International Journal of Basic and Applied Sciences, 14 (2) (2025) 691-702

International Journal of Basic and Applied Sciences

Website: www.sciencepubco.com/index.php/IJBAS
https://doi.org/10.14419/c62avh78
Research paper

Adaptive Cheetah Optimization-Driven CNN: A Hybrid Approach for Robust Image Segmentation

Rajendra V. Patil [1] *, Rahul Manohar Patil [2], Deepak Yashwantrao Bhadane [3], Govind Mohanlal Poddar [4], Anil L. Wakeker [5], Shravani R. Patil [6]

[1] Department of Computer Engineering, SSVPS Bapusaheb Shivajirao Deore College of Engineering, Dhule (MS), India
[2] Department of E and TC, NES's Gangamai College of Engineering. Nagaon, Dhule, Maharashtra, India
[3] Department of Computer Engineering, R. C. Patel College of Engineering and Polytechnic Shirpur. Maharashtra, India
[4] Department of Computer Engineering, NES's Gangamai College of Engineering. Nagaon, Dhule, Maharashtra, India
[5] Department of Computer Engineering, Devi Mahalakshmi College of Engineering & Technology, Mhaskal, Titwala (MS). India
[6] Department of Computer Engineering, SSVPS Bapusaheb Shivajirao Deore College of Engineering, Dhule (MS), India
**Corresponding author E-mail: patilrajendra.v@gmail.com*

Received: May 26, 2025, Accepted: July 4, 2025, Published: July 9, 2025

Abstract

Image segmentation plays a crucial role in various computer vision applications, including medical imaging, autonomous driving, and remote sensing. However, achieving high segmentation accuracy remains a challenge due to issues such as noise, occlusion, complex backgrounds, and class imbalance. Traditional segmentation methods often struggle with these challenges, necessitating the development of advanced deep learning-based approaches. To address these issues, we propose a CNN hybrid with Adaptive Cheetah Optimization (ACO) for improved image segmentation. The Convolutional Neural Network (CNN) serves as the backbone for feature extraction, while ACO optimizes the hyperparameters and fine-tunes the segmentation process to enhance accuracy and robustness. ACO, inspired by the hunting behavior of cheetahs, dynamically balances exploration and exploitation to avoid local optima and improve convergence speed. Experimental results demonstrate that our hybrid approach outperforms conventional CNN-based segmentation models in terms of precision, recall, and segmentation accuracy, particularly in challenging environments. This proposed method contributes to the advancement of image segmentation by addressing common challenges and improving performance through an adaptive optimization strategy.

Keywords: *Computer Vision;mImage Segmentation; Convolutional Neural Network; Adaptive Cheetah Optimization; Atrous Spatial Pyramid Pooling.*

1. Introduction

The core task of computer vision requires image segmentation because this process divides visual data into segments that enable better analysis procedures [1]. Multiple applications, including autonomous vehicles and medical imaging, utilize image segmentation as a vital process for their operations [2]. When images get segmented, viewers can identify scene objects and boundaries so they can make enhanced interpretations and decisions [3]. Image segmentation is essential because conventional images often contain multiple objects, textures, and noise, making direct analysis challenging [4]. Segmentation techniques become essential because they allow researchers to extract certain features while making tasks easier to compute and recognition and classification systems more precise [5]. Autonomous and intelligent systems demand robust and efficient segmentation methods because the growth of visual data in practical applications continues to increase.

Image segmentation encompasses three primary techniques, including thresholding segmentation, edge detection, and region-based approaches with k-means clustering [6-8]. The algorithms function with speed yet face obstacles during the processing of images that contain either noisy components or shadowed areas, or objects with intersecting shapes [9]. The development of machine learning enabled the integration of decision trees alongside random forests into segmental processes, although human interaction was necessary to develop the necessary features [10], [11]. Segmentation improved significantly with the incorporation of deep learning techniques in CNNs, where it learned hierarchy features from large labeled datasets with massive computing resources [12]. The implementation of deep transformers has advanced segmentation efficiency at the cost of extensive system requirements, together with elevated training costs [13]. The solution space for image obscuration issues continues to advance, with ongoing problems regarding image adaptation across different datasets, as well as real-time system operations [14]. Learning models that approach self-supervision and weak supervision reduce the need for data annotations, yet achieve results with lower precision levels compared to fully supervised practices [15]. The focus of continued research is on the creation of simple segmentation models showing explainable results alongside adaptability features.

The graph-based segmentation methods, including normalized cuts and random walks, effectively describe spatial relationships, although they cause high computational expenses for big images [15]. The watershed segmentation method enables boundary detection, although it

produces excessive segmentation for regions with texture patterns. Snakes used for active contour modeling help enhance object edges, yet their accuracy depends on noise levels and initialization accuracy. Users can enhance their computation speed by dividing images into superpixels, yet this approach may result in the loss of subtle details present in challenging imaging scenarios. The feature set of Fully Convolutional Networks (FCNs) allows point-level region segmentation, while they fail to maintain extensive inter-element relations. U-Net provides better segmentation for biomedical applications, though the system heavily depends on extensive labeled data. Segmentation models that use attention-based methods select better features but require large amounts of processing power and memory resources. Hybrid models implement various techniques together, but achieving both efficiency and accuracy and generalization capability continues to present difficulties.

The key contribution is

- Before training the model, practitioners should employ Normalization to standardize input data, which improves both variability control and model training speed.
- Through the implementation of CNN as the backbone feature extraction becomes more effective at extracting hierarchical image features that enhance complex pattern representation.
- The segmentation process gets enhanced with ACO by using dynamic hyperparameter optimization, which delivers better model performance together with improved robustness.
- The ACO achieves exploration and exploitation balance through its cheetah hunting behavior simulation, which stops local optima and speeds up convergence during training.
- The hybrid method leads to better performance metrics according to experimental results when applied to situations with noise, occlusion, and complex backgrounds.

The remainder of this paper is organized as follows: Section 2 reviews the findings from the literature study. Section 3 provides a detailed description of the proposed method. Section 4 presents the experimental results, demonstrating the effectiveness of the proposed method and validating its superiority. Finally, Section 5 concludes the paper with a summary of the findings and suggestions for future research.

2. Literature review

Numerous researchers have proposed advanced methods for image segmentation. A selection of key contributions is summarized below:

Zhang et al. [16] proposed affinity graph-based segmentation algorithms, which have emerged as a significant trend in computer vision. They introduced an adaptive fusion affinity graph (AFA-graph) including noise-free low-rank representation for live natural image segmentation. The AFA-graph system demonstrates efficient performance that aligns with existing benchmarks spanning the BSD300, BSD500, MSRC, SBD, and PASCAL VOC datasets. The approach fails to use discriminative information from texture shapes and past knowledge, compounded by its deficiency in deep unsupervised learning for improvement.

Zhang et al. [17] shown that deep convolutional neural networks have attained significant success in medical image segmentation. They provided an overview of the recent advancements in 2.5D techniques for volumetric medical image segmentation. Furthermore, to evaluate the performance and efficacy of these approaches, an empirical investigation is conducted on three sample segmentation tasks, including various modalities and targets. The proposed 2.5D segmentation methods achieve equivalent performance to 3D CNNs by employing efficient computational techniques that reduce model parameters and training costs. The study methodology employs 2D/3D U-Net as its principal algorithm; nevertheless, this technique may not yield optimal outcomes for certain segmentation tasks, and its efficacy varies across different datasets.

Chen et al. [18] presented the Segment Anything Model (SAM), a foundational model for broad image segmentation, which has exhibited remarkable zero-shot performance across many natural image segmentation challenges. They presented a modality-agnostic SAM adaptation system, termed MA-SAM, suitable for diverse volumetric and video medical data. Our approach is based on a parameter-efficient fine-tuning strategy that modifies just a limited subset of weight increments while maintaining the integrity of the bulk of SAM's pre-trained weights. This surpasses cutting-edge three-dimensional medical image segmentation methods by demonstrating superior generalization across several modalities without requiring prompts. The approach has challenges in segmenting small or ambiguous structures that require human intervention.

Roy et al. [19] introduced the adoption of Transformer-based architectures for medical image segmentation. They presented MedNeXt, a Transformer-inspired large kernel segmentation network that features 1) a completely ConvNeXt 3D Encoder-Decoder Network for medical image segmentation. 2) Residual ConvNeXt upsampling and downsampling blocks to maintain semantic richness across scales. An innovative method for progressively enlarging kernel sizes through the upsampling of tiny kernel networks, aimed at circumventing performance saturation in constrained medical datasets. 4) Compound scaling over many dimensions (depth, breadth, kernel size) of MedNeXt. MedNeXt achieves superior outcomes with its ConvNeXt-based architecture across four intricate medical image segmentation tasks, concurrently surpassing seven existing benchmarks. The utilization of constrained medical datasets impedes the generalisation of the system to novel domains or diverse imaging situations.

Li et al. [20] presented the Segment Anything Model (SAM), which has been swiftly adopted for the segmentation of many natural pictures. A unique approach, AutoSAM Adapter, was devised exclusively for 3D multi-organ CT-based segmentation. They utilise parameter-efficient adaptation strategies to create an autonomous prompt learning paradigm, enhancing the SAM model's capabilities for 3D medical image segmentation and obviating the necessity for manually created prompts. The AutoSAM Adapter facilitates SAM adaptation for 3D multi-organ medical image segmentation, enhancing efficiency using minimal parameters and automated, fast creation. The system encounters challenges in digesting intricate anatomical differences while also striving for enhancements in the effective calculation of extensive 3D medical data sets.

Despite considerable progress in segmentation designs, optimization methods for hyperparameter adjustment are still largely unexplored. Metaheuristic methods, including Particle Swarm Optimization (PSO), Genetic methods (GA), and Grey Wolf Optimizer (GWO), have demonstrated efficacy in improving the performance of learning models through the optimization of critical parameters [21]. Nonetheless, some techniques experience early convergence or incur substantial computational expenses. Adaptive Cheetah Optimization (ACO), a novel bio-inspired algorithm, establishes a dynamic equilibrium between exploration and exploitation, emulating the strategic hunting tactics of cheetahs [22], [23]. Its low cost and flexibility render it especially appropriate for CNN-based segmentation jobs where conventional grid search or random search may prove inadequate.

Existing segmentation methods work well; however, real-world applications often introduce additional challenges. 2.5D approaches balance 2D speed and 3D precision but struggle with volumetric data spatial continuity [17]. Transformer-based models like MedNeXt are accurate yet computationally intensive, restricting their scalability. Graph-based methods are computationally costly and susceptible to

noise in big photos [19]. We suggest integrating CNN with ACO for robust segmentation since these trade-offs highlight the need for hybrid systems that increase accuracy, efficiency, and generalization [16].

3. Proposed method of segmentation

The performance enhancement via hyperparameter optimization uses ACO, which adjusts learning rate values together with filter sizes and activation functions [21] The performance optimization process of ACO utilizes characteristics from cheetah hunting patterns to choose between exploration and exploitation methods, which avoid becoming trapped at suboptimal solutions [22]. The optimized CNN model reaches greater accuracy, together with better stability in its segmentation outcomes, because of this fine-tuning process. Implementation of CNN with ACO results in superior segmentation results for challenging medical imaging situations featuring noisy data alongside occluded areas and unbalanced classes. Adaptive optimization enables the proposed method to achieve better performance than Standard CNN-based segmentation models when evaluating precision and recall values, and total segmentation accuracy. Figure 1 shows a flow diagram of image segmentation [22].

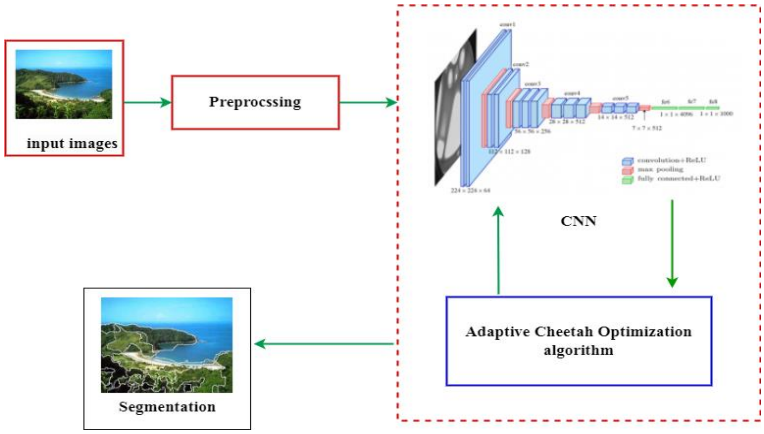


Fig. 1: Flow Diagram of Image Segmentation.

3.1. Preprocessing

The RGB images in the collection come in various sizes. The photos are preprocessed using the subsequent procedures. To obtain a uint8 tensor and three-color channels, decode a JPEG-encoded image. Adjust each image's dimensions to 299 by 299 pixels. The image needs to be normalized to value ranges between 0 and 1 and converted to a type.

The process of image segmentation stands as a necessary step in both computer vision and machine learning, primarily for normal image datasets. The process of segmenting images into distinct areas produces better features and enhanced model performance. Segmented regions need normalization to achieve consistency through pixel value standardization, which controls variations in lighting and sensor quality as well as intensity.

Segmented images require normalization as a preprocessing step, which removes scale variations and improves comparability between data values and avoids inflated prominence of large numerical features. Machine learning models with deep learning components need additional time to converge and become unstable when input data has big numerical value differences. The model learns effectively from divided data through normalization because the process stabilizes training procedures.

Different parts of the image show diverse levels of pixel brightness during image segmentation. The normalization process prevents segmentation frameworks from concentrating their efforts on select regions because this practice distorts the definition of classification areas. The model processes all image regions with equal understanding when the normalization procedure is applied to segmented data.

Pixel intensities need normalization to operate within specific ranges between 0 and 1 or -1 to 1 for better activation functions in deep learning networks. The research implemented min-max scaling normalization on the separated image areas containing objects of interest. Maximum and minimum pixel values transform values between 0 and 1 using the min-max scaling technique. The normalization procedure starts by locating sectioned areas needing normalization then maintains uniform lighting.

The normalization method operates by deducting pixel minimum values from every pixel and then splitting the range through intensity values according to Equation (1).

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (1)$$

Employing min-max scaling to characteristic image data rescales pixel intensities to 0 to 1. This is advantageous when there is a significant variation in pixel intensities across distinct photos, since it ensures consistency in the dataset.

3.2. Image segmentation processing based on the CNN algorithm

3.2.1. CNN framework

The segmentation algorithm uses the Figure 1 sub-network structure for its neural network operations. This technique improves multiple convolutional network efficiency by modeling neighboring tags' inter-relations in segmentation. The decision from the last prediction layer receives data influence from neighboring tags. The CNN architecture contains a cascade structure because the first convolutional network output functions as an additional input for the second. Training effectiveness receives an enhancement through the introduction of residual connections that establish an independent parameter layer bypass. Figure 2 shows a segmentation method based on CNN.

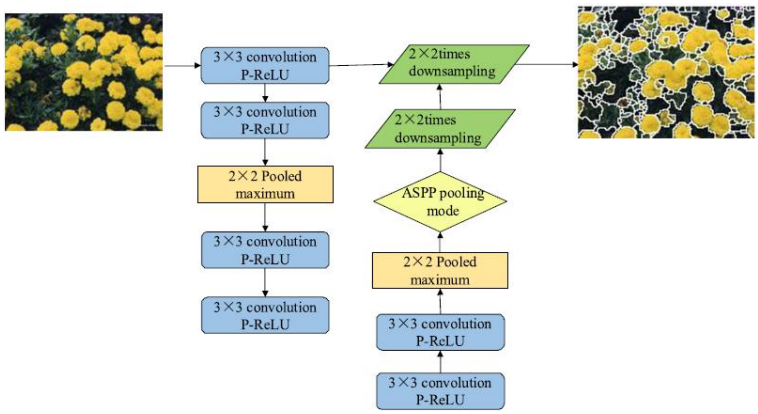


Fig. 2: Segmentation Method Based on CNN.

a) Convolutional layer

The convolutional layer consists of three tensors: two feature maps and one convolution kernel, as shown in the Equation.

$$x_j^0 = \sum_{i \in M} x_{j-1} \otimes k_{ij}^0 + b_j^0 \tag{2}$$

Where, x_{j-1} is the input feature map, k is the convolution kernel, b is the bias term, and x_j It is the output feature map.

The convolutional operation applies a kernel that moves across images or feature maps to compute weighted sum values from local areas, which produce new feature map elements.

b) Pooling Layer

The pooling layer delivers two benefits by decreasing computational complexity and reducing overfitting through decreased connections between convolutional layers. Max pooling serves as the conventional pooling approach in neural network systems.

$$x_j^0 = \max\left(x_{j-1}\right) \tag{3}$$

This selects the maximum value from each region, helping to retain important texture details while reducing parameter sensitivity in convolutional layers. Max pooling is preferred to random pooling because its gradients support the detection of local texture details.

c) Activation Layer

An introduction of non-linearity in CNNs occurs when Sigmoid and Tanh activation functions are applied. The activation functions enable more efficient optimization procedures during the training process.

Sigmoid function:

$$S(x) = \frac{1}{1+e^{-x}} \tag{4}$$

Tanh function (faster convergence):

$$\text{Tanh}(x) = \frac{1-e^{-2x}}{1+e^{-2x}} \tag{5}$$

Both functions, however, can suffer from gradient issues in deep networks due to saturation.

d) Deconvolution Layer

The input layout stays the same size through unpooling before convolution takes place. The deconvolution process involves:

- Unpooling functions as an activation position tracker, which expands the initial input into a sparse image.
- The method of filling sparse output zones through specific selection resembles the principle behind bilinear interpolation.

3.2.2. Loss layer

To optimize the network, a scale-invariant loss function is used.

$$L\left(y, y^*\right) = \frac{1}{n}\sum_i d_i^2 - \frac{\lambda}{n^2}\left(\sum_i d_i\right)^2 \tag{6}$$

This function considers both Euclidean distance loss and 3D structure preservation, making the CNN less sensitive to scene changes and improving depth estimation accuracy.

3.2.3. Cascade structure

Traditional CNNs predict segmentation labels separately, making it difficult to model spatial dependencies [22].

- It dynamically adjusts the number and depth of neurons.
- It allows different methods to work together, improving overall performance.

In this approach:

- The first CNN segment key regions from the original image.
- The second CNN further refines these regions into specific substructures.
- This step-by-step segmentation improves accuracy and simplifies complex tasks.

During training, standard data is used to define segmentation areas. During testing, the segmented output from previous steps helps refine further segmentation.

3.2.3. Residual connection

The deep convolutional neural network (CNN) training efficiency gets enhanced through the implementation of a residual network structure [24], [25]. Direct information transmission through residual connections enables both easier optimization and better network performance across growing depths.

The proposed design includes three blocks of residuals that apply two convolutional layers with direct connections between layers. The input information from a residual block becomes part of its output through direct addition, which helps the block learn what it should transform rather than a whole transformation. This shapes and accelerates the learning process while protecting against gradient vanishing through this technique.

3.2.4. Atrous spatial pyramid pooling (ASPP)

The ASPP mechanism operates as a solution to the input challenges [26]. Atrous convolution was developed for enhancing wavelet transform performance before researchers found its application in deep convolutional neural networks (DCNNs). The standard practice of using DCNNs demands that images enter the network at predetermined dimensions, forcing users to perform adjustments through cropping or resizing, which subsequently reduces diagnostic precision. The introduction of ASPP enabled networks to analyze images of any dimensions while retaining their spatial details.

The Atrous convolution method extends the detection area of filters by using optimized algorithms that maintain parameter efficiency and reduce computation time. The improved control of the feature resolution enhances both the segmentation of fine details and the region classification of larger structures.

ASPP takes its inspiration from spatial pyramid pooling through its application of atrous convolutions with varying dilation rates to extract features at different scales. The merged outputs lead to increased classification precision in the system. An updated version of ASPP combines global average pooling technology with 1×1 and 3×3 atrous convolutions for optimal feature mapping yet minimal computational strain.

Segmentation Prediction (Softmax for Multi-Class Segmentation)

For a given input image X, the CNN predicts a segmentation mask $\hat{Y}$, where each pixel belongs to a class:

$$P\left(y_i = c \mid X\right) = \frac{e^{W_c \cdot X_i + b_c}}{\sum_{c' \in C} e^{W_{c'} \cdot X_i + b_{c'}}} \tag{7}$$

Where:

- y_i is the predicted label for pixel i,
- W_c and b_c are the weights and biases for class c,
- C is the set of possible classes,
- $P\left(y_i = c \mid X\right)$ is the probability of pixel i belonging to class c.

3.3. Adaptive Cheetah Optimization Algorithm

To enhance the performance of the Convolutional Neural Network (CNN), Adaptive Cheetah Optimization (ACO) [23] is utilized for parameter optimization. This method efficiently explores the parameter space to identify optimal weights and configurations, thereby improving the model's accuracy. ACO is a recently developed meta-heuristic inspired by the hunting strategy of cheetahs. It offers key advantages such as fast convergence, fewer required parameter adjustments, and low computational overhead. The algorithm progresses through three core stages: exploration (searching), observation (waiting), and execution (attacking).

a) Initialization

In Adaptive Cheetah Optimization (ACO), the population is represented by a group of candidate solutions, where each cheetah's position within the search space signifies a possible solution. During the optimization process, key hyperparameters such as learning rate (L_r batch size (B_S), loss function (L_f), and the number of epochs (N_e) are taken into account to ensure effective tuning of the model. Each cheetah symbolizes a unique solution, and its position indicates the quality of that solution within the parameter space. The algorithm iteratively updates these positions to converge toward the optimal solution..

$$W = \left\{w_1, w_2, w_3, \ldots\ldots, w_n\right\} \tag{8}$$

Where, W is the population and w_n represents the n^{th} solution.

$$w_1 = [L_r, B_s, L_f, N_e] \tag{9}$$

In which, τ represents the current iteration; $\gamma \in (0,1)$

b) Fitness calculation for Segmentation

Dice Coefficient (Fitness Accuracy)

$$Fitness = Dice = \frac{2TP}{2TP + FP + FN} \tag{10}$$

- TP (True Positive) – Correctly predicted segmented pixels.
- FP (False Positive) – Pixels wrongly classified as part of the segment.
- FN (False Negative) – Pixels missed in the segmentation.
- Ranges from 0 (worst) to 1 (perfect segmentation).

c) Updation process

i) Searching phase: - The cheetah carefully observes its environment, relying on its innate hunting instincts to identify the most suitable prey by analyzing the dynamics of its surroundings. This behavior is mathematically modeled as follows:

ii)

$$C_{i,j}^{\tau+1} = C_{i,j}^{\tau} + rand \times \ell_{i,j}^{\tau} \tag{11}$$

$$\ell_{i,j}^{\tau} = 0.001 + \frac{\tau}{\tau_{\max}}\left(U_L - L_L\right) \tag{12}$$

Where $C_{i,j}^{\tau}$ denotes the current position of cheetah-iat iteration τ, $C_{i,j}^{\tau+1}$ represents the updated position of cheetah-iat iteration $\tau+1$, $rand$ is a random number selected from the interval (0, 1), $\ell_{i,j}^{\tau}$ indicates random step length; U_L *and* L_L represents the upper and lower bounds of the variable, respectively; τ and $\tau_{\max}$ denote the current and maximum iteration numbers.

iii) Sitting and waiting phase: Every motion a cheetah makes when hunting has the danger of alerting and possibly shocking its prey, which could result in an escape. Cheetahs wait patiently for prey to approach them from a distance by remaining low or hiding in bushes. The cheetah keeps its location constant during the sitting and waiting phase, which can be expressed numerically as follows:

$$C_{i,j}^{\tau+1} = C_{i,j}^{\tau} \tag{13}$$

iv) Attacking phase: A cheetah's hunting success relies heavily on precisely timed attacks, leveraging its remarkable speed and agility. In the attacking phase, the cheetah rapidly reduces the distance to its target, aiming to disrupt and capture it with precision and velocity. Whether hunting individually or in coordination with others, the attack requires strategic positioning based on the prey's movement and group dynamics. This phase can be mathematically represented as follows:

$$C_{i,j}^{\tau+1} = C_{H,j}^{\tau} + M_{i,j} \cdot N_{i,j}^{\tau} \tag{14}$$

$$M_{i,j} = \left|rand_{i,j}\right|^{\exp\left(\frac{rand_{i,j}}{2}\right)} \times \sin(2\pi\, rand_{i,j}) \tag{15}$$

$$N_{i,j}^{\tau} = C_{q,j}^{\tau+1} - C_{i,j}^{\tau} \tag{16}$$

Where, $C_{H,j}^{\tau}$ denotes the position of the prey, $M_{i,j}$ *and* $N_{i,j}^{\tau}$ stand for the turning and interaction factors, respectively, and $rand_{i,j}$ represents a value selected at random from a normal distribution. Furthermore, the symbols $C_{q,j}^{\tau+1}$ *and* $C_{i,j}^{\tau}$ represent the locations of cheetahs q and i at iteration τ, respectively.

Tent chaotic mapping

Following the introduction of Tent chaotic mapping, the Tent chaotic map was used to initialize the cheetah instead of the randomly generated approach used in the regular CO.

$$C_{i,j}^{\tau+1} = C_{i,j}^{\tau} + T_c^{\tau} \tag{17}$$

$$T_c^{\tau} = \begin{cases} \dfrac{T_c^{\tau-1}}{\gamma}, & T_c^{\tau-1} \in [0,\gamma] \\ \dfrac{(1-T_c^{\tau-1})}{(1-\gamma)}, & T_c^{\tau-1} \in [\gamma,1] \end{cases} \tag{18}$$

Dynamic weighting factor

During this phase, the cheetah continuously updates its position using a dynamic weighting factor denoted by λ. At the beginning of the iteration process, λ is assigned a relatively high value to facilitate an effective global search across the solution space. As the algorithm progresses, λ adaptively decreases, allowing the search to transition from exploration to exploitation, thereby refining the solution in the later stages.

$$C_{i,j}^{\tau+1} = C_{H,j}^{\tau} + \lambda_{i,j} \cdot N_{i,j}^{\tau} \tag{19}$$

$$\lambda = \frac{e^{4(1-\delta)} - e^{-4(1-\delta)}}{\left[e^{2(1-\delta)} + e^{-2(1-\delta)}\right]^2}, \quad \delta = \frac{\tau}{\tau_{\max}} \tag{20}$$

$$N_{i,j}^{\tau} = C_{q,j}^{\tau} - C_{i,j}^{\tau} \tag{21}$$

These modifications enhance the core functionality of Adaptive Cheetah Optimization (ACO) by improving its search efficiency and maintaining a balanced trade-off between exploration and exploitation. As a result, the algorithm is better equipped to avoid local optima and converge toward a more optimal global solution.

d) Termination

The best solution found during the procedure is returned by the ACO algorithm after the execution [27].

Algorithm 1: Adaptive Cheetah Optimization (ACO)
Input:
Learning Rate (L_r batch size (B_s), Loss Function (L_f), Number of Epochs (N_e)
Output: Optimal output
Start
– Initialize the solution population, total number of iterations, and population size.
– Use the Tent chaotic map to generate initial positions of cheetahs within the search space to enhance diversity.
– Randomly distribute cheetahs across the search area.
– For (each iteration):
– Evaluate the fitness of each cheetah.
– If (no prey is detected):
- Update cheetah positions using step length and a random factor.
- Keep the position fixed during the waiting phase.
– Else (prey is detected):
- Adjust cheetah positions relative to the prey using a turning factor.
- Adapt the Tent chaotic map and the dynamic weighting factor (λ) to balance global and local search.
Repeat until the maximum iterations are reached or the desired fitness value is achieved.
Return the best solution found.
End

4. Results and discussion

The proposed model demonstrates a clear performance advantage over baseline image segmentation methods. Experimental results across various social media datasets confirm the model's robustness and scalability, consistently achieving higher accuracy in detection tasks. It shows notable improvements in key performance metrics, including accuracy, precision, recall, and F-measure, when compared to existing techniques. The experiments were conducted on a system with an Intel(R) Core(TM) i5-4570S CPU @ 2.90 GHz, 8 GB of RAM, and running a 64-bit Windows operating system. The machine, manufactured by Acer and identified as SSM107.smg.local, utilized the MATLAB software environment for implementation. The experimental setup also includes two data centers comprising four hosts, collectively equipped with 8 GB of RAM and a network bandwidth of 2800 Mbps.

4.1. Dataset description

The dataset consists of 500 natural images, ground-truth human annotations, and benchmarking code. The data is explicitly separated into disjoint train, validation, and test subsets. The dataset is an extension of the BSDS300, where the original 300 images are used for training/validation, and 200 fresh images, together with human annotations, are added for testing. Each image was segmented by five different subjects on average [28].

4.2. Evaluation metrics

The experimental data were produced from a CNN-based segmentation methodology. Our deep learning model performed automatic extraction of features together with segmentation tasks. A post-processing technique enhances accuracy through a refinement procedure that operates on insufficient initial segmentation results. Users can operate the GUI application to draw indicators for object regions and

background areas through an easy-to-use interface. The object markers are represented through blue lines, while background markers are indicated by green lines during this procedure. The CNN model uses the marked regions to learn spatial context information, which results in better accuracy outcomes.

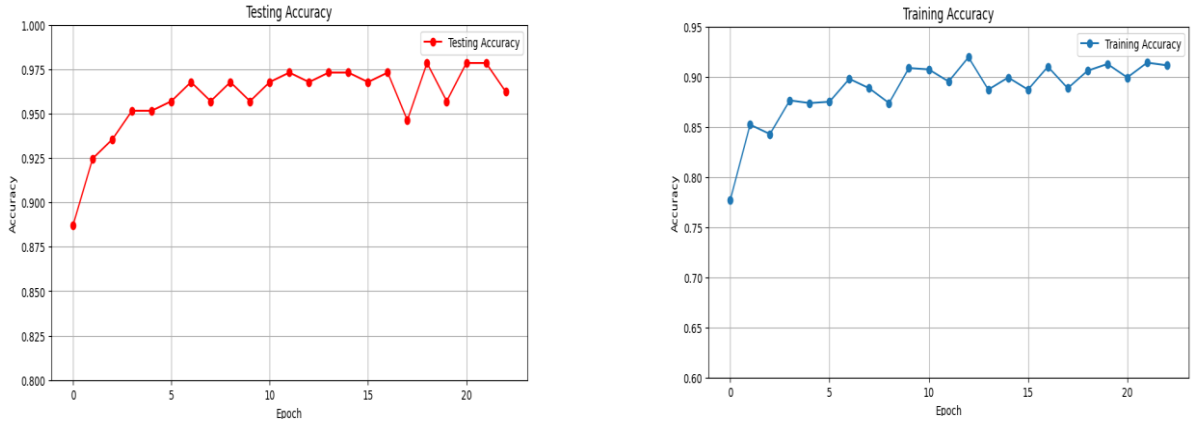


Fig. 3: Testing and Training Accuracy.

Multiple Epochs show how training accuracy performs alongside testing accuracy through graphs depicted in Figure 3. The model learns effectively from training data as indicated by the first graph line, which shows continuous improvement of training accuracy. The accuracy rate stabilizes at 90% after hitting temporary instabilities during the first several epochs. The second graph shows testing accuracy reaching 97% consistency during all epochs while maintaining higher accuracy than training accuracy. Generalization of unseen data points is well demonstrated by the model's performance. The robust testing accuracy proves that the training process has been effective by demonstrating strong generalization capabilities in an independent testing environment.

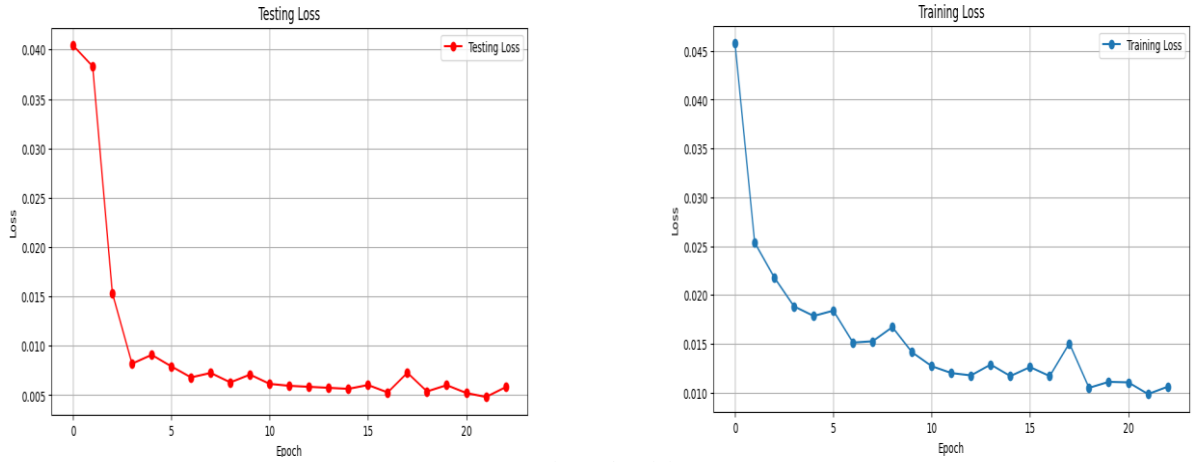
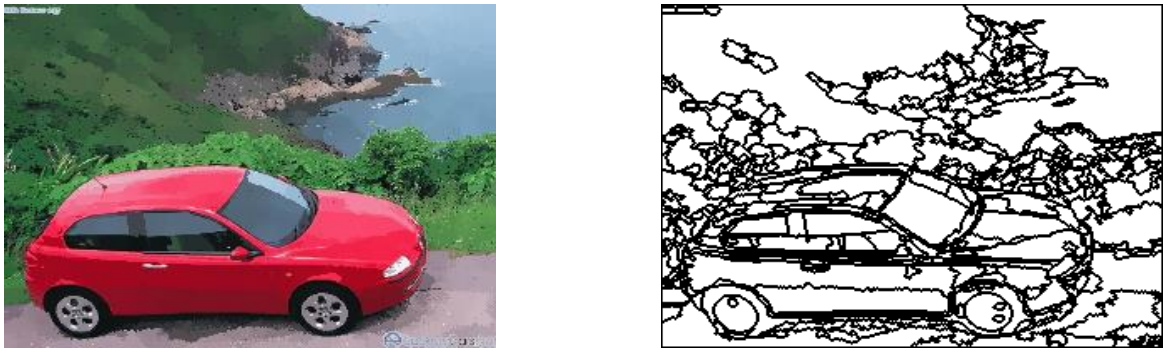


Fig. 4: Testing and Training Loss.

Figure 4 illustrates training and testing loss evolution over consecutive epochs. Training loss experiences a fast decline during the beginning, followed by stable performance in the first graph, which shows the model consistently improves its performance. The model demonstrates efficient pattern learning from training data because the loss stabilizes at a reduced value during the epochs. A substantial decline in testing loss becomes evident during the first phase and stabilizes through a progressive downward trend until it reaches its lowest point. The model achieves its lowest generalization point, which exceeds the training loss stabilization point, because of its excellent ability to predict new data points. The loss values presented in both graphs reveal a gradual decrease through each epoch because the model successfully acquires knowledge and proves its success for training and testing data.



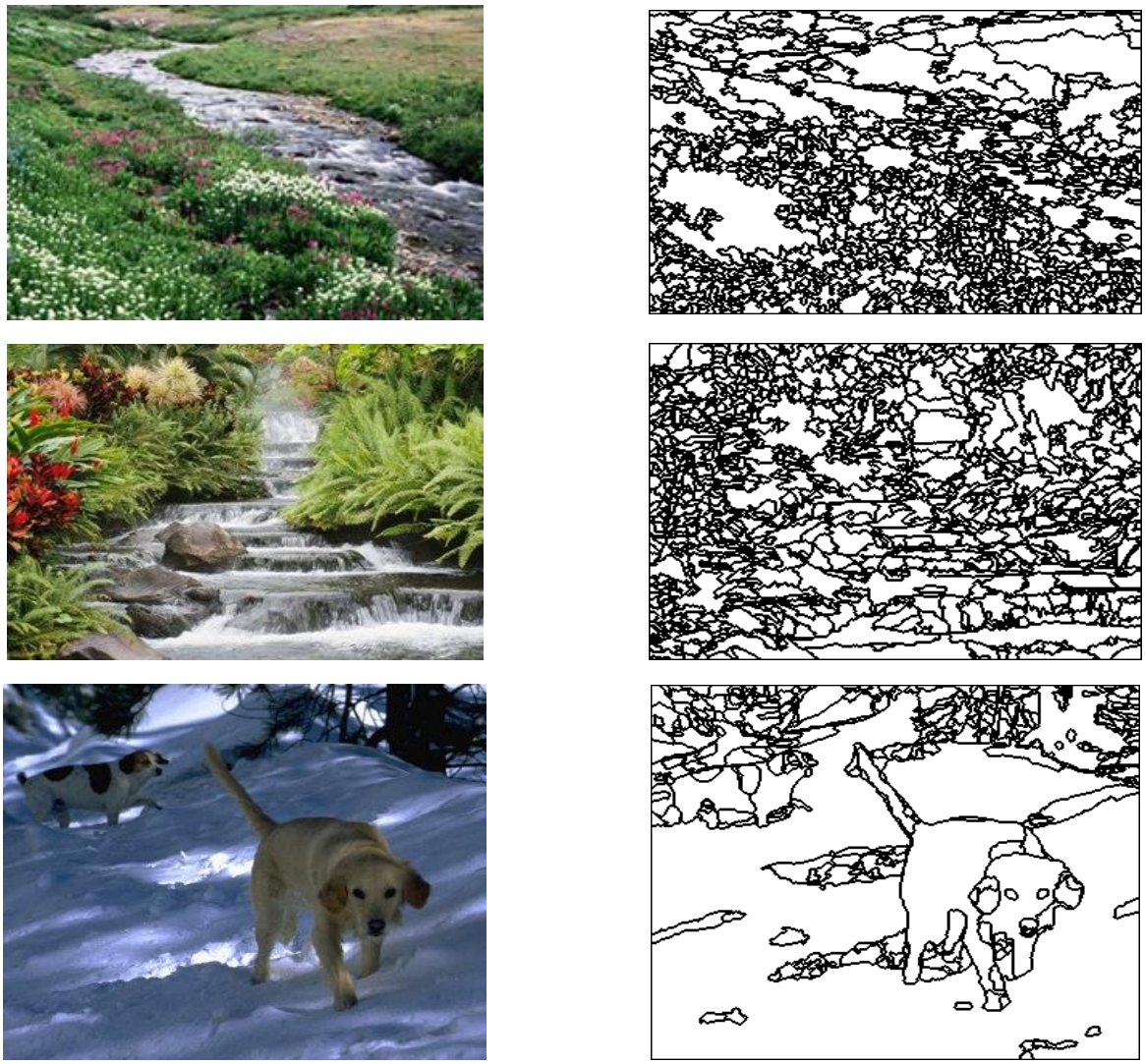
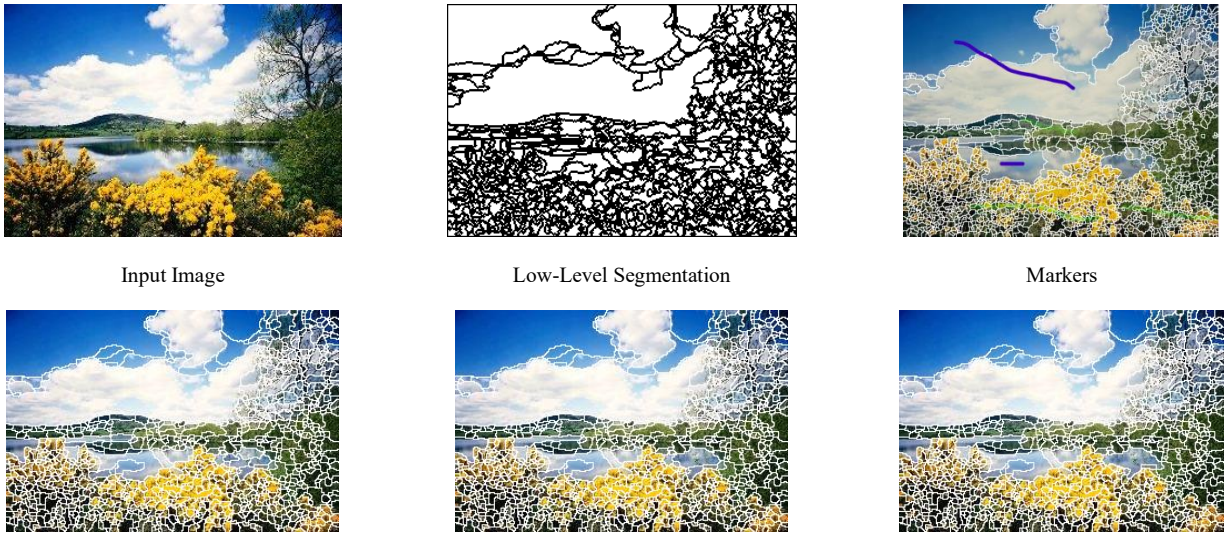


Fig. 5: Original Image and Corresponding Low-Level Segmentation Image.

Each pair in Figure 5 contains both an original photograph and its equivalent low-level segmentation illustration. The originals present several different images featuring a red car and flowers, a waterfall, followed by a dog in a snow environment. The segmented imagery illustrates basic outlines of main objects throughout simplified pictures by diminishing other visual elements. The simplified low-level segmentation helps algorithms to better process shapes and boundaries in visual content. Through this comparison, one can observe how segmentation transforms complex images into simplified outlines, which demonstrates its strength in image analytical operations.



Input Image Low-Level Segmentation Markers

Fig. 6: Segmentation Output: Nature.

Figure 6 depicts the nature scene segmentation development through three distinct parts that include the input image and output, along with markers. The initial image "a" contains a vivid mountain and flower landscape. The model generates a segmentation output that shows the contours of different elements throughout the image while conserving some part of the original detail.

The segmentation process gets refined through visual indicators called markers (c) that mark regions of interest. Each subsequent row in the visualization displays an incremental improvement in segmentation quality until the individual scene elements receive optimized boundary recognition. The segmentation techniques successfully separate multiple parts inside complex images through this repetitive method.

Table 1: Performance Evaluation of Proposed and Existing Methods

Methods	Accuracy	Precision	Recall	F-measure
RNN	88.2	89.00	88.3	88.39
SVM	94.09	94.89	93.05	94.59
NB	98.6	98.64	98.45	98.39
Proposed	99.30	99.56	99.36	99.39

An evaluation of the proposed and existing methods for segmentation appears in Table 1 using accuracy, precision, recall and F-measure metrics. The Naïve Bayes (NB) method uses traditional methods to bring forth the highest accuracy of 98.6% while performing better than Support Vector Machine (SVM) and Recurrent Neural Network (RNN). The proposed approach demonstrates the best performance by achieving an accuracy of 99.30% accompanied by a precision of 99.56% and a recall reaching 99.36%. The improved performance level demonstrates the proposed method's capability to handle classification tasks effectively. The F-measure outcome reaches 99.39% which indicates balanced performance across all precision and recall measures. Testing results show that the proposed model performs the best in present-day segmentation measurements.

An internal ablation investigation examined how Adaptive Cheetah Optimization (ACO) affected the proposed model's performance. We examined a baseline CNN without optimization, a CNN with manually tuned hyperparameters, and the suggested CNN optimized with ACO. Despite minor gains from human tuning, the ACO-integrated model outperformed the baseline in accuracy, precision, recall, and F-

measure. This improvement shows ACO's ability to find optimal parameter combinations, improving segmentation accuracy and model generalization.

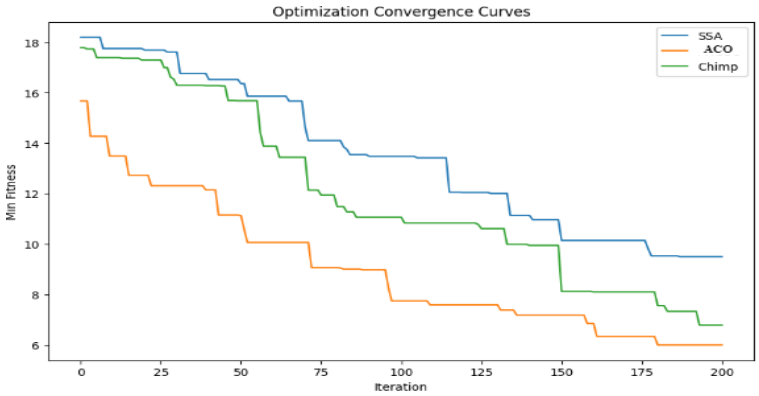


Fig. 7: Convergence Curve.

Figure 7 shows convergence curves representing different optimization approaches that include Adaptive Cheetah Optimization (ACO) among them. The minimum fitness values recorded by the Y-axis show how well each algorithm optimizes its problems. The number of iterations represents the period in which fitness measurements improve. The minimum fitness values decrease consistently through ACO executions, which confirms its effective optimization capabilities toward finding optimal solutions. Different optimization algorithms shown in the plot demonstrate changing improvement levels, although ACO consistently leads its counterparts in multiple iterations. The convergence curve proves that through its comparison with various optimization methods, ACO demonstrates its superiority in reaching optimal results efficiently.

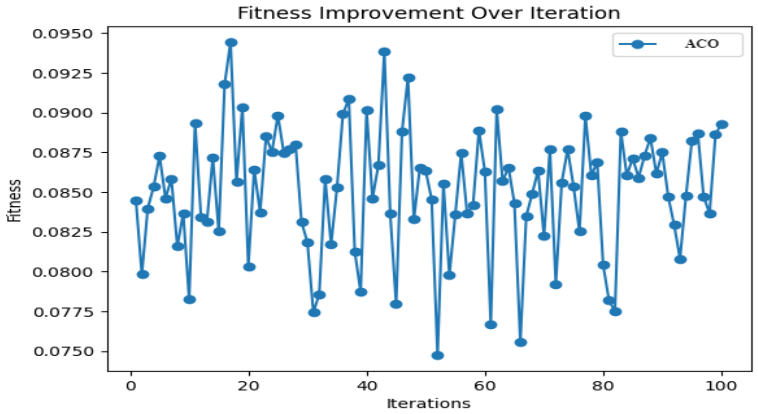


Fig. 8: Fitness Improvement Over Iteration.

The fitness enhancement of Adaptive Cheetah Optimization (ACO) appears in Figure 8 across 100 iterations. The graph shows fitness values on its Y-axis, which contains performance metrics of the solutions as they evolve throughout the iterations. The x-axis shows the optimization process through a series of steps. The optimization dynamically alters fitness values through time, which produces occasional peaks that signal improvements during iterations. ACO demonstrates strong effectiveness in solution optimization through its steady progress towards higher fitness values. The graph reveals how ACO efficiently handles solution space exploration and exploitation, which results in stable performance.

4.3. Limitations of the proposed method

The proposed CNN-ACO hybrid model exhibits enhanced segmentation accuracy and robustness across various evaluation metrics; however, it is essential to acknowledge certain limitations to provide a comprehensive perspective on its practical applicability. One major concern is the computational intensity associated with the Adaptive Cheetah Optimization algorithm. While ACO improves model convergence and performance through dynamic hyperparameter tuning, its iterative optimization process results in extended training times and increased resource consumption. This may pose challenges for real-time applications or deployment on low-power devices.

Another limitation lies in the scope of experimental validation. The model has been evaluated exclusively on the BSDS500 dataset, which, although diverse, may not adequately reflect the complexity of specialized imaging tasks such as medical diagnostics or satellite-based remote sensing. As a result, the generalizability of the proposed approach to other domains remains uncertain and warrants further investigation.

The ACO also depends on precise initialization parameters, including population size and the number of iterations. These settings can influence the stability and reproducibility of the segmentation outcomes. The absence of a formal sensitivity analysis on these parameters in the current study limits our understanding of the model's robustness under varying optimization configurations.

Finally, the study does not account for adversarial scenarios, such as label noise or input perturbations. Evaluating the model’s performance under such conditions would offer deeper insights into its reliability in real-world environments, where data imperfections are common.

5. Conclusion

This research project brought together CNN hybridization with ACO to produce enhanced results for image segmentation accuracy as well as robustness. The integration of CNN features with ACO adaptive hyperparameter optimization in our method produces a solution that effectively handles problems such as noise and occlusion, as well as complex backgrounds and data distribution issues. The ACO control mechanism repeatedly explores different solutions while also exploiting the most promising ones, so it prevents the model from getting stuck in local optima, thus producing better segmentation results. Our method reaches higher precision and recall levels in conjunction with better overall accuracy than conventional CNN-based segmentation models, specifically when operating in complex settings. The research advances image segmentation by combining deep learning with adaptive optimization methods to develop efficient analysis techniques that benefit different computer vision applications. To increase feature extraction in complex images, future research may integrate attention processes like self-attention or transformer blocks into the model. In medical imaging and remote sensing, multi-modal data, like depth or thermal imaging, may improve segmentation accuracy. Lightweight ACO variations or hardware-friendly optimization algorithms may improve the model for real-time or embedded applications. Finally, assessing the impact of input noise and adversarial interference on the model's output may uncover potential vulnerabilities and guide improvements in its reliability.

References

[1] S. Minaee, Y. Y. Boykov, F. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image Segmentation Using Deep Learning: A Survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–1, 2021, https://doi.org/10.1109/TPAMI.2021.3059968.

[2] G. Bathla *et al.*, "Autonomous Vehicles and Intelligent Automation: Applications, Challenges, and Opportunities," *Mob. Inf. Syst.*, vol. 2022, pp. 1–36, Jun. 2022, https://doi.org/10.1155/2022/7632892.

[3] T. Kanade, "Region segmentation: Signal vs semantics," *Comput. Graph. Image Process.*, vol. 13, no. 4, pp. 279–297, Aug. 1980, https://doi.org/10.1016/0146-664X(80)90030-1.

[4] S. S. Chouhan, A. Kaul, and U. P. Singh, "Soft computing approaches for image segmentation: a survey," *Multimed. Tools Appl.*, vol. 77, no. 21, pp. 28483–28537, Nov. 2018, https://doi.org/10.1007/s11042-018-6005-6.

[5] S. Ghosh, N. Das, I. Das, and U. Maulik, "Understanding Deep Learning Techniques for Image Segmentation," *ACM Comput. Surv.*, vol. 52, no. 4, pp. 1–35, Jul. 2020, https://doi.org/10.1145/3329784.

[6] R. V. Patil and R. Aggarwal, "Edge Information based Seed Placement Guidance to Single Seeded Region Growing Algorithm," *Int. J. Intell. Syst. Appl. Eng.*, vol. 12, no. 12s, Art. no. 12s, Jan. 2024. https://doi.org/10.2139/ssrn.5095458.

[7] R. V. Patil, R. Aggarwal, G. M. Poddar, M. Bhowmik, and M. K. Patil, "Embedded Integration Strategy to Image Segmentation Using Canny Edge and K-Means Algorithm," *Int. J. Intell. Syst. Appl. Eng.*, vol. 12, no. 13s, Art. no. 13s, Jan. 2024. https://doi.org/10.2139/ssrn.5095501.

[8] R. V. Patil and K. C. Jondhale, "Edge based technique to estimate number of clusters in k-means color image segmentation," in *2010 3rd International Conference on Computer Science and Information Technology*, Jul. 2010, pp. 117–121. https://doi.org/10.1109/ICCSIT.2010.5563647.

[9] S. Badrloo, M. Varshosaz, S. Pirasteh, and J. Li, "Image-Based Obstacle Detection Methods for the Safe Navigation of Unmanned Vehicles: A Review," *Remote Sens.*, vol. 14, no. 15, p. 3824, Aug. 2022, https://doi.org/10.3390/rs14153824.

[10] S. Kianoush, S. Savazzi, V. Rampa, L. Costa, and D. Tolochenko, "A Random Forest Approach to Body Motion Detection: Multisensory Fusion and Edge Processing," *IEEE Sens. J.*, vol. 23, no. 4, pp. 3801–3814, Feb. 2023, https://doi.org/10.1109/JSEN.2022.3232085.

[11] R. V. Patil, R. Aggarwal, and S. Shivaji Deore, "Edge Segmentation based on Illumination Invariant Feature Detector Phase Congruency," in *2024 5th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI)*, Jan. 2024, pp. 91–96. https://doi.org/10.1109/ICMCSI61536.2024.00020.

[12] F. Lateef and Y. Ruichek, "Survey on semantic segmentation using deep learning techniques," *Neurocomputing*, vol. 338, pp. 321–348, Apr. 2019, https://doi.org/10.1016/j.neucom.2019.02.003.

[13] Z. Xu, W. Zhang, T. Zhang, Z. Yang, and J. Li, "Efficient Transformer for Remote Sensing Image Segmentation," *Remote Sens.*, vol. 13, no. 18, p. 3585, Sep. 2021, https://doi.org/10.3390/rs13183585.

[14] D. Ngo, S. Lee, Q.-H. Nguyen, T. M. Ngo, G.-D. Lee, and B. Kang, "Single Image Haze Removal from Image Enhancement Perspective for Real-Time Vision-Based Systems," *Sensors*, vol. 20, no. 18, p. 5170, Sep. 2020, https://doi.org/10.3390/s20185170.

[15] K. C. Ciesielski, P. A. V. Miranda, A. X. Falcão, and J. K. Udupa, "Joint graph cut and relative fuzzy connectedness image segmentation algorithm," *Med. Image Anal.*, vol. 17, no. 8, pp. 1046–1057, Dec. 2013, https://doi.org/10.1016/j.media.2013.06.006.

[16] Y. Zhang, M. Liu, H. Zhang, G. Sun, and J. He, "Adaptive Fusion Affinity Graph with Noise-free Online Low-rank Representation for Natural Image Segmentation," 2021, *arXiv*.

[17] Y. Zhang, Q. Liao, L. Ding, and J. Zhang, "Bridging 2D and 3D Segmentation Networks for Computation Efficient Volumetric Medical Image Segmentation: An Empirical Study of 2.5D Solutions," 2020, *arXiv*.

[18] C. Chen *et al.*, "MA-SAM: Modality-agnostic SAM Adaptation for 3D Medical Image Segmentation," 2023, *arXiv*. https://doi.org/10.1016/j.media.2024.103310.

[19] S. Roy *et al.*, "MedNeXt: Transformer-driven Scaling of ConvNets for Medical Image Segmentation," 2023, *arXiv*. https://doi.org/10.1007/978-3-031-43901-8_39.

[20] C. Li, P. Khanduri, Y. Qiang, R. I. Sultan, I. Chetty, and D. Zhu, "AutoProSAM: Automated Prompting SAM for 3D Multi-Organ Segmentation," 2023, *arXiv*.

[21] S. Chen, Y. Ji, and X. Sun, "Multi-User Detection Based on Improved Cheetah Optimization Algorithm," *Electronics*, vol. 13, no. 10, p. 1842, May 2024, https://doi.org/10.3390/electronics13101842.

[22] M. A. Akbari, M. Zare, R. Azizipanah-abarghooee, S. Mirjalili, and M. Deriche, "The cheetah optimizer: a nature-inspired metaheuristic algorithm for large-scale optimization problems," *Sci. Rep.*, vol. 12, no. 1, p. 10953, Jun. 2022, https://doi.org/10.1038/s41598-022-14338-z.

[23] D. Kumari, P. Pranav, A. Sinha, and S. Dutta, "A hybrid cheetah and grey wolf optimization algorithm for network intrusion detection," *Eng. Res. Express*, vol. 7, no. 1, p. 015256, Mar. 2025, https://doi.org/10.1088/2631-8695/adb47b.

[24] C. Gode, B. M. Nanche, D. Dhabliya, R. D. Shelke, R. V. Patil, and S. Bhosle, "Dynamic neural architecture search : A pathway to efficiently optimized deep learning models," *J. Inf. Optim. Sci.*, vol. 46, no. 4-A, pp. 1117–1127, 2025, https://doi.org/10.47974/JIOS-1896.

[25] J. S. R, R. V. Patil, Y. S, P. M, S. S, and R. Maranan, "Spilled Deep Capsule Neural Network with Skill Optimization Algorithm for Breast Cancer Recognition in Mammograms," in *2025 International Conference on Inventive Computation Technologies (ICICT)*, Apr. 2025, pp. 632–637. https://doi.org/10.1109/ICICT64420.2025.11004784.

[26] R. Sharma, M. Saqib, C. T. Lin, and M. Blumenstein, "Enhanced Atrous Spatial Pyramid Pooling Feature Fusion for Small Ship Instance Segmentation," *J. Imaging*, vol. 10, no. 12, p. 299, Nov. 2024, https://doi.org/10.3390/jimaging10120299.

[27] S. Kumari and A. Malik, "Transforming Sanskrit: Natural Text-to-Speech with Optimized Encoders," *Int. J. Intell. Syst. Appl. Eng.*, vol. 12, no. 23s, pp. 2637–2958, Nov. 2024.

[28] H. Mittal, A. C. Pandey, M. Saraswat, S. Kumar, R. Pal, and G. Modwel, "A comprehensive survey of image segmentation: clustering methods, performance parameters, and benchmark datasets," *Multimed. Tools Appl.*, vol. 81, no. 24, pp. 35001–35026, Oct. 2022. https://doi.org/10.1007/s11042-021-10594-9.