Advances in Artificial Intelligence and Machine Learning; Research 4 (1) 2113-2127 Received 18-01-2024; Accepted 23-03-2024; Published 29-03-2024

# A Contour-Based Feature Extraction Algorithm

**Wenjiao Liuă** liu2wo@mail.uc.edu
*Computer Science Department*
*University of Cincinnati, OH 45221-0030*
*USA*

**Anca Ralescu** ralescal@ucmail.uc.edu
*Department of Computer Science*
*University of Cincinnati*
*Cincinnati, OH 45221-0030,*
*USA*

**Corresponding Author:** Wenjiao Liuă



## Abstract

Human body recognition is a common problem in computer vision. Currently, most artificial intelligence methods are dependent on the scale of the extracted feature values from the target images during classification, thereby limiting the image size. However, in image recognition, the object's contour alone is sufficient for humans to distinguish image content. This paper proposes a feature extraction algorithm based on image contours. By utilizing only the contour features of the target image and combining them with machine learning algorithms for classification, it achieves good results even on relatively complex images.

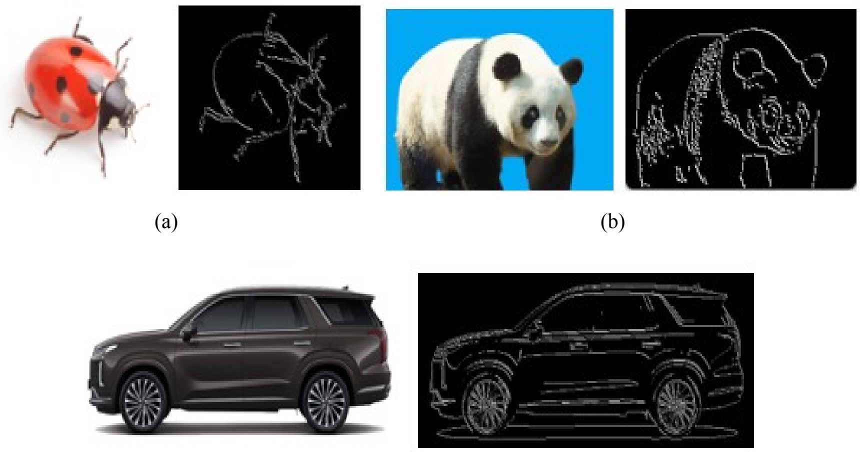
**Keywords:** Classification recognition, Feature extraction, Computer vision, Machine learning.

## 1. INTRODUCTION

In recent years, image recognition and image classification tasks, as essential components of the field of computer vision, have been at the forefront of research in this field. Various methods for image feature extraction have been proposed as prerequisites for these tasks. M. K. Hu introduced the concept of Hu Moments in 1962 [1], which showed promising results in recognizing simple shapes. Torres-Mendez and others, including L. A., Ruiz-Suarez, successfully applied Hu moments to real-world car license plate recognition [2], achieving favorable results. M. Ionescu and A. Ralescu proposed a method to calculate Hamming distance based on features obtained from image texture, color, and other information for measuring image similarity [3]. Gevers T and Smeulders A. W. M. presented an approach for recognition based on lighting and color [4]. The eigenvalues obtained by the SIFT method [5, 6] proposed by Lowe D G. have excellent performance, while the ORB algorithm [7, 8] proposed by Rublee E and others has achieved even better performance. Dalal N and Triggs B's Histograms of Oriented Gradients (HOG) have also demonstrated excellent performance



**Citation:** Wenjiao Liuă and Anca Ralescu. A Contour-Based Feature Extraction Algorithm. Advances in Artificial Intelligence and Machine Learning. 2024;4(1):121.



(a) (b)

(c)

Figure 1: RGB image and contour image. a is Coccinellidae, b is Giant Panda, c is a car.

in human body classification [9–12]. The widely recognized image convolution technique has also significantly contributed to image classification tasks [13–17]. Yet, many of the mentioned approaches also have significant limitations. For example, approaches based on Hu Moments have limitations when dealing with complex shape recognition tasks, as their accuracy may not be satisfactory [18]. High-precision techniques like image histograms and convolution features often result in large-scale feature data, which can pose challenges when dealing with large-sized images, increasing the difficulty of subsequent training and classification tasks. Turning our attention back to the image itself, for a given image, there exists a spatial relationship [19] between each pixel and other pixels. This positional relationship forms the spatial structure of pixel interactions across the entire image. For the task of object recognition, this spatial structure is already sufficient, as illustrated in FIGURE 1. Furthermore, focusing attention on the spatial relationships between pixels themselves can effectively reduce the scale of image features.

This paper introduces an algorithm for *natural feature extraction*, The advantage of the algorithm lies in obtaining smaller size of feature for the same image, extracting features more quickly, while maintaining the classification effectiveness of the classifier. This means that with our algorithm, larger-sized images can be handled, and it can also be applied in video image processing to achieve better real-time performance. From this point on this article is organized as follows: After Introduction (the current section), Section 2 introduces the natural feature extraction algorithm. Section 3 presents the experiments with the proposed algorithm; Section 4 discusses the results. The article concludes with Section 5 which summarizes the results and discusses future directions.

## 2. ALGORITHM

Perceptually, an image generally consists of two major components: shape and color. In many cases, when classifying images, it is often sufficient to consider only the shape, more specifically the contour of the objects/regions in the image. Ignoring RGB color information and converting a

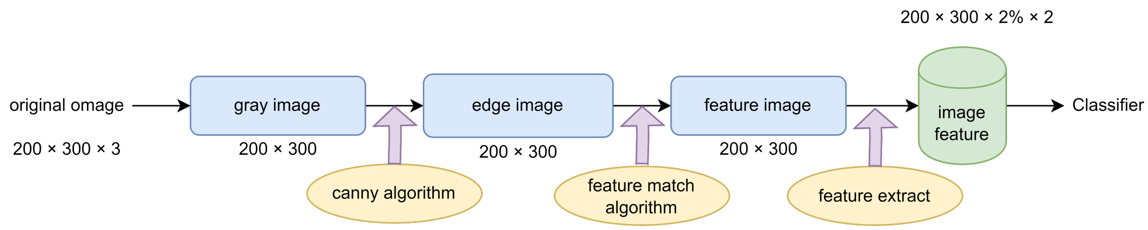



Figure 2: Contour-based feature extraction framework.

color image to grayscale can effectively reduce the computational load for feature extraction. This is one of the reasons why many feature extraction algorithms utilize grayscale images. However, in many scenarios, the grayscale intensity levels are negligible information in the image recognition. In view of this, our algorithm omits the intensity information of the grayscale image. Instead of extracting both contour positions and intensities, it now only preserves the contour positions. This reduction in feature values by one-third is achieved while retaining the essential features necessary for image recognition tasks. Deep learning models or neural network models, widely used recently in computer vision, typically have relatively strict requirements for input data and often necessitate data normalization. The scale of contour feature information varies among different images, and the feature values obtained from simple contour extraction exhibit significant randomness. This randomness poses a major challenge in coordinating contour feature values with machine learning models. To meet the standardization requirements of various models for input data, it is necessary to process the contour information of different images. This ensures that the final contour feature values become a consistently predictable set, addressing the challenge of compatibility with machine learning models.

The natural feature extraction algorithm proposed in this study consists of the following three main steps, illustrated in FIGURE 2:

1 Transform the image into grayscale

2 Extract contour information, standardizing the contour information, and

3 Use Support Vector Machines (SVM) [20, 21] for image classification training. FIGURE 1 illustrates the image processing steps.

## 2.1 Contour Extraction

The Canny algorithm is, to date, the most advanced edge detection algorithm, introduced by John Canny in 1986 [22]. It consists of the following steps:

1 Gaussian filtering for noise removal (using Sobel operators to compute gradients),

2 non-maximum suppression to eliminate edge false positives,

3 double thresholding, and

4 connecting weak edges to strong edges.

The Canny algorithm can detect fine edges [23] in images with low false positive rates, making it a widely used tool for edge detection, the reason for which it is selected for this study. The low and high threshold parameters are set to 70 and 140 in the double thresholding step: these values were obtained experimentally and produced good results after multiple adjustments. They also conform to the parameter combination proportions(H:L = 2:1) discussed by Canny in his paper.

### 2.2 Contour Image Standardization

In image recognition, the intensity levels in grayscale images provide limited valuable information. Therefore, we disregard the brightness and darkness information of contours in the contour image: we consider pixel values greater than 0 in the contour image as effective pixels (EPS).The analysis of many enhanced contour images shows that the proportion of EPS in images is concentrated between 1.5% and 2.5% of the total number of pixels. The issue lies in the fact that the number of EPS varies among different images, while the classifier model imposes strict requirements on the image feature size. It is not feasible to directly use contours as feature values for classification. To address the problem of varying numbers of EPS among different images, our algorithm adjusts the number of EPS in the images without altering their basic contours. This ensures that any image processed by our algorithm yields contour feature values of uniform quantity, facilitating classification by the classifier.To prepare the images for further processing, the EPS in the images are standardized to achieve a uniform 2% proportion. This is done as follows:

Let E denote the number of EPS in the image, T the threshold to be used for standardization. Then E', the result of standardization, is computed as shown in equation 1:

$$E' = \begin{cases} \lfloor \frac{T}{E} \rfloor E & E \leq 0.5T \\ E + a & 0.5T < E < T, a = T - E \\ E - b & E > T, b = E - T \end{cases} \tag{1}$$

When the number of EPS is significantly lower than the threshold, 2E≤T, for all EPS at their surrounding eight adjacent pixel locations, M positions are randomly selected, and their values are set to 255. Here, M is the ratio of the threshold to the number of EPS, and it is rounded down to the nearest integer. This ensures that after processing, 0.5T < E'≤T.

When the number of EPS is higher than the threshold, all EPS are processed. Their surrounding eight locations are analyzed to find other EPS, and this information S (S ranging from 0 to 8) is recorded. The pixels with the highest S values are selected first. Then, one pixel is randomly chosen from this group, and one effective pixel is randomly selected from its neighbors and assigned a value of 0, effectively removing that effective pixel. This process continues until all EPS with the highest S values have been processed once or until the number of EPS matches the threshold. The standardization step is repeated until the number of EPS matches the threshold. When the total number of EPS is less than the threshold but greater than half the threshold, non-EPS are randomly selected according to a standard normal distribution. The number of non-EPS, denoted as N, is equal to the difference between the threshold and the number of EPS. These non-EPS are assigned a value

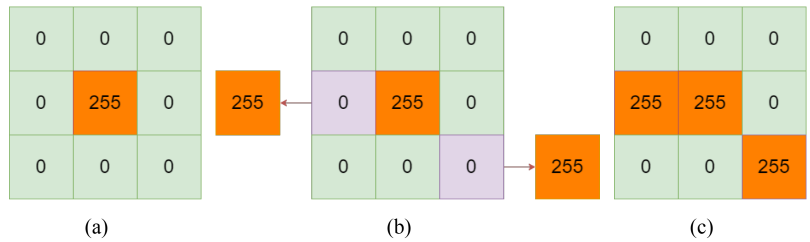



Figure 3: Pixel value change schematic diagram, assuming M=2. (a) Before the operation, (b) shows randomly selecting 2 neighboring pixels, (c) represents the result after the operation.

of 255, thereby transforming them into EPS (FIGURE 3). Following these operations, the number of EPS in the image is adjusted to the given threshold while ensuring that the shape and contour of the image remain largely unchanged. The pseudocode (FIGURE 4), for threshold matching and the algorithm flowchart (FIGURE 5) are shown below:

### 2.3 Feature Extraction

After standardization, the image is systematically traversed in a serpentine fashion on a row-by-row basis. Subsequently, the coordinates corresponding to non-zero pixel values are meticulously documented. The ensuing step involves the normalization of both horizontal and vertical coordinates in accordance with the dimensions of the image. The organized sequence of these coordinates yields the definitive set of feature values encapsulating the image contour. Equation 2 elucidates the formulation for this normalization process. And FIGURE 6, and FIGURE 7, illsutrates these steps and displays the image results at different stages.

$$(X\_nor, Y\_nor) = (X(image.width), Y(image.high)) \tag{2}$$

## 3. DATASET AND CLASSIFICATION TRAINING

To test the model, we collected a positive dataset consisting of 1200 images. This dataset includes 450 images of pedestrians extracted from a segment of security surveillance cameras and 250 images of models. To expand the dataset, we performed data augmentation on a portion of these images by flipping them, resulting in a total of 1200 images in the positive dataset. Similarly, the negative dataset also consists of 1200 images, sourced from random crops of some images and crops from the Flickr2K [24] high-definition dataset. The test dataset comprises 244 images of pedestrians captured by surveillance cameras and 244 negative images. We used four different classifiers to test the extracted features.

```
Algorithm: Threshold Match
Input: strong edge image, difference
Output: result image
Create NL[8] storage for neighbor information
For each pixel p[i][j] in the gray image do:
    If p[i][j] = 255 then
        Record neighbor information according to the number of effective pixels
    End If
End For
If difference < 0 then
    If EPS(strong edge image) <= 0.5 * absolute(difference) then
        For each effective pixel do:
            Randomly choose one non-effective pixel and set it to 255
        End For
    Else if 0.5 * difference < EPS(strong edge image) < absolute(difference) then
        Randomly choose (difference - EPS(strong edge image)) non-effective
pixels and set them to 255
    End If
Else
    Loop:
        I = 8
        s = size of NL[i] - difference
        If I > 0 then
            Randomly choose difference effective pixels from NL[i] and
randomly choose one effective pixel to set to 0
            Stop the loop
        Else
            For every NL[i] do:
                Randomly choose one effective pixel and set it to 0
                I = I - 1
                difference = absolute(size of NL[i] - difference)
            End For
        End If
    End Loop
End If
```

Figure 4: Algorithm. Ignore the brightness information in the grayscale edge image, and adjust all pixel values greater than 0 to 255, creating a strong edge image;The difference represents the difference between the number of EPS in the image and the given threshold. A negative value means an increase in the number of EPS is needed, while the opposite indicates a need to reduce the number of EPS.

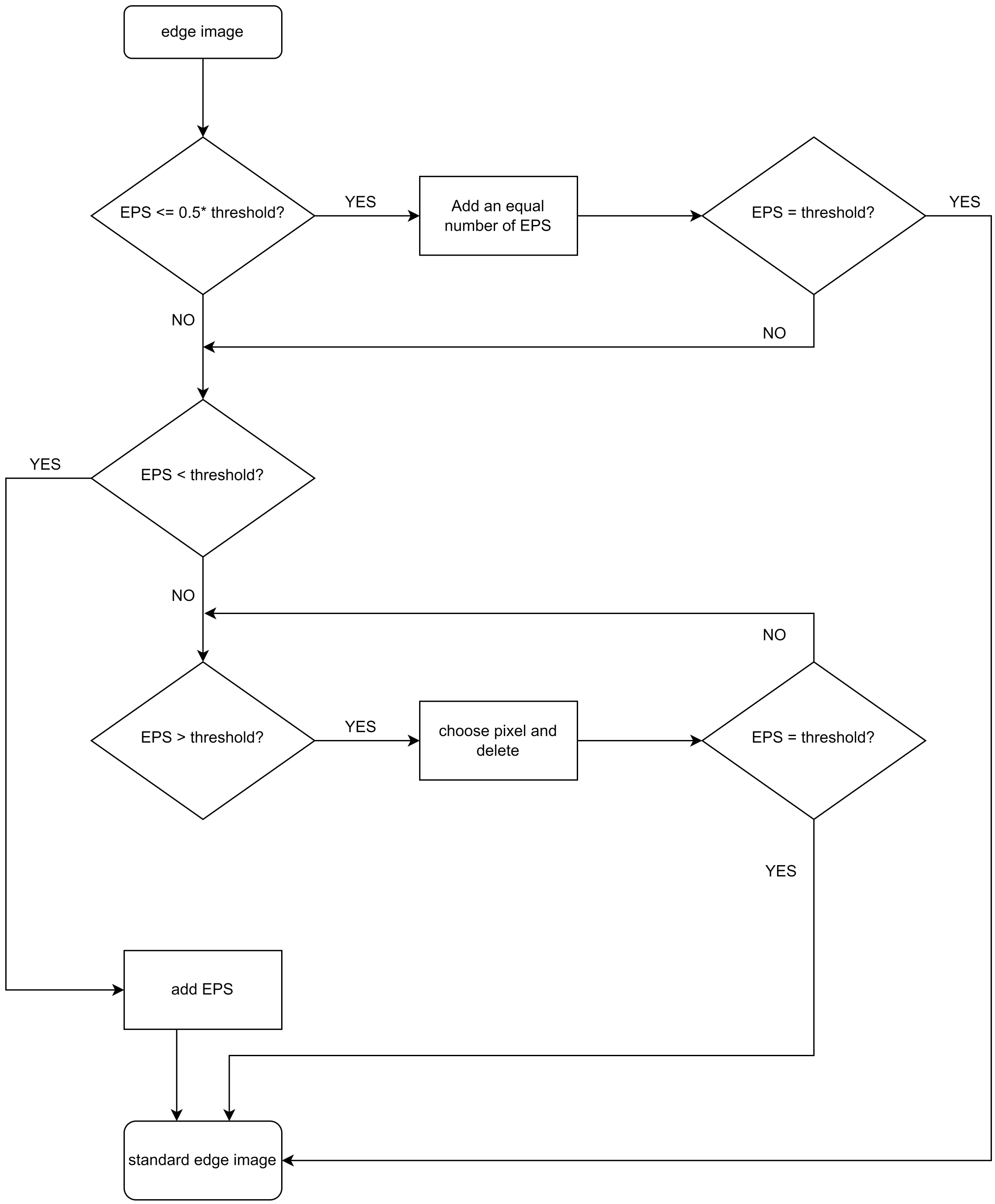



Figure 5: Threshold match flowchart.

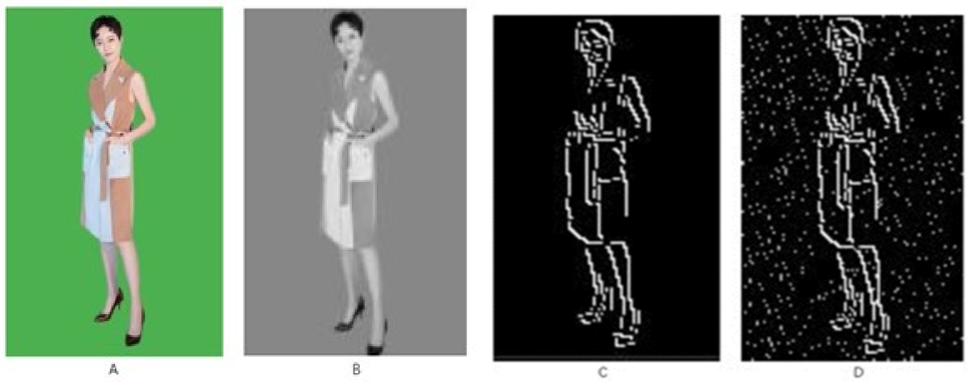



Figure 6: Increase EPS. A is the original image, B is the gray image, C is the edge image, D is the standard feature image.

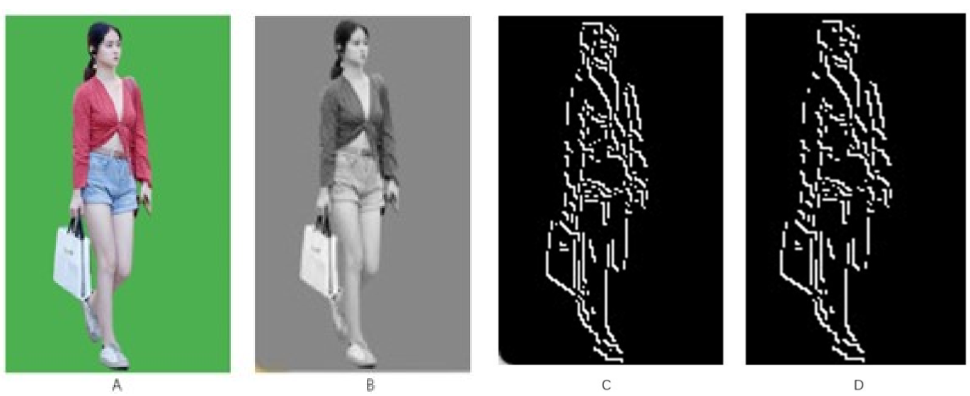



Figure 7: Reduce EPS. A is the original image, B is the gray image, C is the edge image, D is the standard feature image.

### 3.1 Decision Tree

Decision tree [25] algorithm is a method for approximating discrete function values. It is an important classification method that initially preprocesses data, uses induction algorithms to generate readable rules, forms decision trees, and then analyzes new data using these rules. The decision tree classifies data by a series of rule-based judgments. The algorithm has advantages such as fast processing speed and the ability to handle unrelated features.

### 3.2 KNN

KNN (K-Nearest Neighbor) [26] is one of the simplest machine learning algorithms, used for both classification and regression, and is a supervised learning algorithm. Its basic idea is that if the majority of the K nearest neighbors of a sample in the feature space belong to a certain category, then the sample also belongs to that category. KNN algorithm has the advantages of short training time and suitability for nonlinear classification. To avoid the influence of sample imbalance on KNN classification, our dataset has an equal number of positive and negative samples. When testing the dataset, considering the square root rule [27, 28], the value of parameter K is set to 35.

### 3.3 Artificial Neural Network (ANN)

Artificial Neural Network (ANN) [29] is a model that mimics the structure of animal neural networks, typically consisting of three parts: input layer, hidden layers, and output layer. Combined with activation functions, loss functions, and backpropagation algorithms, ANN forms a highly effective artificial intelligence model, widely used in various machine learning tasks. ANN excels in solving nonlinear classification problems. We use a four-layer neural network to classify features, with a sigmoid activation function. The input layer has the same size as the feature set, the first hidden layer has 200 nodes, the second hidden layer has 10 nodes, and the output layer has one node.

### 3.4 Support Vector Machines

Support Vector Machines (SVMs) exhibit strong performance in traditional binary classification problems. Dalal N and Triggs B [9] have also confirmed in their paper that SVMs can deliver excellent performance in image classification tasks. We employ support vector machines for classification training [21] with the Radial Basis Function (RBF) kernel. The kernel coefficient is set to 0.026, and the regularization parameter is set to 1.05.

## 4. TEST RESULT

The test results are shown in TABLE 1, and TABLE 2. From the test results, it can be seen that the features extracted by the algorithm combined with different classifiers have achieved good results. Under multiple effective pixel parameter settings, the decision tree classifier, artificial neural network, and support vector machine all achieved an accuracy of over 90%, indicating the effectiveness of the algorithm. The combination of features extracted by the algorithm and the support vector machine performed the best in classification tasks. We further investigated the performance of the algorithm using the support vector machine model.

In the algorithm, increasing the threshold to increase the number of EPS can also enhance the algorithm's performance. However, when the threshold reaches a certain value, the performance starts to decline. This is mainly because the threshold balances between describing the detailed contour of the image and adding ineffective pixels. Therefore, as the threshold increases, the image features can better reflect the contour information, improving the classifier's performance. However, when the threshold is too high, ineffective pixels may be excessively added, resulting in an abundance of invalid information in the image features and thereby reducing the classifier's performance.

Table 1: Accuracy(defined as (TP + TN) / Total samples) of different effective pixel settings and classifier combinations.

| | Decision tree | KNN | ANN | SVM |
|---|---|---|---|---|
| 800 EPS | 0.907787 | 0.844262 | 0.936475 | 0.950820 |
| 1000 EPS | 0.909836 | 0.854508 | 0.936475 | 0.948770 |
| 1200 EPS | 0.918033 | 0.870902 | 0.924180 | 0.944672 |
| 1500 EPS | 0.913934 | 0.860656 | 0.928279 | 0.940574 |

To evaluate the model's resistance to interference [30], various levels of noise were introduced to the positive class test images. The results are shown in FIGURE 8.

The small impact of the mean on the test results suggests that the algorithm is insensitive to changes in the noise mean, even when the noise mean is not equal to zero. This may indicate that the algorithm exhibits good offset robustness, that is, it is not easily disturbed by variations in the noise mean.

The results also demonstrate that the algorithm exhibits strong resistance to changes in the variance of Gaussian noise. Noise variance represents the strength of the noise, with higher variances indicating stronger noise and lower variances indicating weaker noise. The algorithm maintains its accuracy nearly unchanged under different variances of Gaussian noise, indicating that it is not easily

Table 2: F1-means of different effective pixel settings and classifier combinations.

| | Decision tree | KNN | ANN | SVM |
|---|---|---|---|---|
| 800 EPS | 0.908722 | 0.818182 | 0.935818 | 0.950820 |
| 1000 EPS | 0.910931 | 0.833724 | 0.936864 | 0.948665 |
| 1200 EPS | 0.920000 | 0.854503 | 0.924949 | 0.944330 |
| 1500 EPS | 0.914980 | 0.842593 | 0.927835 | 0.938947 |

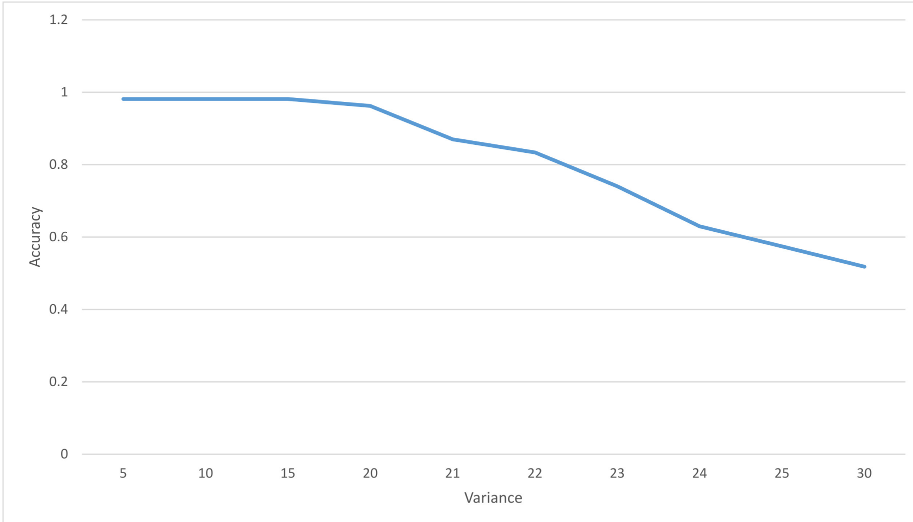



Figure 8: Results: effect of Gaussian noise (mean 0) on accuracy.

Table 3: Runtime for 500 image and SVM classification

| | Contour-Based Feature Extraction | HOG feature extraction |
|---|---|---|
| Runtime | 8984.35ms | 18782.6ms |
| Accuracy | 0.950820 | 0.964381 |

affected by changes in the intensity of Gaussian noise. This is reasonable because Gaussian noise can blur the image, but blurring within a certain range does not significantly affect the image contour, thus not affecting the algorithm's results. Specifically, it maintains almost constant accuracy, close to 1, for noise variance lower than 22, suggesting that the algorithm performs relatively steadily until the variance exceeds 22. For noise variance greater than 22 but less than 30, the accuracy drops in an approximately linear fashion, but it is still greater than 50%, which again suggest that the algorithm remains robust to variations in noise intensity within a certain range.

In summary, these characteristics indicate that the algorithm performs well in the presence of Gaussian noise, showing resilience to changes in both mean and variance. This aspect is highly valuable for many practical applications that involve handling noisy data. By contrast to the Gaussian noise, the salt-and-pepper noise image test reveals that the algorithm is highly sensitive to salt-and-pepper noise. Indeed, the addition of even very low levels of salt-and-pepper noise leads to a rapid decline in the algorithm's accuracy. This sensitivity is due to the method used to adjust the number of EPS, which involves adding random pixel values. This approach makes salt-and-pepper noise within the image susceptible of being mistakenly interpreted as valid information, thereby reducing the algorithm's effectiveness. FIGURE 9 illustrates the accuracy deterioration under the salt-and-pepper noise within the range [0.002, 0.03].

In classification and object detection tasks, runtime and deployment cost are also crucial factors. Handcrafted features [31, 32] have clear advantages over deep learning features when it comes to lightweight deployment on devices with limited runtime and computational resources. The His-

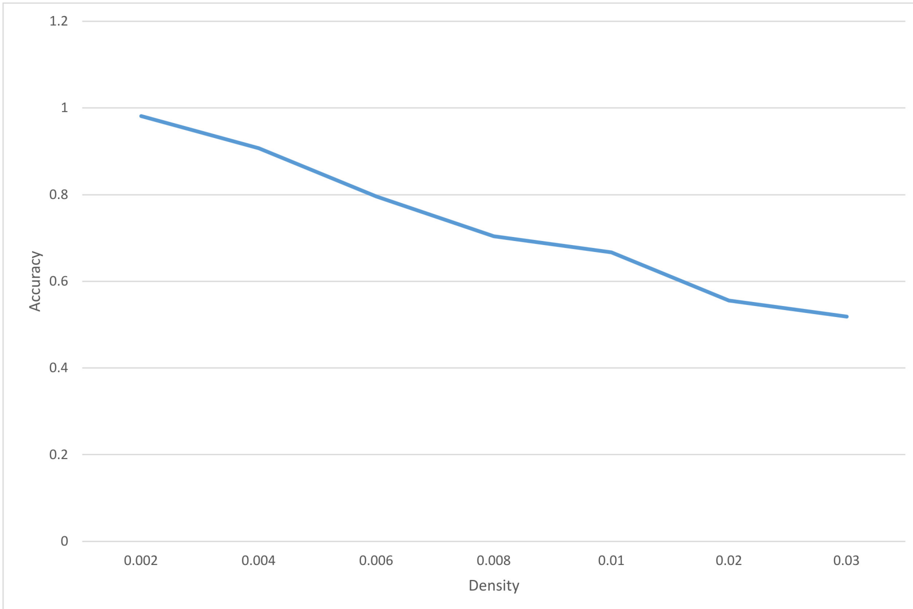



Figure 9: Test results under different densities of salt and pepper noise.

togram of Oriented Gradients (HOG) [9] is a feature descriptor used for object recognition. In handcrafted features algorithms, HOG features are one of the best-performing algorithms in object detection, especially in human detection. When combined with Support Vector Machines (SVM), it becomes one of the most widely applied algorithms for pedestrian detection. To measure the time efficiency of the algorithm, we used the HOG feature extraction algorithm for comparison. TABLE 3 displays the test results of our algorithm and the HOG algorithm after extracting features and using SVM for classification. Although our algorithm slightly lags behind HOG in accuracy, it has a significant advantage in the time required for feature computation. Our algorithm exhibits a slight disadvantage in terms of accuracy, but it performs exceptionally well in terms of runtime.The runtime required to extract features from 500 images of size 200x300(number of EPS = 800). The results indicate that our algorithm excels in feature extraction speed, being 52% faster than traditional HOG feature extraction methods. This significant improvement in time efficiency may have a positive impact on the real-time requirements of practical applications, making our algorithm more appealing.

## 5. CONCLUSION

We introduce the concept of effective pixel, with the aim of defining natural image features, namely object contours, and an image classification algorithm based on such features was proposed. The feasibility of the contour feature extraction algorithm, has been illustrated for image recognition tasks, Especially in images with complex target contours but simple backgrounds. While it is sensitive to salt-and-pepper noise, the algorithm exhibits good resistance to Gaussian noise. Furthermore, due to its computational simplicity, it can be applied to larger-scale images. Further research could investigate an alternative approach for defining EPS such that the contour features are not as sensitive to salt-and-pepper noise. The algorithm has several features which recommend it, including speed and noise resistance. Given its speed, the next step would involve combining it with the graph difference model [33, 34] for real-time video recognition, potentially achieving real-time results in the processing of larger-sized videos.

*Data Acquisition: https://github.com/AscendedBeings/Human-Detection-Classification-Dataset-1./*

## References

[1] Hu MK. Visual Pattern Recognition by Moment Invariants. IRE Trans Inf Theor. 1962;8:179-187.

[2] Abril Torres-Mendez LA, Ruiz-Suarez JC, Sucar LE, Gómez G. Translation, Rotation, and Scale-Invariant Object Recognition. IEEE Trans Syst Man Cybern C (Applications and Reviews). 2000;30:125-130.

[3] Ionescu M, Ralescu A. Fuzzy Hamming Distance in a Content-Based Image Retrieval System. IEEE Int Conf Fuzzy Syst. 2004;3:1721-1726. (IEEE. Cat. No. 04CH37542).

[4] Gevers T, Smeulders AWM. Color-Based Object Recognition. Pattern Recognit. 1999;32:453-464.

[5] Lowe DG. Distinctive Image Features From Scale-Invariant Keypoints. Int J Comput Vis. 2004;60:91-110.

[6] Lowe DG. Object Recognition From Local Scale-Invariant Features. In: Proceedings of the seventh IEEE international conference on computer vision. Ieee Publications. 1999;2:1150-1157.

[7] Rublee E, Rabaud V, Konolige K, Bradski G. ORB: An Efficient Alternative to SIFT or SURF. In: International conference on computer vision. Ieee Publications; 2011;2564-2571.

[8] Calonder M, Lepetit V, Strecha C, Fua P. Brief: Binary Robust Independent Elementary Features. In Computer Vis–ECCV. Proceedings, Part IV: 11th European Conference on Computer Vision, Heraklion, Crete, Greece. Springer. 2010;11:778-792.

[9] Dalal N, Triggs B. Histograms of Oriented Gradients for Human Detection. In: IEEE computer society conference on computer vision and pattern recognition (CVPR'05). Ieee Publications; 2005;1:886-893.

[10] https://theses.hal.science/tel-00390303/document

[11] Zhu Q, Yeh MC, Cheng KT, Avidan S. Fast Human Detection Using a Cascade of Histograms of Oriented Gradients. In: IEEE computer society conference on computer vision and pattern recognition (CVPR'06). IEEE Publications; 2006:1491-1498.

[12] Wang X, Han TX, Yan Shuicheng. An Hog-Lbp Human Detector With Partial Occlusion Handling. In: 12th international conference on computer vision. IEEE Publications. 2009:32-39.

[13] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-Based Learning Applied to Document Recognition. Proc IEEE. 1998;86:2278-2324.

[14] Krizhevsky A, Sutskever I, Hinton GE. Imagenet Classification With Deep Convolutional Neural Networks. Adv Neural Inf Process Syst. 2012;25.

[15] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016:770-778.

[16] Huang G, Liu Z, Van Der Maaten L, Weinberger KQ. Densely Connected Convolutional Networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017:2261-2269.

[17] Ren S, He K, Girshick R, Sun J, Faster R-Cnn. Towards Real-Time Object Detection With Region Proposal Networks. Adv Neural Inf Process Syst (NIPS 2015). 2015;28.

[18] Liang CH, Chang Q. Weighted Modified HU Moment in Human Behavior Recognition. Adv Mater Res. 2013;765-767:2603-2607.

[19] Noma A, Graciano ABV, Cesar RM, Consularo LA, Bloch I. Interactive Image Segmentation by Matching Attributed Relational Graphs. Pattern Recognit. 2012;45:1159-1179.

[20] Cortes C, Vapnik V. Support-Vector Networks. Mach Learn. 1995;20:273-297.

[21] Platt JC. Fast Training of Support Vector Machines Using Sequential Minimal Optimization. Adv Kernel Methods. 1999:185-208.

[22] Canny J. A Computational Approach to Edge Detection. IEEE Trans Pattern Anal Mach Intell. 1986;8:679-698.

[23] McIlhagga W. The Canny Edge Detector Revisited. Int J Comput Vis. 2011;91:251-261.

[24] Lim B, Son Sanghyun, Kim Heewon, Nah S, Lee KM. Enhanced Deep Residual Networks for Single Image Super-Resolution. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops. 2017:1132-1140.

[25] Quinlan JR. Induction of Decision Trees. Mach Learn. 1986;1:81-106.

[26] Cover T, Hart P. Nearest Neighbor Pattern Classification. IEEE Trans Inf Theor. 1967;13:21-27.

[27] Hassanat AB, Abbadi MA, Altarawneh GA, Alhasanat AA. Solving the Problem of the K Parameter in the Knn Classifier Using an Ensemble Learning Approach. 2014. ArXiv Preprint: https://arxiv.org/ftp/arxiv/papers/1409/1409.0919.pdf

[28] Zhang S, Li Xuelong, Zong M, Zhu X, Cheng D. Learning K for KNN Classification. ACM Trans Intell Syst Technol. 2017;8:1-19.

[29] Werbos PJ. Generalization of Backpropagation With Application to a Recurrent Gas Market Model. Neural Netw. 1988;1:339-356.

[30] Castleman KR. Digital Image Processing. NJ: Prentice Hall. 1996.

[31] Lin W, Hasenstab K, Cunha GM, Schwartzman A. Comparison of Handcrafted Features and Convolutional Neural Networks for Liver MR Image Adequacy Assessment. Sci Rep. 2020;10:20336.

[32] Tianyu Z, Zhenjiang M, Jianhu Z. Combining CNN With Hand-Crafted Features for Image Classification. In: 14th IEEE international conference on signal processing (ICSP). IEEE Publications; 2018;554-557.

[33] Piccardi M. Background Subtraction Techniques: A Review. In: IEEE international conference on systems, man and cybernetics. IEEE Publications; 2004;4:3099-3104.(IEEE. Cat. No. 04CH37583).

[34] Zhao J, Gong M, Liu J, Jiao L. Deep Learning to Classify Difference Image for Image Change Detection. In: International Joint Conference on Neural Networks (IJCNN). IEEE Publications; 2014:411-417.