical operating conditions.

5. CONCLUSION

The proposed methods demonstrate the effectiveness of deep learning architectures for achieving accurate and robust image segmentation. The comparative evaluation of U-Net, DeepLabV3+, and SegFormer shows that DeepLabV3+ achieved the highest Dice Score (0.91), IoU (0.84), and Precision (0.92), while SegFormer attained the highest Recall (0.91). These results highlight the effectiveness of the proposed optimization strategies and the importance of selecting appropriate architectural configurations for pixel-level segmentation. Overall, the proposed methods provide a reliable framework for addressing complex image segmentation challenges. Future research may focus on larger and more diverse datasets, advanced attention mechanisms, Transformer-based architectures, and foundation models to further enhance segmentation accuracy, robustness, computational efficiency, and generalization.

REFERENCES

- [1] Long, J., Shelhamer, E., & Darrell, T. (2017). Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4), 640–651. <https://doi.org/10.1109/TPAMI.2016.2572683>.
- [2] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention: MICCAI 2015* (pp. 234–241). Springer. https://doi.org/10.1007/978-3-319-24574-4_28.
- [3] Garcia-Garcia, A., Orts-Escolano, S., Oprea, S. O., Villena-Martinez, V., & Garcia-Rodriguez, J. (2018). A review on deep learning techniques applied to semantic segmentation. *arXiv*. <https://doi.org/10.48550/arXiv.1704.06857>.
- [4] Minaee, S., Boykov, Y. Y., Porikli, F., Plaza, A. J., Kehtarnavaz, N., & Terzopoulos, D. (2021). Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7), 3523–3542. <https://doi.org/10.1109/TPAMI.2021.3059968>.
- [5] Pandey, J. K., Verma, S. K., Kumar, J., Perwej, Y., Jha, S. K., Panchal, B. Y., Ferdouse, R., Sindhu, V., Banerjee, S., Tiwari, M., Badal, R., Mandal, P., & Baghel, J. S. (2026). Transformative role of advanced neural computation in clinical image diagnostics: A review of key concepts and applications. *Seminars in Ultrasound, CT and MRI*. Advance online publication. <https://doi.org/10.1053/j.sult.2026.06.010>
- [6] Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1), 62–66. <https://doi.org/10.1109/TSMC.1979.4310076>.
- [7] Vincent, L., & Soille, P. (1991). Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6), 583–598. <https://doi.org/10.1109/34.87344>.

- [8] Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 801–818). Springer. https://doi.org/10.1007/978-3-030-01234-2_49.
- [9] Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2881–2890). IEEE. <https://doi.org/10.1109/CVPR.2017.660>.
- [10] Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., & Luo, P. (2021). SegFormer: Simple and efficient design for semantic segmentation with transformers. In *Advances in neural information processing systems* (Vol. 34, pp. 12077–12090).
- [11] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521, 436–444. <https://doi.org/10.1038/nature14539>.
- [12] Baron, D. (2019). Machine learning in astronomy: A practical overview. *Annual Review of Astronomy and Astrophysics*, 57, 1–34.
- [13] <https://doi.org/10.1146/annurev-astro-081817-051840>.
- [14] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 580–587). IEEE. <https://doi.org/10.1109/CVPR.2014.81>
- [15] Gonzalez, R. C., & Woods, R. E. (2018). *Digital image processing* (4th ed.). Pearson.
- [16] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* (Vol. 1, pp. 281–297). University of California Press.
- [17] LeCun, Y., & Bengio, Y. (1998). Convolutional networks for images, speech, and time series. In M. A. Arbib (Ed.), *The handbook of brain theory and neural networks* (2nd ed., pp. 255–258). MIT Press.
- [18] LeCun, Y., Bottou, L., Orr, G. B., & Müller, K.-R. (1998). Efficient backprop. In G. B. Orr & K. Müller (Eds.), *Neural networks: Tricks of the trade* (pp. 9–50). Springer. https://doi.org/10.1007/3-540-49430-8_2
- [19] Cireşan, D. C., Giusti, A., Gambardella, L. M., & Schmidhuber, J. (2012). Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in neural information processing systems* (Vol. 25, pp. 2843–2851).
- [20] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, M., Berg, A. C., Lo, W.-Y., Dollár, P., & Girshick, R. (2023). Segment Anything. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 4015–4026.
- [21] Li, X., Ding, H., Yuan, H., Zhang, W., Pang, J., Cheng, G., Chen, K., Liu, Z., & Loy, C. C. (2024). Transformer-Based Visual Segmentation: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12), 10138–10163. <https://doi.org/10.1109/TPAMI.2024.3434373>
- [22] Zhou, T., Xia, W., Zhang, F., Chang, B., Wang, W., Yuan, Y., Konukoglu, E., & Cremers, D. (2024). Image Segmentation in Foundation Model Era: A Survey. *arXiv preprint arXiv:2408.12957*.
- [23] Zhang, Y., Shen, Z., & Jiao, R. (2024). Segment Anything Model for Medical Image Segmentation: Current Applications and Future Directions. *Computers in Biology and Medicine*, 171, 108238. <https://doi.org/10.1016/j.combiomed.2024.108238>
- [24] Lee, H. H., Gu, Y., Zhao, T., Xu, Y., Yang, J., Usuyama, N., Wong, C., Wei, M., Landman, B. A., Huo, Y., Santamaria-Pang, A., & Poon, H. (2024). Foundation Models for Biomedical Image Segmentation: A Survey. *arXiv preprint arXiv:2401.07654*.
- [25] Zhou, T., et al. (2024). Image Segmentation in Foundation Model Era: A Survey. *arXiv preprint arXiv:2408.12957*.
- [26] Pandey, J. K., Kumar, S., Lamin, M., Gupta, S., Dubey, R. K., & Sammy, F. (2022). A metaheuristic autoencoder deep learning model for intrusion detector system. *Mathematical Problems in Engineering*, 2022, Article 3859155. <https://doi.org/10.1155/2022/3859155>
- [27] From CNN to Transformer: A Review of Medical Image Segmentation Models (2024). This recent review compares representative CNN- and Transformer-based segmentation architectures, including FCN, U-Net, DeepLab, TransUNet, Swin-Unet, and SAM.